

Inverse Rendering for Geometry and Material Reconstruction

Dejan Azinović

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz:

Prof. Dr. Rüdiger Westermann

Prüfer der Dissertation:

1. Prof. Dr. Matthias Nießner
2. Prof. Dr. Marc Stamminger
Friedrich-Alexander-Universität
Erlangen-Nürnberg

Die Dissertation wurde am 26.04.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 08.11.2023 angenommen.

To my family and friends

Acknowledgement

This thesis marks the successful completion of my PhD journey, which would not have been possible without the support of my supervisors, collaborators and colleagues.

I would like to thank Prof. Dr. Matthias Nießner for accepting me into one of the leading computer vision and AI research labs in the world. My utmost gratitude goes to Prof. Dr. Justus Thies for his amazing supervision and detailed feedback during two projects, as well as for all the fun conversations during lunch and coffee breaks. This thesis would not have been possible without him. Furthermore, I would like to thank all remaining co-authors: Tzu-Mao Li for his help with both supervision and implementation of the first project, Anton Kaplanyan, Ricardo Martin-Brualla and Dan B Goldman for their valuable feedback during project meetings, and lastly, Olivier Maury and Christophe Hery for mentoring my internship and continued collaboration on my final project.

I would also like to thank Prof. Dr. Rüdiger Westermann and Prof. Dr. Marc Stamminger for serving as members of the examination committee, as well as the lab's administrative staff for their support during my PhD: Susanne Weitz, Assia Franzman, Sebastian Wohner, Christoph Weiler and Marc Benedí.

All my colleagues will have a special place in my heart for making the journey a more pleasant experience. A big thanks goes to Manuel Dahnert, for many pleasant conversations and being a great friend, to Guy Gafni for the great coffee and memes, and to Yawar Siddiqui for being a human-kind developer and the face of my last paper. I would like to thank Andreas Rössler, Aljaž Božič, Armen Avetisyan, Ji Hou, Norman Müller, Andrei Burov and Artem Sevastopolsky for their invaluable feedback during challenging periods of the PhD. I would also like to thank everyone else in the lab for great conversations, bar nights and boardgame sessions: Barbara Rössle, Can Gümeli, Jiapeng Tang, Lukas Höllein, Peter Kocsis, Shenhan Qian, Shivangi Aneja, Simon Giebenhain, Tobias Kirschstein, Yujin Chen, Dave Zhenyu Chen, Ziya Erkoc, Alexey Artemov, Yinyu Nie, Angela Dai, Christian Diller, Alexey Bokhovkin, David Rozenberszki and Pablo Palafox. The help of colleagues from the labs of Prof. Westermann and Prof. Thuerey, as well as friends not affiliated with the university, also deserves mention. Alexander Kumpf, Kevin Höhle, Tina Höhle, Michael Kern, Marie-Lena Eckert, Steffen Wiewel, Lukas Prantl, Mathias Kanzler and Kiwon Um have organized fun barbecues, bouldering and lasertag events, and made life abroad much more enjoyable.

Finally, I would like to thank my parents, Krešimir and Jasna, on whom I could always count, both in good times and in difficult times. Thank you for your unconditional love and support throughout the PhD.

Abstract

Modern computer vision algorithms have contributed significantly towards digitization of the real world and automatic creation of virtual assets. 3D reconstruction algorithms enable the creation of detailed geometric models of real-world objects, but have limited performance when it comes to visualizing view-dependent appearance, such as specular highlights, of everyday objects. Other methods enable specific applications, such as image synthesis from novel viewpoints or relighting, but do not reconstruct an explicit scene representation suitable for more general use cases, which require simultaneous manipulation of viewpoint, lighting and scene properties. This thesis explores ways of bridging the gap between explicit scene reconstruction algorithms and use case-specific solutions with the goal of providing a means to create a versatile digital representation of real-world scenes.

Existing 3D reconstruction methods often ignore illumination and material properties. These are, however, indispensable for photo-realistic visualization of the reconstructed 3D scenes. This thesis proposes an inverse path tracing formulation to accurately estimate illumination and material properties of objects in a scene, given the scene geometry and a set of RGB observations of the scene, with the corresponding camera poses. Furthermore, a neural rendering-based method is proposed to increase the quality of the geometry reconstruction and estimated camera poses in an indoor scene. This method can be employed as a pre-processing step for any subsequent method that relies on accurate geometry and camera poses.

Finally, we propose a practical end-to-end capture pipeline, comprising both capture setup and reconstruction algorithm, for reconstruction of geometry and physically accurate materials and choose reconstruction of human faces as a use case to demonstrate its effectiveness. In contrast to expensive proprietary studio setups, our proposed setup consists of only a smartphone and inexpensive polarization foils, making it accessible to the broader computer vision community. The setup enables creation of high-quality virtual assets with potential use cases ranging from virtual and augmented reality to movie and video game production.

In conclusion, this thesis discusses digitization of real-world objects and scenes, beginning from the data capture and ending with a complete geometry, material and illumination model of the captured scene. Extensive quantitative and qualitative validation, on both synthetic and real datasets, demonstrates the effectiveness of the proposed methods. Finally, discussion of potential research avenues aims to encourage future progress in digitization of the real world.

Zusammenfassung

Moderne Computer Vision Algorithmen haben einen beachtlichen Beitrag zur Digitalisierung der realen Welt und zur automatischen Erstellung von virtuellen Modellen geleistet. Algorithmen für 3D Rekonstruktion ermöglichen die Erstellung von detaillierten geometrischen Modellen von echten Objekten, haben allerdings einige Einschränkungen bei der Visualisierung von blickwinkelabhängigen Effekten, wie z.B., glänzenden Spiegelungen, die bei Objekten aus der echten Welt allgegenwärtig sind. Andere Methoden ermöglichen spezifische Anwendungen, wie Bildsynthese oder Beleuchtungsänderungen, aber rekonstruieren kein explizites Modell der echten Szene, was zu Einschränkungen in deren Anwendung führt. Diese Dissertation erforscht wie sich diese Lücke zwischen expliziter Rekonstruktion und anwendungsspezifischen Lösungen schließen lässt, um vielseitige Anwendungen für die rekonstruierten Szenen zu ermöglichen.

Existierende Methoden für 3D Rekonstruktion lassen oft Beleuchtung und Materialeigenschaften außer Acht. Diese sind jedoch für eine fotorealistische Visualisierung der rekonstruierten Szene unerlässlich. Diese Dissertation schlägt die Invertierung des Path Tracing Algorithmuses vor, um die Beleuchtung und Materialeigenschaften von Objekten in einer Szene zu schätzen, vorausgesetzt die Geometrie ist bereits bekannt und Farbbilder mit den entsprechenden Kamerapositionen sind gegeben. Darüber hinaus wird eine auf neuronalem Rendering basierende Methode vorgeschlagen, um die Qualität der Geometrierekonstruktion und der geschätzten Kamerapositionen in einer Innenraumszene zu verbessern. Diese Methode kann als Vorverarbeitungsschritt für Methoden, die auf eine genaue Geometrie und Kamerapositionen angewiesen sind, eingesetzt werden.

Abschließend schlagen wir auch eine praktische Methode zur Datenaufzeichnung, einschließlich Algorithmus für Rekonstruktion von Geometrie und Materialeigenschaften, vor. Hierbei wird die Rekonstruktion von menschlichen Gesichtern als Anwendungsfall gewählt. Im Gegensatz zu teuren Studioeinrichtungen, schlagen wir vor ein Smartphone mit günstigen Polarisationsfolien auszustatten, was auch anderen Forschern ermöglicht die Methode anzuwenden. Die vorgeschlagene Methode ermöglicht die Erstellung von virtuellen Modellen sehr hoher Qualität mit potenziellen Anwendungsfällen in der Film- oder Videospieleindustrie.

Diese Dissertation behandelt das Thema der Digitalisierung von Szenen aus der echten Welt, von der Datenerfassung bis hin zur Rekonstruktion eines vollständigen Geometrie-, Material- und Beleuchtungsmodells. Ausführliche Experimente, sowohl an synthetischen als auch an echten Datensätzen, zeigen die Effektivität der vorgeschlagenen Methoden. Zum Schluss werden mögliche Forschungsrichtungen, als Anreiz für zukünftige Fortschritte bei der Digitalisierung der echten Welt, diskutiert.

Contents

Acknowledgement	v
Abstract	vii
Zusammenfassung	ix
I Introduction	1
1 Introduction	3
1.1 Dissertation Overview	5
1.2 Contributions	6
1.3 List of Publications	7
2 Fundamentals and Methods	9
2.1 Sensors	9
2.1.1 Pinhole Camera Model	9
2.1.2 RGB Sensors	10
2.1.3 Depth Sensors	11
2.2 3D Geometry Representation	14
2.2.1 Triangle Mesh	14
2.2.2 Point Cloud	15
2.2.3 Voxel Grid	16
2.2.4 Implicit Representation	16
2.3 Geometry Reconstruction	18
2.3.1 Volumetric Integration	18
2.3.2 Structure-from-Motion and Multi-View Stereo	20
2.3.3 Reconstruction Quality Metrics	21
2.4 Rendering & Light Transport	23
2.4.1 Rasterization	24
2.4.2 Ray Tracing	24
2.4.3 Material Representation	26
2.4.4 Light Representation	27
2.4.5 Differentiable Rendering	28
2.4.6 Neural Rendering	28

II	Inverse Rendering for Geometry and Material Reconstruction	31
3	Inverse Path Tracing for Joint Material and Lighting Estimation	33
3.1	Introduction	34
3.2	Related Work	35
3.3	Method	36
3.3.1	Light Transport Simulation	37
3.3.2	Optimizing for Illumination and Materials	38
3.3.3	Computing Gradients with Path Tracing	38
3.3.4	Multiple Captured Images	39
3.4	Optimization Parameters and Methodology	40
3.4.1	Parametric Material Model	40
3.4.2	Scene Parameterization	41
3.4.3	Emission Parameterization	41
3.4.4	Regularization	41
3.4.5	Optimization Parameters	41
3.5	Results	43
3.6	Qualitative Evaluation of Design Choices	47
3.6.1	Choice of Batch Size	47
3.6.2	Variance Reduction	47
3.6.3	Number of Bounces	50
3.7	Results on Scenes with Textures	50
3.8	Additional Comparison to Data-driven Approaches	51
3.9	Object Insertion in Mixed-reality Settings	52
3.10	Implementation Details	53
3.11	Conclusion	54
4	Neural RGB-D Surface Reconstruction	55
4.1	Introduction	56
4.2	Related Work	58
4.3	Method	59
4.3.1	Hybrid Scene Representation	59
4.3.2	Optimization	61
4.4	Results	62
4.5	Implementation Details	68
4.6	Per-scene Quantitative Evaluations	69
4.7	Ablation Studies	70
4.7.1	Effect of the Photometric Energy Term	71
4.7.2	Number of Input Frames	72
4.7.3	Robustness to Noisy Pose Initialization	72
4.7.4	Batch Size	72
4.7.5	Truncation Size	72
4.8	Comparison to RGB-based methods	73
4.9	Color Reproduction of Classic and NeRF-style Methods	73

4.10	Runtime and Memory Requirements	73
4.11	Conclusion	74
5	High-Resolution Face Capture from Polarized Smartphone Images	83
5.1	Introduction	84
5.2	Related Work	85
5.3	Method	87
5.3.1	Capturing Polarized Data with a Smartphone	88
5.3.2	Geometry Reconstruction	89
5.3.3	Rendering Equation & BRDF	89
5.3.4	Optimization	91
5.4	Results	91
5.5	Calibration	97
5.6	Geometry Estimation	99
5.7	Comparison to Prior Work	100
5.8	Conclusion	101
III	Conclusion & Outlook	103
6	Conclusion	105
7	Limitations and Future Work	107
7.1	Inverse Path Tracing for Joint Material and Lighting Estimation	107
7.2	Neural RGB-D Surface Reconstruction	107
7.3	High-Res Facial Appearance Capture from Polarized Smartphone Images	108
	Bibliography	109
	Appendix	125
A	Open-source Code & Videos	127
A.1	Inverse Path Tracing for Joint Material and Lighting Estimation	127
A.2	Neural RGB-D Surface Reconstruction	127
A.3	High-Res Facial Appearance Capture from Polarized Smartphone Images	127
B	Authored and Co-authored Publications	129
C	Original Publications	131
C.1	Inverse Path Tracing for Joint Material and Lighting Estimation	131
C.2	Neural RGB-D Surface Reconstruction	142
C.3	High-Res Facial Appearance Capture from Polarized Smartphone Images	155
	Acronyms	167
Contents		xiii

List of Tables	169
List of Figures	171

Part I

Introduction

1 Introduction

In the past couple of decades, we have witnessed tremendous technological progress. The Internet allows us to almost instantaneously connect to people in remote parts of the world; movie production has reached the level where it's impossible to distinguish fiction from reality; smartphones give us access to information from all over the globe, regardless of our current location. However, the human desire for technological advancement never rests as we keep pushing the boundaries of what is technologically possible. Some of the technological achievements that we are likely to see in the foreseeable future include self-driving vehicles, photo-realistic VR teleconferencing, photo-realistic digital image synthesis of real-world objects from arbitrary views and automatic digital asset creation from real-world objects for use in movie or video game production. The road is still a long one and paved with difficulties. At the heart of the aforementioned applications lie the research fields of computer vision and graphics. This thesis proposes solutions to some of the problems in these fields in the hopes of bringing us one step closer towards a better future. In particular, this thesis examines potential solutions for 3D geometry reconstruction of real-world objects and estimation of the material properties of the reconstructed objects.

The computer graphics and computer vision fields have seen tremendous advancement in the past two decades. As a brief explanation, one could say that computer graphics is a field that strives to invent better methods for photo-realistic rendering of virtual objects. Given a scene description that comprises geometry, material and light information, the goal is to produce 2D images which cannot be distinguished from images of the real world. Computer vision is a broad field encompassing many smaller sub-fields, such as scene reconstruction and understanding, forgery detection or natural language processing. The focus of this thesis is on scene reconstruction and understanding. One could say that this sub-field of computer vision strives to solve the opposite problem of the problem that computer graphics is trying to solve, i.e., we are trying to understand the real world based on 2D observations. In particular, given a series of 2D observations of a real-world scene, this thesis examines how to reconstruct the geometry of the scene, deduce the material properties of the objects in the scene, as well as the illumination of the scene.

The practical applications of scene reconstruction are numerous. For instance, one could create a 3D model of their apartment to show to potential tenants. Such 3D models are present in datasets, such as the ScanNet [1] or Matterport3D [2] datasets. Several methods have been proposed for creating these 3D models. Methods, such as COLMAP [3] use Structure-from-Motion and Multi-view Stereo to produce a geometric model from a series of color images. Methods such as KinectFusion [4], [5] or BundleFusion [6], which was used to create the ScanNet dataset, make use of a stream of depth images for the geometry reconstruction. The depth images are captured directly using

a hardware sensor, such as the Microsoft Kinect or the Intel RealSense camera, and contain per-pixel range measurements to nearby objects in the scene. By tracking the sensor movement it is possible to reconstruct geometry from these range measurements, as first described in [7]. In contrast to using only color images, reconstructing geometry from range data preserves the real-world scale of the scene. One of the limitations of the aforementioned approaches is the static nature of the reconstruction. Appearance is a function of the view direction, meaning that the same object changes appearance when viewed from different directions (think of a shiny desk when viewed towards a bright light). Recent neural rendering methods, such as NeRF [8], have achieved tremendous progress towards realistic image synthesis. In particular, NeRF optimizes a multi-layer perceptron [9] to predict color and density samples at arbitrary locations in 3D space and for arbitrary view directions. These samples are then combined into a single color value using volume rendering [10] to produce convincing 2D images. Some of the limitations of this approach include the difficulty of extracting high-quality geometry data and the lack of editability. In this work, we attempt to tackle both problems by first obtaining a high-quality geometric representation of a real-world scene and then optimizing the material properties of the objects that are present in the scene, in the hopes of achieving both photo-realistic image synthesis and editability, such as relighting or object manipulation.

We show that the path tracing algorithm, primarily used to synthesize photo-realistic images from a scene description, can be inverted to optimize emission and material parameters from 2D observations. To this end, we implement a differentiable path tracer that can track derivatives of the rendered color w.r.t. material parameters and emission at every bounce. Starting from an initial estimate of these parameters, we can use the derivatives in a stochastic gradient descent optimization loop to iteratively optimize the parameters until our rendered images match the provided target images. This is an extremely ill-posed problem with many local minima. Importance sampling and several regularizers are implemented to guide the optimization towards a plausible solution. The optimized parameters allow us to synthesize images of the scene from novel views, to change the lighting or to manipulate the objects in the scene. The major limitation of this approach is the heavy reliance on high-quality geometry data. Differentiable rendering has remained a relevant line of research to this day, with several later works [11]–[13] proposing more general frameworks for optimizing shape and material.

To improve the quality of the reconstructed geometry, we propose a neural rendering-based approach in which we reformulate the NeRF pipeline to optimize a signed distance field instead of a density field. We found that jointly learning geometry and color can fill holes and thus improve geometry reconstruction in areas with missing observations from the depth sensor. Furthermore, by also jointly optimizing per-frame camera poses we can significantly reduce misalignment artifacts, such as loop closure artifacts. This naturally leads to improved image synthesis results if the reconstructed geometry is used as a base for material and illumination reconstruction.

Finally, we show a practical application of some of the aforementioned techniques in an end-to-end system for human face reconstruction. We propose a low-cost capture

setup for human faces. Using only 2D color observations, we reconstruct the geometry of the face along a set of high-resolution skin textures which can be used to relight the face and to visualize it from arbitrary viewpoints. Our proposed solution democratizes the creation of digital assets for use in, e.g., movie or video game production. While the focus of this work was on human faces, the described techniques can be extended to various categories of objects.

This thesis describes a method for material and lighting estimation in indoor scenes given 2D observations along with their corresponding camera poses and a base geometry. It further proposes a method for obtaining such a high-quality base geometry and camera poses. Finally, we show a practical implementation of the discussed techniques in an end-to-end system that is focused on generating digital human assets. In summary, we provide the following contributions to the field of 3D reconstruction:

- An end-to-end differentiable inverse path tracing formulation for joint material and lighting estimation, with a flexible stochastic optimization framework with extensibility and flexibility for different materials and regularization terms.
- An RGB-D scene reconstruction framework that cleverly employs color observations to fill gaps in the geometry reconstruction, caused by missing depth information. In addition, our camera refinement technique is able to compensate for misalignments in the input data, resulting in state-of-the-art reconstruction quality which we demonstrate on synthetic, as well as on real data from the ScanNet [1] dataset.
- A commodity capture setup that combines a smartphone’s camera and flashlight with polarization foils. The polarization allows us to separate diffuse from specular parts, leading to a high-quality reconstruction of diffuse albedo, specular albedo and normal maps.

1.1 Dissertation Overview

This thesis is structured in 7 chapters that are grouped into three parts as follows:

- **Part I:** Introduction (Chapters 1–2)
 - Chapter 1 (Introduction) emphasizes the importance of 3D scene reconstruction, introduces recent developments and describes our contributions to the community.
 - Chapter 2 (Fundamentals and Methods) explains basic concepts of 3D reconstruction and light transport to assist in understanding of this thesis.
- **Part II:** Inverse Rendering for Geometry and Material Reconstruction (Chapters 3–5)
 - Chapter 3 introduces our work on inverse path tracing for joint material and lighting estimation for indoor scenes.

- Chapter 4 introduces our work on neural RGB-D surface reconstruction for scene reconstruction using both color and depth observations.
- Chapter 5 introduces our practical capture setup for reconstruction of high-resolution textures for human faces and presents a real-world use case for techniques described in chapters 3 and 4.
- **Part III: Conclusion & Outlook (Chapters 6–7)**
 - Chapter 6 (Conclusion) summarizes our proposed methods and concludes our contributions.
 - Chapter 7 (Outlook) discusses limitations of our proposed methods and possible future research directions.

1.2 Contributions

This thesis proposes solutions to several current challenging problems in the field of 3D scene reconstruction and novel-view synthesis. Current geometry reconstruction algorithms do not provide reconstructions that are directly suitable for photo-realistic image synthesis. On the other hand, novel-view synthesis and relighting methods have limited editability. We propose a method for estimating physically-based material properties in real-world scenes. These can be used in conjunction with the reconstructed geometry to synthesize images of a scene from novel views or under novel illumination, while still allowing scene manipulation or extraction of digital assets. Furthermore, we improve upon existing geometry reconstruction approaches and also present an end-to-end system for geometry and material capture in the context of creating digital assets of human faces. Structured by publications, the contributions of this thesis are:

- We propose “Inverse Path Tracing for Joint Material and Lighting Estimation”, an end-to-end differentiable inverse path tracing formulation for joint material and lighting estimation. To achieve this, we also introduce a flexible stochastic optimization framework with extensibility and flexibility for different materials and regularization terms. The correctness of the method was tested on several synthetic and real scenes. Qualitative and quantitative comparisons to baseline methods, that existed at the time of publication, prove the effectiveness of our proposed method. The method development and implementation was done by the first two authors. Discussion with the other co-authors led to the final publication [14].
- We propose “Neural RGB-D Surface Reconstruction”, a neural rendering framework tailored for geometry reconstruction from depth and color images. By effective incorporation of depth measurements into the optimization of a neural radiance field, using a signed distance-based surface representation to store the scene geometry, we are able to improve upon both the geometry reconstruction and estimated camera pose accuracy compared to similar works. This is supported by several

experiments on both synthetic and real data. The method development and implementation was done by the first author. Discussion with other co-authors led to the final publication [15].

- We propose “High-Res Facial Appearance Capture from Polarized Smartphone Images”, an end-to-end framework for facial geometry and high-resolution texture reconstruction. Specifically, we propose a commodity capture setup that combines a smartphone’s camera and flashlight with polarization foils. The polarization allows us to separate diffuse from specular parts, and to reconstruct the user’s face textures, such as diffuse albedo, specular albedo and normal maps. Our proposed coarse-to-fine optimization strategy with mip-mapping further increases sharpness of the reconstructed appearance textures, while the proposed capture setting with the co-located camera and light enables separation of skin properties from illumination, which is of key importance for realistic rendering of faces. The development and implementation of the method was done by the first author. Discussion with other co-authors led to the final publication [16].

1.3 List of Publications

D. Azinović, T.-M. Li, A. Kaplanyan, and M. Niessner, “Inverse path tracing for joint material and lighting estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2447–2456

D. Azinović, R. Martin-Brualla, D. B. Goldman, M. Nießner, and J. Thies, “Neural rgb-d surface reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 6290–6301

D. Azinović, O. Maury, C. Hery, M. Nießner, and J. Thies, “High-res facial appearance capture from polarized smartphone images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 16 836–16 846

2 Fundamentals and Methods

There has been a substantial amount of prior work, not only in the broader computer graphics and computer vision fields, but also specifically on the topics of 3D geometry reconstruction, inverse rendering and novel view synthesis. This chapter presents an overview of a number of fundamental concepts and explains some existing methods which are used extensively in this thesis.

Section 2.1 explains sensors used to capture color or geometry information; section 2.2 explains the various ways of representing 3D geometry; section 2.3 explains existing geometry reconstruction algorithms; section 2.4 introduces the basics of rendering and light transport.

2.1 Sensors

Sensors are hardware devices used to measure real-world properties such as light intensity or distance. They are an integral component of many computer vision algorithms. For example, a method for novel view synthesis will typically rely on data captured by an RGB sensor, while a geometry reconstruction algorithm might use data captured by a depth sensor. In this thesis, we rely on two types of sensors: RGB sensors to capture wavelength-dependent light intensity and depth sensors to capture distance to the nearest object in a scene.

Both types of sensors come in a variety of different sizes and formats. RGB sensors have been an integral part of smartphones for over a decade. While significantly less popular, depth sensors can also be found on some recent smartphone models. Apart from commodity hardware like smartphones, both types of sensors also exist as proprietary hardware that is used for a range of scientific or industrial applications. We explain both RGB and depth sensors in more detail in the following subsections.

The complex optical phenomena inside sensors are difficult to accurately model. However, a simplified model, that will be explained next, is sufficient for many computer vision algorithms.

2.1.1 Pinhole Camera Model

The pinhole camera model is a simplified model of how the real world is perceived through an optical lens and sensor. In fact, instead of a lens, the model assumes an infinitely small aperture. Thus, light passes through a single point of the camera to create an image on the image sensor, as depicted in Figure 2.1. Due to its simplicity, this model is a popular choice for many computer vision and computer graphics algorithms.

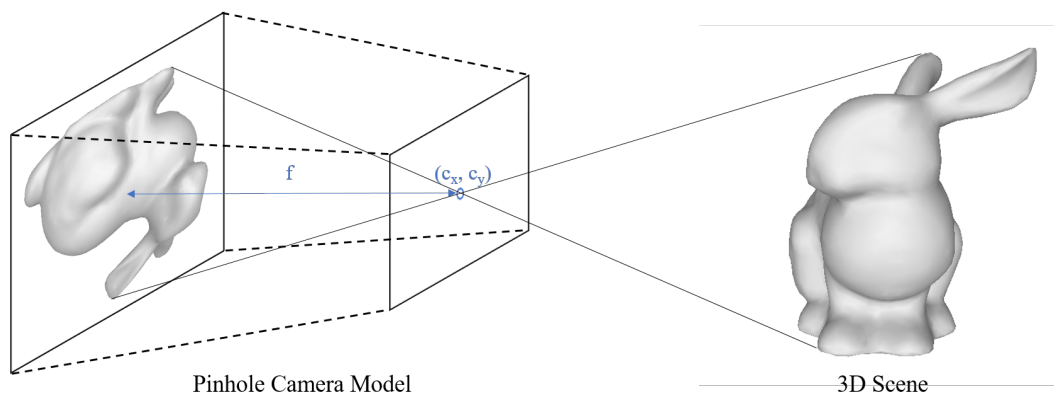


Figure 2.1: Pinhole camera model. The image plane, i.e., image sensor, is located at a distance f away from the aperture, whose location corresponds to the principle point (c_x, c_y) on the image plane. The 3D scene is projected through the aperture onto the image plane. In case of a virtual camera (e.g., in a computer graphics rendering pipeline), the image plane is defined to be at distance f in front of the aperture, as opposed to behind the aperture. In that case, the projected image is no longer flipped w.r.t. the 3D scene.

The image formation process is often described as a projection of the 3D world onto an image plane. This is a perspective projection since faraway objects will appear smaller on the image plane compared to objects closer to the camera. This perspective projection can be mathematically explained by the camera's *intrinsic matrix*:

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}. \quad (2.1)$$

The model parameters are the focal length (i.e., distance between aperture and image plane) f and principal point (i.e., image center) (c_x, c_y) . To allow non-square image pixels, the focal length is split into f_x and f_y . While the physical unit for the focal length is m , in practice it is often expressed in *pixels*. This is a result of the inability to measure the actual physical distance without disassembling the camera. On the other hand, it is possible to estimate the distance in pixels in a calibration procedure. In a similar manner, the principal point is also usually expressed in pixels.

2.1.2 RGB Sensors

RGB sensors are what the term ‘camera’ typically refers to. These sensors measure the intensity of light at specific wavelengths. A typical camera consists of one or multiple lenses that focus light onto the camera’s sensor, which is divided into individual elements, called pixels. Each pixel typically consists of four subpixels: two to capture the green wavelengths of light, one for blue and one for red wavelengths. Such a color filter is called a *Bayer filter* and the process of computing the final pixel color *demosaicing*.

Since larger sensors are capable of capturing more light, the quality of the recorded images depends on sensor size. A larger sensor results in better images, especially in low-light conditions. The intensity of each pixel is directly proportional to the number of photons hitting the camera sensor. One way of controlling this is by changing the aperture size of the camera. However, since the aperture size is directly related to depth of field, changing it may cause parts of the image to become out of focus. The amount of light can also be controlled by changing exposure time, i.e., the length of time the camera's sensor is exposed to light. Longer exposure results in more light being captured, but it can also result in motion blur if the camera or the objects in the scene are moving.

Most digital cameras use a CMOS sensor. An amplifier is attached to each pixel of the sensor and can be controlled by setting an ISO value. This allows the user to influence the pixel intensity before the analog-to-digital conversion of the captured signal. When taking photographs in low-light conditions, it is important to keep in mind that increasing the ISO value also amplifies the noise in the captured data.

Given the benefits and drawbacks of each of the described settings, it is important to carefully adjust the settings to the individual scenes that are being captured. A practical computer vision application where the settings had to be carefully calibrated is described in Chapter 5.

2.1.3 Depth Sensors

Depth sensors measure distance between the camera and objects in the scene that is being recorded. The value that is stored in the pixels of the depth image (also often called depth map) is usually the distance projected onto the optical axis (view direction of the camera). This projected distance is called *depth*. This is also the reason why the sensor is referred to as a *depth* instead of *distance* sensor and the recorded image is a *depth* map instead of a *distance* map.

Depth sensors can be either *passive* or *active*. Passive depth sensing technology is usually based on a stereo capture. Two images are captured by two different sensors with a small offset in space. Matching features need to be detected in both images so that the depth can be computed by triangulation. This works well in highly-textured scenes or scenes with a lot of prominent features. However, in featureless regions, such as white walls in a room, no depth can be computed.

Active depth sensing involves projection of some signal into the scene and the subsequent detection of the reflected signal. The two most common active depth sensing technologies are *structured light* and *time-of-flight*. Both involve projection of IR light into the scene. IR light is chosen, so that the projected light is invisible to the human eye. In the case of structured light, a known IR pattern is emitted into the scene by the camera's IR emitter and depth is computed based on the distorted pattern that is detected by the IR sensor. An example of a structured light depth sensor is the Microsoft Kinect V1. A time-of-flight camera measures, as the name implies, the time it takes for the emitted light to travel between the emitter, scene surface and back to the IR sensor. Since the speed of light is a known constant, it is straightforward to compute distance. Examples of time-of-flight cameras are the Microsoft Kinect V2 and Azure Kinect.

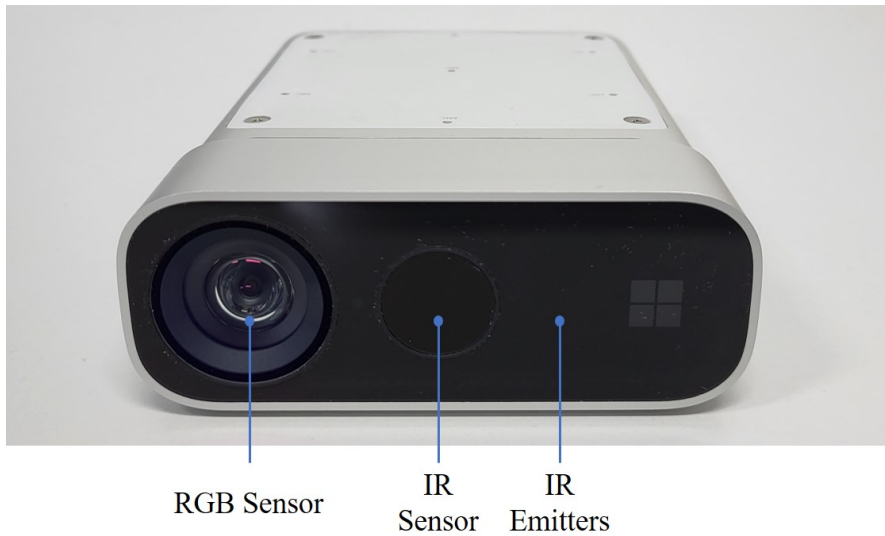


Figure 2.2: The Microsoft Azure Kinect is an RGB-D camera that consists of an RGB sensor, IR emitter and a depth sensor that can detect the reflected IR light.

Active illumination of the scene resolves the issue that passive sensing has in texture-less regions. However, one major drawback is that the sun’s IR light is significantly stronger than the light emitted by the camera, limiting usage to indoor environments. Other limitations include noise in the measured depth, missing measurements on very dark surfaces or surfaces that are thinner than the projected IR pattern. Some cameras, like the Intel RealSense, try to overcome some of these limitations by combining both active and passive depth sensing.

Many consumer-grade devices contain both RGB and depth sensors. This also applies to the two aforementioned Microsoft Kinect cameras and to the Intel RealSense. For this reason, the term RGB-D camera is often used when referring to these cameras. The data that is captured is referred to as RGB-D data. Figure 2.2 shows the Microsoft Azure Kinect RGB-D camera.

The range data recorded by depth sensors is stored in a 2D format (see Figure 2.3 for an example), but since the camera intrinsics are usually known, it can be converted to 3D in a process called *back-projection*. This is achieved by multiplying image-space coordinates with the captured depth and then pre-multiplying with the inverse of the camera intrinsic matrix:

$$\mathbf{p}^{\text{cam}} = K^{-1} \cdot d \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} d \cdot \frac{x - c_x}{f_x} \\ d \cdot \frac{y - c_y}{f_y} \\ d \end{pmatrix}. \quad (2.2)$$

Here, \mathbf{p}^{cam} refers to the 3D coordinates that correspond to the back-projected depth pixel, K is the depth camera’s intrinsic matrix, (x, y) are the image-space coordinates

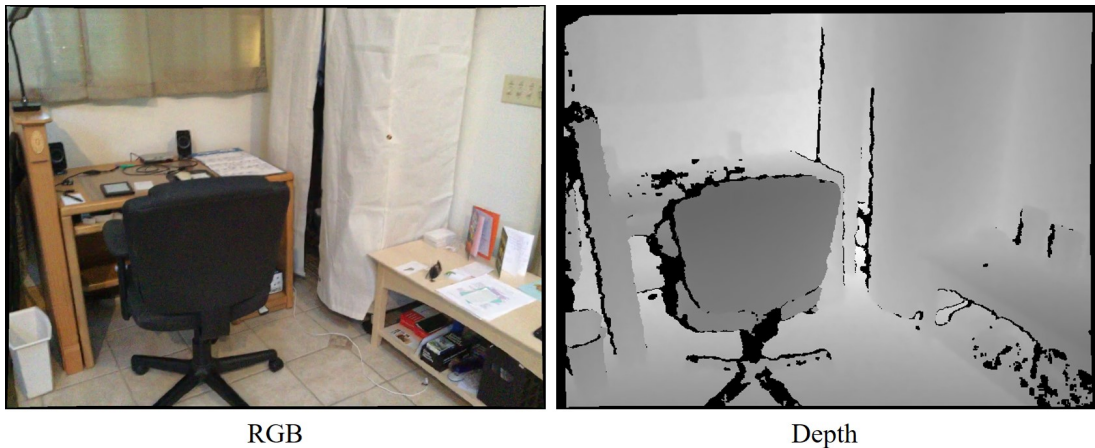


Figure 2.3: Example RGB and depth frame from the ScanNet [1] dataset. The RGB and depth cameras have already been calibrated so that each pixel from the RGB frame corresponds to the depth pixel at the same image coordinates in the depth frame. Furthermore, the captured raw images suffer from lens distortion. In this figure, the images have been undistorted (as visible by the undistortion artifacts on the border of both images), making it possible to use the pinhole camera model. The raw depth data is noisy and has missing regions, primarily in very dark regions, on thing structures and along object edges.

of the back-projected depth pixel, d is the depth value, i.e., the recorded depth, f_x and f_y the focal length, and finally, (c_x, c_y) is the principal point of the depth camera. Back-projection of a depth map results in a point cloud defined in 3D camera space (origin and orientation of the coordinate frame align with the camera origin and orientation). An example of such a point cloud is given in Figure 2.5.

Apart from the inner workings of a camera, its position and orientation in 3D space are also of great importance for computer vision and graphics problems. Given multiple back-projected depth frames, we often need them to be consistently aligned with respect to some reference point in the observed scene. Thus, we would like to further project \mathbf{p}^{cam} from the camera's coordinate frame (i.e., camera-space), to the *world's* coordinate frame (i.e., world-space). This can be achieved by pre-multiplying \mathbf{p}^{cam} with the camera's *extrinsic* matrix, which defines the camera's position and orientation in world-space:

$$\mathbf{p}^{\text{world}} = T \cdot \mathbf{p}^{\text{cam}} = T \cdot \begin{pmatrix} p_x^{\text{cam}} \\ p_y^{\text{cam}} \\ p_z^{\text{cam}} \\ 1 \end{pmatrix}. \quad (2.3)$$

The camera extrinsic matrix T is a 4×4 transformation matrix comprised of a 3×3 rotation matrix R , defining camera orientation and a 3×1 translation vector \vec{t} that defines the camera's position in world-space:

$$T = \begin{pmatrix} R & \vec{t} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} r_x & u_x & l_x & t_x \\ r_y & u_y & l_y & t_y \\ r_z & u_z & l_z & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}. \quad (2.4)$$

The vectors $\vec{r} = (r_x, r_y, r_z)^\top$, $\vec{u} = (u_x, u_y, u_z)^\top$ and $\vec{l} = (l_x, l_y, l_z)^\top$ correspond to the camera's right, up and look (sometimes called view) vectors. Please note that there exist multiple conventions for the actual direction of these vectors (e.g., in OpenGL while the the right and up vectors do point to the right and above the camera, the look vector is oriented from the scene towards the camera).

2.2 3D Geometry Representation

There are several ways to represent 3D geometric data. This section provides a brief explanation of the most popular 3D geometry representations, along with some of their advantages and disadvantages. Visualization methods specific to the particular representations are also discussed.

2.2.1 Triangle Mesh

A triangle mesh is a special case of polygonal meshes. Formally, meshes are graphs comprised of vertices and edges. A triangle mesh is usually implemented by storing a list of vertices, i.e., their 3D coordinates in space, and an index list of triangles (the term 'face' is also often used when referring to a triangle). Each entry in the latter list references three vertices, i.e., three elements from the vertex list. In other words, each element in the index list represents one triangle of the mesh. An example of a triangle mesh is shown in Figure 2.4.

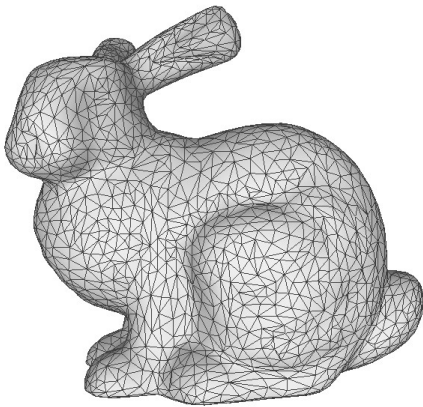


Figure 2.4: Triangle mesh of the Stanford bunny.

Triangle meshes are by far the most popular geometry representation used in computer graphics. Their simplicity allows for dedicated hardware support for accelerated mathematical operations on triangles (e.g., matrix-vector multiplication for transforming the triangle's position in space) on modern graphics cards. Some of the disadvantages are the discretization of the geometry, requiring a very high number of triangles to visualize detail, and the difficulty of computing gradients w.r.t. geometry, which is of high importance for computer vision problems.

2.2.2 Point Cloud

A point cloud is an unstructured collection of surface points in 3D space. Point clouds are frequently used in computer vision problems because they pretty much represent the raw data captured by depth sensors (see Section 2.1.3 for details on back-projecting depth maps to point clouds). Apart from their position in space, each point may also contain other attributes, such as color or surface normal. Figure 2.5 shows a point cloud that has been generated by back-projecting a single depth image.

Point clouds do not store any structure, and thus the connectivity between individual points in a point cloud is undefined. Points are often visualized as squares or circles. If surface normals are present, they can be used to add perspective or shading to the visualized squares or circles.

It is possible to convert point clouds into meshes. If a normal is assigned to each point, Screened Poisson Surface Reconstruction [17] can be employed for this task. In the absence of normals, point clouds that were created from a single depth image can be meshed by connecting points that correspond to neighboring pixels in the depth image, while taking some distance threshold into consideration. This often leads to poor results. However, recent neural networks-based methods like [18] propose more sophisticated solutions, that significantly increase the quality of the reconstructed meshes.

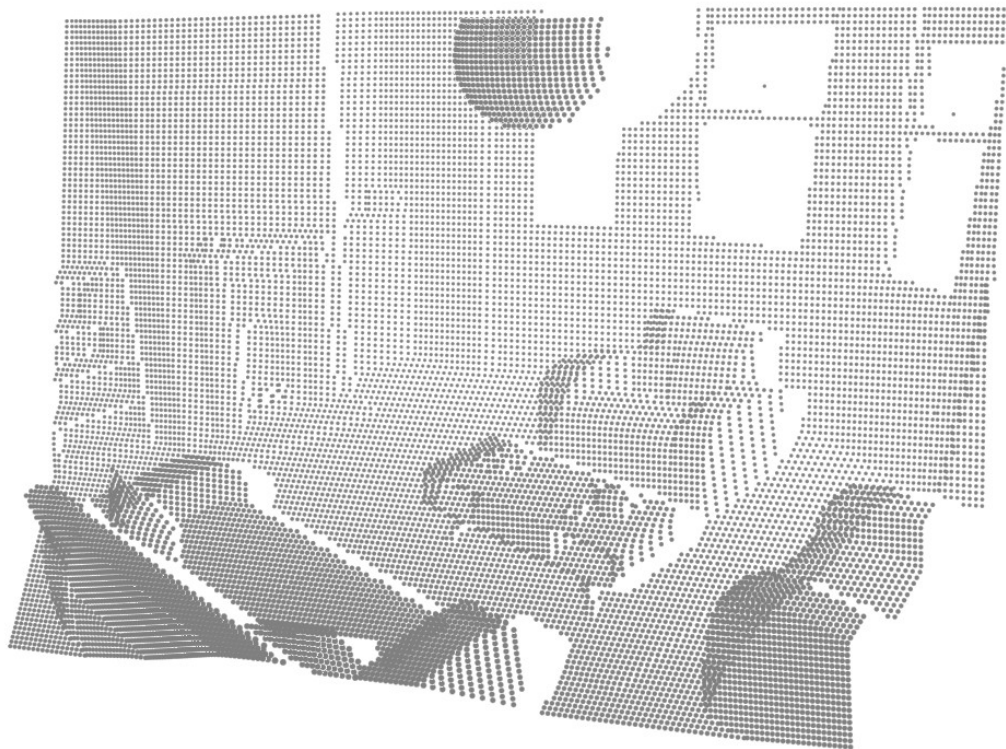


Figure 2.5: Point cloud of a synthetic scene. Back-projecting values from a depth map gives us positions in 3D space.

2.2.3 Voxel Grid

Voxel grids are 3D arrays of elements. The term “voxel” means volume element. One possibility of storing geometry in a voxel grid is the storage of occupancy information in 3D space. This geometry representation is called an occupancy grid and each voxel contains the information whether or not the space in which it resides is occupied by some object or not. An example is shown in Figure 2.6. Another popular way of representing geometry in a voxel grid is to store the distance to the nearest surface in each voxel. This is the discretized version of an implicit surface representation, called distance field, which will be explained in more detail in the next subsection.

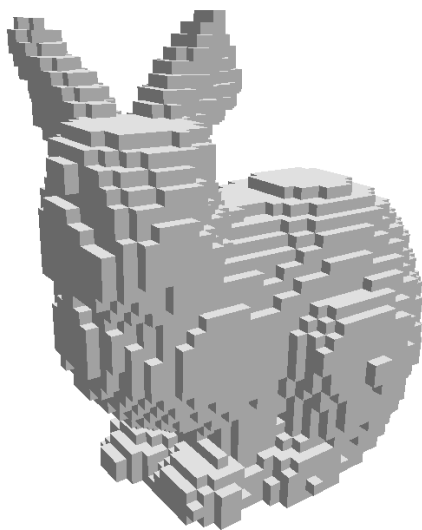


Figure 2.6: Low-resolution occupancy grid of the Stanford bunny.

Voxel grids, as a data structure, are indispensable in many geometry reconstruction algorithms. The major disadvantage of using voxel grids is the prohibitive memory cost. Being a 3D structure, memory grows cubically with the resolution of the grid. This leads to a trade-off between the size of the scene and the detail that can be represented. Since most of the space in many scenes is empty space, alternatives, such as octrees or sparse voxel grids, can be considered to reduce the memory footprint.

Visualization methods differ based on the exact type of underlying information that the voxel grid stores. One way of visualizing geometry stored in a voxel grid is through ray marching. Starting from the camera’s origin, one can skip space equal to the length of one voxel, until reaching the surface. In case the distance to the nearest surface is known, this

information can be used to skip larger distances, such as in the sphere tracing [19] algorithm, visualized in Figure 2.7. Another common approach is to convert the voxel grid into a triangle mesh using the Marching Cubes [20] algorithm.

2.2.4 Implicit Representation

A surface can also be defined implicitly, i.e., as the zero level-set of some mathematical function $F(x, y, z) = 0$. This is analogous to how one could define a sphere of radius 5 and with its center at (a, b, c) as $(x - a)^2 + (y - b)^2 + (z - c)^2 - 5^2 = 0$.

A common implicit representation for geometry is a distance field, which tells us the distance of each point in space to the closest surface. Distance fields come in a number of flavors, such as the signed distance field (SDF) or the unsigned distance field (UDF). An unsigned distance field does not differentiate between the inside and the outside of an object and distances are always positive values. In a signed distance field, positive

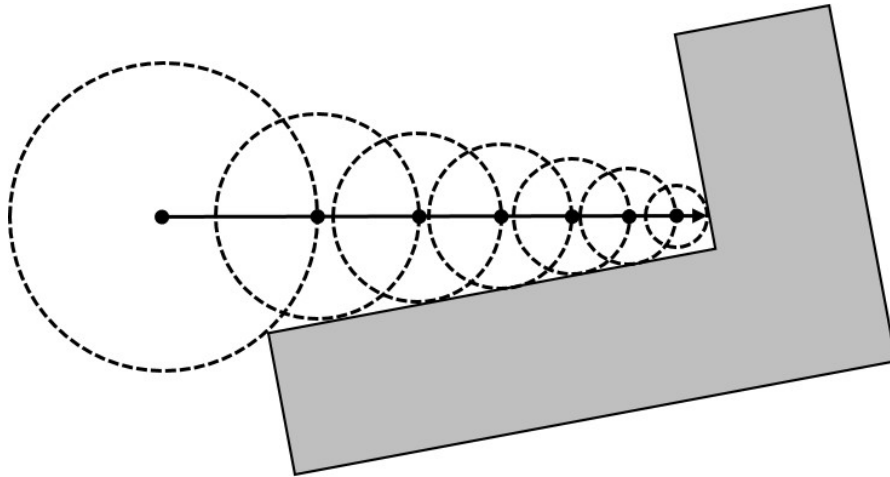


Figure 2.7: Visualization of the sphere tracing algorithm. Starting from the camera origin, we march along a ray towards the nearest surface. At each step, the marching distance is equal to the distance to the nearest surface. The marching stops once the distance falls below a certain threshold.

values typically mean that the point lies outside of an object, while negative values are used to represent the interior region of objects. The surface is thus represented as the zero level-set of the SDF. In some cases, distance to the closest surface is stored only for points within a thin region around the surface, called the truncation region. This representation is then referred to as the truncated signed distance field (TSDF).

Distance fields are a popular surface representation in many computer vision problems. They are easily computed from depth images if the camera location is known. They are also easily differentiable, which is of huge benefit in shape optimization problems.

Historically, the most popular way to store a distance field was a voxel grid. Each voxel would contain the distance to the nearest surface. Other attributes, such as the surface color, could also be stored inside the voxels. With the advent of deep learning and neural rendering, optimizing a neural network, that predicts the distance to the nearest surface at arbitrary points in space, has become a popular alternative. This thesis proposes one such optimization method in Chapter 4.

2.3 Geometry Reconstruction

Geometry can be modeled by hand, using tools, such as Blender [21]. This is the preferred way of creating assets for some applications, like video games. However, in the past decade, there has been a substantial amount of research into reconstructing 3D models from recordings of real-world objects, either for direct use or as a base that digital artists could work with.

Geometry can be reconstructed directly from RGB images using Structure-from-Motion. However, having access to depth images (sometimes also referred to as range images) makes the process significantly easier and provides information about the true scale of an object, as opposed to a reconstruction only from RGB images, which always has arbitrary scale.

2.3.1 Volumetric Integration

Volumetric integration is a method for reconstructing geometry from depth images. First proposed by Curless and Levoy [7], this method iteratively optimizes a signed distance field in a voxel grid over a series of depth images. The method is also often referred to as SDF integration, or TSDF integration in case of a truncated signed distance field. Upon optimization, a mesh can be extracted using Marching Cubes [20].

The signed distance function $D(\mathbf{x})$ of the full scene is optimized by combining individual signed distance functions $d_1(\mathbf{x}), d_2(\mathbf{x}), \dots, d_n(\mathbf{x})$ and weight functions $w_1(\mathbf{x}), w_2(\mathbf{x}), \dots, w_n(\mathbf{x})$ that correspond to depth images $1, 2, \dots, n$. The surface is then extracted at $D(\mathbf{x}) = 0$.

The integration weights need to reflect uncertainty in the range measurements and should be specific to the employed scanning technology. Some typical choices include scaling the weights by the dot product of the view direction and surface normal or by the distance to the surface, since range measurements are less accurate at greater distances. To prevent interference between surfaces on opposite sides of the same object, the weight function falls off within a narrow region behind the surface. To increase computational efficiency, a similar strategy is also applied to both the signed distance and weight functions in front of the surface. It is, however, important that this thin truncation region around the surface that is being optimized is larger than the uncertainty interval in the range measurements.

The final signed distance function $D(\mathbf{x})$ is computed as the weighted sum of the individual distance functions:

$$D(\mathbf{x}) = \frac{\sum w_i(\mathbf{x})d_i(\mathbf{x})}{W(\mathbf{x})}, \quad (2.5)$$

$$W(\mathbf{x}) = \sum w_i(\mathbf{x}). \quad (2.6)$$

The signed distance function is usually computed frame-by-frame in an incremental fashion using the following integration rules:



Figure 2.8: Reconstructed room from the ScanNet [1] dataset.

$$D_{i+1}(\mathbf{x}) = \frac{W_i(\mathbf{x})D_i(\mathbf{x}) + w_{i+1}(\mathbf{x})d_{i+1}(\mathbf{x})}{W_i(\mathbf{x}) + w_{i+1}(\mathbf{x})}, \quad (2.7)$$

$$W_{i+1}(\mathbf{x}) = W_i(\mathbf{x}) + w_{i+1}(\mathbf{x}), \quad (2.8)$$

with $D_i(\mathbf{x})$ and $W_i(\mathbf{x})$ being the cumulative signed distance and weight at point \mathbf{x} after integrating i depth frames.

KinectFusion [4], [5] later proposes a real-time scanning method. The sensor position is tracked using a coarse-to-fine iterative closest point (ICP) algorithm, while the depth measurements are integrated using the previously described approach of Curless and Levoy. Real-time performance is reached with a GPU implementation of the method. Furthermore, it has been shown that tracking the sensor position by aligning the current

depth frame to the growing global scene model has higher accuracy compared to aligning only to the previous depth frame. Follow-up work enables scans of larger scenes [22], uses RGB images for more accurate tracking [6] or enables scanning of dynamic objects [23].

Some of the limitations of these approaches are the reliance on specialized hardware (depth measurement), overly smooth reconstruction results and holes in the reconstruction due to missing depth measurements. Figure 2.8 shows a scene from the ScanNet [1] dataset, that was reconstructed with BundleFusion [6].

2.3.2 Structure-from-Motion and Multi-View Stereo

Structure-from-Motion (SfM) is a method of reconstructing 3D geometry from a series of RGB-only observations. The input are images of an object or scene, while the output are the camera intrinsic and extrinsic parameters and a sparse point cloud of the observed geometry. The algorithm can be divided in three stages.

In the first stage, distinctive features are found in the input images. The second stage tries to match features across multiple images, as shown in Figure 2.9. The final stage then tries to reconstruct the structure or motion by using bundle adjustment, and outputs camera parameters and the matched features in form of a sparse point cloud.

The Multi-View Stereo (MVS) stage takes the SfM output to compute dense depth and normal maps. These can then be fused into a geometric model by algorithms, such as Screened Poisson Surface Reconstruction [17], as shown in Figure 2.10.

Commonly used software for SfM and MVS reconstruction includes COLMAP [3], [24], [25], Agisoft Metashape [26] and RealityCapture [27].

The major disadvantage of the described reconstruction approach is its inability to reconstruct textureless regions. Due to the feature matching step, a large number of views of the same object is needed to provide enough multi-view constraints for the reconstruction. Furthermore, view-dependent effects like specular highlights or changing illumination may lead to incorrect feature matches and degrade the quality of the reconstruction.

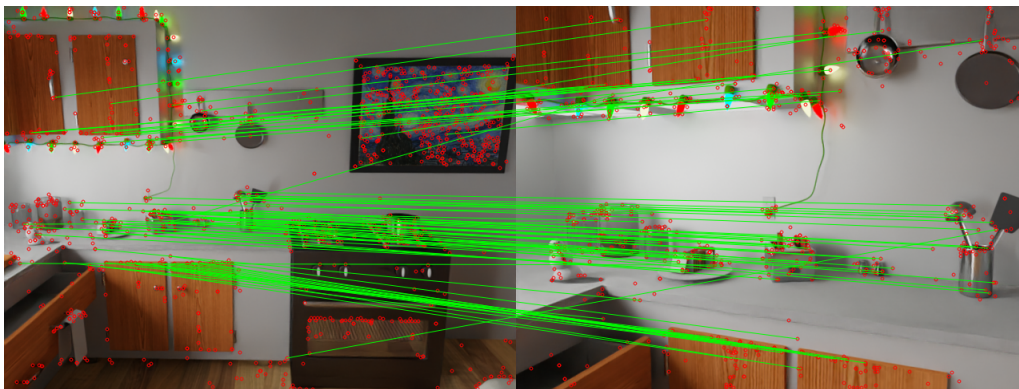


Figure 2.9: The feature matching stage tries to match distinctive features from multiple images. These matches are subsequently used to compute the position of the features in 3D space.

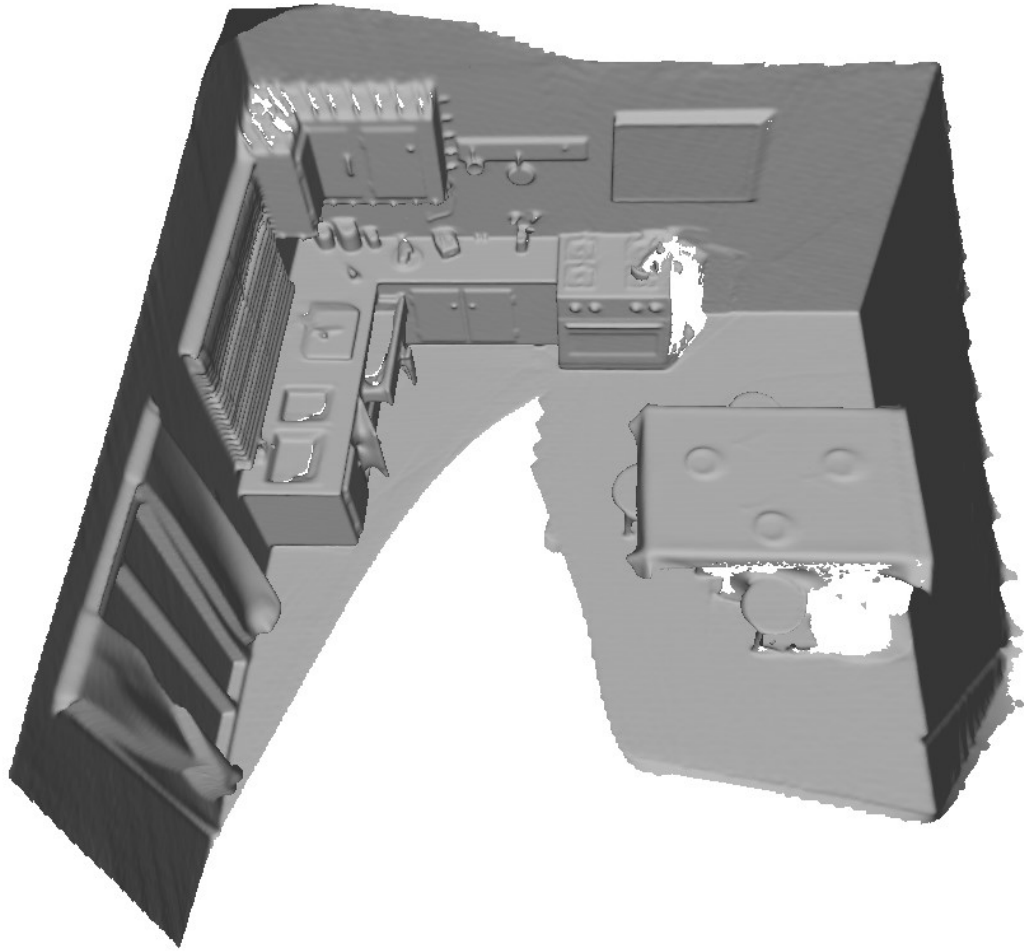


Figure 2.10: Screened Poisson Surface Reconstruction [17] reconstructs a mesh from the point cloud produced by COLMAP.

2.3.3 Reconstruction Quality Metrics

Measuring the quality of geometry reconstruction is not a simple task and there exists no single metric to measure every aspect of it. This is further complicated by the fact that the ground truth geometry of real-world scenes is usually unknown. In these cases, a reconstruction from a method with higher accuracy than the methods being compared is often taken as the ground truth. For example, a high-precision laser scan may be used as ground truth against which to measure the reconstruction quality from a commodity sensor. Different reconstruction methods are typically compared on a series of different metrics, each measuring a different quality aspect of the reconstruction. This section describes the most commonly used metrics, which were also the basis for the quantitative evaluation in Chapter 4.

A simple method to measure reconstruction quality is to first sample a large number of points on the reconstructed and ground truth meshes, and then measure the average ℓ_1 or ℓ_2 distance between points on the reconstructed mesh and the closest point on the ground truth mesh, or vice-versa. A popular metric is the *Chamfer distance* which considers both how well the ground truth has been approximated and how close each of the predicted points is to the target surface. The Chamfer distance between two point clouds \mathcal{P}_1 and \mathcal{P}_2 is calculated as:

$$CD(\mathcal{P}_1, \mathcal{P}_2) = \frac{1}{|\mathcal{P}_1|} \sum_{x \in \mathcal{P}_1} \min_{y \in \mathcal{P}_2} \|x - y\|_2 + \frac{1}{|\mathcal{P}_2|} \sum_{y \in \mathcal{P}_2} \min_{x \in \mathcal{P}_1} \|x - y\|_2. \quad (2.9)$$

One drawback of the Chamfer distance is its susceptibility to outliers. A single outlier point far away from the ground truth might outweigh a completely correct reconstructed shape close to the real surface. There is also no way of measuring noise in the reconstruction. A reconstructed wall with high-frequency noise will have a low Chamfer distance as long as the amplitude of the noise is low. The latter issue can be solved by additionally measuring the *normal consistency* between the ground truth and the reconstruction:

$$NC(\mathcal{P}_1, \mathcal{P}_2) = \frac{1}{|\mathcal{P}_1|} \sum_{x \in \mathcal{P}_1} |\mathbf{n}_x^\top \cdot \mathbf{n}_y^\top|, \quad y = \operatorname{argmin}_{i \in \mathcal{P}_2} \|x - i\|_2, \quad (2.10)$$

where \mathbf{n}_x^\top is the normal of point x and \mathbf{n}_y^\top the normal of the closest point $y \in \mathcal{P}_2$.

Given a voxelized representation of geometry, the *Intersection-over-Union (IoU)* metric can be computed for two voxel grids V_1 and V_2 as:

$$IoU(V_1, V_2) = \frac{|V_1 \cap V_2|}{|V_1 \cup V_2|}. \quad (2.11)$$

IoU depends on voxel size. Thus, if the voxels are too large, there may be a lot of overlap between shapes that do not actually match. If the voxels are too small, slight misalignment errors might lead to a low IoU even for similar shapes. In general, in case of low IoU values it is difficult to reason about performance of competing methods since vastly different shapes may result in similar IoU values.

The *F-score* [28] measures the percentage of points (or surface area) that were correctly reconstructed. It was proposed to overcome drawbacks of existing metrics and is defined as the harmonic mean between *precision* and *recall* at some threshold d :

$$F_d = \frac{2 \cdot \textit{Precision} \cdot \textit{Recall}}{\textit{Precision} + \textit{Recall}}. \quad (2.12)$$

Precision measures the percentage of reconstructed points that are closer than d to the closest ground truth point (i.e., how accurate the reconstruction is), while recall measures the percentage of ground truth points for which the closest reconstructed point is closer than d (i.e., how complete the reconstruction is). Thus, the F-score measures both how accurate and how complete a reconstruction is.



Figure 2.11: An image of the ‘whiteroom’ synthetic scene rendered in Blender [21].

2.4 Rendering & Light Transport

This section gives an overview of rasterization and ray tracing, the two most commonly used methods for rendering scenes into images. Since appearance is a function of the object materials and scene illumination, we also briefly describe the most commonly used ways to represent materials and lighting.

Irrespective of the chosen rendering method, the goal is to solve the rendering equation, first posed by Immel et al. [29] and Kajiyama [30] in concurrent work. The rendering equation defines the outgoing radiance from a point \mathbf{x} in space in a specific direction ω_o as the sum of the emitted light L_e and reflected light L_r at point \mathbf{x} :

$$\begin{aligned} L_o(\mathbf{x}, \omega_o) &= L_e(\mathbf{x}, \omega_o) + L_r(\mathbf{x}, \omega_i, \omega_o) \\ &= L_e(\mathbf{x}, \omega_o) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o) L_i(\mathbf{x}, \omega_i) (\omega_i \cdot \mathbf{n}^\top) d\omega_i. \end{aligned} \quad (2.13)$$

The reflected light is integrated over all incoming light directions ω_i within the hemisphere Ω defined on a surface with normal \mathbf{n}^\top . The reflected light is a function of the bidirectional reflectance distribution function f_r (see Section 2.4.3), incoming light L_i and the dot product between ω_i and \mathbf{n}^\top (also known as the foreshortening effect). While the rendering equation is also a function of the light’s wavelength, this thesis assumes a slightly simplified model that treats all wavelengths in the same manner. Furthermore, the thesis assumes reflective materials and does not deal with light transmission or subsurface scattering effects.

2.4.1 Rasterization

Rasterization is a rendering approach in which geometry is first projected into screen-space coordinates before solving the visibility problem. In case of a triangle mesh, the vertices of each triangle can be projected into screen-space. Knowing the screen-space coordinates of the triangle vertices, it is easy to check whether a pixel lies inside or outside of the triangle. A further depth test is then performed to find out which triangle is closest to the camera in each pixel of the image.

Several optimization strategies are used to make the described procedure as fast as possible. Rasterization is part of modern rendering frameworks such as DirectX, OpenGL or Vulkan. Modern GPUs provide hardware support for all of these frameworks, allowing execution of many geometry and shading operations directly in the hardware. A full rasterization pipeline (like the one shown in Figure 2.12) will usually involve vertex, tessellation and geometry shaders before the actual rasterization step. These are used for transformations and projections between different coordinate systems, as well as for geometry pre-processing. A fragment shader (in some literature pixel shader) will follow the rasterization step and is responsible for computing the final color of the pixel.

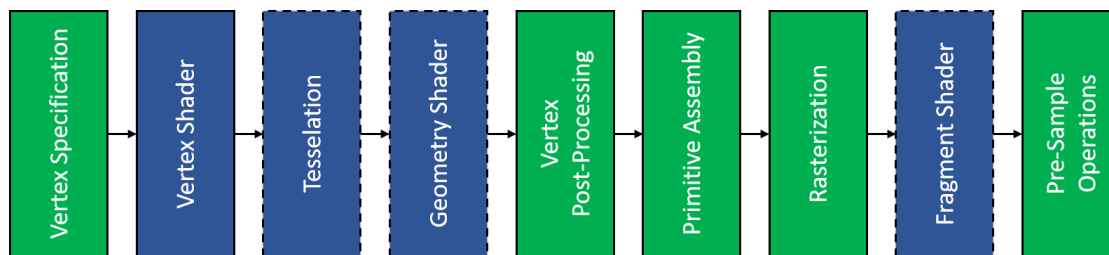


Figure 2.12: Visualization of the full OpenGL rasterization pipeline [31]. Programmable stages of the pipeline are depicted in blue. Stages with a dashed line border are optional.

With the advent of differentiable rasterization frameworks, such as nvdiffrast [32], rasterization has also become a viable approach for solving inverse rendering problems. Specifically, gradients can be computed w.r.t. vertex attributes, the index buffer or textures, enabling joint reconstruction of shape and appearance.

2.4.2 Ray Tracing

Ray tracing is a rendering approach in which a ray is traced into the scene through each pixel of the image that is to be rendered. In contrast to rasterization, where there exists an outer loop over geometry primitives and an inner loop over pixels, ray tracing has an outer loop over pixels and an inner loop over geometry primitives, usually triangles. To solve the visibility problem, it's necessary to check whether or not the ray intersects each of the triangles. The triangle with the shortest intersection distance will be the one that is visible through the particular pixel of the image. To increase performance, the scene geometry is usually stored inside an acceleration structure, such as a bounding

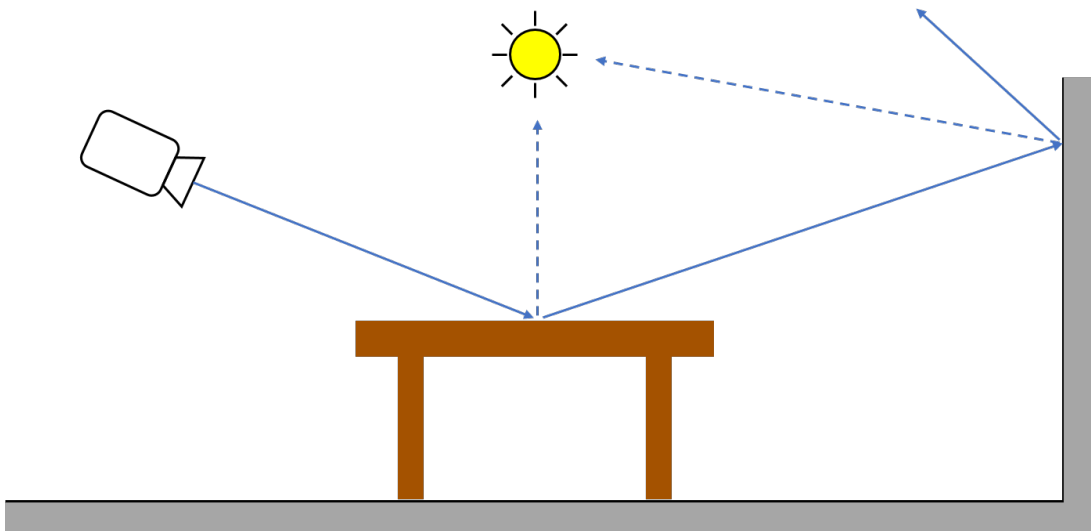


Figure 2.13: Illustration of the path tracing algorithm. A ray is cast from the camera into the scene. At the intersection with the scene, a bounce occurs. The direction of the bounced ray is sampled based on the material (BRDF sampling) at the intersection point. This is repeated until the ray either hits the light sources, leaves the scene bounds (in which case an environment illumination map can be sampled) or the maximum number of bounces is reached. To increase performance, the light source can also be directly sampled and a *shadow ray* cast towards its position.

volume hierarchy [33]. In case a ray does not intersect the bounding volume, there is no need to perform an intersection test with any of the geometry inside that volume.

While conceptually simple, ray tracing has for a long time not experienced widespread use in the graphics industry, apart from movies with a high production cost. The reason for this is the prohibitive runtime of ray tracing-based algorithms, which is several orders of magnitude slower than rasterization. In recent years, this has started to change, as hardware improvements have made ray tracing a viable rendering approach in video games.

The implementation of algorithms for photo-realistic rendering is much easier with ray tracing than with rasterization. Ray tracing allows for a relatively simple physically-based light transport simulation, e.g., using the *path tracing* algorithm. In this algorithm, each ray is shot through a pixel of the image plane into the scene. At the intersection with scene geometry, a new direction for the ray to be bounced in is sampled. This process is repeated until the ray hits a light source or escapes the scene (see Figure 2.13). The surface material and angle between the surface normal and incoming (or outgoing) direction of the ray determines the intensity of the light that is retained after the bounce. In practice, a finite number of bounces is used. This does not present an issue, since the light contribution is usually negligible after a certain number of bounces and implementation tricks like *Russian roulette* ensure that the light transport simulation remains unbiased. By tracing many paths for each pixel in a Monte Carlo simulation, we

obtain the final pixel color as the average amount of light reaching the camera over all of the sampled paths. Several variations and improvements to the path tracing algorithm, such as bidirectional path tracing or Metropolis light transport, were proposed over the past decades to improve performance or to handle difficult cases. Figure 2.11 shows an image rendered with the described algorithm.

2.4.3 Material Representation

An object’s material determines in what way incoming light is reflected off the object’s surface. In computer graphics, the material is usually defined with a bidirectional scattering distribution function (BSDF), a function of two angles: the angle between the surface normal and incoming light direction, and the angle between the surface normal and outgoing light direction. The BSDF serves as a generalization of the bidirectional transmittance distribution function (BTDF), which is a measure of transmitted light and the bidirectional reflectance distribution function (BRDF), a measure of reflected light. If subsurface scattering of the light is taken into account, the term bidirectional scattering surface reflectance distribution function (BSSRDF) is often used instead of the term BRDF.

The scope of this work is limited to non-transmitting surfaces, so the focus will be mainly on the BRDF. For dielectric materials, the BRDF $f(\omega_i, \omega_o)$ is usual split into its diffuse part $f_d(\omega_i)$ and specular part $f_s(\omega_i, \omega_o)$. The diffuse part represents the portion of light that penetrated the surface, bounced within the surface, and then exited the surface again (not necessarily at the same point). The specular part represents the portion of the light that bounces directly off the surface, without entering it.

Implementation-wise, f_d is often modulated by a diffuse or base color parameter, while f_s is modulated by a specular color parameter and the surface roughness. Many different BRDFs have been proposed to photo-realistically model various real-world materials. For example, Disney [34], [35] proposes a BRDF of eleven parameters, including parameters such as clearcoat or sheen for precisely modelling a material’s specular response, or a metallic parameter to blend between dielectric and metallic objects.

The diffuse reflection of a material is typically implemented so that it is independent of the view direction. It is defined as:

$$f_d(\omega_i) = \frac{k_d}{\pi}, \quad (2.14)$$

where ω_i is the light direction and k_d the diffuse color of the material. Some implementations, like the aforementioned Disney BRDF, further modulate it by a roughness and view direction parameter to more accurately model the Fresnel effect. Microfacet models are usually used to model specular reflections. Conceptually, each surface is comprised of a set of microfacets with their individual surface normal, thus defining the material roughness. Some practical implementations include the Cook-Torrance BRDF [36]:

$$f_s(\omega_o, \omega_i, \alpha) = k_s \frac{D(\omega_o, \omega_i, \alpha)G(\omega_o, \omega_i)F(\omega_o, \omega_i)}{4 \cdot \omega_o \cdot \omega_i}, \quad (2.15)$$

where k_s is the specular gain, ω_o is the outgoing light direction, ω_i the incoming light direction and α the roughness of the material. D is the distribution term that describes the statistical distribution of the microfacet normals, G the shadowing-masking term that describes what portion of the microfacets is visible from both ω_o and ω_i and F the Fresnel term that defines the ratio between specular reflection and total incoming light. Multiple mathematical models have been proposed for each of these terms, e.g., the GGX model [37] for D and G or the Fresnel-Schlick approximation [38] for F .

2.4.4 Light Representation

Light is what is responsible for perception of color in a scene. A light source emits light, which interacts with the scene, before hitting the camera sensor. There are numerous ways of representing light sources within a 3D scene.

A *point light* source is, as the name implies, a point in space that is emitting light. The intensity of light is equal in all directions and falls off with the square of the distance from the light source. Due to their simplicity and the speed by which objects can be shaded by a point light source in a fragment shader, point light sources have been a popular light representation choice in video games. They can effectively approximate illumination from objects like light bulbs or candles.

A *spot light* is a light source that shines in a cone from a point in space. It is used as approximation for a flashlight or a car's headlights.

Directional light is in theory light emitted by a source that is infinitely far away. In practice, it is used to approximate light coming from a light source that has a finite, but very large distance from the object being illuminated. The most common use of a directional light would be illumination from the sun. Due to the enormous distance between the shaded scene and the sun, it can be assumed that the light hits every point in the scene from the same direction.

The common property of a point, spot and directional light is that they do not have explicit geometry assigned to them. This can be an issue for rendering algorithms, such as path tracing. In these case, 3D objects in the scene can be made emissive, turning them into *area lights*. Area lights also enable effects, such as soft shadows, increasing the photo-realism of the rendered scene.

Very often, a scene is simultaneously illuminated from many directions. This is the case, for example, for an object in an outdoor environment. Strictly speaking, the only source of light is the sun, whose light then bounces within the environment, before finally reaching the object in question. However, since it is impractical to model the entire environment and run a complete light transport simulation, the illumination from the environment can be approximated by an *environment map*. This can be represented as a latitude-longitude 2D texture or as a cube map, where the environment is projected onto faces of a cube. In case the environment illumination is of low frequency, it can also be efficiently approximated using *spherical harmonics*.

2.4.5 Differentiable Rendering

Differentiable rendering refers to the idea of tracking gradients with respect to scene parameters during the rendering process. Gradients can be computed w.r.t. object shape, object material and illumination. These gradients are an essential component in any inverse rendering pipeline that uses an analysis-by-synthesis approach to infer scene parameters from RGB scene observations. Differentiable implementations have been proposed for both ray tracing [11]–[13], [39] and rasterization [32], [40], enabling the reconstruction of shape and material of captured objects. Access to the gradients is, however, only one of the necessary components required for such a reconstruction. The problem of jointly reconstructing shape, material and lighting of an environment is extremely ill-posed with many local minima. Thus, current solutions are often limited to synthetic data and single objects. In some cases object masks may also be required [40]. In Chapter 3, we introduce an inverse path tracing formulation for joint estimation of material and lighting and explain how to solve some of the challenges of reconstructing real-world scenes.

2.4.6 Neural Rendering

Deep learning methods have been successfully utilized to solve complex rendering problems. Thies et al. [41] propose optimizing a neural texture and a corresponding neural renderer for the purpose of novel view synthesis. The texture resides on the surface of the scene geometry and may have an arbitrary number of channels. The neural renderer learns to interpret the sampled texels and render images that match the target images. The neural renderer helps overcome challenges, such as missing geometry, faced by traditional algorithms.

Multi-layer perceptrons (MLPs) have also proven to be capable of learning an implicit scene representation, as shown by the recent neural radiance field (NeRF [8]) model, that has seen tremendous success in rendering scenes from novel views. Optimization of NeRFs requires a set of target RGB images along with their respective camera poses. A ray is generated for each pixel of each image and points are sampled on each ray. The positional encoding of the 3D position of the samples and 2D ray directions is passed through an MLP to predict density and color values at each sample. These are then used in a volume rendering [10] formulation to render the final pixel color, which is compared to the target to compute gradients for the MLP weight updates. The value of the volume rendering integral is approximated with a finite number of samples as follows:

$$C(\mathbf{r}) = \sum_{i=1}^N T_i (1 - e^{-\sigma_i \delta_i}) c_i, \quad (2.16)$$

$$T_i = e^{-\sum_{j=1}^{i-1} \sigma_j \delta_j}, \quad (2.17)$$

where $C(\mathbf{r})$ is the final rendered color for the ray \mathbf{r} , N is the number of samples on the ray, T_i the accumulated transmittance (i.e., probability that the ray will not have

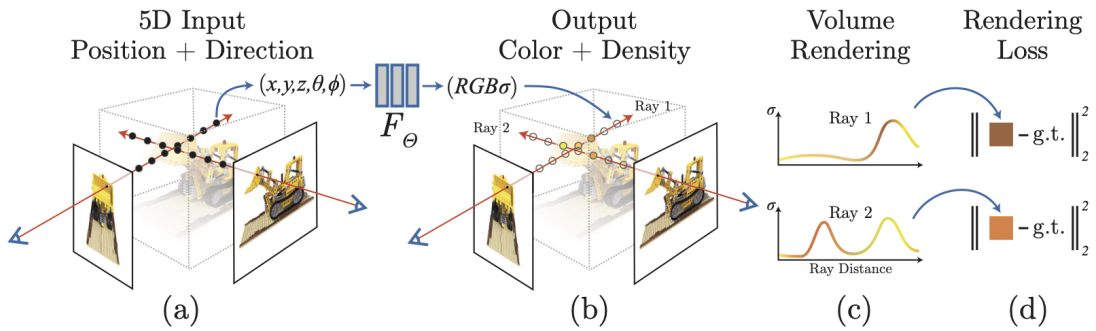


Figure 2.14: NeRF pipeline as visualized in the original paper [8]. A number of points is sampled on rays that are cast through each pixel in a set of input images (a). The position of the points, along with the corresponding ray direction, is passed to the MLP F_θ to obtain per-sample RGB color and density σ values (b). Based on this output, a per-pixel color value is computed using volume rendering (c). Finally, a loss between the rendered and input image is calculated (d) and gradients propagated back to update the weights of F_θ .

terminated) up to point i , σ_i the volume’s density at point i , c_i the color of point i and δ_i the distance between samples i and $i + 1$. The full NeRF pipeline is visualized in Figure 2.14.

While NeRF is able to synthesize photo-realistic images from novel viewpoints, the underlying scene representation has some limitations. For instance, it is not possible to extract smooth geometry from the optimized model. The density-based scene representation is noisy, regardless of the isolevel that is chosen for geometry extraction. This problem persists even in the presence of depth maps and an additional optimization objective that minimizes the difference between predicted and measured depth. We address this in Chapter 4 by reformulating NeRF to use a signed distance field instead of a density field.

Other limitations include the lengthy optimization and rendering times and diminished performance from extrapolated viewpoints (in contrast to interpolated viewpoints, i.e., those that lie between viewpoints used for optimization). Numerous follow-up works try to alleviate these limitations [42]–[46].

Part II

Inverse Rendering for Geometry and Material Reconstruction

3 Inverse Path Tracing for Joint Material and Lighting Estimation

This chapter introduces the following paper:

D. Azinović, T.-M. Li, A. Kaplanyan, and M. Niessner, “Inverse path tracing for joint material and lighting estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2447–2456

Abstract of paper Modern computer vision algorithms have brought significant advancement to 3D geometry reconstruction. However, illumination and material reconstruction remain less studied, with current approaches assuming very simplified models for materials and illumination. We introduce Inverse Path Tracing, a novel approach to jointly estimate the material properties of objects and light sources in indoor scenes by using an invertible light transport simulation. We assume a coarse geometry scan, along with corresponding images and camera poses. The key contribution of this work is an accurate and simultaneous retrieval of light sources and physically based material properties (e.g., diffuse reflectance, specular reflectance, roughness, etc.) for the purpose of editing and re-rendering the scene under new conditions. To this end, we introduce a novel optimization method using a differentiable Monte Carlo renderer that computes derivatives with respect to the estimated unknown illumination and material properties. This enables joint optimization for physically correct light transport and material models using a tailored stochastic gradient descent.

Contribution The method development and implementation were done by the first and second author. Discussions with the co-authors led to the final paper.

Revised layout and minor adaptations. Accepted version of the original publication [14] and detailed disclaimer are included in Chapter C.1 in the appendix.

3.1 Introduction

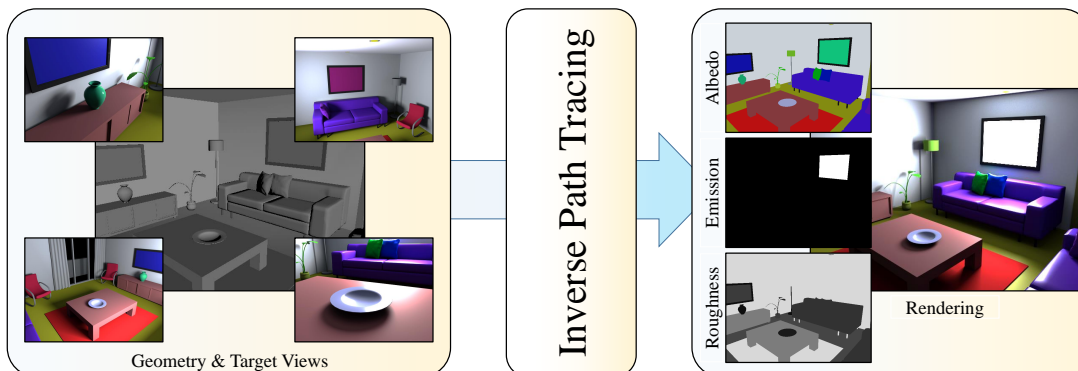


Figure 3.1: Our Inverse Path Tracing algorithm takes as input a 3D scene and up to several RGB images (left), and estimates material as well as the lighting parameters of the scene. The main contribution of our approach is the formulation of an end-to-end differentiable inverse Monte Carlo renderer which is utilized in a nested stochastic gradient descent optimization.

With the availability of inexpensive, commodity RGB-D sensors, such as the Microsoft Kinect, Google Tango, or Intel RealSense, we have seen incredible advances in 3D reconstruction techniques [4]–[6], [47], [48]. While tracking and reconstruction quality have reached impressive levels, the estimation of lighting and materials has often been neglected. Unfortunately, this presents a serious problem for virtual- and mixed-reality applications, where we need to re-render scenes from different viewpoints, place virtual objects, edit scenes, or enable telepresence scenarios where a person is placed in a different room.

This problem has been viewed in the 2D image domain, resulting in a large body of work on intrinsic images or videos [49]–[51]. However, the problem is severely underconstrained on monocular RGB data due to lack of known geometry, and thus requires heavy regularization to jointly solve for lighting, material, and scene geometry. We believe that the problem is much more tractable in the context of given 3D reconstructions. However, even with depth data available, most state-of-the-art methods, e.g., shading-based refinement [52], [53] or indoor re-lighting [54], are based on simplistic lighting models, such as spherical harmonics (SH) [55] or spatially-varying SH [56], which can cause issues on occlusion and view-dependent effects (Figure 3.4).

In this work, we address this shortcoming by formulating material and lighting estimation as a proper inverse rendering problem. To this end, we propose an Inverse Path Tracing algorithm that takes as input a given 3D scene along with a single or up to several captured RGB frames. The key to our approach is a differentiable Monte Carlo path tracer which can differentiate with respect to rendering parameters constrained on the difference of the rendered image and the target observation. Leveraging these derivatives, we solve for the material and lighting parameters by nesting the Monte Carlo path tracing process into a stochastic gradient descent (SGD) optimization. The main

contribution of this work lies in this SGD optimization formulation, which is inspired by recent advances in deep neural networks.

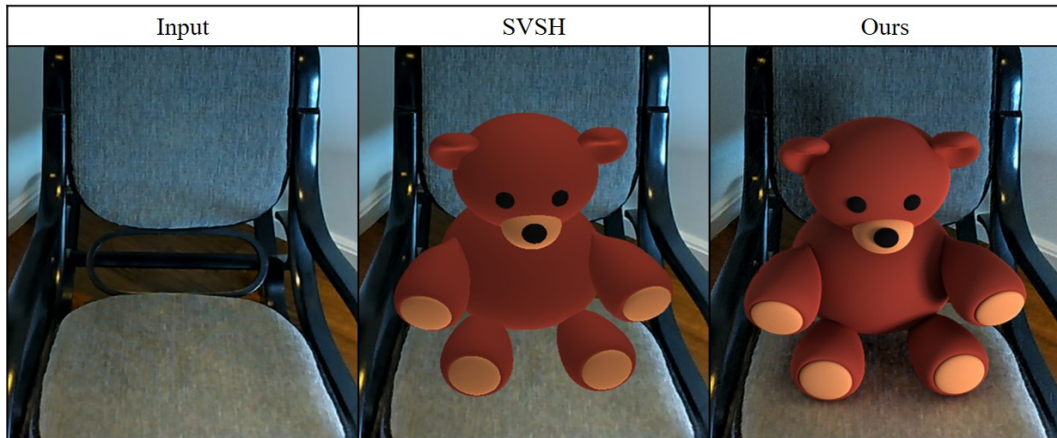


Figure 3.2: Inserting virtual objects in real 3D scenes; the estimated lighting and material parameters of our approach enable convincing image compositing in AR settings.

We tailor this Inverse Path Tracing algorithm to 3D scenes, where scene geometry is (mostly) given but the material and lighting parameters are unknown. In a series of experiments on both synthetic ground truth and real scan data, we evaluate the design choices of our optimizer. In comparison to current state-of-the-art lighting models, we show that our inverse rendering formulation and its optimization achieves significantly more accurate results.

In summary, we contribute the following:

- An end-to-end differentiable inverse path tracing formulation for joint material and lighting estimation.
- A flexible stochastic optimization framework with extensibility and flexibility for different materials and regularization terms.

3.2 Related Work

Material and illumination reconstruction has a long history in computer vision (e.g., [57], [58]). Given scene geometry and observed radiance of the surfaces, the task is to infer the material properties and locate the light source. However, to our knowledge, none of the existing methods handle non-Lambertian materials with near-field illumination (area light sources), while taking interreflection between surfaces into account.

3D approaches. A common assumption in reconstructing material and illumination is that the light sources are infinitely far away. Ramamoorthi and Hanrahan [55] project both material and illumination onto spherical harmonics and solve for their coefficients

using the convolution theorem. Dong et al. [59] solve for spatially-varying reflectance from a video of an object. Kim et al. [60] reconstruct the reflectance by training a convolutional neural network operating on voxels constructed from RGB-D video. Maier et al. [56] generalize spherical harmonics to handle spatial dependent effects, but do not correctly take view-dependent reflection and occlusion into account. All these approaches simplify the problem by assuming that the light sources are infinitely far away, in order to reconstruct a single environment map shared by all shading points. In contrast, we model the illumination as emission from the surfaces, and handle near-field effects such as the squared distance falloff or glossy reflection better.

Image-space approaches (e.g., [49], [51], [61], [62]). These methods usually employ sophisticated data-driven approaches, by learning the distributions of material and illumination. However, these methods do not have a notion of 3D geometry, and cannot handle occlusion, interreflection and geometry factors such as the squared distance falloff in a physically based manner. These methods also usually require a huge amount of training data, and are prone to errors when subjected to scenes with different characteristics from the training data.

Active illumination (e.g., [63]–[65]). These methods use highly-controlled lighting for reconstruction, by carefully placing the light sources and measuring the intensity. These methods produce high-quality results, at the cost of a more complicated setup.

Inverse radiosity (e.g., [54], [66]) achieves impressive results for solving near-field illumination and Lambertian materials for indoor illumination. It is difficult to generalize the radiosity algorithm to handle non-Lambertian materials (Yu et al. handle it by explicitly measuring the materials, whereas Zhang et al. assume Lambertian).

Differentiable rendering. Blanz and Vetter utilized differentiable rendering for face reconstruction using 3D morphable models [67], which is now leveraged by modern analysis-by-synthesis face trackers [68]. Gkioulekas et al. [69], [70] and Che et al. [71] solve for scattering parameters using a differentiable volumetric path tracer. Kasper et al. [72] developed a differentiable path tracer, but focused on distant illumination. Loper and Black [73] and Kato [74] developed fast differentiable rasterizers, but do not support global illumination. Li et al. [11] showed that it is possible to compute correct gradients of a path tracer while taking discontinuities introduced by visibility into consideration.

3.3 Method

Our *Inverse Path Tracing* method employs physically based light transport simulation [30] to estimate derivatives of all unknown parameters w.r.t. the rendered image(s). The rendering problem is generally extremely high-dimensional and is therefore usually solved using stochastic integration methods, such as Monte Carlo integration. In this work, we nest differentiable path tracing into stochastic gradient descent to solve for the unknown scene parameters. Figure 3.3 illustrates the workflow of our approach. We start from the captured imagery, scene geometry, object segmentation of the scene, and an arbitrary initial guess of the illumination and material parameters. Material and

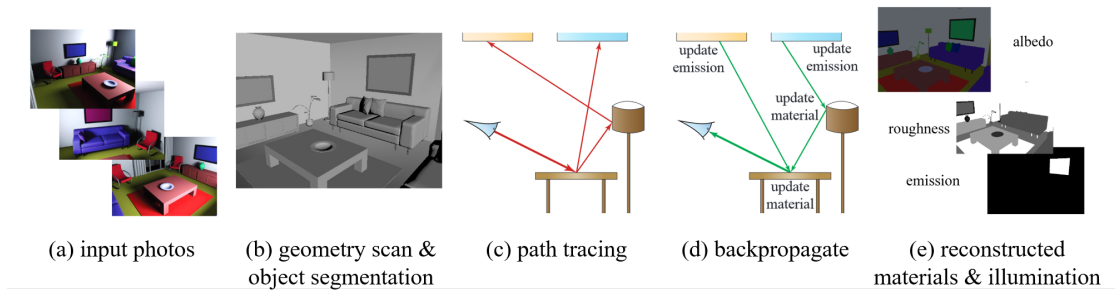


Figure 3.3: Overview of our pipeline. Given (a) a set of input photos from different views, along with (b) an accurate geometry scan and proper segmentation, we reconstruct the material properties and illumination of the scene, by iteratively (c) rendering the scene with path tracing, and (d) backpropagating to the material and illumination parameters in order to update them. After numerous iterations, we obtain the (e) reconstructed material and illumination.

emission properties are then estimated by optimizing for rendered imagery to match the captured images.

The path tracer renders a noisy and undersampled version of the image using Monte Carlo integration and computes derivatives of each sampled light path w.r.t. the unknowns. These derivatives are passed as input to our optimizer to perform a single optimization step. This process is performed iteratively until we arrive at the correct solution. Path tracing is a computationally expensive operation, and this optimization problem is non-convex and ill-posed. To this end, we employ variance reduction and novel regularization techniques (Sec. 3.4.4) for our gradient computation to arrive at a converged solution within a reasonable amount of time, usually a few minutes on a modern 8-core CPU.

3.3.1 Light Transport Simulation

If all scene and image parameters are known, an expected linear pixel intensity can be computed using light transport simulation. In this work, we assume that all surfaces are opaque and there is no participating media (e.g., fog) in the scene. In this case, the rendered intensity I_R^j for pixel j is computed using the path integral [75]:

$$I_R^j = \int_{\Omega} h_j(\mathbf{X}) f(\mathbf{X}) d\mu(\mathbf{X}), \quad (3.1)$$

where $\mathbf{X} = (\mathbf{x}_0, \dots, \mathbf{x}_k)$ is a light path, i.e., a list of vertices on the surfaces of the scene starting at the light source and ending at the sensor; the integral is a path integral taken over the space of all possible light paths of all lengths, denoted as Ω , with a product area measure $\mu(\cdot)$; $f(\mathbf{X})$ is the measurement contribution function of a light path \mathbf{X} that computes how much energy flows through this particular path; and $h_j(\mathbf{X})$ is the pixel

filter kernel of the sensor’s pixel j , which is non-zero only when the light path \mathbf{X} ends around the pixel j and incorporates sensor sensitivity at this pixel. We refer interested readers to the work of Veach [75] for more details on the light transport path integration.

The most important term of the integrand to our task is the path measurement contribution function f , as it contains the material parameters as well as the information about the light sources. For a path $\mathbf{X} = (\mathbf{x}_0, \dots, \mathbf{x}_k)$ of length k , the measurement contribution function has the following form:

$$f(\mathbf{X}) = L_e(\mathbf{x}_0, \overline{\mathbf{x}_0\mathbf{x}_1}) \prod_{i=1}^k f_r(\mathbf{x}_i, \overline{\mathbf{x}_{i-1}\mathbf{x}_i}, \overline{\mathbf{x}_i\mathbf{x}_{i+1}}), \quad (3.2)$$

where L_e is the radiance emitted at the scene surface point \mathbf{x}_0 (beginning of the light path) towards the direction $\overline{\mathbf{x}_0\mathbf{x}_1}$. At every interaction vertex \mathbf{x}_i of the light path, there is a *bidirectional reflectance distribution function* (BRDF) $f_r(\mathbf{x}_i, \overline{\mathbf{x}_{i-1}\mathbf{x}_i}, \overline{\mathbf{x}_i\mathbf{x}_{i+1}})$ defined. The BRDF describes the material properties at the point \mathbf{x}_i , i.e., how much light is scattered from the incident direction $\overline{\mathbf{x}_{i-1}\mathbf{x}_i}$ towards the outgoing direction $\overline{\mathbf{x}_i\mathbf{x}_{i+1}}$. The choice of the parametric BRDF model f_r is crucial to the range of materials that can be reconstructed by our system. We discuss the challenges of selecting the BRDF model in Sec. 3.4.1.

Note that both the BRDF f_r and the emitted radiance L_e are unknown and the desired parameters to be found at every point on the scene manifold.

3.3.2 Optimizing for Illumination and Materials

We take as input a series of images in the form of real-world photographs or synthetic renderings, together with the reconstructed scene geometry and corresponding camera poses. We aim to solve for the unknown material parameters \mathcal{M} and lighting parameters \mathcal{L} that will produce rendered images of the scene that are identical to the input images.

Given the un-tonemapped captured pixel intensities I_C^j at all pixels j of all images, and the corresponding noisy estimated pixel intensities \tilde{I}_R^j (in linear color space), we seek all material and illumination parameters $\Theta = \{\mathcal{M}, \mathcal{L}\}$ by solving the following optimization problem using stochastic gradient descent:

$$\operatorname{argmin}_{\Theta} E(\Theta) = \sum_j^N \left| I_C^j - \tilde{I}_R^j \right|_1, \quad (3.3)$$

where N is the number of pixels in all images. We found that using an L_1 norm as a loss function helps with robustness to outliers, such as extremely high contribution samples coming from Monte Carlo sampling.

3.3.3 Computing Gradients with Path Tracing

In order to efficiently solve the minimization problem in Eq. 3.3 using stochastic optimization, we compute the gradient of the energy function $E(\Theta)$ with respect to the set

of unknown material and emission parameters Θ :

$$\nabla_{\Theta} E(\Theta) = \sum_j^N \nabla_{\Theta} \tilde{I}_R^j \operatorname{sgn}(I_C^j - \tilde{I}_R^j), \quad (3.4)$$

where $\operatorname{sgn}(\cdot)$ is the sign function, and $\nabla_{\Theta} \tilde{I}_R^j$ the gradient of the Monte Carlo estimate with respect to all unknowns Θ .

Note that this equation for computing the gradient now has two Monte Carlo estimates for each pixel j : (1) the estimate of pixel color itself \tilde{I}_R^j ; and (2) the estimate of its gradient $\nabla_{\Theta} \tilde{I}_R^j$. Since the expectation of product only equals the product of expectation when the random variables are independent, it is important to draw independent samples for each of these estimates to avoid introducing bias.

In order to compute the gradients of a Monte Carlo estimate for a single pixel j , we determine what unknowns are touched by the measurement contribution function $f(\mathbf{X})$ for a sampled light path \mathbf{X} . We obtain the explicit formula of the gradients by differentiating Eq. 3.2 using the product rule (for brevity, we omit some arguments for emission L_e and BRDF f_r):

$$\nabla_{\Theta_{\mathcal{L}}} f(\mathbf{X}) = \nabla_{\Theta_{\mathcal{L}}} L_e(\mathbf{x}_0) \prod_i^k f_r(\mathbf{x}_i) \quad (3.5)$$

$$\nabla_{\Theta_{\mathcal{M}}} f(\mathbf{X}) = L_e(\mathbf{x}_0) \sum_l^k \nabla_{\Theta_{\mathcal{M}}} f_r(\mathbf{x}_l) \prod_{i, i \neq l}^k f_r(\mathbf{x}_i) \quad (3.6)$$

where the gradient vector $\nabla_{\Theta} = \{\nabla_{\Theta_{\mathcal{M}}}, \nabla_{\Theta_{\mathcal{L}}}\}$ is very sparse and has non-zero values only for unknowns touched by the path \mathbf{X} . The gradients of emissions (Eq. 3.5) and materials (Eq. 3.6) have similar structure to the original path contribution (Eq. 3.2). Therefore, it is natural to apply the same path sampling strategy; see the appendix for details.

3.3.4 Multiple Captured Images

The single-image problem can be directly extended to multiple images. Given multiple views of a scene, we aim to find parameters for which rendered images from these views match the input images. A set of multiple views can cover parts of the scene that are not covered by any single view from the set. This proves important for deducing the correct position of the light source in the scene. With many views, the method can better handle view-dependent effects such as specular and glossy highlights, which can be ill-posed with just a single view, as they can also be explained as variations of albedo texture.

3.4 Optimization Parameters and Methodology

In this section we address the remaining challenges of the optimization task: what are the material and illumination parameters we actually optimize for, and how to resolve the ill-posed nature of the problem.

3.4.1 Parametric Material Model

We want our material model to satisfy several properties. First, it should cover as much variability in appearance as possible, including such common effects as specular highlights, multi-layered materials, and spatially-varying textures. On the other hand, since each parameter adds another unknown to the optimization, we would like to keep the number of parameters minimal. Since we are interested in re-rendering and related tasks, the material model needs to have interpretable parameters, so the users can adjust the parameters to achieve the desired appearance. Finally, since we are optimizing the material properties using first-order gradient-based optimization, we would like the range of the material parameters to be similar.

To satisfy these properties, we represent our materials using the Disney material model [34], the state-of-the-art physically based material model used in movie and game rendering. It has a “base color” parameter which is used by both diffuse and specular reflectance, as well as 10 other parameters describing the roughness, anisotropy, and specularity of the material. All these parameters are perceptually mapped to $[0, 1]$, which is both interpretable and suitable for optimization.

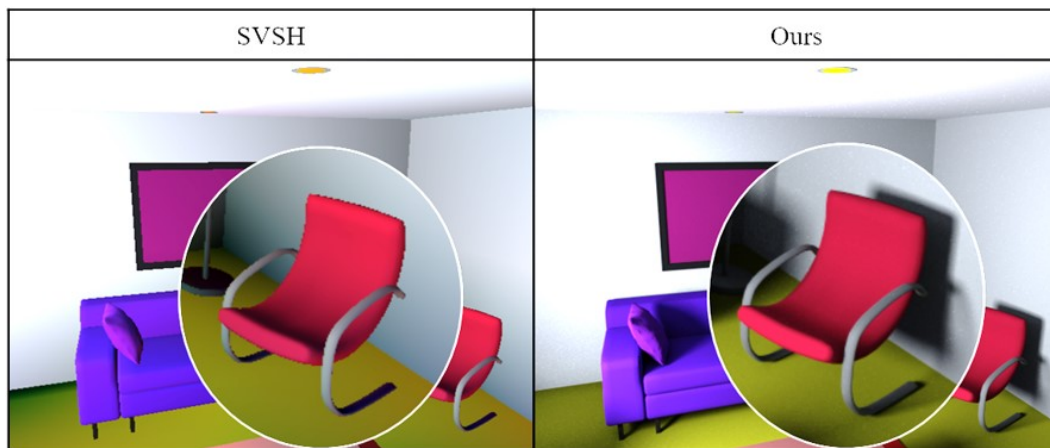


Figure 3.4: Methods based on spherical harmonics have difficulties handling sharp shadows or lighting changes due to the distant illumination assumption. A physically based method, such as Inverse Path Tracing, correctly reproduces these effects.

3.4.2 Scene Parameterization

We use triangle meshes to represent the scene geometry. Surface normals are defined per-vertex and interpolated within each triangle using barycentric coordinates. The optimization is performed on a per-object basis, i.e., every object has a single unknown emission and a set of material parameters that are assumed constant across the whole object. We show that this is enough to obtain accurate lighting and an average constant value for the albedo of an object.

3.4.3 Emission Parameterization

For emission reconstruction, we currently assume all light sources are scene surfaces with an existing reconstructed geometry. For each emissive surface, we currently assume that emitted radiance is distributed according to a view-independent directional emission profile $L_e(\mathbf{x}, \mathbf{i}) = e(\mathbf{x})(\mathbf{i} \cdot \mathbf{n}(\mathbf{x}))_+$, where $e(\mathbf{x})$ is the unknown radiant flux at \mathbf{x} ; \mathbf{i} is the emission direction at surface point \mathbf{x} , $\mathbf{n}(\mathbf{x})$ is the surface normal at \mathbf{x} and $(\cdot)_+$ is the dot product (cosine) clamped to only positive values. This is a common emission profile for most of the area lights, which approximates most of the real soft interior lighting well. Our method can also be extended to more complex or even unknown directional emission profiles or purely directional distant illumination (e.g., sky dome, sun) if needed.

3.4.4 Regularization

The observed color of an object in a scene is most easily explained by assigning emission to the triangle. This is only avoided by differences in shading of the different parts of the object. However, it can happen that there are no observable differences in the shading of an object, especially if the object covers only a few pixels in the input image. This can be a source of error during optimization. Another source of error is Monte Carlo and SGD noise. These errors lead to incorrect emission parameters for many objects after the optimization. The objects usually have a small estimated emission value when they should have none. We tackle the problem with an L1-regularizer for the emission. The vast majority of objects in the scene is not an emitter and having such a regularizer suppresses the small errors we get for the emission parameters after optimization.

3.4.5 Optimization Parameters

We use ADAM [76] as our optimizer with batch size $B = 8$ estimated pixels and learning rate $5 \cdot 10^{-3}$. To form a batch, we sample B pixels uniformly from the set of all pixels of all images. Please see the appendix for an evaluation regarding the impact of different batch sizes and sampling distributions on the convergence rate. While a higher batch size reduces the variance of each iteration, having smaller batch sizes, and therefore faster iterations, proves to be more beneficial.

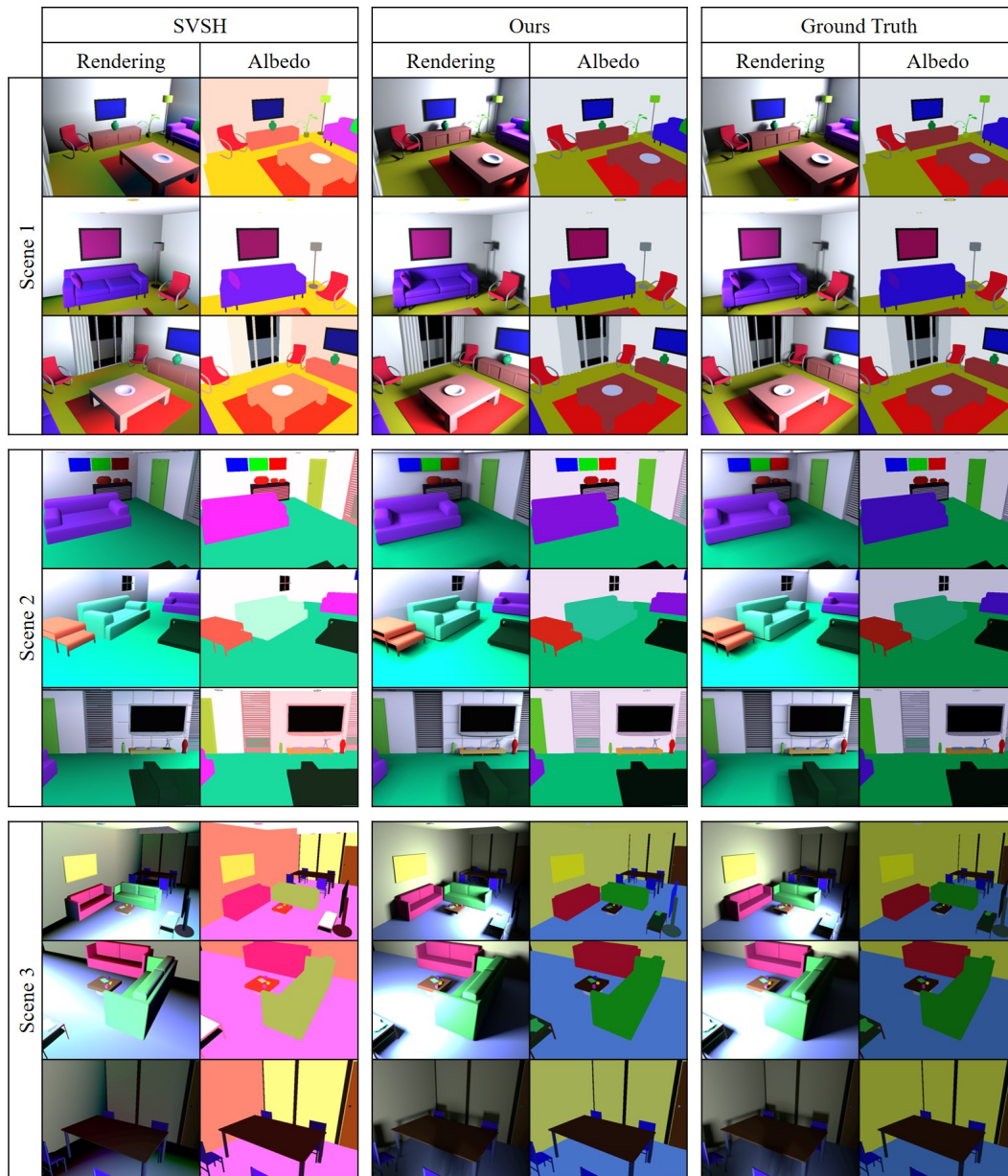


Figure 3.5: Evaluation on synthetic scenes. Three scenes have been rendered from different views with both direct and indirect lighting (right). An approximation of the albedo lighting with spatially-varying spherical harmonics is shown (left). Our method is able to detect the light source even though it was not observed in any of the views (middle). Notice that we are able to reproduce sharp lighting changes and shadows correctly. The albedo is also closer to the ground truth albedo.

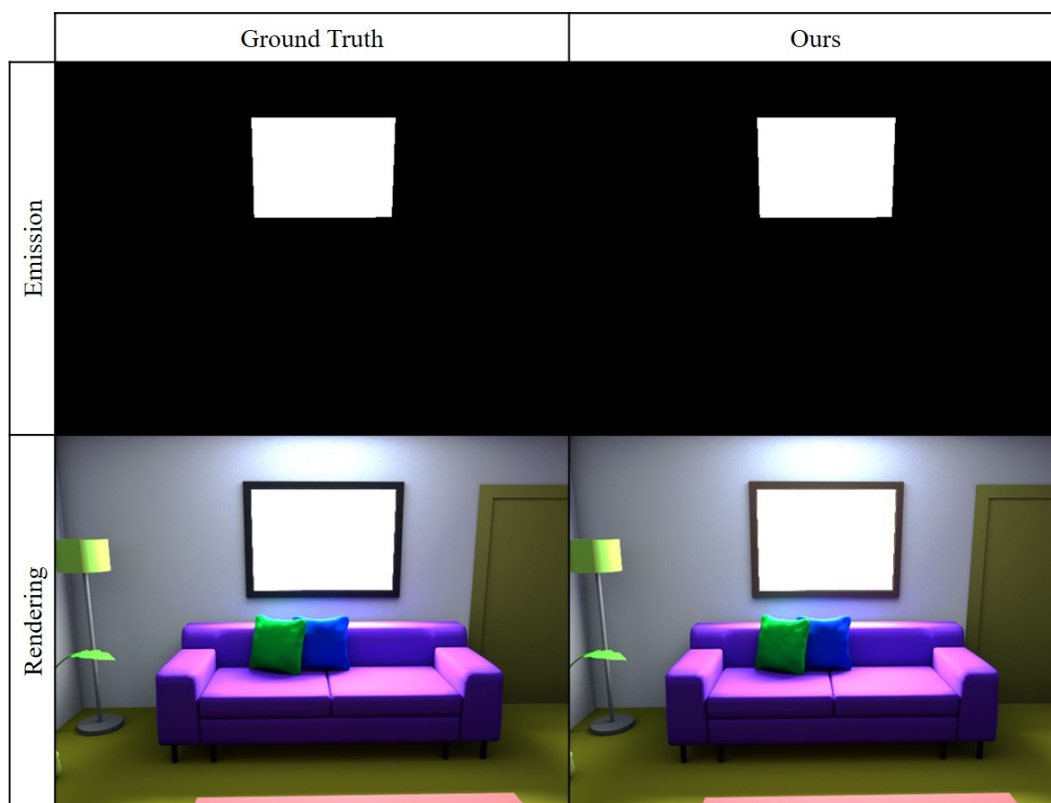


Figure 3.6: Inverse Path Tracing is able to correctly detect the light emitting object (top). The ground truth rendering and our estimate is shown on the bottom. Note that this view was not used during optimization.

3.5 Results

Evaluation on synthetic data. We first evaluate our method on multiple synthetic scenes, where we know the ground truth solution. Quantitative results are listed in Tab. 3.1, and qualitative results are shown in Figure 3.5. Each scene is rendered using a path tracer with the ground truth lighting and materials to obtain the “captured images”. These captured images and scene geometry are then given to our Inverse Path Tracing algorithm, which optimizes for unknown lighting and material parameters. We compare to the closest previous work based on spatially-varying spherical harmonics (SVSH) [56]. SVSH fails to capture sharp details such as shadows or high-frequency lighting changes. A comparison of the shadow quality is presented in Figure 3.4.

Our method correctly detects light sources and converges to a correct emission value, while the emission of objects that do not emit light stays at zero. Figure 3.6 shows a novel view, rendered with results from an optimization that was performed on input views from Figure 3.5. Even though the light source was not visible in any of the input views, its emission was correctly computed by Inverse Path Tracing.

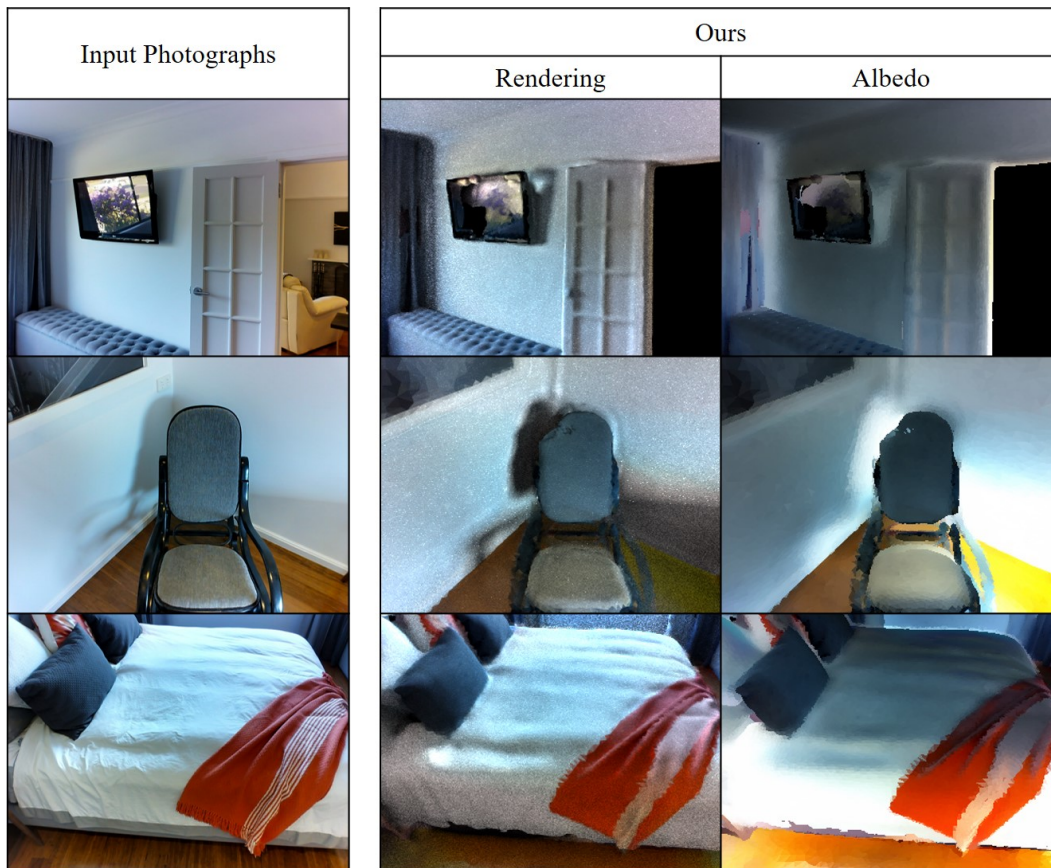


Figure 3.7: We can resolve object textures by optimizing for the unknown parameters per triangle. Higher resolution textures can be obtained by further subdividing the geometry.

In addition to albedo, our Inverse Path Tracer can also optimize for other material parameters such as roughness. In Figure 3.8, we render a scene containing objects of varying roughness. Even when presented with the challenge of estimating both albedo and roughness, our method produces the correct result as shown in the re-rendered image.

Evaluation on real data. We use the Matterport3D [2] dataset to evaluate our method on real captured scenes obtained through 3D reconstruction. The scene was parameterized using the segmentation provided in the dataset. Due to imperfections in the data, such as missing geometry and inaccurate surface normals, it is more challenging to perform an accurate light transport simulation. Nevertheless, our method produces impressive results for the given input. After the optimization, the optimized light direction matches the captured light direction and the rendered result closely matches the photograph. Figure 3.11 shows a comparison to the SVSH method.

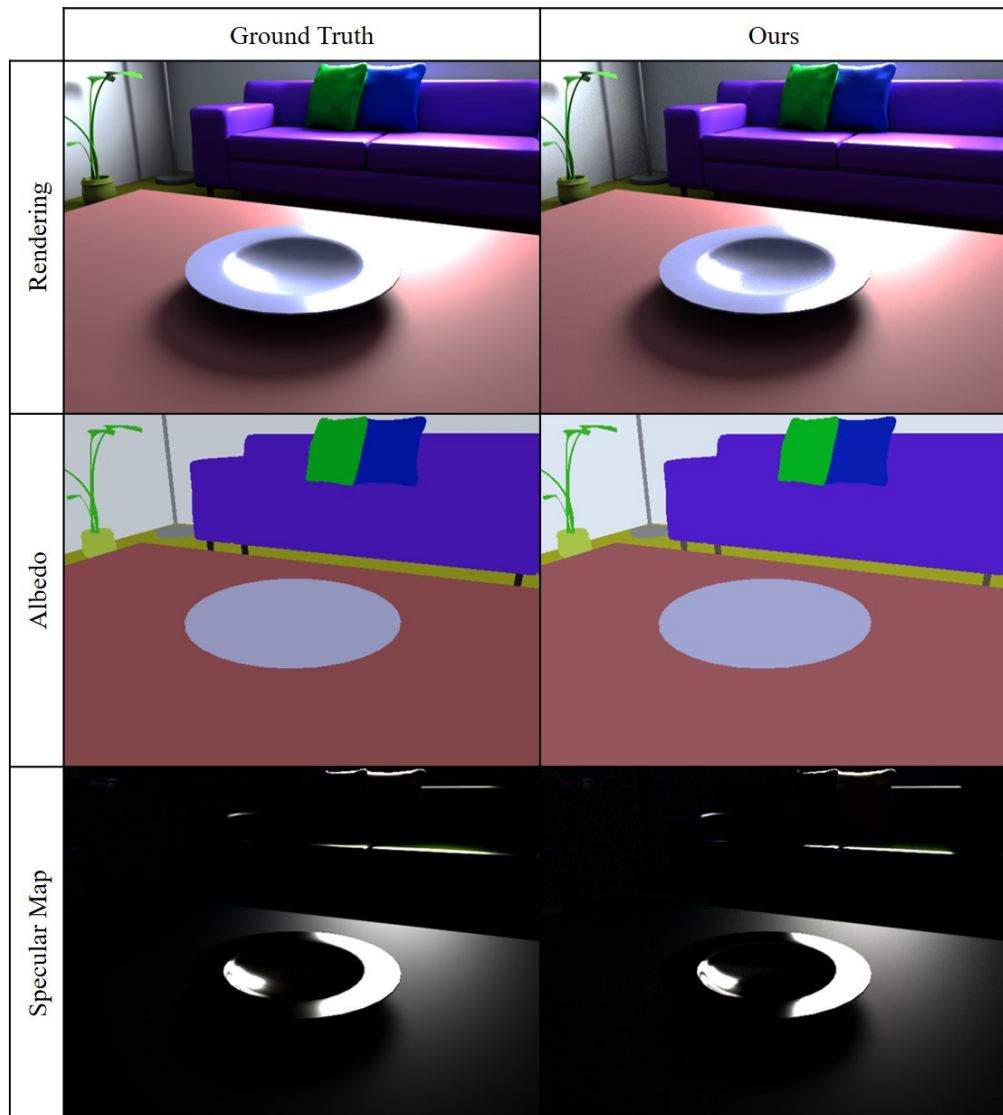


Figure 3.8: Inverse Path Tracing is agnostic to the underlying BRDF; e.g., here, in a specular case, we are able to correctly estimate both the albedo and the roughness of the objects. The ground truth rendering and our estimate is shown on top, the albedo in the middle and the specular map on the bottom.

The albedo of real-world objects varies across its surface. Inverse Path Tracing is able to compute an object’s average albedo by employing knowledge of the scene segmentation. To reproduce fine texture, we refine the method to optimize for each individual triangle of the scene with adaptive subdivision where necessary. This is demonstrated in Figure 3.7.

Optimizer Ablation. There are several ways to reduce the variance of our optimizer. One obvious way is to use more samples to estimate the pixel color and the derivatives, but this also results in slower iterations. Figure 3.9 shows that the method does not converge if only a single path is used. A general recommendation is to use between 2^7 and 2^{10} depending on the scene complexity and number of unknowns.

Another important aspect of our optimizer is the sample distribution for pixel color and derivatives estimation. Our tests in Figure 3.10 show that minimal variance can be achieved by using one sample to estimate the derivatives and the remaining samples in the available computational budget to estimate the pixel color.

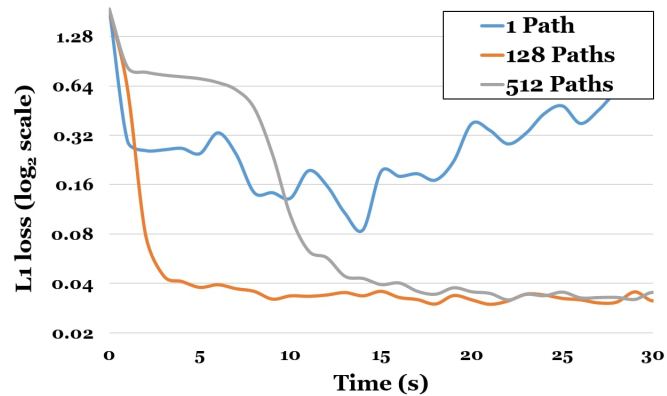


Figure 3.9: Convergence with respect to the number of paths used to estimate the pixel color. If this is set too low, the algorithm will fail.

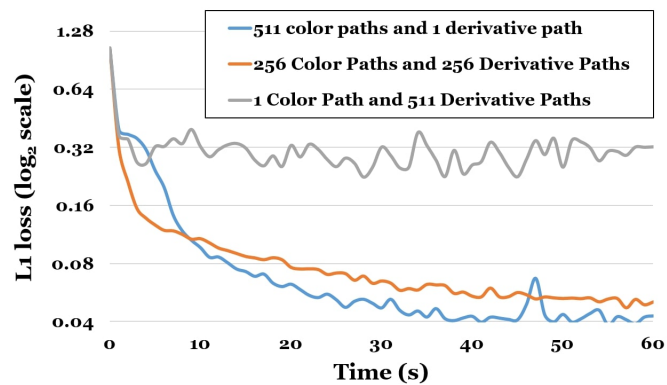


Figure 3.10: Convergence with respect to distributing the available path samples budget between pixel color and derivatives. It is best to keep the number of paths high for pixel color estimation and low for derivative estimation.

Method	Scene 1	Scene 2	Scene 3
SVSH Rendering Loss	0.052	0.048	0.093
Our Rendering Loss	0.006	0.010	0.003
SVSH Albedo Loss	0.052	0.037	0.048
Our Albedo Loss	0.002	0.009	0.010

Table 3.1: Quantitative evaluation for synthetic data. We measure the L1 loss with respect to the rendering error and the estimated albedo parameters. Note that our approach achieves a significantly lower error on both metrics.

Limitations. Inverse Path Tracing assumes that high-quality geometry is available. However, imperfections in the recovered geometry can have big impact on the quality of material estimation as shown in Figure 3.11. Our method also does not compensate for the distortions in the captured input images. Most cameras, however, produce artifacts such as lens flare, motion blur or radial distortion. Our method can potentially account for these imperfections by simulating the corresponding effects and optimize not only for the material parameters, but also for the camera parameters, which we leave for future work.

3.6 Qualitative Evaluation of Design Choices

3.6.1 Choice of Batch Size

In Figure 3.13, we evaluate the choice of the batch size for the optimization. To this end, we assume the compute budget for all experiments, and plot the results with respect to time on the x -axis and the ℓ_1 loss of our problem (log scale) on the y -axis. If the batch size is too low (blue curve), then the estimated gradients are noisy, which leads to a slower convergence; if batches are too large (gray curve), then we require too many rays for each gradient step, which would be used instead to perform multiple gradient update steps.

3.6.2 Variance Reduction

In order to speed up the convergence of our algorithm, we must aim to reduce the variance of the gradients as much as possible. There are two sources of variance: the Monte Carlo integration in path tracing and the SGD, since we path trace only a small fraction of captured pixels in every batch.

As mentioned in the main paper, the gradients of the rendering integral have similar structure to the original integral, therefore we employ the same importance sampling strategy as in usual path tracing. The path tracing variance is reduced using Multiple Importance Sampling (i.e., we combine BRDF sampling with explicit light sampling) [75]. We follow the same computation for estimating the gradients with respect to our un-

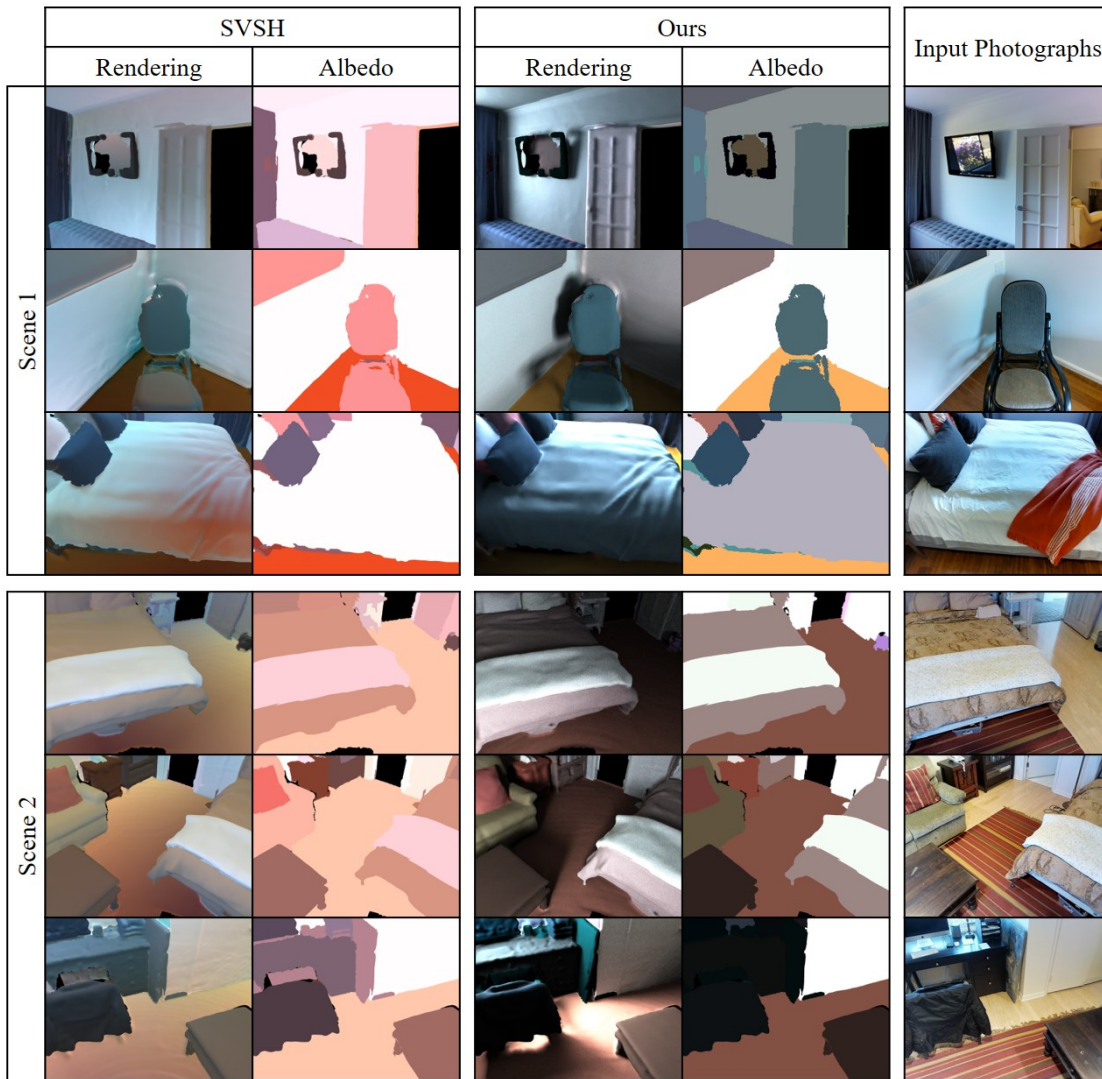


Figure 3.11: Evaluation on real scenes: (right) input is 3D scanned geometry and photographs. We employ object instance segmentation to estimate the emission and the average albedo of every object in the scene. Our method is able to optimize for the illumination and shadows. Other methods usually do not take occlusions into account and fail to model shadows correctly. Views 1 and 2 of Scene 2 show that if the light emitters are not present in the input geometry, our method gives an incorrect estimation.

knowns. A comparison between implementation with and without MIS is shown in Figure 3.14.

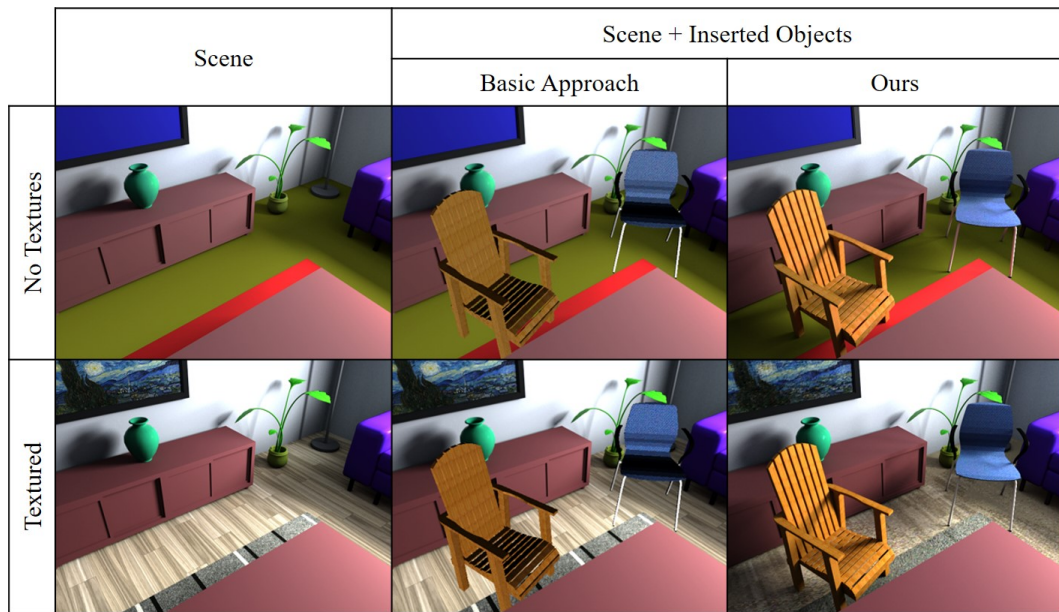


Figure 3.12: Mixed-reality setting: we insert two new 3D objects (chairs) into an existing 3D scene. Our goal is to find a consistent lighting between the existing and newly-inserted content. In the middle column, we show a naive compositing approach; on the right the results of our approach. The naive approach does not take the 3D scene and light transport into consideration, and fails to photo-realistically render the chair.

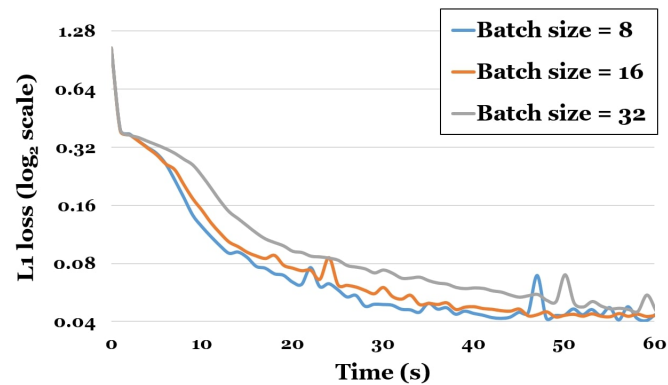


Figure 3.13: Convergence with respect to the batch size: in this experiment, we assume the same compute/time budget for all experiments (x -axis), but we use different distributions of rays within each batch; i.e., we try different batch sizes.

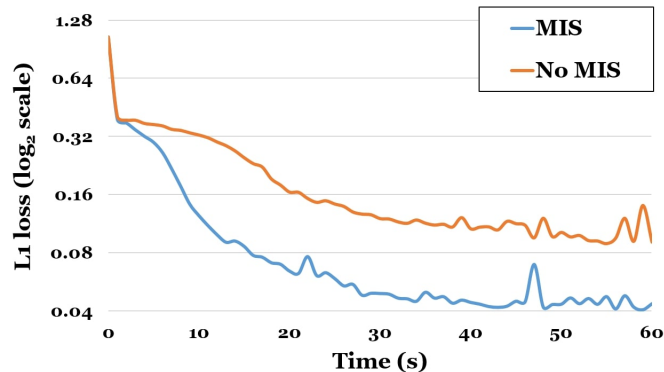


Figure 3.14: Use of Multiple Importance Sampling during path tracing significantly improves the convergence rate.

3.6.3 Number of Bounces

We argue that most diffuse global illumination effects can be approximated by as few as two bounces of light. To this end, we render an image with 10 bounces and use it as ground truth for our optimization. We try to approximate the ground truth by renderings with one, two, and three bounces, respectively (see Figure 3.15). One bounce corresponds to direct illumination; adding more bounces allows us to take into account indirect illumination as well. Optimization with only a single bounce is the fastest, but the error remains high even after convergence. Having more than two bounces leads to high variance and takes a lot of time to converge. Using two bounces strikes the balance between convergence speed and accuracy.

3.7 Results on Scenes with Textures

In order to evaluate surfaces with high-frequency surface signal, we consider both real and synthetic scenes with textured objects. To this end, we optimize first for the light sources and material parameters on the coarse per-object resolution. Once converged, we keep the light sources fixed, and we subdivide all other regions based on the surface texture where the re-rendering error is high; i.e., we subdivide every triangle based on the average ℓ_2 error of the pixels it covers, and continue until convergence. This coarse-to-fine strategy allows us to first separate out material and lighting in the more well-conditioned setting; in the second step, we then obtain high-resolution material information. Results on synthetic data [77] are shown in Figure 3.16, and results on real scenes from Matterport3D [2] are shown in Figure 3.17.

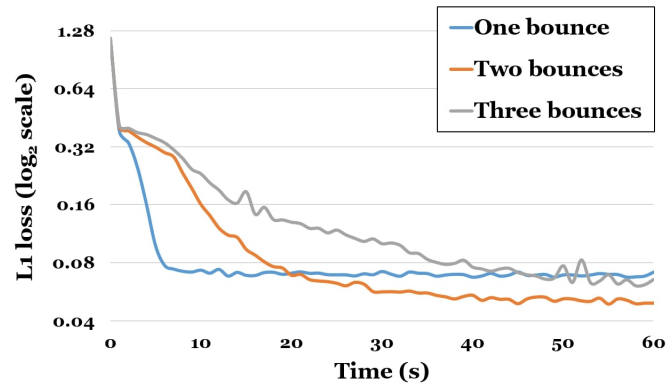


Figure 3.15: A scene rendered with 10 bounces of light is given as input to our algorithm. We estimate emission and material parameters by using one, two, and three bounces during optimization. Two bounces are enough to capture most of the diffuse indirect illumination in the scene.

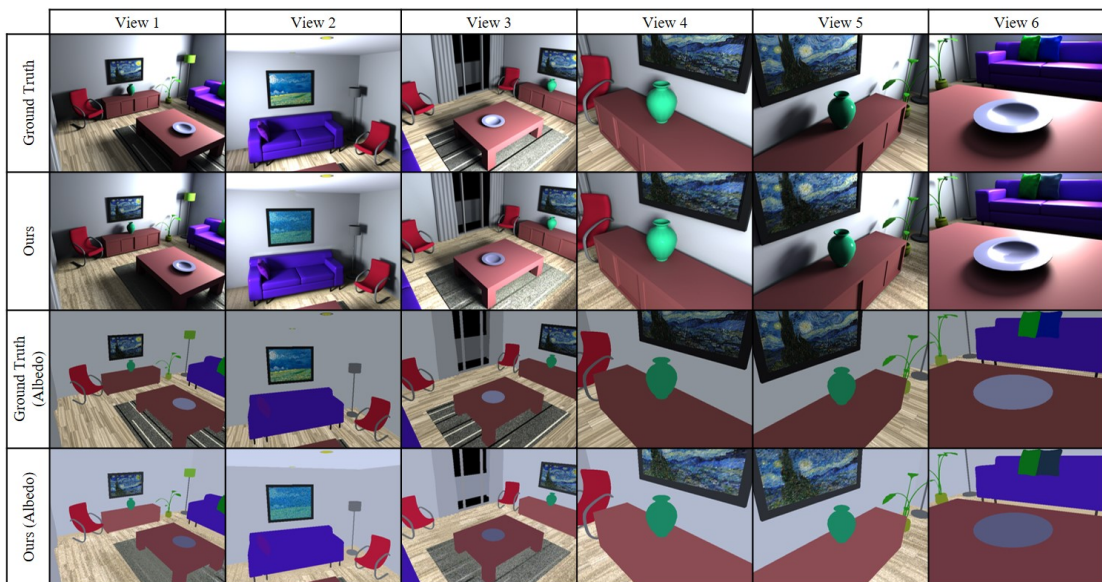


Figure 3.16: Results of our approach on synthetic scenes with textured objects. Our optimization is able to recover the scene lighting in addition to high-resolution surface texture material parameters.

3.8 Additional Comparison to Data-driven Approaches

We compare our approach to Meka et al. [51] and present quantitative results in Tab. 3.2. Please note that our approach is not limited to a single material of a single object at a

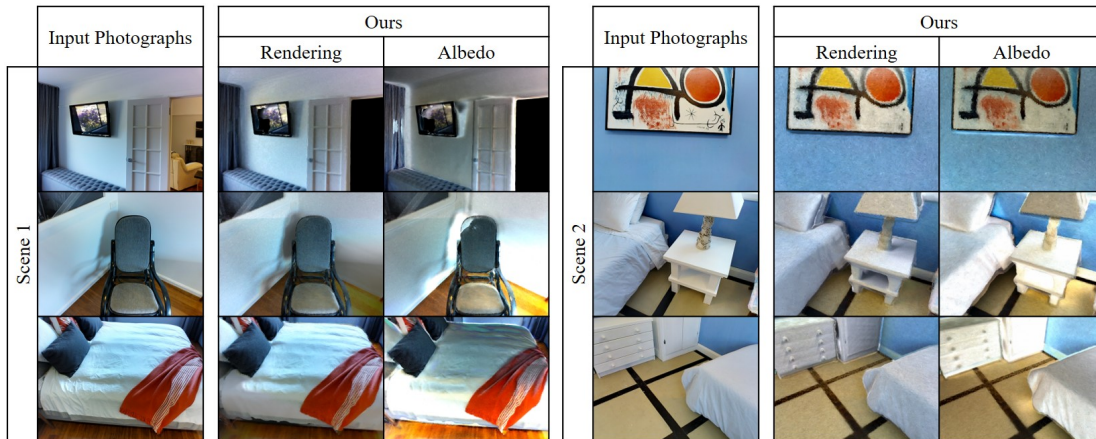


Figure 3.17: Examples from Matterport3D [2] (real-world RGB-D scanning data) where we reconstruct emission parameters, as well as high-resolution surface texture material parameters. We are able to reconstruct fine texture detail by subdividing the geometry mesh and optimizing on individual triangle parameters. Since not all light sources are present in the reconstructed geometry, some inaccuracies are introduced into our material reconstruction. Albedo in shadow regions can be overestimated to compensate for missing illumination (visible behind the chair in Scene 1), specular effects can be baked into the albedo (reflection of flowers on the TV) or color may be projected onto the incorrect geometry (part of the chair is missing, so its color is projected onto the floor and wall).

time. The other data-driven references are mostly on planar surfaces only and/or assume specific lighting conditions, such as a single point light close to the surface.

3.9 Object Insertion in Mixed-reality Settings

One of the primary target applications of our method is insertion of virtual objects into an existing scene while maintaining a coherent appearance. Here, the idea is to first estimate the lighting and material parameters of a given 3D scene or 3D reconstruction. We then insert a new 3D object into the environment, and re-render the scene using both the estimated lighting and material parameters for the already existing content, and the

Method	Object 1	Object 2
LIME [51]	0.45%	1.37%
Ours	0.00037%	0.14%

Table 3.2: We compare the relative error between the estimated diffuse albedo for two objects. We outperform LIME even though our method is not restricted to the estimation of only a single material at a time.

known intrinsic parameters for the newly-inserted object. A complete 3D knowledge is required to produce photorealistic results, in order to take interreflection and shadow between objects into consideration.

In Figure 3.12, we show an example on a synthetic scene where we virtually inserted two new chairs. As a baseline, we consider a naive image compositing approach where the new object is first lit by spherical harmonics lighting and then inserted while not considering the rest of the scene; this is similar to most existing AR applications on mobile devices. We can see that a naive compositing approach (middle) is unable to produce a consistent result, and the two inserted chairs appear somewhat out of place. Using our approach, we can estimate the lighting and material parameters of the original scene, composite the scene in 3D, and then re-render. We are able to show that we can produce consistent results for both textured and non-textured optimization results (right column).

In Figure 3.2, we show a real-world example on the Matterport3D [2] dataset, where we insert a virtual teddy into the environment. To this end, we first estimate lighting and surface materials in a 3D scan; we then insert a new virtual object, render it, and then apply the delta image to the original input. Compared to the SVSH baseline, our approach achieves significantly better compositing results.

3.10 Implementation Details

We implement our inverse path tracer in C++, and all of our experiments run on an 8-core CPU. We use Embree [78] for the ray casting operations. For efficient implementation, instead of employing automatic differentiation libraries, the light path gradients are computed using manually-derived derivatives.

We use ADAM [76] as our optimizer of choice with an initial learning rate of $5 \cdot 10^{-3}$. We further use an initial batch size of 8 pixels which are uniformly sampled from the set of all pixels of all images. We found marginal benefit of having larger batches, but we believe there is high potential in investigating better sampling strategies. In all our experiments, the emission and albedo parameters are initialized to zero.

For every pixel in the batch, we need to compute an estimate of the pixel color based on the current value of the unknown material and emission parameters. This estimated color is compared against the ground truth color and a gradient is computed depending on the choice of the loss function. For most commonly used loss functions, this gradient will involve a multiplication of the estimated pixel color and its derivative with respect to the unknown parameters. Since these are random variables (approximated by Monte Carlo integration), it is important that they are calculated from independent samples to avoid bias. We use path tracing with multiple importance sampling for the computation of the pixel color, but any unbiased light transport method will produce the correct result.

We extend our path tracer to analytically compute derivatives w.r.t. emission and materials parameters as defined by Eq. 3.5 and 3.6. To this end, we pass a reference to a structure holding the derivatives to our ray casting function. The product of BSDFs

in Eq. 3.5 is incrementally calculated at each bounce. Given that $L_e(x_i)$ is the unknown emission parameter on surface i , the derivative w.r.t. this emission parameter is equal to the product of the BSDFs at each surface intersection from surface i to the sensor. The derivatives w.r.t. to the materials are computed in similar manner. As per chain rule, we multiply the throughput by the derivative of the BSDF w.r.t. the unknown material parameters to obtain the derivative of the pixel color w.r.t. the unknown material parameters.

We implement multiple importance sampling, a combination of light source sampling and BRDF importance sampling. The importance for light source sampling is based on the unknown emission parameters which may change in every iteration of our optimization. An efficient data structure is needed to store the sampling probabilities for every object. We implement a binary indexed tree (also known as Fenwick tree) for this purpose. This provides logarithmic complexity for both reading and updating the probabilities.

Finally, to make the optimization more robust, we propose a coarse-to-fine approach, where we first optimize for one emission and one material parameter per object instance. Most scenes have only a few emitters, so we employ an L1-regularizer on all the emission parameters. After convergence, the result is refined by optimizing for material parameters of individual object triangles. The light sources stay fixed in this phase, but their emission value may still change. In the end, the triangles may be subdivided as explained in Sec. 3.7 to further improve the results.

3.11 Conclusion

We present Inverse Path Tracing, a novel approach for joint lighting and material estimation in 3D scenes. We demonstrate that our differentiable Monte Carlo renderer can be efficiently integrated in a nested stochastic gradient descent optimization. In our results, we achieve significantly higher accuracy than existing approaches. High-fidelity reconstruction of materials and illumination is an important step for a wide range of applications such as virtual and augmented reality scenarios. Overall, we believe that this is a flexible optimization framework for computer vision that is extensible to various scenarios, noise factors, and other imperfections of the computer vision pipeline. We hope to inspire future work along these lines, for instance, by incorporating more complex BRDF models, joint geometric refinement and completion, and further stochastic regularizations and variance reduction techniques.

4 Neural RGB-D Surface Reconstruction

This chapter introduces the following paper:

D. Azinović, R. Martin-Brualla, D. B. Goldman, M. Nießner, and J. Thies, “Neural rgb-d surface reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 6290–6301

Abstract of paper Obtaining high-quality 3D reconstructions of room-scale scenes is of paramount importance for upcoming applications in AR or VR. These range from mixed reality applications for teleconferencing, virtual measuring, virtual room planing, to robotic applications. While current volume-based view synthesis methods that use neural radiance fields (NeRFs) show promising results in reproducing the appearance of an object or scene, they do not reconstruct an actual surface. The volumetric representation of the surface based on densities leads to artifacts when a surface is extracted using Marching Cubes, since during optimization, densities are accumulated along the ray and are not used at a single sample point in isolation. Instead of this volumetric representation of the surface, we propose to represent the surface using an implicit function (truncated signed distance function). We show how to incorporate this representation in the NeRF framework, and extend it to use depth measurements from a commodity RGB-D sensor, such as a Kinect. In addition, we propose a pose and camera refinement technique which improves the overall reconstruction quality. In contrast to concurrent work on integrating depth priors in NeRF which concentrates on novel view synthesis, our approach is able to reconstruct high-quality, metrical 3D reconstructions.

Contribution The method development and implementation were done by the first author. Discussions with the co-authors led to the final paper.

Revised layout and minor adaptations. Accepted version of the original publication [15] and detailed disclaimer are included in Chapter C.2 in the appendix.

4.1 Introduction

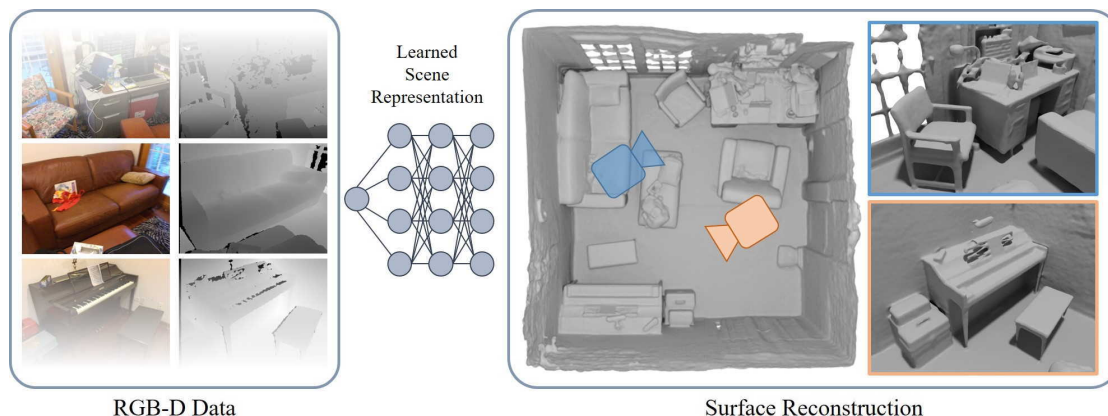


Figure 4.1: Our method obtains a high-quality 3D reconstruction from an RGB-D input sequence by training a multi-layer perceptron. The core idea is to reformulate the neural radiance field definition in NeRF [8], and replace it with a differentiable rendering formulation based on signed distance fields which is specifically tailored to geometry reconstruction.

Research on neural networks for scene representations and image synthesis has made impressive progress in recent years [79]. Methods that learn volumetric representations [8], [80] from color images captured by a smartphone camera can be employed to synthesize near photo-realistic images from novel viewpoints. While the focus of these methods lies on the reproduction of color images, they are not able to reconstruct metric and clean (noise-free) meshes. To overcome these limitations, we show that there is a significant advantage in taking additional range measurements from consumer-level depth cameras into account. Inexpensive depth cameras are broadly accessible and are also built into modern smartphones. While classical reconstruction methods [4], [7], [22] that purely rely on depth measurements struggle with the limitations of physical sensors (noise, limited range, transparent objects, etc.), a neural radiance field-based reconstruction formulation allows to also leverage the dense color information. Methods like BundleFusion [6] take advantage of color observations to compute sparse SIFT [81] features for re-localization and refinement of camera poses (loop closure). For the actual geometry reconstruction (volumetric fusion), only the depth maps are taken into account. Missing depth measurements in these maps, lead to holes and incomplete geometry in the reconstruction. This limitation is also shared by learned surface reconstruction methods that only rely on the range data [82], [83]. In contrast, our method is able to reconstruct geometry in regions where only color information is available. Specifically, we adapt the neural radiance field (NeRF) formulation of Mildenhall et al. [8] to learn a truncated signed distance field (TSDF), while still being able to leverage differentiable volumetric integration for color reproduction. To compensate for noisy initial camera poses which we compute based on the depth measurements, we jointly optimize our scene representation network with the camera poses. The implicit function represented by the scene

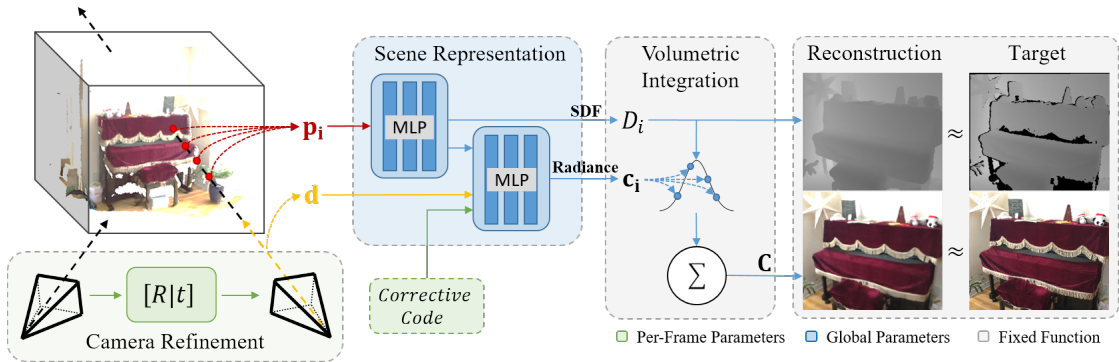


Figure 4.2: Differentiable volumetric rendering is used to reconstruct a scene that has been captured using an RGB-D camera. The scene is represented using multi-layer perceptrons (MLPs), encoding a signed distance value D_i and a viewpoint-dependent radiance value c_i per point \mathbf{p}_i . We perform volumetric rendering by integrating the radiance along a ray, weighing the samples as a function of their signed distance D_i and their visibility. We also learn a per-frame latent corrective code to account for exposure or white balance changes throughout the capture, which is passed to the radiance MLP alongside the ray direction \mathbf{d} . We optimize the scene representation’s MLPs, together with the per-frame corrective codes, the input camera poses, and an image-plane deformation field (not shown) by computing losses for the signed distance D_i of the samples, and the final integrated color \mathbf{C} with respect to the input depth and color views.

representation network allows us to predict signed distance values at arbitrary points in space which is used to extract a mesh using Marching Cubes.

Concurrent work that incorporates depth measurements in NeRF focuses on novel view synthesis [84]–[86], and uses the depth prior to restrict the volumetric rendering to near-surface regions [85], [86] or adds an additional constraint on the depth prediction of NeRF [84]. NeuS [87] is also a concurrent work on novel view synthesis which uses a signed distance function to represent the geometry, but takes only RGB images as input, and thus fails to reconstruct the geometry of featureless surfaces, like white walls. In contrast, our method aims for high-quality 3D reconstructions of room-scale scenes using an implicit surface representation and direct SDF-based losses on the input depth maps. Comparisons to state-of-the-art scene reconstruction methods show that our approach improves the quality of geometry reconstructions both qualitatively and quantitatively.

In summary, we propose an RGB-D based scene reconstruction method that leverages both dense color and depth observations. It is based on an effective incorporation of depth measurements into the optimization of a neural radiance field using a signed distance-based surface representation to store the scene geometry. It is able to reconstruct geometry detail that is observed by the color images, but not visible in the depth maps. In addition, our pose and camera refinement technique is able to compensate for misalignments in the input data, resulting in state-of-the-art reconstruction quality which we demonstrate on synthetic as well as on real data from ScanNet [1].

4.2 Related Work

Our approach reconstructs geometry from a sequence of RGB-D frames, leveraging both dense color and depth information. It is related to classical fusion-based 3D reconstruction methods [5]–[7], [22], [88], learned 3D reconstruction [82], [89]–[92], as well as to recent coordinate-based scene representation models [8], [93], [94].

Classical 3D Reconstruction. There exists a wide range of methods for RGB and RGB-D based 3D reconstruction that are not based on deep learning. Reconstructing objects and scenes can be done using passive stereo systems that rely on stereo matching from two or multiple color views [95], [96], Structure-from-Motion [25], or SLAM-based [97]–[99] methods. These approaches may use disjoint representations, like oriented patches [100], volumes [101], or meshes [102] to reconstruct the scene or object. Zollhöfer et al. [88] review the 3D reconstruction methods that rely on range data from RGB-D cameras like the Kinect. Most of these methods are based on [7], where multiple depth measurements are fused using a signed distance function (SDF) which is stored in a uniform 3D grid. E.g., KinectFusion [5] combines such representation with real-time tracking to reconstruct objects and small scenes in real-time. To handle large scenes Nießner et al. [22] propose a memory-efficient storage of the SDF grid using spatial hashing. To handle the loop closure problem when scanning large-scale scenes, bundle adjustment can be used to refine the camera poses [6]. In addition, several regularization techniques have been proposed to handle outliers during reconstruction [103]–[105].

Deep Learning for 3D Reconstruction. To reduce artifacts from classical reconstruction methods, a series of methods was proposed that use learned spatial priors to predict depth maps from color images [106]–[108], to learn multi-view stereo using 3D CNNs on voxel grids [109]–[111], or multi-plane images [112], to reduce the influence of noisy depth values [92], to complete incomplete scans [91], [113], to learn image features for SLAM [114]–[116] or feature fusion [117]–[119], to predict normals [120], or to predict objects or parts of a room from single images [121]–[125]. Most recently coordinate-based models have become popular [94]. These models use a scene representation that is based on a deep neural network with fully connected layers, i.e., a multi-layer perceptron (MLP) [79], [94]. As input the MLP takes a 3D location in the model space and outputs for example, occupancy [82], [89], [90], [126]–[129], density [8], radiance [8], color [130], or the signed distance to the surface [131]–[133]. Scene Representation Networks [93] combine such a representation with a learned renderer which is inspired by classical sphere tracing, to reconstruct objects from single RGB images. Instead, Mildenhall et al. [8] propose a method that represents a scene as a neural radiance field (NeRF) using a coordinate-based model, and a classical, fixed volumetric rendering formulation [134]. Based on this representation, they show impressive novel view synthesis results, while only requiring color input images with corresponding camera poses and intrinsics. Besides the volumetric image formation, a key component of the NeRF technique is a positional encoding layer, that uses sinusoidal functions to improve the learning proper-

ties of the MLP. In follow-up work, alternatives to the positional encoding were proposed, such as Fourier features [135] or sinusoidal activation layers [83]. NeRF has been extended to handle in-the-wild data with different lighting and occluders [136], dynamic scenes [137], [138], avatars [139], and adapted for generative modeling [140], [141] and image-based rendering [142], [143]. Others have focused on resectioning a camera given a learned NeRF [144], and optimizing for the camera poses while learning a NeRF [145], [146].

In our work, we take advantage of the volumetric rendering of NeRF and propose the usage of a hybrid scene representation that consists of an implicit surface representation (SDF) and a volumetric radiance field. We incorporate depth measurements in this formulation to achieve robust and metric 3D reconstructions. In addition, we propose a camera refinement scheme to further improve the quality of the reconstruction. In contrast to NeRF which uses a density based volumetric representation of the scene, our implicit surface representation leads to high quality geometry estimates of entire scenes.

Concurrent Work. In concurrent work, Wang et al. [87] present NeuS which uses an implicit surface representation to improve novel view synthesis of NeRF. Wei et al. [86] propose a multi-view stereo approach to estimate dense depth maps which they use to constrain the sampling region when optimizing a NeRF. Similarly, Neff et al. [85] restrict the volumetric rendering to near surface regions. Additional constraints on the depth predictions of NeRF were proposed by Deng et al. [84]. In contrast to these, our method focuses on accurate 3D reconstructions of room-scale scenes, with explicit incorporation of depth measurements using an implicit surface representation.

4.3 Method

We propose an optimization-based approach for geometry reconstruction from an RGB-D sequence of a consumer-level camera (e.g., a Microsoft Kinect). We leverage both the N color frames \mathcal{I}_i as well as the corresponding aligned depth frames \mathcal{D}_i to optimize a coordinate-based scene representation network. Specifically, our hybrid scene representation consists of an implicit surface representation based on a truncated signed distance function (TSDF) and a volumetric representation for the radiance. As illustrated in Figure 4.2, we use differentiable volumetric integration of the radiance values [134] to compute color images from this representation. Besides the scene representation network, we optimize for the camera poses and intrinsics. We initialize the camera poses \mathcal{T}_i using BundleFusion [6]. At evaluation time, we use Marching Cubes [20] to extract a triangle mesh from the optimized implicit scene representation.

4.3.1 Hybrid Scene Representation

Our method is built upon a hybrid scene representation which combines an implicit surface representation with a volumetric appearance representation. Specifically, we

implement this representation using a multi-layer perceptron (MLP) which can be evaluated at arbitrary positions \mathbf{p}_i in space to compute a truncated signed distance value D_i and view-dependent radiance value \mathbf{c}_i . As a conditioning to the MLP, we use a sinusoidal positional encoding $\gamma(\cdot)$ [8] to encode the 3D query point \mathbf{p}_i and the viewing direction \mathbf{d} .

Inspired by the recent success of volumetric integration in neural rendering [8], we render color as a weighted sum of radiance values along a ray. Instead of computing the weights as probabilities of light reflecting at a given sample point based on the density of the medium [8], we compute weights directly from signed distance values as the product of two sigmoid functions:

$$w_i = \sigma\left(\frac{D_i}{tr}\right) \cdot \sigma\left(-\frac{D_i}{tr}\right), \quad (4.1)$$

where tr is the truncation distance. This bell-shaped function has its peak at the surface, i.e., at the zero-crossing of the signed distance values. A similar formulation is used in concurrent work [87], since this function produces unbiased estimates of the signed distance field. The truncation distance tr directly controls how quickly the weights fall to zero as the distance from the surface increases. To account for the possibility of multiple intersections, weights of samples beyond the first truncation region are set to zero. The color along a specific ray is approximated as a weighted sum of the K sampled colors:

$$\mathbf{C} = \frac{1}{\sum_{i=0}^{K-1} w_i} \sum_{i=0}^{K-1} w_i \cdot \mathbf{c}_i. \quad (4.2)$$

This scheme gives the highest integration weight to the point on the surface, while points farther away from the surface have lower weights. Although such an approach is not derived from a physically-based rendering model, as is the case with volumetric integration over density values, it represents an elegant way to render color in a signed distance field in a differentiable manner, and we show that it helps deduce depth values through a photometric loss (see Sec. 4.4). In particular, this approach allows us to predict hard boundaries between occupied and free space which results in high-quality 3D reconstructions of the surface. In contrast, density-based models [8] can introduce semi-transparent matter in front of the actual surface to represent view-dependent effects when integrated along a ray. This leads to noisy reconstructions and artifacts in free space, as can be seen in Sec. 4.4.

Network Architecture Our hybrid scene representation network is composed of two MLPs which represent the shape and radiance, as depicted in Figure 4.2. The shape MLP takes the encoding of a queried 3D point $\gamma(\mathbf{p})$ as input and outputs the truncated signed distance D_i to the nearest surface. The task of the second MLP is to produce the surface radiance for a given encoded view direction $\gamma(\mathbf{d})$ and an intermediate feature output of the shape MLP. The view vector conditioning allows our method to deal with view-dependent effects like specular highlights, which would otherwise have to be modeled by deforming the geometry. Since color data is often subject to varying exposure

or white-balance, we learn a per-frame latent corrective code vector as additional input to the radiance MLP [136].

Pose and Camera Refinement The camera poses \mathcal{T}_i , represented with Euler angles and a translation vector for every frame, are initialized with BundleFusion [6] and refined during the optimization. Inspired by [147], an additional image-plane deformation field in form of a 6-layer ReLU MLP is added as a residual to the pixel location before unprojecting into a 3D ray to account for possible distortions in the input images or inaccuracies of the intrinsic camera parameters. Note that this correction field is the same for every frame. During optimization, camera rays are first shifted with the 2D vector retrieved from the deformation field, before being transformed to world space using the camera pose \mathcal{T}_i .

4.3.2 Optimization

We optimize our scene representation network by randomly sampling a batch of P_b pixels from the input dataset of color and depth images. For each pixel p in the batch, a ray is generated using its corresponding camera pose and S_p sample points are generated on the ray. Our global objective function $\mathcal{L}(\mathcal{P})$ is minimized w.r.t. the unknown parameters \mathcal{P} (the network parameters Θ and the camera poses \mathcal{T}_i) over all B input batches and is defined as:

$$\mathcal{L}(\mathcal{P}) = \sum_{b=0}^{B-1} \lambda_1 \mathcal{L}_{rgb}^b(\mathcal{P}) + \lambda_2 \mathcal{L}_{fs}^b(\mathcal{P}) + \lambda_3 \mathcal{L}_{tr}^b(\mathcal{P}). \quad (4.3)$$

$\mathcal{L}_{rgb}^b(\mathcal{P})$ measures the squared difference between the observed pixel colors \hat{C}_p and predicted pixel colors C_p of the b -th batch of rays:

$$\mathcal{L}_{rgb}^b(\mathcal{P}) = \frac{1}{|P_b|} \sum_{p \in P_b} (C_p - \hat{C}_p)^2. \quad (4.4)$$

\mathcal{L}_{fs}^b is a ‘free-space’ objective, which forces the MLP to predict a value of tr for samples $s \in S_p^{fs}$ which lie between the camera origin and the truncation region of a surface:

$$\mathcal{L}_{fs}^b(\mathcal{P}) = \frac{1}{|P_b|} \sum_{p \in P_b} \frac{1}{|S_p^{fs}|} \sum_{s \in S_p^{fs}} (D_s - tr)^2. \quad (4.5)$$

For samples within the truncation region ($s \in S_p^{tr}$), we apply $\mathcal{L}_{tr}^b(\mathcal{P})$, the signed distance objective of samples close to the surface:

$$\mathcal{L}_{tr}^b(\mathcal{P}) = \frac{1}{P_b} \sum_{p \in P_b} \frac{1}{|S_p^{tr}|} \sum_{s \in S_p^{tr}} (D_s - \hat{D}_s)^2, \quad (4.6)$$

where D_s is the predicted signed distance of sample s , and \hat{D}_s the signed distance observed by the depth sensor, along the optical axis. In our experiments, we use a truncation distance $tr = 5$ cm, and scale the scene so that the truncation region maps to $[-1, 1]$ (positive in front of the surface, negative behind).

The S_p sample points on the ray are generated in two steps. In the first step S'_c sample points are generated on the ray using stratified sampling. Evaluating the MLP on these S'_c sample points allows us to get a coarse estimate for the ray depth by explicitly searching for the zero-crossing in the predicted signed distance values. In the second step, another S'_f sample points are generated around the zero-crossing and a second forward pass of the MLP is performed with these additional samples. The output of the MLP is concatenated to the output from the first step and color is integrated using all $S'_c + S'_f$ samples, before computing the objective loss. It is important that the sampling rate in the first step is high enough to produce samples within the truncation region of the signed distance field, otherwise the zero-crossing may be missed.

We implement our method in Tensorflow using the ADAM optimizer [148] with a learning rate of 5×10^{-4} and set the loss weights to $\lambda_1 = 0.1$, $\lambda_2 = 10$ and $\lambda_3 = 6 \times 10^3$. We run all of our experiments for 2×10^5 iterations, where in each iteration we compute the gradient w.r.t. $|P_b| = 1024$ randomly chosen rays. We set the number of S'_f samples to 16. S'_c is chosen such that there is on average one sample for every 1.5 cm of the ray length. The ray length itself needs to be greater than the largest distance in the scene that is to be reconstructed and ranges from 4 to 8 meters in our scenes.

4.4 Results

In the following, we evaluate our method on real, as well as on synthetic data. For the shown results, we use Marching Cubes [20] with a spatial resolution of 1 cm to extract a mesh from the reconstructed signed distance function.

Results on real data. We test our method on the ScanNet dataset [1] which provides RGB-D sequences of room-scale scenes. The data has been captured with a StructureIO camera which provides quality similar to that of a Kinect v1. The depth measurements are noisy and often miss structures like chair legs or other thin geometry. To this end our method proposes the additional usage of a dense color reconstruction loss, since regions that are missed by the range sensor are often captured by the color camera. To compensate for the exposure and white balancing of the used camera, our approach learns a per-frame latent code as proposed in [136]. In Figure 4.3, we compare our method to the original ScanNet BundleFusion reconstructions which often suffer from severe camera pose misalignment. Our approach jointly optimizes for the scene representation network as well as the camera poses, leading to substantially reduced misalignment artifacts in the reconstructed geometry.

Quantitative evaluation. We perform a quantitative evaluation of our method on a dataset of 10 synthetic scenes for which the ground truth geometry and camera trajectory

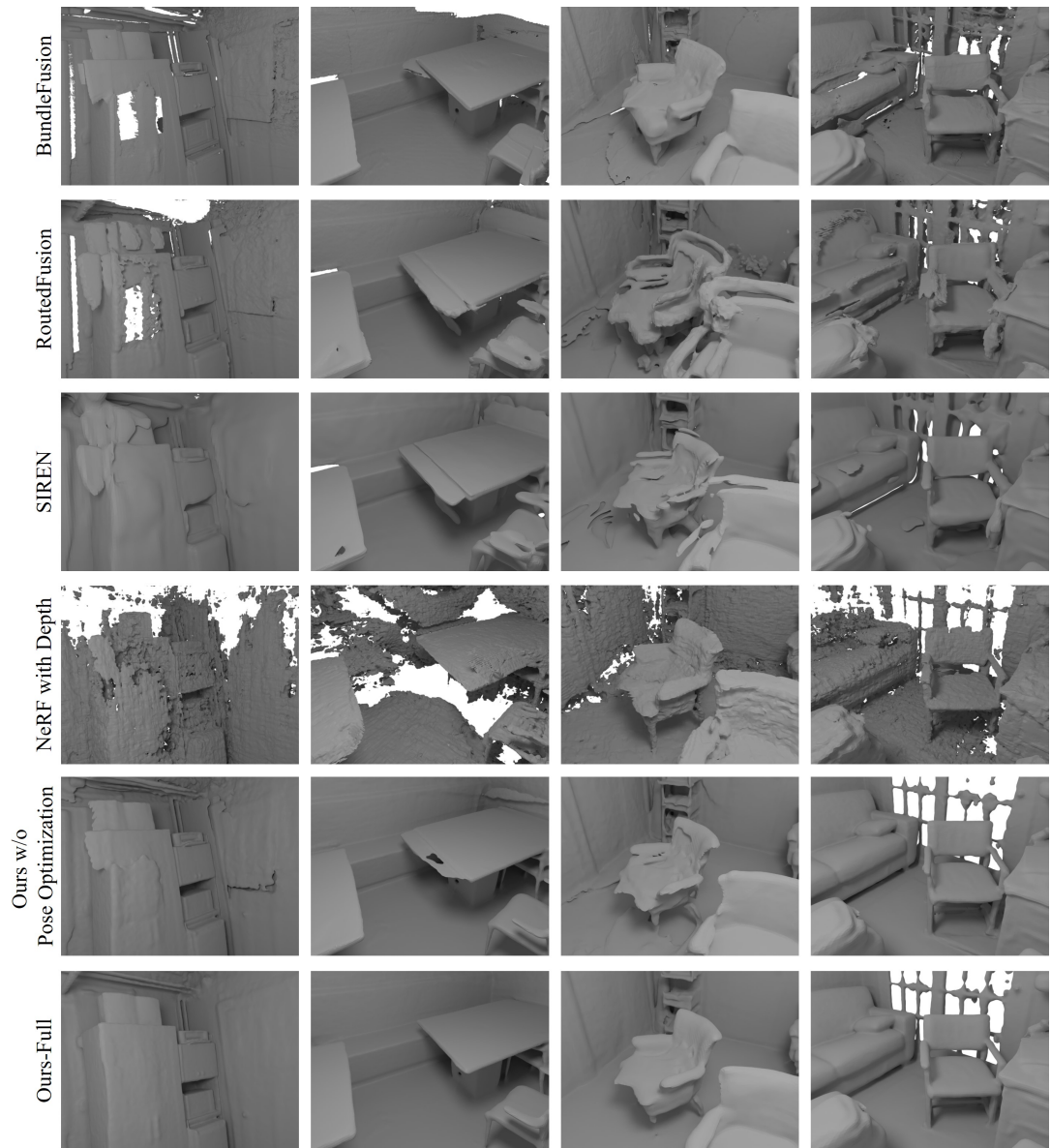


Figure 4.3: We compare our model without pose optimization and our full model with both the pose optimization and image-plane deformation field to BundleFusion, RoutedFusion, SIREN and a NeRF optimized with depth supervision in scenes 2, 5, 12, and 50 of the ScanNet dataset. Our model without pose optimization recovers smoother meshes than the density-based NeRF model, but still suffers from misalignment artifacts. These are solved by our full model to recover a clean reconstruction.

Method	C- ℓ_1 ↓	IoU ↑	NC ↑	F-score ↑
BundleFusion	0.062	0.594	0.892	0.805
RoutedFusion	0.057	0.615	0.864	0.838
COLMAP + Poisson	0.057	0.619	0.901	0.839
Conv. Occ. Nets	0.077	0.461	0.849	0.643
SIREN	0.060	0.603	0.893	0.816
NeRF + Depth	0.065	0.550	0.768	0.782
Ours (w/o pose)	0.049	0.655	0.908	0.868
Ours	0.044	0.747	0.918	0.924

Table 4.1: Reconstruction results on a dataset of 10 synthetic scenes. The Chamfer ℓ_1 distance, normal consistency and the F-score [28] are computed between point clouds sampled with a density of 1 point per cm^2 , using a threshold of 5 cm for the F-score. We voxelize the mesh to compute the intersection-over-union (IoU) between the predictions and ground truth.

are known. Note that the ground truth camera trajectory is only used for the rendering and evaluation, and not for the reconstruction. For each frame, we render a photo-realistic image using Blender [21], [149]. We apply noise and artifacts, similar to those of a real depth sensor [150]–[153]. On this data, we compare our technique to several state-of-the-art methods that use either depth input only, or both color and depth data to reconstruct geometry (see Tab. 4.1).

BundleFusion. BundleFusion [6] uses the color and depth input to reconstruct the scene. It is a classical depth fusion approach [7] which compensates misalignments using a global bundle adjustment approach.

RoutedFusion. RoutedFusion [92] uses a routing network which filters sensor-specific noise and outliers from depth maps and computes pixel-wise confidence values, which are used by a fusion network to produce the final SDF. It takes the depth maps and camera poses as input.

COLMAP with screened Poisson surface reconstruction. We obtain camera poses using COLMAP [3], [24], [25] and use these to back-project depth maps into world space. We obtain a mesh by applying screened Poisson surface reconstruction [17] on the resulting point cloud.

Convolutional Occupancy Networks. We accumulate the point clouds from the depth maps using BundleFusion poses and evaluate the pre-trained convolutional occupancy networks model [89] provided by the authors (which has been used on similar data [2]).

SIREN. We optimize a SIREN [83] per scene using the back-projected point cloud data. The ICL-NUIM [153] scene on which the method was originally tested, is also included in our synthetic dataset.

NeRF with an additional depth loss. NeRF [8] proposes using the expected ray termination distance as a way to visualize the depth of the scene. In our baseline, we add an

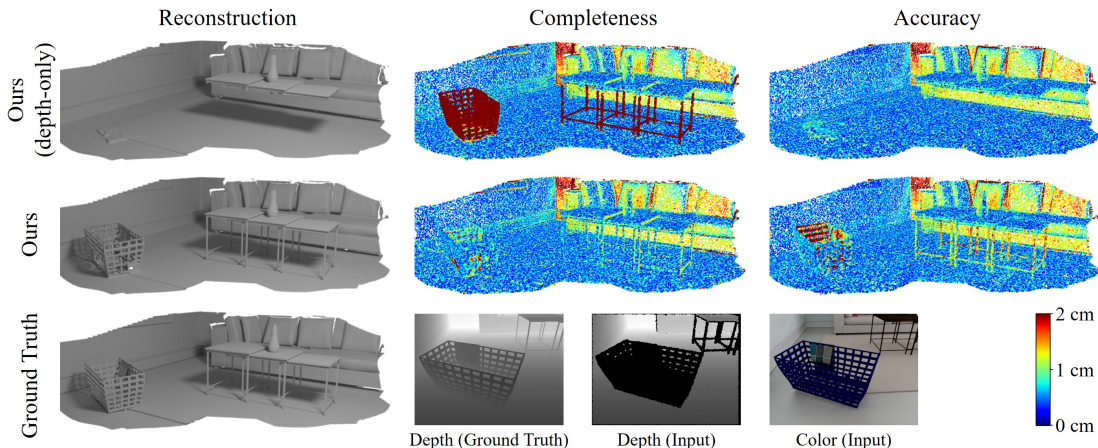


Figure 4.4: Accuracy shows how close ground truth points are to predicted points, while completeness shows how close predicted points are to ground truth points. Geometry reconstructed purely through the photometric loss has slightly lower accuracy than geometry for which depth observations were also available. Furthermore, the accuracy and completeness drop in distant areas, which had less multi-view constraints and more noise in the depth measurements.

additional loss to NeRF, where this depth value is compared to the input depth using an L2 loss. Note that this baseline still uses NeRF’s density field to represent geometry.

As can be seen in Tab. 4.1, our approach with camera refinement results in the lowest Chamfer distance, and the highest IoU, normal consistency (mean of the dot product of the ground truth and predicted normals), and F-score [28]. Especially, the comparison to the density-based NeRF with an additional depth constraint shows the benefit of our proposed hybrid scene representation.

Ablation studies. We conduct ablation studies to justify our choice of network architecture and training parameters. In Figure 4.3, we show the difference between a volumetric representation (density field, ‘NeRF with Depth’) to an implicit surface representation (signed distance field, ‘Ours-Full’) on real data from ScanNet [1]. While representing scenes with a density field works great for color integration, extracting the geometry itself is a challenging problem. Although small variations in density may not affect the integrated color much, they cause visible noise in the extracted geometry and produce floating artifacts in free space. These artifacts can be reduced by choosing a different iso-level for geometry extraction with Marching Cubes, but this leads to less complete reconstructions. In contrast, a signed distance field models a smooth boundary between occupied and free space, and we show that it can be faithfully represented by an MLP. However, the reconstruction quality is still limited by the provided camera poses, as can be seen in Figure 4.3 (e.g., the cabinet in the left column). Optimizing for pose corrections further improves the quality of our reconstructions.

Method	Pos. error (meters) ↓	Rot. error (degrees) ↓
BundleFusion	0.033	0.571
COLMAP	0.038	0.692
Ours	0.021	0.144

Table 4.2: Based on our synthetic dataset, we evaluate the average positional and rotational errors of the estimated camera poses. Our method is able to further increase the pose estimation accuracy compared to its BundleFusion initialization.

Effect of the photometric term. A fundamental component of our method is the use of a photometric term to infer depth values which are missing from camera measurements. We analyze the effect of this term on the synthetic scene in Figure 4.4, where we simulate missing geometry of the table legs and the meshed basket. In the figure, we visualize the completeness and accuracy. In contrast to a model without the photometric term, our method is still able to reconstruct the missing geometry leveraging the RGB observations.

For our full approach, we also separately evaluate the reconstruction quality of geometry where depth measurements were available and where they were missing. Regions that relied only on color have a somewhat worse average accuracy of 11 mm, compared to 8 mm for regions that had access to depth measurements. We refer the reader to the supplemental material for more details and a qualitative comparison on real data.

Effect of pose refinement. We show that initial camera pose estimates can be further improved by jointly optimizing for the rotation and translation parameters of the cameras which are initialized with BundleFusion [6]. We quantitatively evaluate this on all scenes in our synthetic dataset. An aggregate of the positional and rotational errors of different methods is presented in Tab. 4.2. A detailed per-scene breakdown is given in the supplemental material. In Tab. 4.1 and Figure 4.3, we show that optimizing camera poses reduces geometry misalignment artifacts and improves the overall reconstruction, both quantitatively and qualitatively.

Effect of the image-plane deformation field. To evaluate the effect of the pixel-space deformation field, we initialize the camera with an incorrect focal length and optimize our model with and without the deformation field. Tab. 4.3 shows that the deformation field mitigates this inaccuracy in the camera’s intrinsic parameters which leads to significantly better reconstruction results compared to the model that does not use the deformation field. Figure 4.5 showcases the effects of our camera pose and image-plane deformation field [6]. Blurry frames and sparse features lead to systematic camera pose errors in BundleFusion. Our method improves these camera poses and the camera distortion model, and, thus, is able to better align scene features, resulting in higher reconstruction quality.

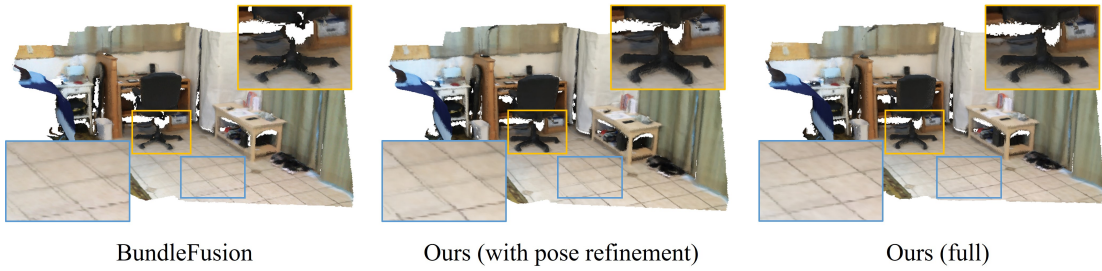


Figure 4.5: Our method improves the camera alignment over the baseline, as visible in the tiles of the floor. The additional image-plane distortion correction results in straight and aligned edges in the reconstruction.

Method	$C\text{-}\ell_1 \downarrow$	IoU \uparrow	NC \uparrow	F-score \uparrow
Ours (w/o IPDF)	0.061	0.266	0.886	0.406
Ours (w/ IPDF)	0.031	0.609	0.911	0.904

Table 4.3: Ablation of the image-plane deformation field (IPDF) which compensates image space distortions and incorrect intrinsic parameters. The experiment is based on a synthetic scene, where we assume an incorrect focal length of 570 instead of 554.26 (GT).

Limitations and future work. Similar to other methods that are based on a scene representation which uses a scene-specific MLP, our method runs offline (around 9 hours for 2×10^5 iterations using an NVIDIA RTX 3090). Recent methods that utilize voxel grids to optimize a radiance field [154], [155] have shown significantly faster convergence compared to earlier MLP-based methods and we believe that they would also be good candidates for improving our method. Nonetheless, our proposed method offers a high-quality scene reconstruction which outperforms online fusion approaches. Another limitation is the global MLP which stores the entire scene information which comes at the cost of missing high-frequency local detail in very large scenes. Approaches like IF-Nets [90] or Convolutional Occupancy Networks [89] benefit from locally-conditioned MLPs and can be integrated in future work. Finally, our method was designed to handle only opaque surfaces.

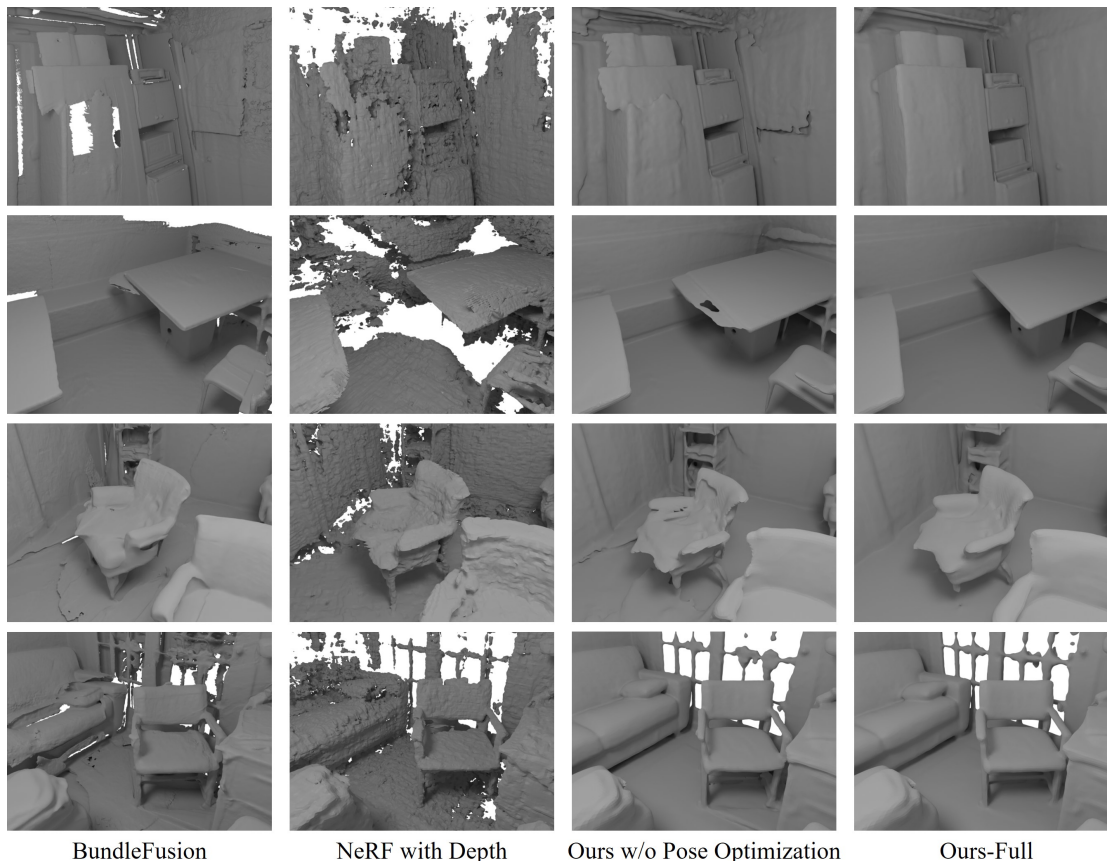


Figure 4.6: Our method obtains a high-quality 3D reconstruction from an RGB-D input sequence by training a multi-layer perceptron. In comparison to state-of-the-art methods like BundleFusion [6] or the theoretical NeRF [8] with additional depth constraints, our approach results in cleaner and more complete reconstructions. As can be seen, the pose optimization of our approach is key to resolving misalignment artifacts.

4.5 Implementation Details

We implement our method in TensorFlow v2.4.1 using the ADAM [148] optimizer with a learning rate of 5×10^{-4} and an exponential learning rate decay of 10^{-1} over 2.5×10^5 iterations. In each iteration, we compute a gradient w.r.t. $|P_b| = 1024$ randomly chosen rays. We set the number of S'_f samples to 16. S'_c is chosen so that there is on average one sample for every 1.5 cm of the ray length. Tab. 4.4 gives an overview of ray length and number of samples for each of the experiments. Internally, we translate and scale each scene so that it lies within a $[-1, 1]^3$ cube. Depending on scene size, our method takes between 9 and 13 hours to converge on a single NVIDIA RTX 3090 (see Sec. 4.10). We set the loss weights to $\lambda_1 = 0.1$, $\lambda_2 = 10$ and $\lambda_3 = 6 \times 10^3$. We use 8 bands for the

Scene	S'_c	ray length (m)	#frames
Scene 0	512	8	1394
Scene 2	256	4	1299
Scene 5	256	4	1159
Scene 12	320	5	1335
Scene 24	512	8	849
Scene 50	256	4	1163
Scene 54	256	4	1250
Breakfast room	320	5	1167
Green room	512	8	1442
Grey-white room	512	8	1493
ICL living room	320	5	1510
Kitchen 1	512	8	1517
Kitchen 2	640	10	1221
Morning apartment	256	4	920
Staircase	512	8	1149
Thin geometry	256	4	395
White room	512	8	1676

Table 4.4: We list the number of samples S'_c and the ray length in meters that were used to reconstruct each of the ScanNet scenes and the synthetic scenes. Note that these settings are dependent on the scene size.

positional encoding of the point coordinates and 4 bands to encode the view direction vector.

To account for distortions or inaccuracies of the intrinsic parameters, a 2D deformation field of the camera pixel space in form of a 6-layer MLP, with a width of 128, is used.

4.6 Per-scene Quantitative Evaluations

In Tab. 4.8 and Tab. 4.9 we present a per-scene breakdown of the quantitative analysis from the main paper (see Sec. 4, Tab. 1 and Tab. 2 in the main paper). The corresponding qualitative results are shown in Figure 4.14 and Figure 4.15.

Reconstruction Evaluation. The goal of our method is to reconstruct a scene from color and depth data, i.e., we do not aim for scene completion. To evaluate the reconstruction quality, we evaluate the quality of reconstructions w.r.t. Chamfer distance ($C-\ell_1$), intersection-over-union (IoU), normal consistency (NC) based on cosine similarity, and F-score. These metrics are computed on surfaces which were visible in the color and depth streams (geometry within the viewing frusta of the input images). Specifically, we subdivide all meshes to have a maximum edge length of below 1.5 cm and use the ground truth trajectory to detect vertices which are visible in at least one camera. Triangles

Scene	URL	License
ScanNet	http://www.scan-net.org/	MIT
Breakfast room	https://blendswap.com/blend/13363	CC-BY
Green room	https://blendswap.com/blend/8381	CC-BY
Grey-white room	https://blendswap.com/blend/13552	CC-BY
ICL living room	https://www.doc.ic.ac.uk/~ahanda/VaFRIC/iclnuim.html	CC-BY
Kitchen 1	https://blendswap.com/blend/5156	CC-BY
Kitchen 2	https://blendswap.com/blend/11801	CC-0
Morning apart.	https://blendswap.com/blend/10350	CC-0
Staircase	https://blendswap.com/blend/14449	CC-BY
Thin geometry	https://blendswap.com/blend/8381	CC-BY
White room	https://blendswap.com/blend/5014	CC-BY

Table 4.5: Source and license information of the used data.

which have no visible vertices, either due to not being in any of the viewing frusta or due to being occluded by other geometry, are culled. This is necessary to avoid computing the error in regions such as occluded geometry in the synthetic ground truth mesh or in regions where the network output is unpredictable because the region was never seen at training time. The culled geometry is sampled with a density of 1 point per cm^2 and the error metrics are evaluated on the sampled point clouds. To evaluate the IoU, we voxelize the reconstruction using voxels with an edge length of 5 cm. The F-score is also computed using a 5 cm threshold.

Synthetic Dataset. Our synthetic dataset which we use for numeric evaluation purposes consists of 10 scenes published under either the CC-BY or CC-0 license (see Tab. 4.5). We define a trajectory by a Catmull-Rom spline interpolation [156] on several manually chosen control points. We use BlenderProc [149] to render color and depth images for each camera pose in the interpolated trajectory. Noise is applied to the depth maps to simulate sensor noise of a real depth sensor [150]–[153]. For the ICL scene [153], we use the color and noisy depth provided by the authors and do not render our own images. The scenes in the dataset have various sizes, complexity and materials like highly specular surfaces or mirrors. BundleFusion [6] is used to get an initial estimate of the camera trajectory. This estimated trajectory is used by all methods other than COLMAP to allow a fair comparison.

4.7 Ablation Studies

In this section, we present additional details for the ablation studies described in the main paper, and show further studies to test the robustness and the limitations of our method. In Figure 4.6, the additional results on real data demonstrate the advantages of the signed distance field and our camera refinement.

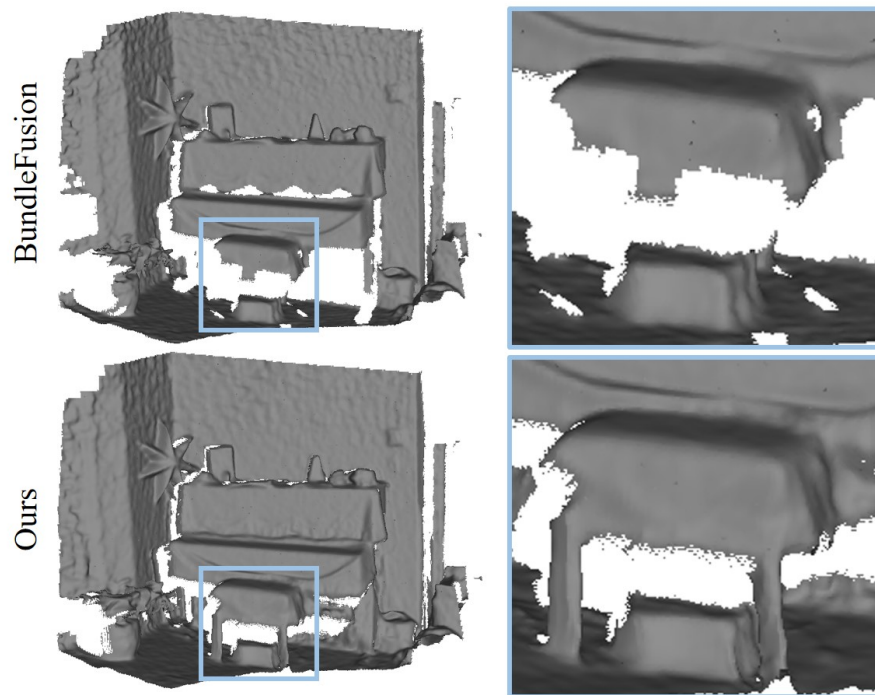


Figure 4.7: The photometric energy term encourages correct depth prediction in areas where the depth sensor did not capture any depth measurements.

Method	$C-l_1 \downarrow$	IoU \uparrow	NC \uparrow	F-score \uparrow
Ours (depth-only)	0.017	0.791	0.910	0.944
Ours (full)	0.009	0.865	0.910	0.995

Table 4.6: Detailed reconstruction results for Figure 4 from the main paper. Our method reconstructs geometry visible only in color images, leading to significantly better reconstruction results in scenes with geometry which is not captured by the depth sensor.

4.7.1 Effect of the Photometric Energy Term

In Tab. 4.6, we list the quantitative evaluation of the experiment on the effectiveness of the photometric energy term from Figure 4 in the main paper. Figure 4.7 shows the effect of the term on a real scene from the ScanNet dataset. The legs of the piano stool were not visible in any of the depth maps. Nevertheless, our method is able to reconstruct them by making use of the corresponding color data.

4.7.2 Number of Input Frames

The reconstruction quality of any reconstruction method is dependent on the number of input frames. We evaluate our method on the ‘whiteroom’ synthetic scene through multiple experiments in which we remove different numbers of frames in the dataset used for optimization. Reconstruction results are presented in Figure 4.8. Note that for these experiments we use the camera poses initialized with BundleFusion which uses all 1676 depth frames.

4.7.3 Robustness to Noisy Pose Initialization

To analyze the robustness of our method w.r.t. presence of inaccuracies in camera alignment, we apply Gaussian noise to every camera’s position and direction in the ‘whiteroom’ scene. In Figure 4.9 we present reconstruction results for poses of increasing inaccuracy. We separately show the pose errors of the refined cameras in Figure 4.10. On the reconstruction metrics, our method is robust to camera position and orientation errors of up to 5 cm and 5° respectively. The pose refinement is robust up to a noise level of 3 cm and 3° . At noise levels with a standard deviation of 10 cm and higher, some cameras are initially positioned inside geometry, preventing our method from refining their position and leading to large errors in geometry reconstruction.

4.7.4 Batch Size

Optimization with a lower batch size leads to more noise and might miss areas without depth supervision due to a lower number of multi-view constraints within the batch. A batch size that is too large will slow down the optimization and consume more GPU memory, while not offering improvements in reconstruction quality (see Figure 4.11).

4.7.5 Truncation Size

The reconstruction quality is dependent on the width of the truncation region, as shown in Tab. 4.7. The truncation region needs to account for the noise in the input (i.e., needs to be greater than the noise of the depth camera). In our experiments a truncation radius of $tr = 5$ cm gives the best results (evaluated based on the mean across multiple scenes).

Truncation (cm)	C- ℓ_1 ↓	IoU ↑	NC ↑	F-score ↑
2	0.053	0.671	0.855	0.862
3	0.023	0.766	0.901	0.930
5	0.021	0.786	0.912	0.933
10	0.024	0.742	0.908	0.912

Table 4.7: Impact of the truncation region width on reconstruction quality.

4.8 Comparison to RGB-based methods

NeuS [87] and VolSDF [133] are concurrent works that propose learning a signed distance field of an object from a set of RGB images. In contrast to these methods, our focus lies on reconstructing indoor scenes which often have large textureless regions (e.g., a white wall). Methods which use only color input will not have enough multi-view constraints to properly reconstruct these regions. In Figure 4.12, we show a case where methods that rely only on color input struggle to reconstruct high-quality geometry.

4.9 Color Reproduction of Classic and NeRF-style Methods

While our focus lies on geometry reconstruction and not accurate view synthesis, we conducted a brief analysis of the advantages and drawbacks of classic reconstruction methods [6], [147] and MLP-based radiance fields [8] when synthesizing unseen views. Classic reconstruction methods usually do not try to decouple intrinsic material parameters [6], [147] and instead optimize a texture that represents the average observation of all the input views. The resulting texture is usually high-resolution (bounded by the resolution of the input images), but does not allow for correct synthesis of view-dependent effects. Furthermore, inaccuracies in camera calibration may lead to visible seams in the optimized texture. Methods like NeRF that focus purely on high-quality novel view synthesis do not explicitly reconstruct geometry and may thus produce images riddled with artifacts for views that are too far from the input views. We believe that it is possible to combine both of these approaches to improve novel view synthesis on views far away from the ones used during the optimization and would like to encourage research in this direction. Figure 4.13 shows an example view synthesis result on the ScanNet dataset, for an out-of-trajectory camera position and orientation.

4.10 Runtime and Memory Requirements

Our method. The runtime and memory requirements of our method are dependent on the scene size. For smaller scenes where it is enough to have $S'_c = 256$ samples, our method completes 2×10^5 iterations in 9 hours on an NVIDIA RTX 3090 and requires 8.5 GB of GPU memory. When S'_c is set to 512, the runtime increases to 13 hours and the memory requirement to 10.5 GB. The memory consumption can be reduced by using smaller batches.

BundleFusion. We run BundleFusion at a voxel resolution of 1 cm for all scenes. On an NVIDIA GTX TITAN Black, depending on the size of the scene and number of frames in the camera trajectory, it takes 10 to 40 minutes to integrate the depth frames into a truncated signed distance field and extract a mesh using Marching Cubes. The memory usage is around 5.8 GB.

RoutedFusion. To train and test RoutedFusion, we used an NVIDIA RTX 3090. The routing network was trained for 24 hours on images with a resolution of 320×240 pixels. As per suggestion of the authors, we train the fusion network for 20 epochs which takes about 1.5 hours. We reconstruct all scenes at a voxel resolution of 1 cm for a fair comparison to other methods. The runtime ranges from 40 minutes to 6 hours depending on scene size and number of frames. The memory usage also heavily depends on scene size and ranges from 5.5 GB to 23 GB.

COLMAP + Poisson. In the COLMAP + Poisson baseline, the bottleneck is the global bundle adjustment process performed by COLMAP. The total runtime depends on the number of frames in the trajectory. Using all 8 cores of an Intel i7-7700K CPU, it took us about 4 hours to align all 1167 cameras in the ‘breakfast room’. The couple of minutes needed to backproject all depth maps at full resolution and run the screened Poisson surface reconstruction are negligible in comparison.

Convolutional Occupancy Networks. We reconstruct each scene using the pre-trained model provided by the authors. This takes about 2 minutes per scene and requires about 10 GB of memory.

SIREN. We train SIREN for 10^4 epochs on each scene. SIREN is trained over the complete point cloud in each epoch, so the runtime depends on the number of points in the point cloud. In our experiments on an NVIDIA RTX 3090, this ranged from 6 to 12 hours with 12 GB of memory being in use.

NeRF + Depth. We optimize NeRF using 64 samples for the coarse network and 128 samples for the fine network. On an NVIDIA RTX 3090 it takes 6 hours for 2×10^5 iterations to run. The memory usage is 4.7 GB.

4.11 Conclusion

We have presented a new method for 3D surface reconstruction from RGB-D sequences by introducing a hybrid scene representation that is based on an implicit surface function and a volumetric representation of radiance. This allows us to efficiently incorporate depth observations, while still benefiting from the differentiable volumetric rendering of the original neural radiance field formulation. As a result, we obtain high-quality surface reconstructions, outperforming traditional and learned RGB-D fusion methods. Overall, we believe our work is a stepping stone towards leveraging the success of implicit, differentiable representations for 3D surface reconstruction.






Scene	Method	C- ℓ_1 ↓	IoU ↑	NC ↑	F-score ↑	P. err. ↓	R. err. ↓
	BundleFusion	0.033	0.698	0.944	0.890	0.037	0.697
	RoutedFusion	0.033	0.714	0.918	0.901	-	-
	COLMAP + Poisson	0.033	0.668	0.935	0.893	0.009	0.210
	Conv. Occ. Nets	0.047	0.474	0.879	0.780	-	-
	SIREN	0.060	0.566	0.922	0.822	-	-
	NeRF + Depth	0.041	0.619	0.811	0.854	-	-
	Ours (w/o pose)	0.031	0.720	0.930	0.914	-	-
	Ours	0.030	0.793	0.934	0.920	0.007	0.135
		BundleFusion	0.024	0.694	0.923	0.926	0.027
RoutedFusion		0.018	0.755	0.904	0.969	-	-
COLMAP + Poisson		0.018	0.849	0.925	0.967	0.014	0.227
Conv. Occ. Nets		0.053	0.554	0.855	0.737	-	-
SIREN		0.023	0.746	0.913	0.940	-	-
NeRF + Depth		0.030	0.668	0.748	0.871	-	-
Ours (w/o pose)		0.014	0.766	0.931	0.982	-	-
Ours		0.013	0.921	0.932	0.990	0.012	0.104
		BundleFusion	0.038	0.567	0.860	0.751	0.056
	RoutedFusion	0.033	0.606	0.850	0.790	-	-
	COLMAP + Poisson	0.029	0.727	0.899	0.899	0.029	0.296
	Conv. Occ. Nets	0.048	0.480	0.841	0.601	-	-
	SIREN	0.033	0.635	0.868	0.812	-	-
	NeRF + Depth	0.040	0.563	0.764	0.697	-	-
	Ours (w/o pose)	0.032	0.640	0.864	0.806	-	-
	Ours	0.015	0.886	0.924	0.987	0.014	0.146
		BundleFusion	0.018	0.743	0.956	0.958	0.022
RoutedFusion		0.019	0.698	0.939	0.976	-	-
COLMAP + Poisson		0.023	0.727	0.947	0.966	0.029	0.836
Conv. Occ. Nets		0.112	0.352	0.841	0.507	-	-
SIREN		0.020	0.768	0.950	0.967	-	-
NeRF + Depth		0.021	0.689	0.900	0.956	-	-
Ours (w/o pose)		0.014	0.790	0.964	0.992	-	-
Ours		0.011	0.905	0.969	0.994	0.007	0.109
		BundleFusion	0.234	0.368	0.860	0.620	0.038
	RoutedFusion	0.265	0.401	0.805	0.680	-	-
	COLMAP + Poisson	0.252	0.459	0.888	0.748	0.103	0.941
	Conv. Occ. Nets	0.262	0.352	0.839	0.483	-	-
	SIREN	0.265	0.357	0.850	0.575	-	-
	NeRF + Depth	0.271	0.336	0.710	0.600	-	-
	Ours (w/o pose)	0.255	0.420	0.887	0.700	-	-
	Ours	0.252	0.447	0.886	0.718	0.030	0.114

Table 4.8: We compare the quality of our reconstruction on several synthetic scenes for which ground truth data is available. The Chamfer ℓ_1 distance, normal consistency and the F-score [28] are computed between point clouds sampled with a density of 1 point per cm^2 . We use a threshold of 5 cm for the F-score. We further voxelize each mesh to compute the intersection-over-union (IoU) between the predictions and ground truth.



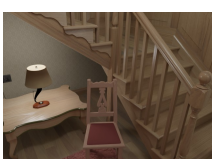


Scene	Method	C- ℓ_1 ↓	IoU ↑	NC ↑	F-score ↑	Pos. error ↓	Rot. error ↓
Kitchen 2 	BundleFusion	0.089	0.441	0.856	0.687	0.050	0.566
	RoutedFusion	0.059	0.572	0.842	0.787	-	-
	COLMAP + Poisson	0.037	0.675	0.919	0.818	0.043	1.154
	Conv. Occ. Nets	0.052	0.484	0.861	0.653	-	-
	SIREN	0.055	0.453	0.898	0.735	-	-
	NeRF + Depth	0.051	0.435	0.708	0.630	-	-
	Ours (w/o pose)	0.034	0.488	0.908	0.796	-	-
	Ours	0.032	0.637	0.903	0.890	0.083	0.450
	Morning apartment 	BundleFusion	0.012	0.767	0.885	0.968	0.008
RoutedFusion		0.013	0.815	0.870	0.976	-	-
COLMAP + Poisson		0.017	0.668	0.877	0.959	0.017	0.380
Conv. Occ. Nets		0.045	0.450	0.802	0.784	-	-
SIREN		0.013	0.727	0.873	0.966	-	-
NeRF + Depth		0.022	0.587	0.838	0.975	-	-
Ours (w/o pose)		0.011	0.787	0.887	0.983	-	-
Ours		0.011	0.716	0.888	0.982	0.005	0.093
Staircase 		BundleFusion	0.091	0.373	0.860	0.623	0.039
	RoutedFusion	0.069	0.340	0.864	0.622	-	-
	COLMAP + Poisson	0.074	0.322	0.895	0.628	0.043	0.305
	Conv. Occ. Nets	0.069	0.315	0.838	0.508	-	-
	SIREN	0.067	0.432	0.885	0.676	-	-
	NeRF + Depth	0.087	0.396	0.644	0.624	-	-
	Ours (w/o pose)	0.057	0.457	0.899	0.704	-	-
	Ours	0.045	0.565	0.920	0.853	0.016	0.123
	Thin geometry 	BundleFusion	0.019	0.764	0.909	0.922	0.009
RoutedFusion		0.023	0.708	0.829	0.881	-	-
COLMAP + Poisson		0.047	0.440	0.820	0.721	0.079	2.400
Conv. Occ. Nets		0.022	0.723	0.882	0.910	-	-
SIREN		0.021	0.733	0.887	0.913	-	-
NeRF + Depth		0.014	0.825	0.847	0.989	-	-
Ours (w/o pose)		0.009	0.857	0.911	0.995	-	-
Ours		0.009	0.865	0.910	0.995	0.010	0.037
White room 		BundleFusion	0.062	0.528	0.869	0.701	0.045
	RoutedFusion	0.038	0.545	0.817	0.799	-	-
	COLMAP + Poisson	0.036	0.652	0.904	0.796	0.018	0.167
	Conv. Occ. Nets	0.061	0.424	0.853	0.470	-	-
	SIREN	0.046	0.617	0.888	0.752	-	-
	NeRF + Depth	0.073	0.385	0.716	0.619	-	-
	Ours (w/o pose)	0.034	0.631	0.902	0.813	-	-
	Ours	0.028	0.738	0.911	0.915	0.028	0.133

Table 4.9: We compare the quality of our reconstruction on several synthetic scenes for which ground truth data is available. The Chamfer ℓ_1 distance, normal consistency and the F-score [28] are computed between point clouds sampled with a density of 1 point per cm^2 . We use a threshold of 5 cm for the F-score. We further voxelize each mesh to compute the intersection-over-union (IoU) between the predictions and ground truth.

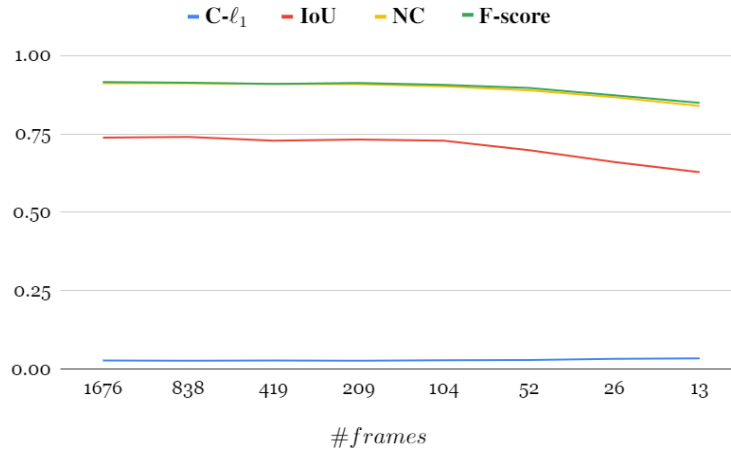


Figure 4.8: We test the robustness of our method by removing frames from the dataset used for optimization. Our method achieves good reconstruction results using as few as 13 frames.

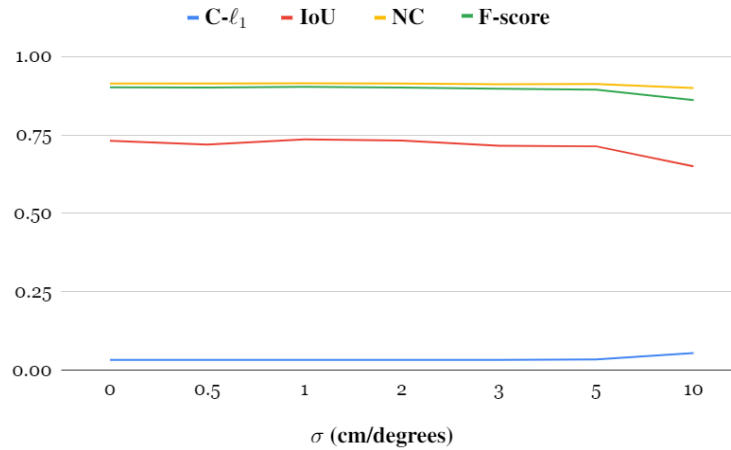


Figure 4.9: We test the robustness of our reconstructions to noise in the initial camera position and direction. Our method achieves good results even in the presence of significant noise. At $\sigma = 10$ cm, some of the cameras intersect geometry, degrading the reconstruction quality.

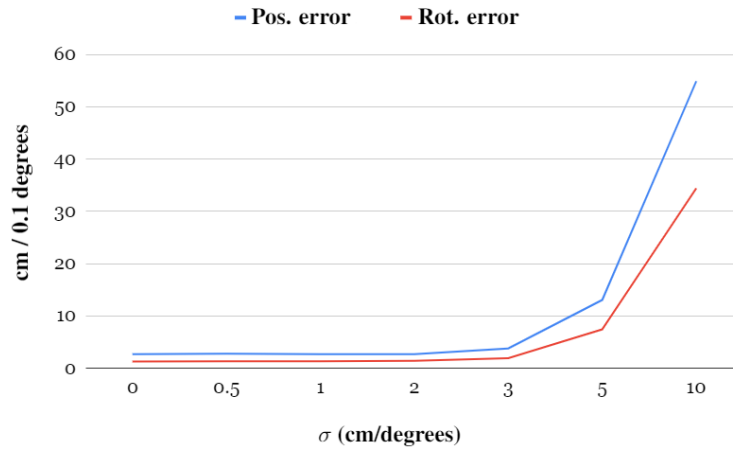


Figure 4.10: We test the robustness of our pose refinement to noise in the initial camera position and direction. The rotation error has been scaled by a factor of 10 for better visibility. Our method is able to correct poses even in the presence of significant noise. At $\sigma = 10$ cm, some of the cameras start intersecting geometry, making refinement impossible.

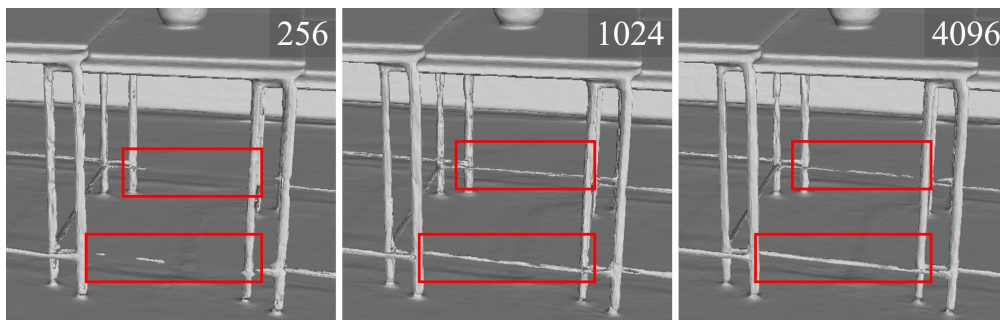


Figure 4.11: Reconstruction quality with varying batch size.

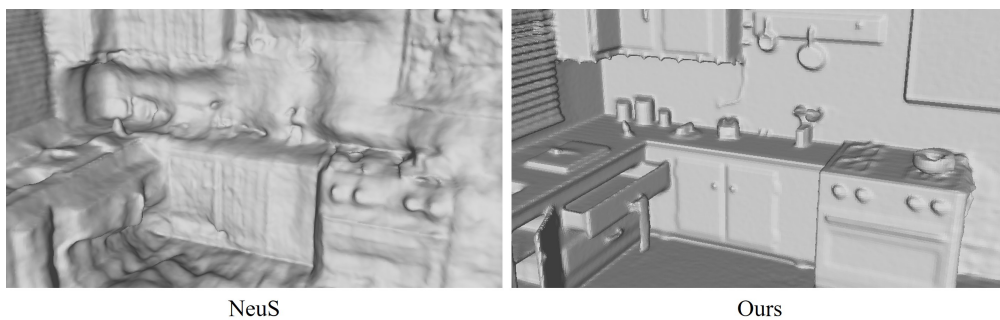


Figure 4.12: Comparison between NeuS and our method on the ‘morning apartment’ scene.

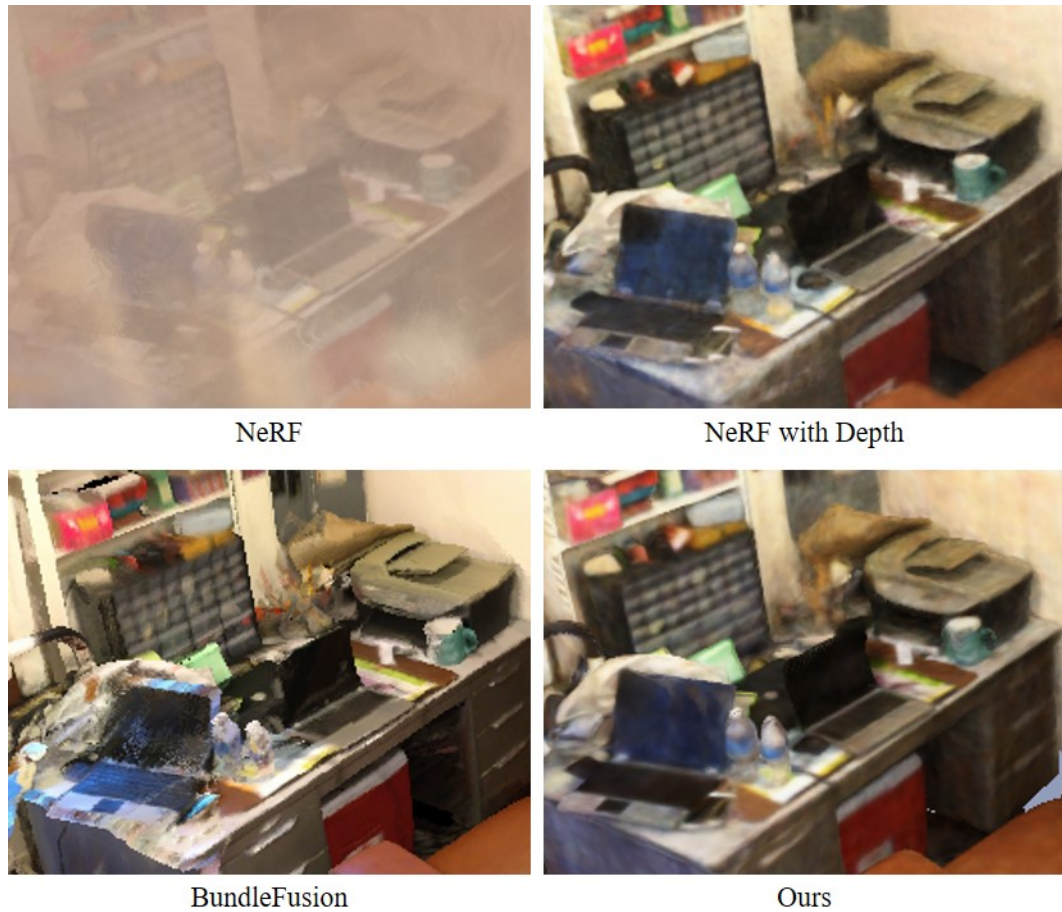


Figure 4.13: We compare the color synthesis of BundleFusion and NeRF-style methods. NeRF without any depth constraints shows severe fogging when rendering an image from a novel view. This gets resolved after adding depth constraints to the optimization. BundleFusion produces the sharpest results, but suffers from incorrect view-dependent effects and misalignment artifacts. Our method produces results similar to NeRF with a depth constraint. A combination of classic and NeRF-style methods may yield both high-quality geometry and high-quality view synthesis and we encourage further research in this direction.



Figure 4.14: We show a qualitative comparison of synthetic scene reconstructions obtained using our method and several baseline methods. The BundleFusion reconstruction is incomplete in some regions, screened Poisson and SIREN attempt to fit noise in the depth data, while the NeRF reconstruction suffers from noise in the density field. Our method manages to fill in gaps in geometry, while maintaining the smoothness of classic fusion approaches.



Figure 4.15: We show a qualitative comparison of synthetic scene reconstructions obtained using our method and several baseline methods. The BundleFusion reconstruction is incomplete in some regions, screened Poisson and SIREN attempt to fit noise in the depth data, while the NeRF reconstruction suffers from noise in the density field. Our method manages to fill in gaps in geometry, while maintaining the smoothness of classic fusion approaches.

5 High-Resolution Face Capture from Polarized Smartphone Images

This chapter introduces the following paper:

D. Azinović, O. Maury, C. Hery, M. Nießner, and J. Thies, “High-res facial appearance capture from polarized smartphone images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 16 836–16 846

Abstract of paper We propose a novel method for high-quality facial texture reconstruction from RGB images using a novel capturing routine based on a single smartphone which we equip with an inexpensive polarization foil. Specifically, we turn the flashlight into a polarized light source and add a polarization filter on top of the camera. Leveraging this setup, we capture the face of a subject with cross-polarized and parallel-polarized light. For each subject, we record two short sequences in a dark environment under flash illumination with different light polarization using the modified smartphone. Based on these observations, we reconstruct an explicit surface mesh of the face using structure from motion. We then exploit the camera and light co-location within a differentiable renderer to optimize the facial textures using an analysis-by-synthesis approach. Our method optimizes for high-resolution normal textures, diffuse albedo, and specular albedo using a coarse-to-fine optimization scheme. We show that the optimized textures can be used in a standard rendering pipeline to synthesize high-quality photo-realistic 3D digital humans in novel environments.

Contribution The method development and implementation were done by the first author. Discussions with the co-authors led to the final paper.

Revised layout and minor adaptations. Accepted version of the original publication [16] and detailed disclaimer are included in Chapter C.3 in the appendix.

5.1 Introduction

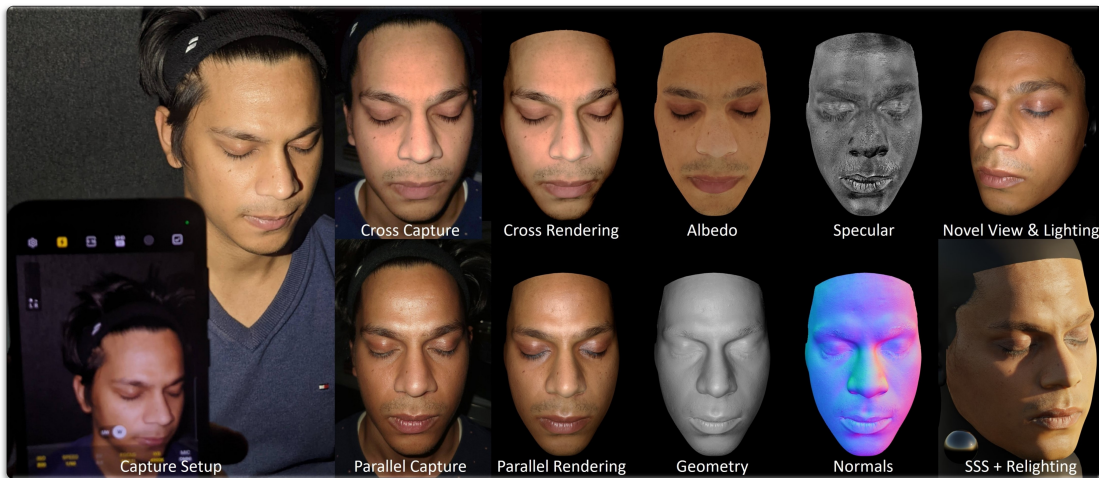


Figure 5.1: Our method obtains high-resolution skin textures from two RGB input sequences captured with polarization foils attached to a smartphone. The core idea is to separate the skin’s diffuse and specular response by capturing one cross-polarized and one parallel-polarized sequence. We recover an accurate geometry with multi-view stereo, fit a parametric head model, and employ a differentiable rendering strategy to recover 4K diffuse albedo, specular gain and normal maps. These can be used with off-the-shelf rendering software, such as Blender, to produce photo-realistic images from novel views, under novel illumination and with subsurface scattering (SSS).

In recent years, we have seen tremendous advances in the development of virtual and mixed reality devices. At the same time, the commercial availability of such hardware has led to a massive interest in the creation of ‘digital human’ assets and photo-realistic renderings of human faces. In particular, the democratization to commodity hardware would open up significant potential for asset creation in video games, other home entertainment applications, or immersive teleconferencing systems. However, rendering a human face realistically in a virtual environment from arbitrary viewpoints with changing lighting conditions is an extremely difficult problem. It involves an accurate reconstruction of the face geometry and skin textures, such as the diffuse albedo, specular gain, or skin roughness. Traditionally, this problem has been approached by recording data in expensive and carefully calibrated light stage capture setups, under expert supervision. We seek to simplify this capture process to allow individuals to reconstruct their own faces, while keeping the quality degradation compared to a light stage to a minimum.

The disentanglement of geometry and material of human faces is an extremely ill-posed problem. Current solutions involve a capture setup with multiple cameras and light sources, with millimeter-accurate calibration. A common approach to disentangling face skin surface from subsurface response is the use of polarization filters [157] in tandem with such expensive capture setups. Given such a carefully calibrated capture setting, one can use differentiable rendering to estimate the individual skin parameters in an

analysis-by-synthesis approach. While these methods do produce visually impressive results, they are limited to high-budget production studios.

In this paper, we propose a capture setup consisting of only a smartphone and inexpensive polarization foils, which can be attached to the camera lens and flashlight. Inspired by light stage capture setups, a user captures two sequences of their face, one with perpendicular filter alignment, and one with parallel alignment. This allows for a two-stage optimization, where we first reconstruct a high-resolution diffuse albedo texture of a user’s face from the cross-polarized capture, followed by recovery of the specular albedo, normal map, and roughness from the parallel-polarized views. Data is captured in a dark room to avoid requiring pre-computation of an environment map. In addition to visually compelling novel view synthesis and relighting results, our method produces editable textures and face geometry.

In summary, the key contributions of our project are:

- We propose a commodity capture setup that combines a smartphone’s camera and flashlight with polarization foils. The polarization allows us to separate diffuse from specular parts, and to reconstruct the user’s face textures, such as diffuse albedo, specular albedo and normal maps.
- Our proposed capture setting with the co-located camera and light enables separation of skin properties from illumination, which is of key importance for realistic rendering of faces.
- We propose a coarse-to-fine optimization strategy with mip-mapping, which increases sharpness of the reconstructed appearance textures.

5.2 Related Work

High-fidelity face appearance capture and reconstruction has received significant attention in the entertainment industry for creating digital humans and more recently in the AR/VR community for generating realistic avatars. In our context, facial appearance reconstruction means recovering a set of high-resolution albedo, specular (gain and roughness) and normal maps. Over the years, physically-based skin scattering models have become ever more sophisticated [158]–[160]; however, their input texture quality remains the single most important factor to photo-realism.

Polarization. For some time, polarization has been used to separate specular from diffuse [161]–[163]. These techniques rely on the fact that single bounce specular reflection does not alter the polarization state of incoming light. Riviere et al. [164] propose an approach to reconstruct reflectance in uncontrolled lighting, using the inherent polarization of natural illumination. Nogue et al. [165] recover SVBRDF maps of planar objects with near-field display illumination, exploiting Brewster angle properties. Deschaintre et al. [166] use polarization to estimate the shape and SVBRDF of an object with normal,

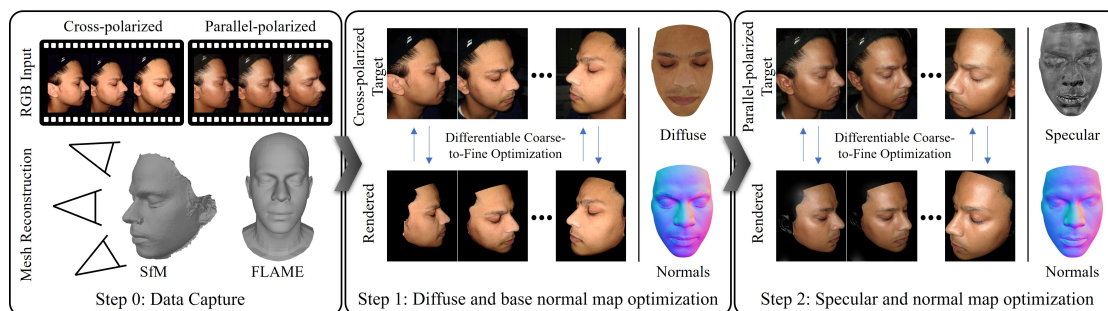


Figure 5.2: Our optimization has three steps: In step 0, we capture data with a handheld smartphone which is equipped with polarization foils (on the camera, as well as on the flashlight; see Figure 5.3). We reconstruct the facial geometry and estimate camera poses based on all captured images using structure-from-motion and multi-view stereo. To ensure consistent texture parameterization across different subjects, we non-rigidly fit a FLAME mesh to the scan. In a subsequent photometric optimization step (step 1), we estimate a high-resolution diffuse texture of the skin from the cross-polarized data, as well as an initial normal map. The reconstructed geometry, diffuse and normal map are used as input for step 2 of the optimization. Using the parallel-polarized sequence, we estimate the specular gain and final normal map in a second photometric optimization. In addition, a global skin roughness value is optimized in this step.

diffuse, specular, roughness and depth maps from a single view. Dave et al. [167] propose a similar approach for multi-view data. In MoRF [168], a studio setup with polarization is used to reconstruct relightable neural radiance fields of a face.

Lightstage capture systems. In their foundational work, Debevec et al. [157] introduced the Lightstage system to capture human face reflectance using a dome equipped with controlled lights, separating the diffuse from the specular component using polarization filters. Follow-up work reconstructs high-resolution normal maps using photometric stereo [169], compensates for motion during the capture [170] and expands the captured area [171]. The proposed capture studios didn’t come without limitations, as the lighting environment needed to be tightly controlled, the lighting patterns involved took a relatively long time, and the polarization filters were challenging to set up for multiple cameras and lights. Fyffe et al. [172]–[175] proposed the use of color gradients and spectral multiplexing to reduce capture time. With the objective of designing a more practical system, Kampouris et al. [176] demonstrate that binary gradients are sufficient for separating diffuse from specular without polarization. Lattas et al. [177] use an array of monitors or tablets for a practical binary gradients capture studio. In line with this thread of research, Gotardo et al. [178] present a multi-view setup for dynamic facial texture acquisition without the need for polarized illumination. Riviere et al. [179] build a similar lightweight system reintroducing polarization without active illumination, and modeling subsurface scattering. This effort was refined to include global illumination and polarization modeling [180]. The proposed solutions deliver impressive visual re-

sults, but require expensive and difficult to use hardware. We propose a solution for high-resolution facial texture reconstruction using commodity devices, such as smartphones.

Differentiable rendering. Recent progress in differentiable rendering [13], [14], [39], [181] has led to the development of mature frameworks [12], [32], [182] and a number of methods that try to jointly estimate appearance and lighting [183]. For an overview of differentiable rendering techniques, see [184]. Luan et al. [185] use a co-located camera and light setup to reconstruct shape and material, relying on a differentiable renderer [13] to produce unbiased gradients for shape estimation on an explicit mesh. With the same co-located setup, Zhang et al. [186] improve results by using a hybrid volume radiance field and neural SDFs for the shape estimation. While using a similar capture configuration to our work, the previous techniques focus on shape reconstruction, while we can lean on an accurate prior for the basis of our face shape. Furthermore, by using polarization, we can properly decouple diffuse from specular textures.

Dib et al. [187]–[189] propose the estimation of face skin textures by modelling the illumination with a virtual light stage, and using a differentiable ray tracer [11]. The method fits a parametric face mask to the observed images and is able to handle self-shadowing, but complex lighting environments can have an impact on separation of lighting and material. Wang et al. [190] propose a capture setup with the sun as the main light source. A FLAME [191] model is fit to the observed data, after which geometry and material are jointly refined using an analysis-by-synthesis approach. As with other methods in uncontrolled lighting, separation of individual textures remains a challenge.

Deep learning-based approaches. A wide range of work proposes learning a neural network from large collections of high-quality light stage data, and subsequently applying the model to new data [192]–[198]. Zhang et al. [199] propose learning a neural light transport model from uv-space light and view direction information. At test time, the model generalizes to novel views and lighting. Several other works propose learning neural rendering models, either from single-view [200]–[203] or multi-view [204], [205] data, for a range of different applications.

5.3 Method

We propose a two-step analysis-by-synthesis approach for the estimation of high resolution face textures, as depicted in Figure 5.2. The user captures two video sequences and a series of photographs of their face under linear-polarized point light illumination using a smartphone. The first sequence has the polarization filters oriented in a perpendicular fashion, i.e., the filter covering the camera lens is perpendicular to the filter covering the smartphone’s flashlight. In accordance with existing literature, we denote this sequence as the cross-polarized sequence. The second video sequence has parallel oriented filters and will be referred to as the parallel-polarized sequence.

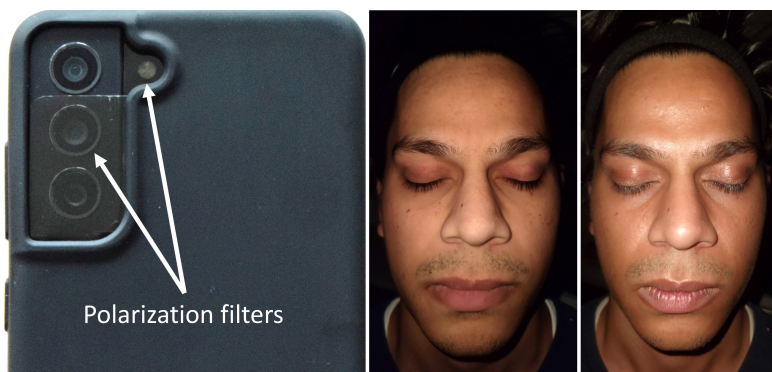


Figure 5.3: Left to right: smartphone equipped with polarization filters, cross-polarized image (perpendicular filter orientation) and parallel-polarized image (parallel filter orientation).

We use structure-from-motion and multiview-stereo on all captured frames jointly to compute the camera alignment and reconstruct coarse geometry in form of a triangle mesh. We then non-rigidly fit the FLAME model [191] to the scan and use it as our base geometry model. This fitting helps us avoid noise from the multiview-stereo and provides a consistent UV-parameterization for all subjects. Based on this geometry, we recover the diffuse albedo texture of the subject using the cross-polarized data and photo-metric optimization. While keeping the diffuse albedo fixed, we estimate the remaining textures based on the parallel-polarized data. Note that we reconstruct textures using only the photographs, as these capture more detail than the video frames. For the geometry reconstruction, we use all captured data, as we found that this leads to more robust results compared to only using a small set of photographs.

5.3.1 Capturing Polarized Data with a Smartphone

We capture one cross-polarized and one parallel-polarized video sequence with a smartphone in a dark room, with the smartphone’s flashlight as the only source of illumination. Such a capture setup has the advantage of not requiring optimization of the scene lighting, leading to better separation of appearance and shading. We assume that the flashlight is co-located with the camera lens and that its color is white. We capture a color-checker under both filter orientations to color-calibrate both sequences. This is important, since the filters introduce wavelength-dependent attenuation which tints the color of the light. We use an affine color calibration scheme to compute the corresponding color correction matrix only once, and apply it to all subsequent sequences. Furthermore, since an arbitrary smartphone’s flashlight does not behave like an ideal point light (e.g., due to occlusion by the phone’s cover along grazing directions), we pre-compute a per-pixel light attenuation map, that is multiplied with the final rendered images during optimization. To this end, we put markers on a flat white surface and record a cross-polarized sequence of the surface. We form an optimization problem with the unknowns being the surface’s diffuse texture and the per-pixel light attenuation map.

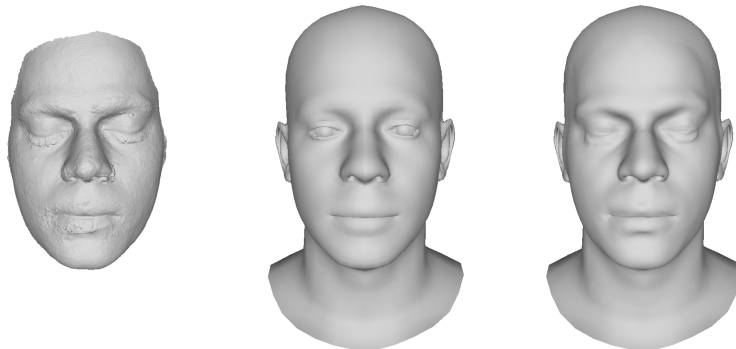


Figure 5.4: Geometry reconstruction for subject from Figure 5.3. From left to right: reconstruction via structure from motion, fitted FLAME [191] mesh, ICP-based refinement of the mesh.

The map is then kept fixed for all future face texture optimizations. We refer to the supplemental material for more detail on this calibration step.

We ensure that all captures have consistent and fixed camera settings: focal length, exposure time and white balance. The sequences are captured at 4K resolution and 30fps, but we select the sharpest frame from every 10-frames window, using variance of the Laplacian as the sharpness metric. In addition to the video data, we capture a set of cross-polarized and a set of parallel-polarized photographs to obtain higher-quality data. Since the flash is much brighter for photographs than for videos, we capture the photographs with shorter exposure and lower ISO to roughly match the brightness of the video frames. The entire capture takes about five minutes.

5.3.2 Geometry Reconstruction

We use Agisoft Metashape [26] on all frames jointly to obtain an initial mesh reconstruction. We provide Metashape with face masks estimated by [206], to make the reconstruction more robust to rigid motion of the head. We then fit the FLAME model [191] to the scanned geometry, first by optimizing the shape parameters of the FLAME face space, and then by an ICP-based as-rigid-as-possible deformation approach (see Figure 5.4). For the non-rigid deformation, we subdivide the triangles of the face region, to obtain detailed geometry. The resulting mesh is used as the base mesh for the subsequent texture optimizations.

5.3.3 Rendering Equation & BRDF

We model the skin with a spatially-varying bidirectional reflectance distribution function (SV-BRDF). Assuming a point light source \mathbf{l} in a dark environment, the rendering equation that defines the outgoing radiance $L_o(\mathbf{x}, \omega)$, at point \mathbf{x} with normal direction

\mathbf{n}^\top in direction ω , has the following simplified form:

$$L_o(\mathbf{x}, \omega) = \frac{f(\mathbf{x}, \omega)(\mathbf{n}^\top \omega)L_i(\mathbf{x}, \omega)}{|\mathbf{x} - \mathbf{l}|_2^2}. \quad (5.1)$$

Here, we make use of the fact that the light direction aligns with the view direction, i.e., $\omega_i = \omega_o = \omega$. The BRDF $f(\mathbf{x}, \omega)$ has a diffuse component f_d , and a specular component f_s . We use the Cook-Torrance [36] BRDF for our specular term:

$$f_s(\mathbf{x}, \omega) = k_s(\mathbf{x}) \frac{D(\omega, \mathbf{n}^\top, \alpha)G(\mathbf{n}, \omega)F(\mathbf{n}, \omega)}{4(\mathbf{n}^\top \omega)(\mathbf{n}^\top \omega)}, \quad (5.2)$$

with k_s being the spatially-varying specular gain and α a global roughness blend factor for the Blinn-Phong distribution term D of the 2-lobe mix (D_{12} and D_{48}) suggested by [179]. G denotes the geometry term of the Cook-Torrance BRDF model. We use Shlick’s approximation [38] for the Fresnel term F :

$$F(\mathbf{n}, \omega) = F_0 + (1 - F_0)(1 - \mathbf{n}^\top \omega)^5. \quad (5.3)$$

To model the skin’s diffuse response, we implement the BRDF model proposed by Ashikhmin and Shirley [207], [208], that accounts for the fact that a portion of the light has already scattered before penetrating the skin surface:

$$f_d(\mathbf{x}, \omega) = \frac{28k_d(\mathbf{x})}{23\pi}(1 - F_0)(1 - (1 - \frac{\mathbf{n}^\top \omega}{2})^5)^2, \quad (5.4)$$

where $F_0 = 0.04$ is the reflectance of the skin at normal incidence. Indirect light bouncing from the capture environment and on the captured face itself might have a significant contribution to pixel intensity at grazing angles, so we also add a Fresnel-modulated ambient term to our BRDF f :

$$f_a(\mathbf{x}, \omega) = k_a(\mathbf{x})(1 - (1 - F_0)(1 - (1 - \frac{\mathbf{n}^\top \omega}{2})^5)^2), \quad (5.5)$$

with an ambient map k_a which is regularized to be smooth via a total variation loss and close to zero.

Note that using a diffuse scattering model for the optimization is compatible with state-of-the-art physically-based subsurface scattering skin shading [158], [159], as shown in Figure 5.1. Production-ready subsurface scattering models typically include an albedo inversion stage, which takes a diffuse albedo as input, and converts it to extinction coefficients for the volume rendering random walk.

5.3.4 Optimization

The objective of the photometric optimization step is to minimize the difference between rendered images \hat{I} and color-corrected target images I :

$$\mathcal{L}(\hat{I}, I) = \left| W \cdot (\hat{I} - I) \right|, \quad (5.6)$$

with $\hat{I} = \mathcal{M} \cdot L_o$, where \mathcal{M} is the pre-computed light attenuation map, that accounts for uneven light distribution in different directions. We apply a per-pixel loss weight W based on the respective mip level and the angle between viewing direction and normal $\mathbf{n}^\top \omega$ to improve sharpness. Specifically, to ensure that distant or grazing angle observations do not blur the resulting textures, for each pixel that is projected from the target image to texture space, we calculate which mip level l would need to be looked up in classical forward rendering. W is set to $(\mathbf{n}^\top \omega)(1 - l)$ if the pixel corresponds to a mip level below 1, and zero otherwise.

We optimize $\mathcal{L}(\hat{I}, I)$ in two steps, using a coarse-to-fine optimization strategy in each. In the first step, we only use the cross-polarized images to optimize the spatially-varying diffuse albedo texture $k_d(\mathbf{x})$ and an initial tangent-space normal map $n(\mathbf{x})$, while assuming $f_s(\cdot) = 0$ for the specular term. In the second step, we fix the diffuse texture and optimize for specular gain $k_s(\mathbf{x})$, specular roughness α , and the final normal map $n(\mathbf{x})$. To account for potentially different light attenuation in the cross and parallel-polarized filter settings, we also optimize per-channel scaling factors for the diffuse texture. The optimization is performed entirely in texture space. In each step, we employ a four-level coarse-to-fine optimization strategy, starting with a texture resolution of 512×512 , and increasing the size by a factor of two after convergence of each level, up to the final resolution of 4096×4096 .

We implement our optimization framework in PyTorch, using nvdiffrast [32] as our differentiable renderer. We optimize on batches of 4 images, using Adam with an initial learning rate $lr_0 = 10^{-3}$ for all parameters at the beginning of every coarse-to-fine step, and updating it to $lr = lr_0 \cdot 10^{-0.001t}$ in every iteration t . We scale the FLAME mesh to unit size and set the light intensity to 10. The total optimization time is about 90 minutes.

5.4 Results

In this section, we present texture reconstruction and rendering results on several subjects. Figure 5.5 shows the texture reconstruction on several actors of different ethnicity. Our method is able to reconstruct pore-level detail in the diffuse, specular and normal maps. Further, we evaluate the quality of our reconstructed textures by rendering the mesh from novel views and under novel illumination. Figure 5.6 shows that our method faithfully reconstructs the skin’s appearance under novel views and lighting.

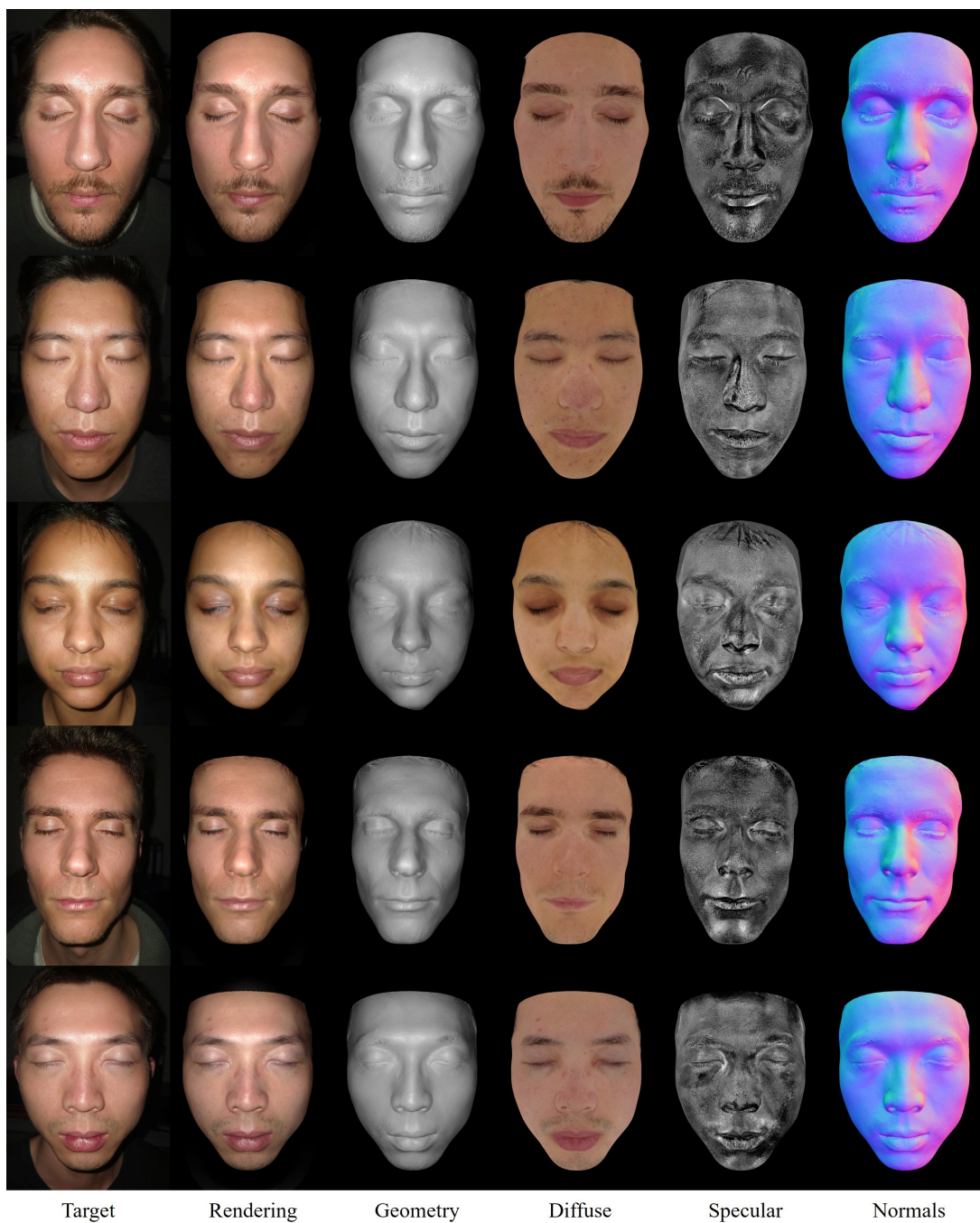


Figure 5.5: We show skin texture reconstructions of several actors of different skin type. The rendered images closely match the reference target images, and we achieve good separation of diffuse and specular textures.

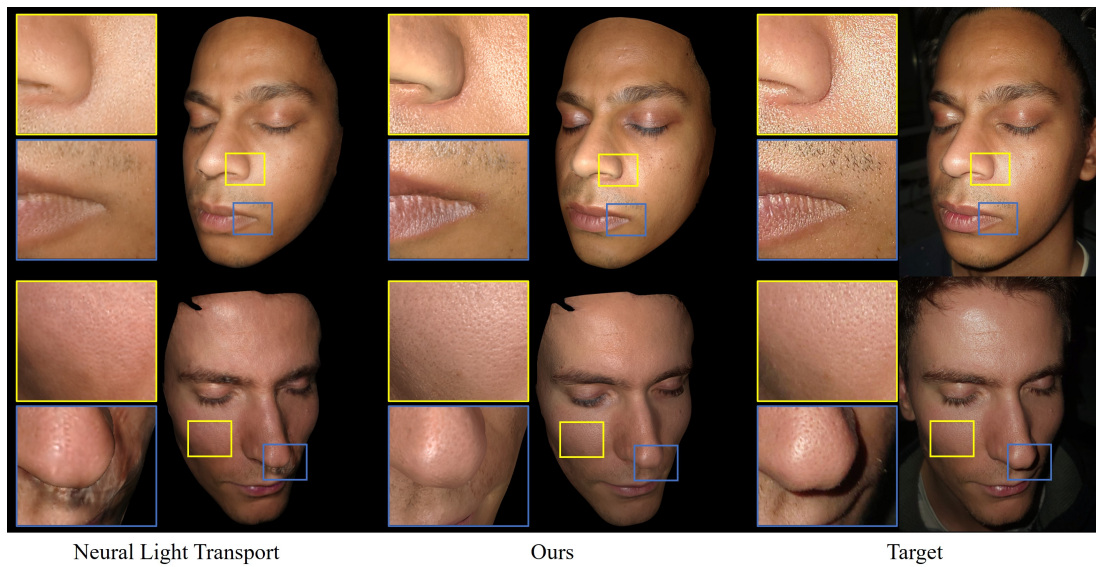


Figure 5.6: We evaluate on a validation frame from a novel viewpoint and with novel lighting that was held out during the optimization. As visible in the crop regions, our method is able to synthesize sharper texture details and specular highlights compared to NLT [199].

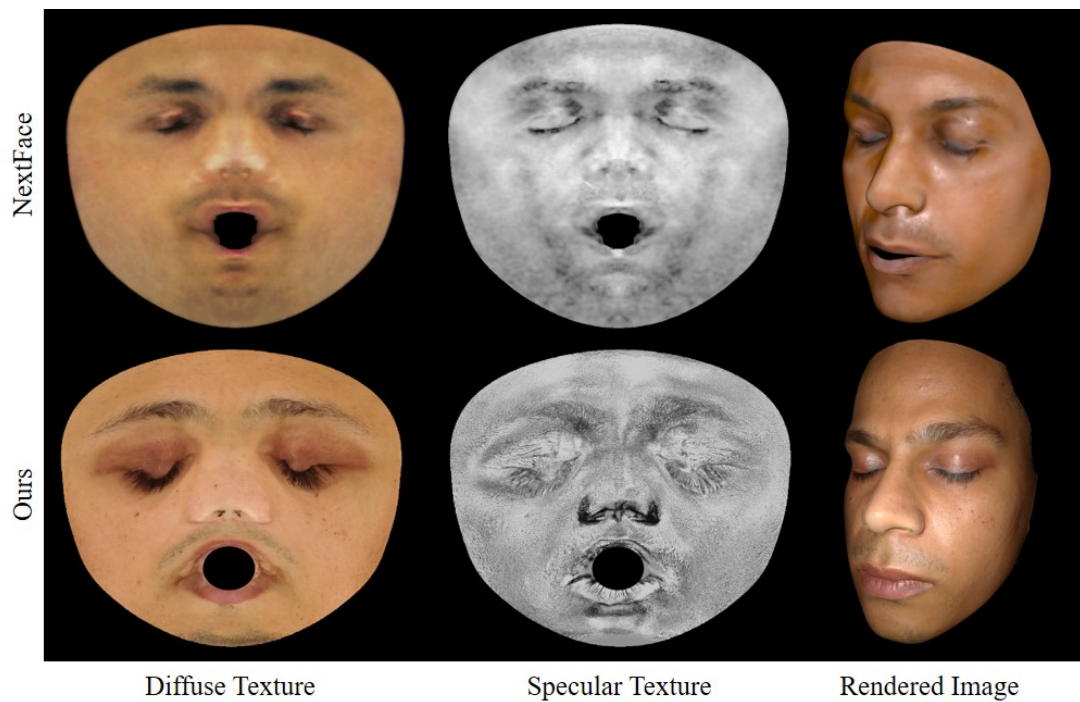


Figure 5.7: Comparison to NextFace [189] in terms of reconstructed appearance textures.

Method	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
NLT [199]	31.51	0.96	0.11
NextFace [189]	22.85	0.89	0.31
Ours	32.37	0.96	0.10

Table 5.1: We compare our method to NLT and NextFace on validation frames over 10 different subjects.

Comparison to state of the art. We perform both a qualitative and quantitative evaluation of our method and compare to state-of-the-art methods for relighting and texture reconstruction. During optimization, we hold out a validation frame on which we compute image metrics.

Neural Light Transport. Neural Light Transport [199] is a deep learning-based method that takes as input pre-computed diffuse base, light-cosine and view-cosine uv-space maps. The diffuse base is computed as the average of all observations. The cosine maps contain per-textel cosines of the angles between the normal vector and the light or view vector. Based on these inputs, as well as nearest neighbor observations, a neural network learns to predict the final shaded image. Since the method does not take light intensity and falloff into account, we optimize the rendered validation image’s brightness to match the target as closely as possible, before computing the rendering error.

NextFace. NextFace [187]–[189] first fits a morphable face model to the input frames, then estimates the face shape, pose, lighting, statistical diffuse and specular albedos by minimizing a photo-consistency loss between the target image and a ray traced estimate. In a final step, the statistical albedos are refined on a per-textel basis. We conducted several experiments with different illumination conditions and number of frames, including an experiment on our data for which we replaced the spherical harmonics lighting representation with a small area light, modelling our flashlight.

As shown in Table 5.1, our approach achieves favorable image metrics. Figure 5.6 compares our method to NLT on novel lighting and viewpoint. NLT closely matches the target by using nearby camera views, but specular highlights are often blurry, and the low number of training views results in the model producing artifacts in shadowed areas. We obtained the best NextFace results in an experiment with uniform illumination using three frames that cover the whole face region. As shown in Figure 5.7, inaccuracies in the face model fitting lead to somewhat blurry textures. This issue is exacerbated by adding more frames. Using fewer frames degraded the separation of the diffuse and specular textures. Our method is able to overcome these issues by accurately fitting a geometric model to the input data and by using polarization to separate the individual textures.

Ablation Studies. We conduct ablation studies to justify our choice of capture setup and training parameters. In Figure 5.8, we show that accounting for the direction-dependent light attenuation of a smartphone’s flashlight leads to an overall lower error in the re-rendered images. In the same figure, we also show the importance of accounting for

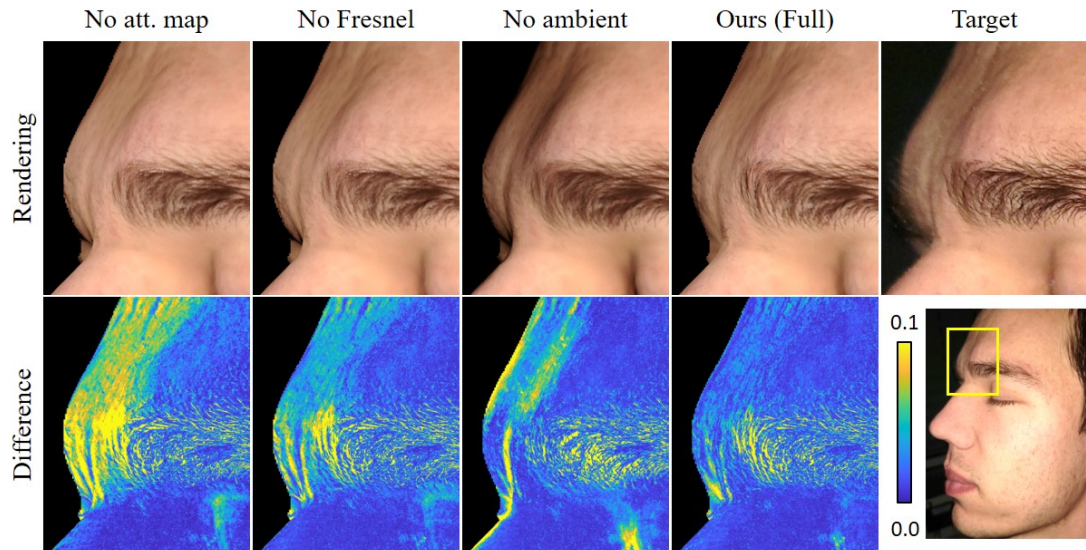


Figure 5.8: Ignoring the angle-dependent flashlight attenuation, the Fresnel effect, or the ambient light leads to an incorrect reconstruction, that can no longer reproduce the shading from all views. We account for these effects to closely match the target data.

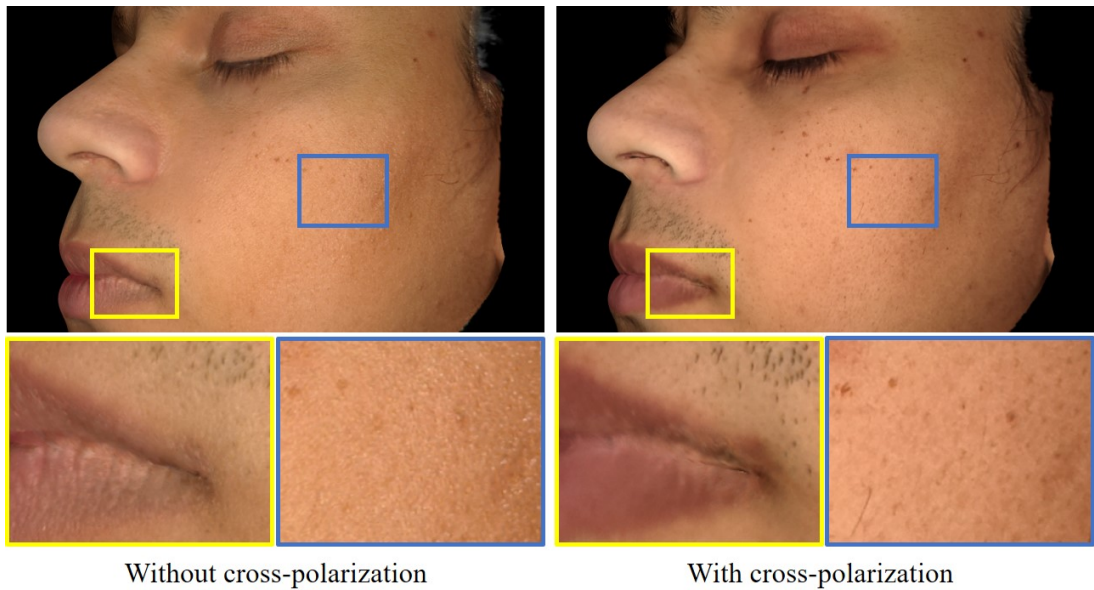


Figure 5.9: We compare joint optimization of all textures to our full approach on a purely diffuse render. Optimizing jointly leaks specular and normal map information into the diffuse texture.

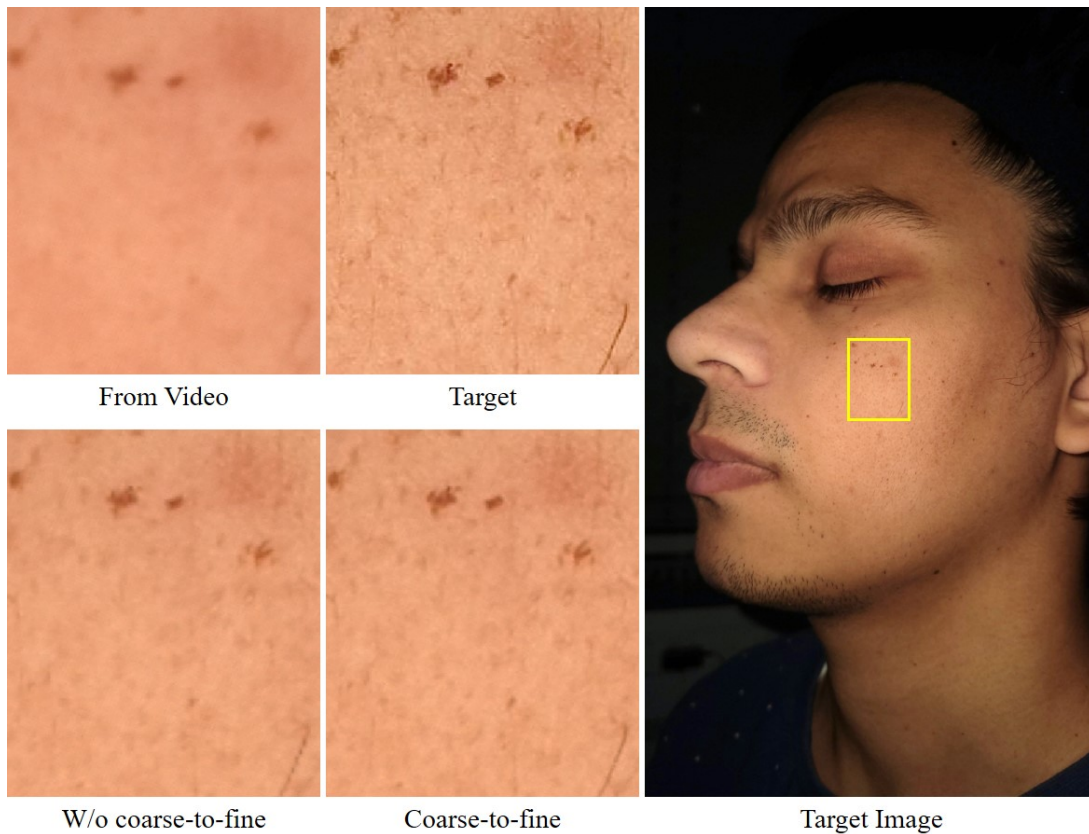


Figure 5.10: To increase sharpness, we optimize from photographs, instead of video frames. Using only pixels of the appropriate mip level in a coarse-to-fine approach further enhances results.

the Fresnel effect when reconstructing the diffuse texture. A purely Lambertian BRDF will not be able to model the skin’s diffuse response at all angles. In Figure 5.9, we show that optimizing textures without cross-polarization will leak specular information into the diffuse texture.

Coarse-to-fine optimization and mipmapping. Pixels of the target images have different footprints in uv-space, depending on distance and angle between camera and surface. Weighting the loss of each pixel equally leads to blur in the reconstruction. Optimizing coarse-to-fine, where at each resolution we use only pixels with the corresponding uv-space footprint, helps us reconstruct additional detail in the textures. Figure 5.10 shows a comparison between our full approach and a direct optimization of the highest resolution texture. We additionally show the decrease in quality when optimizing only on video frames (w/o photographs).

Runtime and memory consumption. Including 1 hour spent on MVS, our method needs about 2.5h to reconstruct a face. Photo-metric skin texture reconstruction takes about 90 minutes on an Nvidia RTX A6000. We reconstruct facial geometry with Metashape using an average of 420 video frames and 70 photographs. At a texture resolution of 4096×4096 and target image resolution of 3840×2160 , the photo-metric optimization requires 30GB of GPU memory. In comparison, NLT takes about 10h and NextFace about 6h given the same number of frames.

Discussion & Limitations. Our method reconstructs high-quality face textures with a low-cost capture routine. However, it is restricted to static expressions, i.e., it does not handle dynamically changing face geometry and textures. An avenue for future research is the reconstruction of dynamic expressions by fitting a parametric model with consistent mesh topology to each frame, and optimizing over the entire non-rigid sequence. Our method does not explicitly handle global illumination. A differentiable path tracer could potentially improve results in the concavities of the eye region. As we assume a static face with closed mouth and closed eyes, we only recover the skin area of a face. Eyes, mouth interior and hair are a subject of future work.

5.5 Calibration

To use a smartphone as a tool to capture high-quality textures of human faces, we apply a calibration step related to the flashlight and camera sensor. Specifically, we compute a light attenuation map to take into account vignetting effects and the fact that the flashlight is not an ideal point light source, and we color-calibrate the cross-polarized and parallel-polarized images.

Light attenuation map. In the general case, a smartphone’s flashlight does not behave like an ideal point light. We observed a significant decrease of light intensity towards grazing angles. To account for this effect, we compute a per-pixel attenuation map that we multiply with our rendered images to match the observations. To this end, we put calibration markers on a white wall and recorded a cross-polarized sequence (see Figure 5.11). The markers allow us to estimate camera poses for the sequence and provide us a sparse point cloud to which we fit a plane. Finally, we pose an optimization problem:

$$\operatorname{argmin}_{\mathcal{M}, k_d} \left| \left(\hat{I} - I \right) \right|, \quad (5.7)$$

with $\hat{I} = \mathcal{M} \cdot L_o$, where \mathcal{M} is the light attenuation map, and k_d the diffuse texture. Once optimized, we keep \mathcal{M} fixed for all subsequent face texture optimizations.

Color correction. We color-calibrate both the cross-polarized images and the parallel-polarized images using pre-recorded images of a Macbeth colorchecker board. We compute an affine color transformation matrix to match these calibration images to a refer-

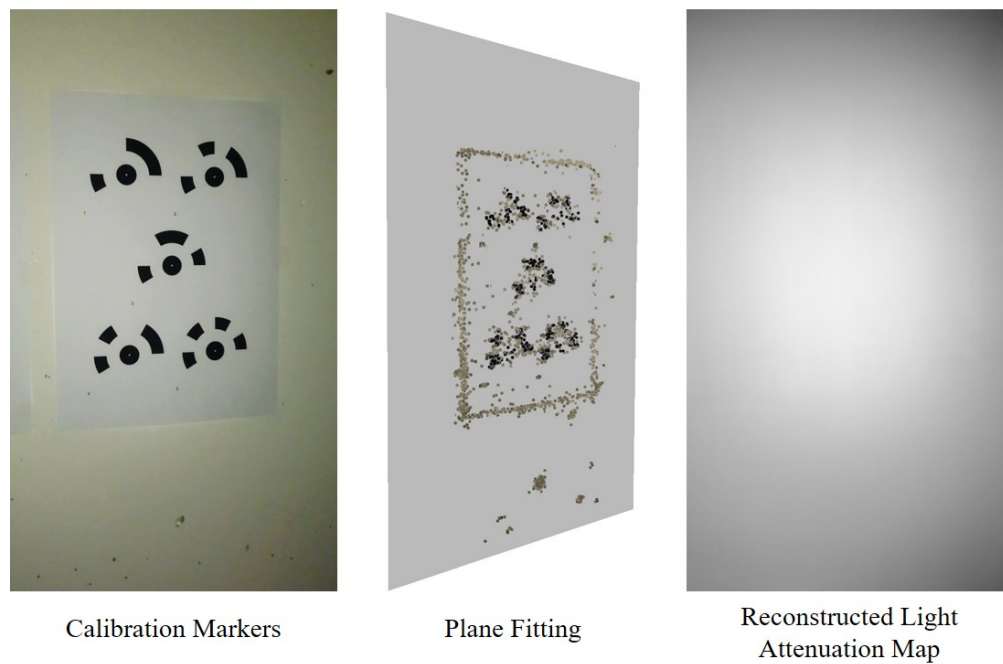


Figure 5.11: To calibrate the light of the smartphone, we record a cross-polarized sequence of a white planar surface with markers for tracking. We fit a UV-parameterized plane to the data and optimize for a light attenuation map which we use for all experiments.



Figure 5.12: We found that the polarization filters introduce a color shift depending on the polarization direction. To this end, we perform a color calibration with a Macbeth colorchecker board which we capture in both scenarios (cross-, and parallel-polarized). We use an affine color correction to match both captures, and apply this transformation to recordings of all subjects.

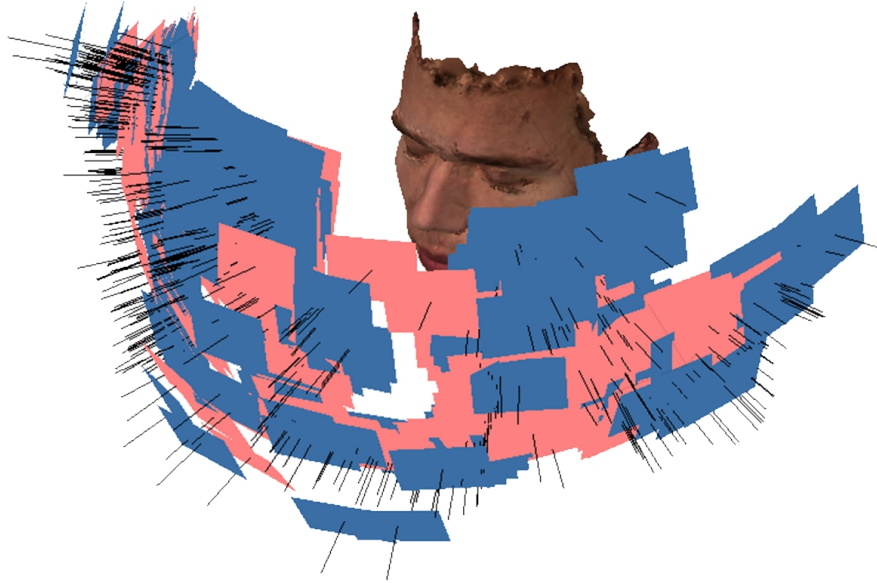


Figure 5.13: Distribution of cross-polarized (red) and parallel-polarized (blue) views.

ence color chart. This calibration step is done once for the smartphone and then used for all recorded sequences. The effect of this calibration step is shown in Figure 5.12.

Camera settings. We record our data using a Samsung Galaxy S21 FE 5G. For the video sequences, we use an ISO of 800 and exposure time of 1/60s. The photographs were shot with an ISO of 200 and exposure time of 1/90s. The smartphone’s white balance was set to 4900K.

5.6 Geometry Estimation

To estimate the geometry of a subject, we use the Structure-from-Motion method from MetaShape [26] on the captured data (see Figure 5.13 for a camera pose visualization). The resulting geometry is noisy and might contain holes, so we fit a 3DMM-based face model to the reconstruction. Specifically, we use PIPNet [209] to detect landmarks on a front-facing image of the face. These are then projected to 3D using the known camera extrinsic and intrinsic matrices. Using Procrustes’s algorithm, we get a coarse alignment between the FLAME face model [191] and the 3D landmarks. We further improve the alignment by optimizing for both a rigid transform between FLAME and nearby scan vertices, as well as the FLAME shape vector to non-rigidly fit the scan. The resulting mesh is subdivided in the face region by a factor of 16, and the eyes are removed from the mesh. Finally, we employ an As-Rigid-As-Possible (ARAP) [210] non-rigid deformation strategy to refine the face mesh, to better align with the reconstruction of MetaShape.

5.7 Comparison to Prior Work

In this section, we explain in more detail the differences between our proposed method and results, and some of the existing solutions for light stage data to which we could not compare directly. Furthermore, we discuss potential benefits of capture setups with independent view and light directions.

MoRF [168] is a generative model trained on a high-quality image database with polarization-based separation of diffuse and specular reflectance. It can generate a volumetric representation of a face based on latent ID codes, which can be optimized to fit new subjects. The database itself is created using the capture setup from [179]. Images of a subject can be rendered by first feeding the subject-specific ID code into a deformation and a canonical MLP. The canonical MLP is composed of a density, diffuse and specular branch, and the output of these branches is used in a volumetric rendering formulation, similar to [8], to render the final image. This is in contrast to our approach, which uses a triangle mesh to represent geometry, and which defines the SVBRDF on the surface of the mesh. The major advantage of MoRF is the fewer number of images it requires at test time and better facial hair and eye handling. This is, however, offset by its limited performance in accurately fitting to faces of new subjects. Furthermore, the material is not separated from lighting and the results are over-smoothed due to the low-order spherical harmonics lighting approximation.

Deep Relightable Appearance Models for Animatable Faces [198] proposes a conditional variational auto-encoder (CVAE) architecture to predict mesh vertices, a corresponding texture warp field and light-dependent textures. A late-conditioned model is first trained on light stage OLAT (one light at a time) data to predict a lit texture map of a subject’s face from its average texture (nearest fully-lit frame averaged across all cameras) and an initial estimate of the mesh vertices (provided by an off-the-shelf face tracker). This model has good generalization ability, but is not suitable for real-time rendering. Making use of the good generalization ability of the trained model, a large dataset of synthetic images is generated and used to train an early-conditioned model which can render faces under complex lighting in real-time. The biggest advantage compared to our approach is the capture and rendering of dynamic sequences. Some of the drawbacks include the necessity of a light stage capture setup and the long training time. Furthermore, the model does not separate lighting from material, so its output can not be used in a standard rendering pipeline, or for the creation of virtual assets.

Near-Instant Capture of High-Resolution Facial Geometry and Reflectance [175] performs multi-view color-space analysis to separate diffuse from specular reflectance. Photometric estimation of specular normals further refines geometry compared to the reconstructed base mesh. Similar to our method, and in contrast to the previously described deep learning-based methods, the output is a set of textures that can be used in a standard rendering pipeline to render photo-realistic images of a person’s face. The

carefully calibrated high-cost capture setup, consisting of 24 DSLR cameras, enables reconstruction of fine-scale detail and cannot be matched by current smartphone camera technology. Nevertheless, we see potential benefit of our method’s flexibility to capture specular highlights from arbitrary viewpoints, compared to a predetermined set of fixed viewpoints. Another drawback is the necessity of a manual cleanup of the reconstructed multi-view stereo mesh, which is avoided by our method’s automated FLAME fitting.

Several prior works [177], [203], [211] on face reconstruction and relighting use a capture setup, in which the light direction is independent from the view direction. While we see potential benefit for convergence speed from the additional constraints provided by such capture setups, given multiple views, our co-located data also provides enough constraints for successful convergence. The shadowing-masking term G is the only term that is directly linked to both the view and light vector. However, by reciprocity of the BRDF, the dependence on view and light direction is the same. Instead of having independent view and light vectors, we found it more important to have a good distribution of the angles between surface normal and view (or light) vector to recover a complete specular and normal map. This is in contrast to [177] and [203] where both camera and light are mostly front-facing.

5.8 Conclusion

We have presented a practical and inexpensive method of capturing high-resolution textures of a person’s face by coupling commodity smartphones and polarization foils. The co-location of the camera lens and light source allows us to reduce the problem complexity and separate material from shading information. As a result, we obtain high-resolution textures of the skin area of the human face. We believe that polarization is a powerful tool for material recovery in the real world, and future smartphones could benefit from including filters directly in the hardware. Overall, we believe that our work is a stepping stone towards democratizing the creation of digital human face assets by making it more accessible to smaller production studios or individual users.

Part III

Conclusion & Outlook

6 Conclusion

In this dissertation, we developed new algorithms for the reconstruction of scene geometry and materials. We focus on three main problems: joint estimation of scene materials and lighting, accurate geometry reconstruction with camera pose estimation and building a complete capture setup for geometry and appearance capture. Each of these problems was introduced in Part II and we present concluding remarks in this chapter.

Inverse Path Tracing for Joint Material and Lighting Estimation We formulate an end-to-end differentiable inverse Monte Carlo renderer which is utilized in a nested stochastic gradient descent optimization to estimate scene material parameters and lighting, given RGB observations, corresponding camera poses and geometry. We retrieve physically-based material properties, such as the diffuse reflectance, specular reflectance or roughness. This enables digital asset creation, as well as important virtual- and mixed-reality applications, like relighting, novel-view synthesis or scene manipulation. The key insight is that the path tracing algorithm can be made differentiable by analytically computing derivatives w.r.t. scene parameters at every ray bounce. These derivatives can then be employed in a stochastic gradient descent optimization, further guided by regularization terms, to estimate material parameters. We show an improvement compared to previous methods in a qualitative and quantitative analysis. We confirm convergence to correct material and lighting parameters in a series of experiments on synthetic data and show the applicability of our method on real data on the Matterport3D [2] dataset.

Neural RGB-D Surface Reconstruction While current volume-based view synthesis methods (e.g., NeRF) show promising results in reproducing the appearance of an object or scene, they do not reconstruct an actual surface. The volumetric representation of the surface based on densities leads to artifacts when a surface is extracted using Marching Cubes. Instead of this density-based representation of the surface, we propose using a truncated signed distance function (TSDF). We show how a TSDF can be incorporated in a density-based volume rendering formulation and propose an optimization that employs both color and depth measurements, leading to a more complete surface reconstruction. We further increase the reconstruction quality by also jointly optimizing camera parameters. Compared to existing methods, like BundleFusion [6], our reconstruction is more complete and misalignment artifacts are significantly reduced. We confirm this with both qualitative and quantitative comparisons, in a series of experiments on synthetic and real data. A thorough ablation study measures the impact of individual method design choices.

High-Res Facial Appearance Capture from Polarized Smartphone Images A discussion on geometry and material capture would not be complete without mention of the data capture process. A carefully calibrated high-cost capture setup, e.g., a light stage, is typically employed to obtain a high-quality reconstruction. To allow individual users to obtain high-quality reconstructions, we decided to explore an alternative, comprising only a smartphone and inexpensive polarization foils. Using an analysis-by-synthesis approach on cross-polarized and parallel-polarized data, captured in a dark room under flashlight illumination, we were able to separate diffuse from specular reflectance and reconstruct high-resolution facial textures. The biggest challenge was finding a solution to numerous challenges that arise when working with real-world data, such as flashlight attenuation or sensor noise. Nevertheless, we present an end-to-end framework to capture the appearance of a human face and provide a method to retrieve textures which can later be modified and used in standard rendering pipelines for novel-view synthesis and relighting. Compared to existing low-cost setups, our method reconstructs more detailed textures and enables more realistic image synthesis, as shown in several experiments.

7 Limitations and Future Work

7.1 Inverse Path Tracing for Joint Material and Lighting Estimation

We devised an end-to-end framework for a physically-based material and lighting parameter estimation. This is however limited to opaque materials. Reconstruction of more complex materials is an avenue of future research. Our method assumes an accurate geometry reconstruction. In case of incomplete geometry, the material and lighting reconstruction quality may also suffer, e.g. the color of one object may be “projected” onto a background object if part of the geometry is missing. Joint reconstruction of both shape and material is an interesting future direction. We also analytically formulate all derivatives required in the optimization. [11]–[13] show that a more general approach is also possible and a direction worth exploring. Finally, progress towards real-time optimization would enable further applications, such as reconstruction of dynamic scenes.

7.2 Neural RGB-D Surface Reconstruction

Our reformulation of neural radiance fields [8] allowed accurate reconstruction of geometry. However, there exist multiple improvement opportunities. The main drawback of our current implementation is the long optimization time. Several methods have been proposed to reduce the optimization time of neural radiance fields. Recent methods that utilize voxel grids, e.g. [154] or [155], have shown significantly faster convergence compared to earlier MLP-based methods and we believe it would be worth exploring their effectiveness in the context of RGB-D surface reconstruction.

The global MLP that stores the surface information comes at the cost of missing high-frequency local geometry detail in large scenes, such as scans of an entire apartment. Approaches like [90] or [89] benefit from locally-conditioned MLPs and could be integrated in future work to improve local detail. The change in the volume rendering formulation unfortunately came at the cost of image synthesis quality. It would, therefore, be worthwhile to devise a method that can accurately reconstruct geometry while conserving the image synthesis quality of neural radiance fields. Lastly, our method is limited to opaque surfaces, and an extension that can handle transparent surfaces would be an interesting research direction.

7.3 High-Res Facial Appearance Capture from Polarized Smartphone Images

We propose a method for high-resolution facial texture reconstruction. The proposed approach is limited to static facial expressions, since the long capture time does not allow reconstruction of dynamic faces, e.g., a talking head. It is, however, possible to trade off detail for a faster capture time. Using only video frames for reconstruction, we could reduce the capture time to roughly 20 seconds. This would, however, still be prohibitively long for dynamic reconstructions. An avenue for future research is the reconstruction of dynamic expressions by fitting a parametric model with consistent mesh topology to each frame, and optimizing over the entire non-rigid sequence. Furthermore, our method does not handle global illumination. Using a differentiable path tracer, it may be possible to obtain a more accurate reconstruction, especially in the eye region and other concave areas of the face. Our proposed method is also limited to just the skin area of the face. Reconstruction of hair and eyes is a subject of future work. Finally, given that we propose a low-cost setup targeted at individual users, it would be interesting to explore alternatives that don't require polarization or that work in a less constrained environment.

Bibliography

- [1] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, “Scannet: Richly-annotated 3d reconstructions of indoor scenes,” in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.
- [2] A. Chang, A. Dai, T. Funkhouser, *et al.*, “Matterport3D: Learning from RGB-D data in indoor environments,” *International Conference on 3D Vision (3DV)*, 2017.
- [3] 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.
- [4] S. Izadi, D. Kim, O. Hilliges, *et al.*, “Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera,” in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, ACM, 2011, pp. 559–568.
- [5] R. A. Newcombe, S. Izadi, O. Hilliges, *et al.*, “Kinectfusion: Real-time dense surface mapping and tracking,” in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, IEEE, 2011, pp. 127–136.
- [6] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt, “Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration,” *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, 76a, 2017.
- [7] B. Curless and M. Levoy, “A volumetric method for building complex models from range images,” in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’96, New York, NY, USA: Association for Computing Machinery, 1996, 303–312, ISBN: 0897917464. DOI: 10.1145/237170.237269. [Online]. Available: <https://doi.org/10.1145/237170.237269>.
- [8] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” in *ECCV*, 2020.
- [9] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychological Review*, no. 65(6), 386–408, 1958. [Online]. Available: <https://doi.org/10.1037/h0042519>.

- [10] J. T. Kajiya and B. P. Von Herzen, “Ray tracing volume densities,” *SIGGRAPH Comput. Graph.*, vol. 18, no. 3, 165–174, 1984, ISSN: 0097-8930. DOI: 10.1145/964965.808594. [Online]. Available: <https://doi.org/10.1145/964965.808594>.
- [11] T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen, “Differentiable monte carlo ray tracing through edge sampling,” *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 37, no. 6, 222:1–222:11, 2018.
- [12] M. Nimier-David, D. Vicini, T. Zeltner, and W. Jakob, “Mitsuba 2: A retargetable forward and inverse renderer,” *ACM Transactions on Graphics (TOG)*, vol. 38, no. 6, pp. 1–17, 2019.
- [13] C. Zhang, Z. Yu, and S. Zhao, “Path-space differentiable rendering of participating media,” *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–15, 2021.
- [14] D. Azinović, T.-M. Li, A. Kaplanyan, and M. Niessner, “Inverse path tracing for joint material and lighting estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2447–2456.
- [15] D. Azinović, R. Martin-Brualla, D. B. Goldman, M. Nießner, and J. Thies, “Neural rgb-d surface reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 6290–6301.
- [16] D. Azinović, O. Maury, C. Hery, M. Nießner, and J. Thies, “High-res facial appearance capture from polarized smartphone images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 16 836–16 846.
- [17] M. Kazhdan and H. Hoppe, “Screened poisson surface reconstruction,” *ACM Trans. Graph.*, vol. 32, no. 3, Jul. 2013, ISSN: 0730-0301. DOI: 10.1145/2487228.2487237. [Online]. Available: <https://doi.org/10.1145/2487228.2487237>.
- [18] Y. Ben-Shabat, C. Hewa Koneputugodage, and S. Gould, “Digs: Divergence guided shape implicit neural representation for unoriented point clouds,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19 323–19 332.
- [19] J. Hart, “Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces,” *The Visual Computer*, vol. 12, Jun. 1995. DOI: 10.1007/s003710050084.
- [20] W. E. Lorensen and H. E. Cline, “Marching cubes: A high resolution 3d surface construction algorithm,” in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH ’87, New York, NY, USA: Association for Computing Machinery, 1987, 163–169, ISBN: 0897912276. DOI: 10.1145/37401.37422. [Online]. Available: <https://doi.org/10.1145/37401.37422>.

- [21] B. O. Community, *Blender - a 3d modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: <http://www.blender.org>.
- [22] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, “Real-time 3d reconstruction at scale using voxel hashing,” *ACM Transactions on Graphics (TOG)*, 2013.
- [23] R. A. Newcombe, D. Fox, and S. M. Seitz, “Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
- [24] J. L. Schönberger, T. Price, T. Sattler, J.-M. Frahm, and M. Pollefeys, “A vote-and-verify strategy for fast spatial verification in image retrieval,” in *Asian Conference on Computer Vision (ACCV)*, 2016.
- [25] J. L. Schonberger and J.-M. Frahm, “Structure-from-motion revisited,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [26] Agisoft, *Agisoft metashape professional (version 1.8.4)*, Agisoft, 2022. [Online]. Available: <http://www.agisoft.com>.
- [27] E. Games, *Realitycapture*, Epic Games, Inc., 2023. [Online]. Available: <https://www.capturingreality.com/>.
- [28] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, “Tanks and temples: Benchmarking large-scale scene reconstruction,” *ACM Trans. Graph.*, vol. 36, no. 4, Jul. 2017, ISSN: 0730-0301. DOI: 10.1145/3072959.3073599. [Online]. Available: <https://doi.org/10.1145/3072959.3073599>.
- [29] D. S. Immel, M. F. Cohen, and D. P. Greenberg, “A radiosity method for non-diffuse environments,” ser. SIGGRAPH ’86, New York, NY, USA: Association for Computing Machinery, 1986, 133–142, ISBN: 0897911962. DOI: 10.1145/15922.15901. [Online]. Available: <https://doi.org/10.1145/15922.15901>.
- [30] J. T. Kajiya, “The rendering equation,” *SIGGRAPH Comput. Graph.*, vol. 20, no. 4, pp. 143–150, Aug. 1986, ISSN: 0097-8930. DOI: 10.1145/15886.15902. [Online]. Available: <http://doi.acm.org/10.1145/15886.15902>.
- [31] K. Group, *OpenGL rendering pipeline overview*, Khronos Group. [Online]. Available: https://www.khronos.org/opengl/wiki/Rendering_Pipeline_Overview.
- [32] S. Laine, J. Hellsten, T. Karras, Y. Seol, J. Lehtinen, and T. Aila, “Modular primitives for high-performance differentiable rendering,” *ACM Transactions on Graphics*, vol. 39, no. 6, 2020.
- [33] T. L. Kay and J. T. Kajiya, “Ray tracing complex scenes,” *SIGGRAPH Comput. Graph.*, vol. 20, no. 4, 269–278, 1986, ISSN: 0097-8930. DOI: 10.1145/15886.15916. [Online]. Available: <https://doi.org/10.1145/15886.15916>.
- [34] B. Burley and W. D. A. Studios, “Physically-based shading at disney.”

- [35] B. Burley, “Extending the disney brdf to a bsdf with integrated subsurface scattering,” *Physically Based Shading in Theory and Practice ‘SIGGRAPH Course*, 2015.
- [36] R. L. Cook and K. E. Torrance, “A reflectance model for computer graphics,” *ACM Trans. Graph.*, vol. 1, no. 1, 7–24, 1982, ISSN: 0730-0301. DOI: 10.1145/357290.357293. [Online]. Available: <https://doi.org/10.1145/357290.357293>.
- [37] B. Walter, S. R. Marschner, H. Li, and K. E. Torrance, “Microfacet models for refraction through rough surfaces,” in *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, ser. EGSR’07, Grenoble, France: Eurographics Association, 2007, 195–206, ISBN: 9783905673524.
- [38] C. Schlick, “An inexpensive brdf model for physically-based rendering,” *Computer Graphics Forum*, vol. 13, no. 3, pp. 233–246, 1994, ISSN: 1467-8659. DOI: 10.1111/1467-8659.1330233.
- [39] D. Vicini, S. Speierer, and W. Jakob, “Path replay backpropagation: Differentiating light paths using constant memory and linear time,” *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–14, 2021.
- [40] J. Munkberg, J. Hasselgren, T. Shen, *et al.*, “Extracting triangular 3d models, materials, and lighting from images,” *arXiv:2111.12503*, 2021.
- [41] J. Thies, M. Zollhöfer, and M. Nießner, “Deferred neural rendering: Image synthesis using neural textures,” *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.
- [42] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt, “Neural sparse voxel fields,” *NeurIPS*, 2020.
- [43] C. Reiser, S. Peng, Y. Liao, and A. Geiger, “Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps,” in *International Conference on Computer Vision (ICCV)*, 2021.
- [44] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, “PlenOctrees for real-time rendering of neural radiance fields,” in *ICCV*, 2021.
- [45] T. Müller, A. Evans, C. Schied, and A. Keller, “Instant neural graphics primitives with a multiresolution hash encoding,” *ACM Trans. Graph.*, vol. 41, no. 4, 102:1–102:15, Jul. 2022. DOI: 10.1145/3528223.3530127. [Online]. Available: <https://doi.org/10.1145/3528223.3530127>.
- [46] 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,” *CVPR*, 2022.
- [47] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, “Real-time 3d reconstruction at scale using voxel hashing,” *ACM Transactions on Graphics (ToG)*, vol. 32, no. 6, p. 169, 2013.

- [48] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, “Elasticfusion: Real-time dense slam and light source estimation,” *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.
- [49] J. T. Barron and J. Malik, “Shape, illumination, and reflectance from shading,” *Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 8, pp. 1670–1687, 2015.
- [50] A. Meka, M. Zollhoefer, C. Richardt, and C. Theobalt, “Live intrinsic video,” *ACM Transactions on Graphics (Proceedings SIGGRAPH)*, vol. 35, no. 4, 2016.
- [51] A. Meka, M. Maximov, M. Zollhoefer, *et al.*, “Lime: Live intrinsic material estimation,” in *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, 2018. [Online]. Available: <http://gvv.mpi-inf.mpg.de/projects/LIME/>.
- [52] C. Wu, M. Zollhöfer, M. Nießner, M. Stamminger, S. Izadi, and C. Theobalt, “Real-time shading-based refinement for consumer depth cameras,” *ACM Transactions on Graphics (TOG)*, vol. 33, no. 6, 2014.
- [53] M. Zollhöfer, A. Dai, M. Innmann, *et al.*, “Shading-based refinement on volumetric signed distance functions,” *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, 2015.
- [54] E. Zhang, M. F. Cohen, and B. Curless, “Emptying, refurbishing, and relighting indoor spaces,” *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, vol. 35, no. 6, 2016.
- [55] R. Ramamoorthi and P. Hanrahan, “A signal-processing framework for inverse rendering,” ser. SIGGRAPH, 2001, pp. 117–128.
- [56] 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 *International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017.
- [57] G. Patow and X. Pueyo, “A survey of inverse rendering problems,” *Computer Graphics Forum*, vol. 22, no. 4, pp. 663–687, 2003.
- [58] N. Bonneel, B. Kovacs, S. Paris, and K. Bala, “Intrinsic decompositions for image editing,” *Computer Graphics Forum (Eurographics State of the Art Reports)*, vol. 36, no. 2, 2017.
- [59] Y. Dong, G. Chen, P. Peers, J. Zhang, and X. Tong, “Appearance-from-motion: Recovering spatially varying surface reflectance under unknown lighting,” *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 33, no. 6, 193:1–193:12, 2014.
- [60] K. Kim, J. Gu, S. Tyree, P. Molchanov, M. Niessner, and J. Kautz, “A lightweight approach for on-the-fly reflectance estimation,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2017.
- [61] H. Barrow, J Tenenbaum, A Hanson, and E Riseman, “Recovering intrinsic scene characteristics,” *Comput. Vis. Syst*, vol. 2, pp. 3–26, 1978.

- [62] V. Deschaintre, M. Aittala, F. Durand, G. Drettakis, and A. Bousseau, “Single-image SVBRDF capture with a rendering-aware deep network,” *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 37, no. 4, 128:1–128:15, 2018.
- [63] S. R. Marschner, “Inverse rendering for computer graphics,” Ph.D. dissertation, 1998.
- [64] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar, “Acquiring the reflectance field of a human face,” ser. SIGGRAPH, 2000, pp. 145–156.
- [65] K. Kang, Z. Chen, J. Wang, K. Zhou, and H. Wu, “Efficient reflectance capture using an autoencoder,” *ACM Trans. Graph. (Proc. SIGGRAPH)*, vol. 37, no. 4, 127:1–127:10, 2018.
- [66] Y. Yu, P. Debevec, J. Malik, and T. Hawkins, “Inverse global illumination: Recovering reflectance models of real scenes from photographs,” in *SIGGRAPH*, 1999, pp. 215–224.
- [67] V. Blanz and T. Vetter, “A morphable model for the synthesis of 3d faces,” in *SIGGRAPH*, 1999, pp. 187–194.
- [68] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, “Face2face: Real-time face capture and reenactment of rgb videos,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2387–2395.
- [69] I. Gkioulekas, S. Zhao, K. Bala, T. Zickler, and A. Levin, “Inverse volume rendering with material dictionaries,” *ACM Trans. Graph.*, vol. 32, no. 6, 162:1–162:13, 2013.
- [70] I. Gkioulekas, A. Levin, and T. Zickler, “An evaluation of computational imaging techniques for heterogeneous inverse scattering,” in *European Conference on Computer Vision*, 2016, pp. 685–701.
- [71] C. Che, F. Luan, S. Zhao, K. Bala, and I. Gkioulekas, “Inverse transport networks,” *arXiv preprint arXiv:1809.10820*, 2018.
- [72] M. Kasper, N. Keivan, G. Sibley, and C. R. Heckman, “Light source estimation with analytical path-tracing,” *CoRR*, vol. abs/1701.04101, 2017. [Online]. Available: <http://arxiv.org/abs/1701.04101>.
- [73] M. M. Loper and M. J. Black, “OpenDR: An approximate differentiable renderer,” in *European Conference on Computer Vision*, vol. 8695, 2014, pp. 154–169.
- [74] H. Kato, Y. Ushiku, and T. Harada, “Neural 3D mesh renderer,” in *Computer Vision and Pattern Recognition*, 2018, pp. 3907–3916.
- [75] E. Veach, “Robust monte carlo methods for light transport simulation,” AAI9837162, Ph.D. dissertation, Stanford, CA, USA, 1998, ISBN: 0-591-90780-1.
- [76] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *International Conference on Learning Representations*, 2015.

- [77] A. Handa, T. Whelan, J. McDonald, and A. Davison, “A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM,” in *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China, 2014.
- [78] I. Wald, S. Woop, C. Benthin, G. S. Johnson, and M. Ernst, “Embree: A kernel framework for efficient CPU ray tracing,” *ACM Trans. Graph. (Proc. SIG-GRAPH)*, vol. 33, no. 4, p. 143, 2014.
- [79] A. Tewari, O. Fried, J. Thies, *et al.*, “State of the art on neural rendering,” in *Computer Graphics Forum*, Wiley Online Library, vol. 39, 2020, pp. 701–727.
- [80] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh, “Neural volumes,” *ACM Transactions on Graphics*, vol. 38, no. 4, 1–14, 2019, ISSN: 1557-7368. DOI: 10.1145/3306346.3323020. [Online]. Available: <http://dx.doi.org/10.1145/3306346.3323020>.
- [81] D. G. Lowe, “Object recognition from local scale-invariant features,” in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, 1150–1157 vol.2. DOI: 10.1109/ICCV.1999.790410.
- [82] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy networks: Learning 3d reconstruction in function space,” in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [83] V. Sitzmann, J. N. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein, “Implicit neural representations with periodic activation functions,” in *arXiv*, 2020.
- [84] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan, “Depth-supervised nerf: Fewer views and faster training for free,” *arXiv preprint arXiv:2107.02791*, 2021.
- [85] T. Neff, P. Stadlbauer, M. Parger, *et al.*, “DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks,” *Computer Graphics Forum*, vol. 40, no. 4, pp. 45–59, 2021, ISSN: 14678659. DOI: 10.1111/cgf.14340. arXiv: 2103.03231.
- [86] Y. Wei, S. Liu, Y. Rao, W. Zhao, J. Lu, and J. Zhou, “Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo,” in *ICCV*, 2021.
- [87] 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,” *arXiv preprint arXiv:2106.10689*, 2021.
- [88] M. Zollhöfer, P. Stotko, A. Görlitz, *et al.*, “State of the Art on 3D Reconstruction with RGB-D Cameras,” *Computer Graphics Forum (Eurographics State of the Art Reports 2018)*, vol. 37, no. 2, 2018.
- [89] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, “Convolutional occupancy networks,” in *European Conference on Computer Vision (ECCV)*, Cham: Springer International Publishing, Aug. 2020.

- [90] J. Chibane, T. Alldieck, and G. Pons-Moll, “Implicit functions in feature space for 3d shape reconstruction and completion,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2020.
- [91] A. Dai, Y. Siddiqui, J. Thies, J. Valentin, and M. Nießner, “Spsg: Self-supervised photometric scene generation from rgb-d scans,” in *Proc. Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2021.
- [92] S. Weder, J. L. Schönberger, M. Pollefeys, and M. R. Oswald, “Routedfusion: Learning real-time depth map fusion,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [93] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, “Scene representation networks: Continuous 3d-structure-aware neural scene representations,” in *Advances in Neural Information Processing Systems*, 2019.
- [94] A. Tewari, J. Thies, B. Mildenhall, *et al.*, *Advances in neural rendering*, 2021. arXiv: 2111.05849 [cs.GR].
- [95] D. Scharstein, R. Szeliski, and R. Zabih, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” in *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, 2001, pp. 131–140. DOI: 10.1109/SMBV.2001.988771.
- [96] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz, “Multi-view stereo for community photo collections,” in *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8. DOI: 10.1109/ICCV.2007.4408933.
- [97] J. Engel, J. Sturm, and D. Cremers, “Semi-dense visual odometry for a monocular camera,” in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 1449–1456. DOI: 10.1109/ICCV.2013.183.
- [98] J. Engel, T. Schöps, and D. Cremers, “LSD-SLAM: Large-scale direct monocular SLAM,” in *European Conference on Computer Vision (ECCV)*, 2014.
- [99] C. Forster, M. Pizzoli, and D. Scaramuzza, “Svo: Fast semi-direct monocular visual odometry,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 15–22. DOI: 10.1109/ICRA.2014.6906584.
- [100] Y. Furukawa and J. Ponce, “Accurate, dense, and robust multiview stereopsis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010. DOI: 10.1109/TPAMI.2009.161.
- [101] K. N. Kutulakos and S. M. Seitz, “A theory of shape by space carving,” *International journal of computer vision*, vol. 38, no. 3, pp. 199–218, 2000.
- [102] V. H. Hiep, R. Keriven, P. Labatut, and J.-P. Pons, “Towards high-resolution large-scale multi-view stereo,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, 2009, pp. 1430–1437.
- [103] C. Zach, T. Pock, and H. Bischof, “A globally optimal algorithm for robust tv- L^1 range image integration,” in *2007 IEEE 11th International Conference on Computer Vision*, 2007, pp. 1–8. DOI: 10.1109/ICCV.2007.4408983.

- [104] N. Savinov, C. Häne, L. Ladický, and M. Pollefeys, “Semantic 3d reconstruction with continuous regularization and ray potentials using a visibility consistency constraint,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5460–5469. DOI: 10.1109/CVPR.2016.589.
- [105] W. Dong, Q. Wang, X. Wang, and H. Zha, “PSDF fusion: Probabilistic signed distance function for on-the-fly 3d data fusion and scene reconstruction,” *CoRR*, vol. abs/1807.11034, 2018. arXiv: 1807.11034. [Online]. Available: <http://arxiv.org/abs/1807.11034>.
- [106] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, “Deeper depth prediction with fully convolutional residual networks,” in *2016 Fourth international conference on 3D vision (3DV)*, IEEE, 2016, pp. 239–248.
- [107] C. Godard, O. Mac Aodha, and G. J. Brostow, “Unsupervised monocular depth estimation with left-right consistency,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 270–279.
- [108] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, “Deep ordinal regression network for monocular depth estimation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2002–2011.
- [109] A. Kar, C. Häne, and J. Malik, “Learning a multi-view stereo machine,” *arXiv preprint arXiv:1708.05375*, 2017.
- [110] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, “Mvsnet: Depth inference for unstructured multi-view stereo,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 767–783.
- [111] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhofer, “Deepvoxels: Learning persistent 3d feature embeddings,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2437–2446.
- [112] J. Flynn, I. Neulander, J. Philbin, and N. Snavely, “Deepstereo: Learning to predict new views from the world’s imagery,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5515–5524.
- [113] A. Dai, C. Diller, and M. Nießner, “Sg-nn: Sparse generative neural networks for self-supervised scene completion of rgb-d scans,” in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2020.
- [114] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, “Codeslam — learning a compact, optimisable representation for dense visual slam,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [115] S. Zhi, M. Bloesch, S. Leutenegger, and A. J. Davison, “Scenecode: Monocular dense semantic reconstruction using learned encoded scene representations,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

- [116] J Czarnowski, T Laidlow, R Clark, and A. Davison, “Deepfactors: Real-time probabilistic dense monocular slam,” *IEEE Robotics and Automation Letters*, vol. 5, pp. 721–728, 2020. DOI: 10.1109/lra.2020.2965415. [Online]. Available: <http://dx.doi.org/10.1109/lra.2020.2965415>.
- [117] S. Weder, J. L. Schonberger, M. Pollefeys, and M. R. Oswald, “Neuralfusion: Online depth fusion in latent space,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 3162–3172.
- [118] J. Sun, Y. Xie, L. Chen, X. Zhou, and H. Bao, “NeuralRecon: Real-time coherent 3D reconstruction from monocular video,” *CVPR*, 2021.
- [119] A. Božič, P. Palafox, J. Thies, A. Dai, and M. Nießner, “Transformerfusion: Monocular rgb scene reconstruction using transformers,” *Proc. Neural Information Processing Systems (NeurIPS)*, 2021.
- [120] Y. Zhang and T. Funkhouser, “Deep depth completion of a single rgb-d image,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 175–185.
- [121] G. Gkioxari, J. Malik, and J. Johnson, “Mesh R-CNN,” *CoRR*, vol. abs/1906.02739, 2019. arXiv: 1906.02739. [Online]. Available: <http://arxiv.org/abs/1906.02739>.
- [122] Y. Nie, X. Han, S. Guo, Y. Zheng, J. Chang, and J. J. Zhang, “Total3DUnderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image,” in *CVPR*, 2020, pp. 55–64.
- [123] M. Dahnert, J. Hou, M. Nießner, and A. Dai, “Panoptic 3D scene reconstruction from a single RGB image,” *NeurIPS*, 2021.
- [124] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, “Pixel2mesh: Generating 3d mesh models from single rgb images,” in *ECCV*, 2018.
- [125] M. Denninger and R. Triebel, “3d scene reconstruction from a single viewport,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
- [126] S. Saito, Z. Huang, *et al.*, “Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization,” *arXiv preprint arXiv:1905.05172*, 2019.
- [127] S. Saito, T. Simon, J. Saragih, and H. Joo, “Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.
- [128] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, “Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [129] M. Oechsle, S. Peng, and A. Geiger, “Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction,” in *International Conference on Computer Vision (ICCV)*, 2021.

- [130] M. Oechsle, L. Mescheder, M. Niemeyer, T. Strauss, and A. Geiger, “Texture fields: Learning texture representations in function space,” in *International Conference on Computer Vision*, Oct. 2019.
- [131] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “Deepsdf: Learning continuous signed distance functions for shape representation,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.
- [132] L. Yariv, Y. Kasten, D. Moran, *et al.*, “Multiview neural surface reconstruction by disentangling geometry and appearance,” *Advances in Neural Information Processing Systems*, vol. 33, 2020.
- [133] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, “Volume rendering of neural implicit surfaces,” *Advances in Neural Information Processing Systems*, vol. 34, 2021.
- [134] N. Max, “Optical models for direct volume rendering,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 1, no. 2, pp. 99–108, 1995. DOI: 10.1109/2945.468400.
- [135] M. Tancik, P. P. Srinivasan, B. Mildenhall, *et al.*, “Fourier features let networks learn high frequency functions in low dimensional domains,” *NeurIPS*, 2020.
- [136] 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 *CVPR*, 2021.
- [137] Z. Li, S. Niklaus, N. Snavely, and O. Wang, “Neural scene flow fields for space-time view synthesis of dynamic scenes,” *arXiv preprint arXiv:2011.13084*, 2020.
- [138] K. Park, U. Sinha, J. T. Barron, *et al.*, “Deformable neural radiance fields,” *arXiv preprint arXiv:2011.12948*, 2020.
- [139] G. Gafni, J. Thies, M. Zollhöfer, and M. Nießner, “Dynamic neural radiance fields for monocular 4d facial avatar reconstruction,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.
- [140] E. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein, “Pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis,” in *arXiv*, 2020.
- [141] K. Schwarz, Y. Liao, M. Niemeyer, and A. Geiger, “Graf: Generative radiance fields for 3d-aware image synthesis,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [142] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, “Pixelnerf: Neural radiance fields from one or few images,” in *CVPR*, 2021.
- [143] Q. Wang, Z. Wang, K. Genova, *et al.*, “Ibrnet: Learning multi-view image-based rendering,” *CVPR*, 2021.
- [144] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin, “Inerf: Inverting neural radiance fields for pose estimation,” 2020.

- [145] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu, *Nerf-: Neural radiance fields without known camera parameters*, 2021. arXiv: 2102.07064 [cs.CV].
- [146] 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.
- [147] Q.-Y. Zhou and V. Koltun, “Color map optimization for 3d reconstruction with consumer depth cameras,” vol. 33, no. 4, Jul. 2014, ISSN: 0730-0301. DOI: 10.1145/2601097.2601134. [Online]. Available: <https://doi.org/10.1145/2601097.2601134>.
- [148] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014. arXiv: 1412.6980. [Online]. Available: <http://arxiv.org/abs/1412.6980>.
- [149] M. Denninger, M. Sundermeyer, D. Winkelbauer, *et al.*, “Blenderproc,” *arXiv preprint arXiv:1911.01911*, 2019.
- [150] A. Handa, *Simulating kinect noise for the icl-nuim dataset*, <https://github.com/ankurhanda/simkinect>, Accessed: 2021-11-15.
- [151] J. T. Barron and J. Malik, “Intrinsic scene properties from a single rgb-d image,” *CVPR*, 2013.
- [152] J. Bohg, J. Romero, A. Herzog, and S. Schaal, “Robot arm pose estimation through pixel-wise part classification,” *ICRA*, 2014.
- [153] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, “A benchmark for rgb-d visual odometry, 3d reconstruction and slam,” *ICRA*, 2014.
- [154] C. Sun, M. Sun, and H.-T. Chen, “Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction,” *arXiv preprint arXiv:2111.11215*, 2021.
- [155] Sara Fridovich-Keil and Alex Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, “Plenoxels: Radiance fields without neural networks,” 2022.
- [156] C. Yuksel, “A class of c2 interpolating splines,” *ACM Trans. Graph.*, vol. 39, no. 5, 2020, ISSN: 0730-0301. DOI: 10.1145/3400301. [Online]. Available: <https://doi.org/10.1145/3400301>.
- [157] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar, “Acquiring the reflectance field of a human face,” in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 145–156.
- [158] M. Wrenninge, R. Villemin, and C. Hery, “Path traced subsurface scattering using anisotropic phase functions and non-exponential free flights,” Tech. Rep. 17-07, Pixar. <https://graphics.pixar.com/library...>, Tech. Rep., 2017.
- [159] M. J.-Y. Chiang, P. Kutz, and B. Burley, “Practical and controllable subsurface scattering for production path tracing,” in *ACM SIGGRAPH 2016 Talks*, 2016, pp. 1–2.

- [160] O. Klehm, F. Rousselle, M. Papas, *et al.*, “Recent advances in facial appearance capture,” in *Computer Graphics Forum*, Wiley Online Library, vol. 34, 2015, pp. 709–733.
- [161] V. Müller, “Polarization-based separation of diffuse and specular surface-reflection,” in *Mustererkennung 1995*, Springer, 1995, pp. 202–209.
- [162] L. B. Wolff and T. E. Boult, “Constraining object features using a polarization reflectance model,” *Phys. Based Vis. Princ. Pract. Radiom*, vol. 1, p. 167, 1993.
- [163] S. Rahmann and N. Canterakis, “Reconstruction of specular surfaces using polarization imaging,” in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, IEEE, vol. 1, 2001, pp. I–I.
- [164] J. Riviere, I. Reshetouski, L. Filipi, and A. Ghosh, “Polarization imaging reflectometry in the wild,” *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, pp. 1–14, 2017.
- [165] E. Nogue, Y. Lin, and A. Ghosh, “Polarization-imaging Surface Reflectometry using Near-field Display,” in *Eurographics Symposium on Rendering*, A. Ghosh and L.-Y. Wei, Eds., The Eurographics Association, 2022, ISBN: 978-3-03868-187-8. DOI: 10.2312/sr.20221154.
- [166] V. Deschaintre, Y. Lin, and A. Ghosh, “Deep polarization imaging for 3d shape and svbrdf acquisition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
- [167] A. Dave, Y. Zhao, and A. Veeraraghavan, “Pandora: Polarization-aided neural decomposition of radiance,” *arXiv preprint arXiv:2203.13458*, 2022.
- [168] D. Wang, P. Chandran, G. Zoss, D. Bradley, and P. Gotardo, “Morf: Morphable radiance fields for multiview neural head modeling,” in *ACM SIGGRAPH 2022 Conference Proceedings*, ser. SIGGRAPH ’22, Vancouver, BC, Canada: Association for Computing Machinery, 2022, ISBN: 9781450393379. DOI: 10.1145/3528233.3530753. [Online]. Available: <https://doi.org/10.1145/3528233.3530753>.
- [169] R. J. Woodham, “Photometric method for determining surface orientation from multiple images,” *Optical engineering*, vol. 19, no. 1, pp. 139–144, 1980.
- [170] C. A. Wilson, A. Ghosh, P. Peers, J.-Y. Chiang, J. Busch, and P. Debevec, “Temporal upsampling of performance geometry using photometric alignment,” *ACM Transactions on Graphics (TOG)*, vol. 29, no. 2, pp. 1–11, 2010.
- [171] A. Ghosh, G. Fyffe, B. Tunwattanapong, J. Busch, X. Yu, and P. Debevec, “Multiview face capture using polarized spherical gradient illumination,” *ACM Trans. Graph.*, vol. 30, no. 6, 1–10, 2011, ISSN: 0730-0301. DOI: 10.1145/2070781.2024163. [Online]. Available: <https://doi.org/10.1145/2070781.2024163>.
- [172] G. Fyffe, “Cosine lobe based relighting from gradient illumination photographs,” in *SIGGRAPH’09: Posters*, 2009, pp. 1–1.

- [173] G. Fyffe, “Single-shot photometric stereo by spectral multiplexing,” in *ACM SIGGRAPH ASIA 2010 Sketches*, 2010, pp. 1–2.
- [174] G. Fyffe and P. Debevec, “Single-shot reflectance measurement from polarized color gradient illumination,” in *2015 IEEE International Conference on Computational Photography (ICCP)*, IEEE, 2015, pp. 1–10.
- [175] G. Fyffe, P. Graham, B. Tunwattanapong, A. Ghosh, and P. Debevec, “Near-instant capture of high-resolution facial geometry and reflectance,” in *Computer Graphics Forum*, Wiley Online Library, vol. 35, 2016, pp. 353–363.
- [176] C. Kampouris, S. Zafeiriou, and A. Ghosh, “Diffuse-specular separation using binary spherical gradient illumination,” in *EGSR (EIEI)*, 2018, pp. 1–10.
- [177] A. Lattas, Y. Lin, J. Kannan, *et al.*, “Practical and scalable desktop-based high-quality facial capture,” 2022.
- [178] P. Gotardo, J. Riviere, D. Bradley, A. Ghosh, and T. Beeler, “Practical dynamic facial appearance modeling and acquisition,” *ACM Trans. Graph.*, vol. 37, no. 6, 2018, ISSN: 0730-0301. DOI: 10.1145/3272127.3275073. [Online]. Available: <https://doi.org/10.1145/3272127.3275073>.
- [179] J. Riviere, P. Gotardo, D. Bradley, A. Ghosh, and T. Beeler, “Single-shot high-quality facial geometry and skin appearance capture,” *ACM Trans. Graph.*, vol. 39, no. 4, 2020, ISSN: 0730-0301. DOI: 10.1145/3386569.3392464. [Online]. Available: <https://doi.org/10.1145/3386569.3392464>.
- [180] Y. Xu, J. Riviere, G. Zoss, P. Chandran, D. Bradley, and P. Gotardo, “Improved lighting models for facial appearance capture,” 2022.
- [181] T. Zeltner, S. Speierer, I. Georgiev, and W. Jakob, “Monte carlo estimators for differential light transport,” *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–16, 2021.
- [182] W. Jakob, S. Speierer, N. Roussel, and D. Vicini, “Dr. jit: A just-in-time compiler for differentiable rendering,” *ACM Transactions on Graphics (TOG)*, vol. 41, no. 4, pp. 1–19, 2022.
- [183] J. Munkberg, J. Hasselgren, T. Shen, *et al.*, “Extracting triangular 3d models, materials, and lighting from images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8280–8290.
- [184] H. Kato, D. Beker, M. Morariu, *et al.*, “Differentiable rendering: A survey,” *arXiv preprint arXiv:2006.12057*, 2020.
- [185] F. Luan, S. Zhao, K. Bala, and Z. Dong, “Unified shape and svbrdf recovery using differentiable monte carlo rendering,” in *Computer Graphics Forum*, Wiley Online Library, vol. 40, 2021, pp. 101–113.
- [186] K. Zhang, F. Luan, Z. Li, and N. Snavely, “Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5565–5574.

- [187] A. Dib, G. Bharaj, J. Ahn, *et al.*, “Practical face reconstruction via differentiable ray tracing,” in *Computer Graphics Forum*, Wiley Online Library, vol. 40, 2021, pp. 153–164.
- [188] A. Dib, C. Thebault, J. Ahn, P.-H. Gosselin, C. Theobalt, and L. Chevallier, “Towards high fidelity monocular face reconstruction with rich reflectance using self-supervised learning and ray tracing,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 819–12 829.
- [189] A. Dib, J. Ahn, C. Thebault, P.-H. Gosselin, and L. Chevallier, “S2f2: Self-supervised high fidelity face reconstruction from monocular image,” *arXiv preprint arXiv:2203.07732*, 2022.
- [190] Y. Wang, A. Holynski, X. Zhang, and X. C. Zhang, “Sunstage: Portrait reconstruction and relighting using the sun as a light stage,” *arXiv preprint arXiv:2204.03648*, 2022.
- [191] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero, “Learning a model of facial shape and expression from 4D scans,” *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, vol. 36, no. 6, 194:1–194:17, 2017. [Online]. Available: <https://doi.org/10.1145/3130800.3130813>.
- [192] A. Meka, C. Haene, R. Pandey, *et al.*, “Deep reflectance fields: High-quality facial reflectance field inference from color gradient illumination,” *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.
- [193] S. Saito, L. Wei, L. Hu, K. Nagano, and H. Li, “Photorealistic facial texture inference using deep neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
- [194] J. Li, Z. Kuang, Y. Zhao, M. He, K. Bladin, and H. Li, “Dynamic facial asset and rig generation from a single scan,” *ACM Trans. Graph.*, vol. 39, no. 6, 2020, ISSN: 0730-0301. DOI: 10.1145/3414685.3417817. [Online]. Available: <https://doi.org/10.1145/3414685.3417817>.
- [195] A. Lattas, S. Moschoglou, B. Gecer, *et al.*, “Avatarme: Realistically renderable 3d facial reconstruction in-the-wild,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 760–769.
- [196] L. Huynh, W. Chen, S. Saito, *et al.*, “Mesoscopic facial geometry inference using deep neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8407–8416.
- [197] S. Yamaguchi, S. Saito, K. Nagano, *et al.*, “High-fidelity facial reflectance and geometry inference from an unconstrained image,” *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.
- [198] S. Bi, S. Lombardi, S. Saito, *et al.*, “Deep relightable appearance models for animatable faces,” *ACM Trans. Graph.*, vol. 40, no. 4, 2021, ISSN: 0730-0301. DOI: 10.1145/3450626.3459829. [Online]. Available: <https://doi.org/10.1145/3450626.3459829>.

- [199] X. Zhang, S. Fanello, Y.-T. Tsai, *et al.*, “Neural light transport for relighting and view synthesis,” *ACM Trans. Graph.*, vol. 40, no. 1, 2021, ISSN: 0730-0301. DOI: 10.1145/3446328. [Online]. Available: <https://doi.org/10.1145/3446328>.
- [200] G. Gafni, J. Thies, M. Zollhöfer, and M. Nießner, “Dynamic neural radiance fields for monocular 4d facial avatar reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 8649–8658.
- [201] P.-W. Grassal, M. Prinzler, T. Leistner, C. Rother, M. Nießner, and J. Thies, “Neural head avatars from monocular rgb videos,” *arXiv preprint arXiv:2112.01554*, 2021.
- [202] Y. Zheng, V. F. Abrevaya, M. C. Bühler, X. Chen, M. J. Black, and O. Hilliges, “I M avatar: Implicit morphable head avatars from videos,” in *IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 13 545–13 555.
- [203] S. Sengupta, B. Curless, I. Kemelmacher-Shlizerman, and S. M. Seitz, “A light stage on every desk,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 2420–2429.
- [204] T. Sun, K.-E. Lin, S. Bi, Z. Xu, and R. Ramamoorthi, “Nelf: Neural light-transport field for portrait view synthesis and relighting,” in *Eurographics Symposium on Rendering*, 2021.
- [205] A. Sevastopolsky, S. Ignatiev, G. Ferrer, E. Burnaev, and V. Lempitsky, “Relightable 3d head portraits from a smartphone video,” *arXiv preprint arXiv:2012.09963*, 2020.
- [206] Y. Zheng, H. Yang, T. Zhang, *et al.*, “General facial representation learning in a visual-linguistic manner,” *arXiv preprint arXiv:2112.03109*, 2021.
- [207] M. Ashikhmin and P. Shirley, “An anisotropic phong light reflection model,” *University of Utah Computer Science Technical Report*, 2000.
- [208] M. Ashikhmin and P. Shirley, “An anisotropic phong brdf model,” *Journal of Graphics Tools* 5 (2), 25–32, 2002.
- [209] H. Jin, S. Liao, and L. Shao, “Pixel-in-pixel net: Towards efficient facial landmark detection in the wild,” *International Journal of Computer Vision*, 2021, ISSN: 1573-1405. DOI: 10.1007/s11263-021-01521-4. [Online]. Available: <http://dx.doi.org/10.1007/s11263-021-01521-4>.
- [210] O. Sorkine and M. Alexa, “As-Rigid-As-Possible Surface Modeling,” in *Geometry Processing*, A. Belyaev and M. Garland, Eds., The Eurographics Association, 2007, ISBN: 978-3-905673-46-3. DOI: 10.2312/SGP/SGP07/109-116.
- [211] N. Maček, B. Usta, E. Eisemann, and R. Marroquim, “Real-time relighting of human faces with a low-cost setup,” *Proc. ACM Comput. Graph. Interact. Tech.*, vol. 5, no. 1, 2022. DOI: 10.1145/3522626. [Online]. Available: <https://doi.org/10.1145/3522626>.

Appendix

A Open-source Code & Videos

A.1 Inverse Path Tracing for Joint Material and Lighting Estimation

- Video: https://www.youtube.com/watch?v=nC_t0t9u6ws

A.2 Neural RGB-D Surface Reconstruction

- Video: <https://www.youtube.com/watch?v=iWuSowPsC3g>
- Project: <https://dazinovic.github.io/neural-rgb-d-surface-reconstruction/>
- Source code: <https://github.com/dazinovic/neural-rgb-d-surface-reconstruction>

A.3 High-Res Facial Appearance Capture from Polarized Smartphone Images

- Video: <https://www.youtube.com/watch?v=jnb4V0qURtc>
- Project: <https://dazinovic.github.io/polface/>

B Authored and Co-authored Publications

1. D. Azinović, T.-M. Li, A. Kaplanyan, and M. Niessner, “Inverse path tracing for joint material and lighting estimation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 2447–2456
2. D. Azinović, R. Martin-Brualla, D. B. Goldman, M. Nießner, and J. Thies, “Neural rgb-d surface reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 6290–6301
3. D. Azinović, O. Maury, C. Hery, M. Nießner, and J. Thies, “High-res facial appearance capture from polarized smartphone images,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 16 836–16 846

C Original Publications

C.1 Inverse Path Tracing for Joint Material and Lighting Estimation

Copyright Notice

©2019 IEEE. Reprinted, with permission, from

Dejan Azinović, Tzu-Mao Li, Anton Kaplanyan and Matthias Nießner

Inverse Path Tracing for Joint Material and Lighting Estimation

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019

DOI: 10.1109/CVPR.2019.00255

In accordance with the IEEE Thesis/Dissertation Reuse Permissions, we include the accepted version of the original publication [14] in the following.

Inverse Path Tracing for Joint Material and Lighting Estimation

Dejan Azinović¹Tzu-Mao Li^{2,3}Anton Kaplanyan³Matthias Nießner¹

¹Technical University of Munich ²MIT CSAIL ³Facebook Reality Labs

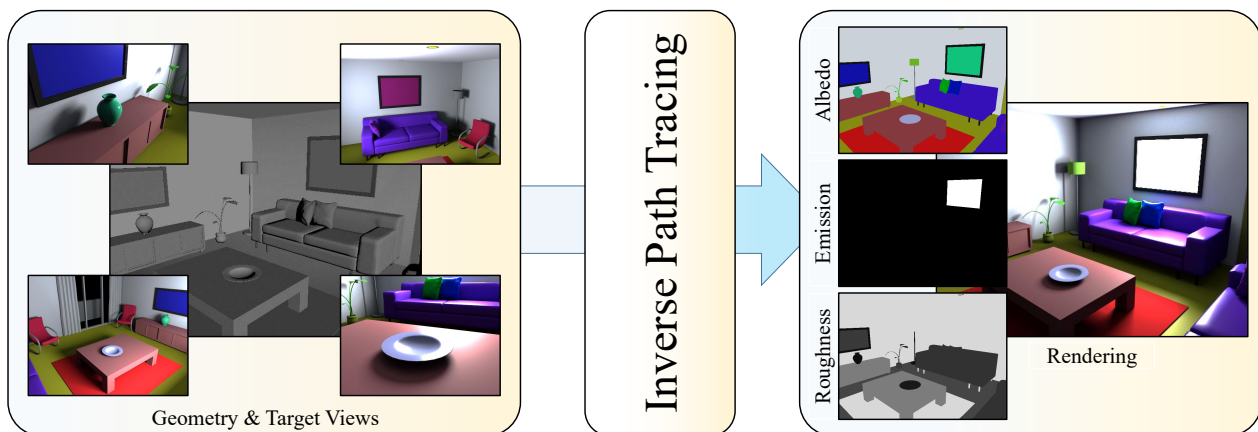


Figure 1: Our Inverse Path Tracing algorithm takes as input a 3D scene and up to several RGB images (left), and estimates material as well as the lighting parameters of the scene. The main contribution of our approach is the formulation of an end-to-end differentiable inverse Monte Carlo renderer which is utilized in a nested stochastic gradient descent optimization.

Abstract

Modern computer vision algorithms have brought significant advancement to 3D geometry reconstruction. However, illumination and material reconstruction remain less studied, with current approaches assuming very simplified models for materials and illumination. We introduce *Inverse Path Tracing*, a novel approach to jointly estimate the material properties of objects and light sources in indoor scenes by using an invertible light transport simulation. We assume a coarse geometry scan, along with corresponding images and camera poses. The key contribution of this work is an accurate and simultaneous retrieval of light sources and physically based material properties (e.g., diffuse reflectance, specular reflectance, roughness, etc.) for the purpose of editing and re-rendering the scene under new conditions. To this end, we introduce a novel optimization method using a differentiable Monte Carlo renderer that computes derivatives with respect to the estimated unknown illumination and material properties. This enables joint optimization for physically correct light transport and material models using a tailored stochastic gradient descent.

1. Introduction

With the availability of inexpensive, commodity RGB-D sensors, such as the Microsoft Kinect, Google Tango, or Intel RealSense, we have seen incredible advances in 3D reconstruction techniques [27, 14, 28, 33, 8]. While tracking and reconstruction quality have reached impressive levels, the estimation of lighting and materials has often been neglected. Unfortunately, this presents a serious problem for virtual- and mixed-reality applications, where we need to re-render scenes from different viewpoints, place virtual objects, edit scenes, or enable telepresence scenarios where a person is placed in a different room.

This problem has been viewed in the 2D image domain, resulting in a large body of work on intrinsic images or videos [1, 26, 25]. However, the problem is severely underconstrained on monocular RGB data due to lack of known geometry, and thus requires heavy regularization to jointly solve for lighting, material, and scene geometry. We believe that the problem is much more tractable in the context of given 3D reconstructions. However, even with depth data available, most state-of-the-art methods, e.g., shading-based refinement [34, 37] or indoor re-lighting [36], are based on simplistic lighting models, such as spher-

ical harmonics (SH) [30] or spatially-varying SH [23], which can cause issues on occlusion and view-dependent effects (Fig. 4).

In this work, we address this shortcoming by formulating material and lighting estimation as a proper inverse rendering problem. To this end, we propose an Inverse Path Tracing algorithm that takes as input a given 3D scene along with a single or up to several captured RGB frames. The key to our approach is a differentiable Monte Carlo path tracer which can differentiate with respect to rendering parameters constrained on the difference of the rendered image and the target observation. Leveraging these derivatives, we solve for the material and lighting parameters by nesting the Monte Carlo path tracing process into a stochastic gradient descent (SGD) optimization. The main contribution of this work lies in this SGD optimization formulation, which is inspired by recent advances in deep neural networks.

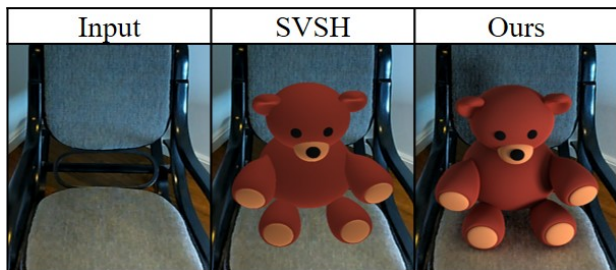


Figure 2: Inserting virtual objects in real 3D scenes; the estimated lighting and material parameters of our approach enable convincing image compositing in AR settings.

We tailor this Inverse Path Tracing algorithm to 3D scenes, where scene geometry is (mostly) given but the material and lighting parameters are unknown. In a series of experiments on both synthetic ground truth and real scan data, we evaluate the design choices of our optimizer. In comparison to state-of-the-art lighting models, we show that our inverse rendering formulation achieves significantly more accurate results.

In summary, we contribute the following:

- An end-to-end differentiable inverse path tracing formulation for joint material and lighting estimation.
- A flexible stochastic optimization framework with extensibility and flexibility for different materials and regularization terms.

2. Related Work

Material and illumination reconstruction has a long history in computer vision (e.g., [29, 4]). Given scene geometry and observed radiance of the surfaces, the task is to infer the material properties and locate the light source. However, to our knowledge, none of the existing methods handle non-Lambertian materials with near-field illumination (area

light sources), while taking interreflection between surfaces into account.

3D approaches. A common assumption in reconstructing material and illumination is that the light sources are infinitely far away. Ramamoorthi and Hanrahan [30] project both material and illumination onto spherical harmonics and solve for their coefficients using the convolution theorem. Dong *et al.* [11] solve for spatially-varying reflectance from a video of an object. Kim *et al.* [19] reconstruct the reflectance by training a convolutional neural network operating on voxels constructed from RGB-D video. Maier *et al.* [23] generalize spherical harmonics to handle spatial dependent effects, but do not correctly take view-dependent reflection and occlusion into account. All these approaches simplify the problem by assuming that the light sources are infinitely far away, in order to reconstruct a single environment map shared by all shading points. In contrast, we model the illumination as emission from the surfaces, and handle near-field effects such as the squared distance falloff or glossy reflection better.

Image-space approaches (e.g., [2, 1, 10, 25]). These methods usually employ sophisticated data-driven approaches, by learning the distributions of material and illumination. However, these methods do not have a notion of 3D geometry, and cannot handle occlusion, interreflection and geometry factors such as the squared distance falloff in a physically based manner. These methods also usually require a huge amount of training data, and are prone to errors when subjected to scenes with different characteristics from the training data.

Active illumination (e.g., [24, 9, 16]). These methods use highly-controlled lighting for reconstruction, by carefully placing the light sources and measuring the intensity. These methods produce high-quality results, at the cost of a more complicated setup.

Inverse radiosity (e.g., [35, 36]) achieves impressive results for solving near-field illumination and Lambertian materials for indoor illumination. It is difficult to generalize the radiosity algorithm to handle non-Lambertian materials (Yu *et al.* handle it by explicitly measuring the materials, whereas Zhang *et al.* assume Lambertian).

Differentiable rendering. Blanz and Vetter utilized differentiable rendering for face reconstruction using 3D morphable models [3], which is now leveraged by modern analysis-by-synthesis face trackers [31]. Gkioulekas *et al.* [13, 12] and Che *et al.* [7] solve for scattering parameters using a differentiable volumetric path tracer. Kasper *et al.* [17] developed a differentiable path tracer, but focused on distant illumination. Loper and Black [22] and Kato [18] developed fast differentiable rasterizers, but do not support global illumination. Li *et al.* [21] showed that it is possible to compute correct gradients of a path tracer while taking discontinuities introduced by visibility into consideration.

3. Method

Our *Inverse Path Tracing* method employs physically based light transport simulation [15] to estimate derivatives of all unknown parameters w.r.t. the rendered image(s). The rendering problem is generally extremely high-dimensional and is therefore usually solved using stochastic integration methods, such as Monte Carlo integration. In this work, we nest differentiable path tracing into stochastic gradient descent to solve for the unknown scene parameters. Fig. 3 illustrates the workflow of our approach. We start from the captured imagery, scene geometry, object segmentation of the scene, and an arbitrary initial guess of the illumination and material parameters. Material and emission properties are then estimated by optimizing for rendered imagery to match the captured images.

The path tracer renders a noisy and undersampled version of the image using Monte Carlo integration and computes derivatives of each sampled light path w.r.t. the unknowns. These derivatives are passed as input to our optimizer to perform a single optimization step. This process is performed iteratively until we arrive at the correct solution. Path tracing is a computationally expensive operation, and this optimization problem is non-convex and ill-posed. To this end, we employ variance reduction and novel regularization techniques (Sec. 4.4) for our gradient computation to arrive at a converged solution within a reasonable amount of time, usually a few minutes on a modern 8-core CPU.

3.1. Light Transport Simulation

If all scene and image parameters are known, an expected linear pixel intensity can be computed using light transport simulation. In this work, we assume that all surfaces are opaque and there is no participating media (*e.g.*, fog) in the scene. In this case, the rendered intensity I_R^j for pixel j is computed using the path integral [32]:

$$I_R^j = \int_{\Omega} h_j(\mathbf{X}) f(\mathbf{X}) d\mu(\mathbf{X}), \quad (1)$$

where $\mathbf{X} = (\mathbf{x}_0, \dots, \mathbf{x}_k)$ is a light path, *i.e.* a list of vertices on the surfaces of the scene starting at the light source and ending at the sensor; the integral is a path integral taken over the space of all possible light paths of all lengths, denoted as Ω , with a product area measure $\mu(\cdot)$; $f(\mathbf{X})$ is the measurement contribution function of a light path \mathbf{X} that computes how much energy flows through this particular path; and $h_j(\mathbf{X})$ is the pixel filter kernel of the sensor’s pixel j , which is non-zero only when the light path \mathbf{X} ends around the pixel j and incorporates sensor sensitivity at this pixel. We refer interested readers to the work of Veach [32] for more details on the light transport path integration.

The most important term of the integrand to our task is the path measurement contribution function f , as it contains

the material parameters as well as the information about the light sources. For a path $\mathbf{X} = (\mathbf{x}_0, \dots, \mathbf{x}_k)$ of length k , the measurement contribution function has the following form:

$$f(\mathbf{X}) = L_e(\mathbf{x}_0, \overline{\mathbf{x}_0\mathbf{x}_1}) \prod_{i=1}^k f_r(\mathbf{x}_i, \overline{\mathbf{x}_{i-1}\mathbf{x}_i}, \overline{\mathbf{x}_i\mathbf{x}_{i+1}}), \quad (2)$$

where L_e is the radiance emitted at the scene surface point \mathbf{x}_0 (beginning of the light path) towards the direction $\overline{\mathbf{x}_0\mathbf{x}_1}$. At every interaction vertex \mathbf{x}_i of the light path, there is a *bidirectional reflectance distribution function* (BRDF) $f_r(\mathbf{x}_i, \overline{\mathbf{x}_{i-1}\mathbf{x}_i}, \overline{\mathbf{x}_i\mathbf{x}_{i+1}})$ defined. The BRDF describes the material properties at the point \mathbf{x}_i , *i.e.*, how much light is scattered from the incident direction $\overline{\mathbf{x}_{i-1}\mathbf{x}_i}$ towards the outgoing direction $\overline{\mathbf{x}_i\mathbf{x}_{i+1}}$. The choice of the parametric BRDF model f_r is crucial to the range of materials that can be reconstructed by our system. We discuss the challenges of selecting the BRDF model in Sec. 4.1.

Note that both the BRDF f_r and the emitted radiance L_e are unknown and the desired parameters to be found at every point on the scene manifold.

3.2. Optimizing for Illumination and Materials

We take as input a series of images in the form of real-world photographs or synthetic renderings, together with the reconstructed scene geometry and corresponding camera poses. We aim to solve for the unknown material parameters \mathcal{M} and lighting parameters \mathcal{L} that will produce rendered images of the scene that are identical to the input images.

Given the un-tonemapped captured pixel intensities I_C^j at all pixels j of all images, and the corresponding noisy estimated pixel intensities \tilde{I}_R^j (in linear color space), we seek all material and illumination parameters $\Theta = \{\mathcal{M}, \mathcal{L}\}$ by solving the following optimization problem using stochastic gradient descent:

$$\operatorname{argmin}_{\Theta} E(\Theta) = \sum_j^N \left| I_C^j - \tilde{I}_R^j \right|_1, \quad (3)$$

where N is the number of pixels in all images. We found that using an L_1 norm as a loss function helps with robustness to outliers, such as extremely high contribution samples coming from Monte Carlo sampling.

3.3. Computing Gradients with Path Tracing

In order to efficiently solve the minimization problem in Eq. 3 using stochastic optimization, we compute the gradient of the energy function $E(\Theta)$ with respect to the set of unknown material and emission parameters Θ :

$$\nabla_{\Theta} E(\Theta) = \sum_j^N \nabla_{\Theta} \tilde{I}_R^j \operatorname{sgn} \left(I_C^j - \tilde{I}_R^j \right), \quad (4)$$

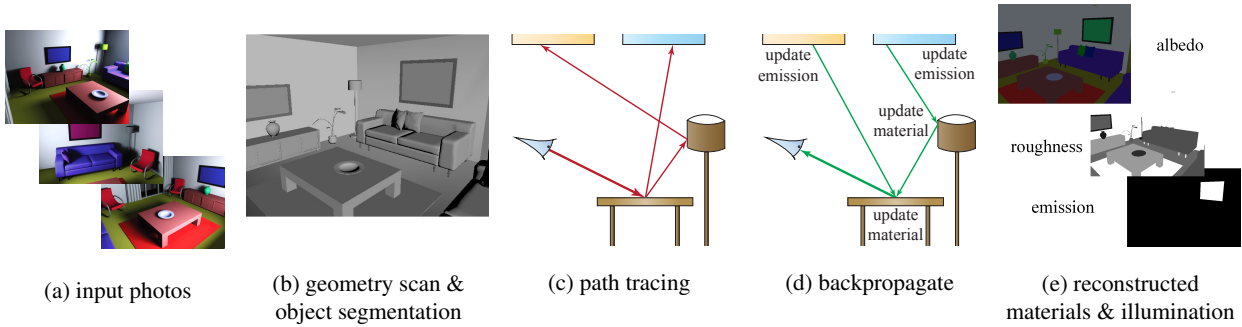


Figure 3: Overview of our pipeline. Given (a) a set of input photos from different views, along with (b) an accurate geometry scan and proper segmentation, we reconstruct the material properties and illumination of the scene, by iteratively (c) rendering the scene with path tracing, and (d) backpropagating to the material and illumination parameters in order to update them. After numerous iterations, we obtain the (e) reconstructed material and illumination.

where $\text{sgn}(\cdot)$ is the sign function, and $\nabla_{\Theta} \tilde{I}_R^j$ the gradient of the Monte Carlo estimate with respect to all unknowns Θ .

Note that this equation for computing the gradient now has two Monte Carlo estimates for each pixel j : (1) the estimate of pixel color itself \tilde{I}_R^j ; and (2) the estimate of its gradient $\nabla_{\Theta} \tilde{I}_R^j$. Since the expectation of product only equals the product of expectation when the random variables are independent, it is important to draw independent samples for each of these estimates to avoid introducing bias.

In order to compute the gradients of a Monte Carlo estimate for a single pixel j , we determine what unknowns are touched by the measurement contribution function $f(\mathbf{X})$ for a sampled light path \mathbf{X} . We obtain the explicit formula of the gradients by differentiating Eq. 2 using the product rule (for brevity, we omit some arguments for emission L_e and BRDF f_r):

$$\nabla_{\Theta_c} f(\mathbf{X}) = \nabla_{\Theta_c} L_e(\mathbf{x}_0) \prod_i^k f_r(\mathbf{x}_i) \quad (5)$$

$$\nabla_{\Theta_M} f(\mathbf{X}) = L_e(\mathbf{x}_0) \sum_l^k \nabla_{\Theta_M} f_r(\mathbf{x}_l) \prod_{i, i \neq l}^k f_r(\mathbf{x}_i) \quad (6)$$

where the gradient vector $\nabla_{\Theta} = \{\nabla_{\Theta_M}, \nabla_{\Theta_c}\}$ is very sparse and has non-zero values only for unknowns touched by the path \mathbf{X} . The gradients of emissions (Eq. 5) and materials (Eq. 6) have similar structure to the original path contribution (Eq. 2). Therefore, it is natural to apply the same path sampling strategy; see the appendix for details.

3.4. Multiple Captured Images

The single-image problem can be directly extended to multiple images. Given multiple views of a scene, we aim to find parameters for which rendered images from these views match the input images. A set of multiple views can cover parts of the scene that are not covered by any single

view from the set. This proves important for deducing the correct position of the light source in the scene. With many views, the method can better handle view-dependent effects such as specular and glossy highlights, which can be ill-posed with just a single view, as they can also be explained as variations of albedo texture.

4. Optimization Parameters and Methodology

In this section we address the remaining challenges of the optimization task: what are the material and illumination parameters we actually optimize for, and how to resolve the ill-posed nature of the problem.

4.1. Parametric Material Model

We want our material model to satisfy several properties. First, it should cover as much variability in appearance as possible, including such common effects as specular highlights, multi-layered materials, and spatially-varying textures. On the other hand, since each parameter adds another unknown to the optimization, we would like to keep the number of parameters minimal. Since we are interested in re-rendering and related tasks, the material model needs to have interpretable parameters, so the users can adjust the parameters to achieve the desired appearance. Finally, since we are optimizing the material properties using first-order gradient-based optimization, we would like the range of the material parameters to be similar.

To satisfy these properties, we represent our materials using the Disney material model [5], the state-of-the-art physically based material model used in movie and game rendering. It has a “base color” parameter which is used by both diffuse and specular reflectance, as well as 10 other parameters describing the roughness, anisotropy, and specularly of the material. All these parameters are perceptually mapped to $[0, 1]$, which is both interpretable and suitable for optimization.

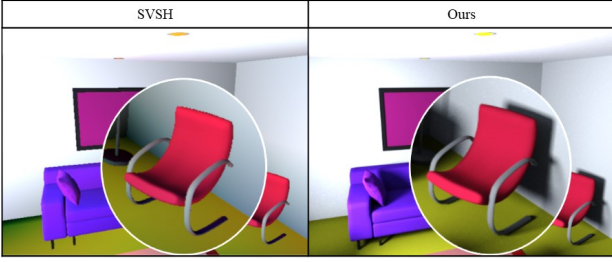


Figure 4: Methods based on spherical harmonics have difficulties handling sharp shadows or lighting changes due to the distant illumination assumption. A physically based method, such as Inverse Path Tracing, correctly reproduces these effects.

4.2. Scene Parameterization

We use triangle meshes to represent the scene geometry. Surface normals are defined per-vertex and interpolated within each triangle using barycentric coordinates. The optimization is performed on a per-object basis, *i.e.*, every object has a single unknown emission and a set of material parameters that are assumed constant across the whole object. We show that this is enough to obtain accurate lighting and an average constant value for the albedo of an object.

4.3. Emission Parameterization

For emission reconstruction, we currently assume all light sources are scene surfaces with an existing reconstructed geometry. For each emissive surface, we currently assume that emitted radiance is distributed according to a view-independent directional emission profile $L_e(\mathbf{x}, \mathbf{i}) = e(\mathbf{x})(\mathbf{i} \cdot \mathbf{n}(\mathbf{x}))_+$, where $e(\mathbf{x})$ is the unknown radiant flux at \mathbf{x} ; \mathbf{i} is the emission direction at surface point \mathbf{x} , $\mathbf{n}(\mathbf{x})$ is the surface normal at \mathbf{x} and $(\cdot)_+$ is the dot product (cosine) clamped to only positive values. This is a common emission profile for most of the area lights, which approximates most of the real soft interior lighting well. Our method can also be extended to more complex or even unknown directional emission profiles or purely directional distant illumination (*e.g.*, sky dome, sun) if needed.

4.4. Regularization

The observed color of an object in a scene is most easily explained by assigning emission to the triangle. This is only avoided by differences in shading of the different parts of the object. However, it can happen that there are no observable differences in the shading of an object, especially if the object covers only a few pixels in the input image. This can be a source of error during optimization. Another source of error is Monte Carlo and SGD noise. These errors lead to incorrect emission parameters for many objects after the optimization. The objects usually have a small estimated emission value when they should have none. We tackle the

problem with an L1-regularizer for the emission. The vast majority of objects in the scene is not an emitter and having such a regularizer suppresses the small errors we get for the emission parameters after optimization.

4.5. Optimization Parameters

We use ADAM [20] as our optimizer with batch size $B = 8$ estimated pixels and learning rate $5 \cdot 10^{-3}$. To form a batch, we sample B pixels uniformly from the set of all pixels of all images. Please see the appendix for an evaluation regarding the impact of different batch sizes and sampling distributions on the convergence rate. While a higher batch size reduces the variance of each iteration, having smaller batch sizes, and therefore faster iterations, proves to be more beneficial.

5. Results

Evaluation on synthetic data. We first evaluate our method on multiple synthetic scenes, where we know the ground truth solution. Quantitative results are listed in Tab. 1, and qualitative results are shown in Fig. 5. Each scene is rendered using a path tracer with the ground truth lighting and materials to obtain the “captured images”. These captured images and scene geometry are then given to our Inverse Path Tracing algorithm, which optimizes for unknown lighting and material parameters. We compare to the closest previous work based on spatially-varying spherical harmonics (SVSH) [23]. SVSH fails to capture sharp details such as shadows or high-frequency lighting changes. A comparison of the shadow quality is presented in Fig. 4.

Our method correctly detects light sources and converges to a correct emission value, while the emission of objects that do not emit light stays at zero. Fig. 6 shows a novel view, rendered with results from an optimization that was performed on input views from Fig. 5. Even though the light source was not visible in any of the input views, its emission was correctly computed by Inverse Path Tracing.

In addition to albedo, our Inverse Path Tracer can also optimize for other material parameters such as roughness. In Fig. 8, we render a scene containing objects of varying roughness. Even when presented with the challenge of estimating both albedo and roughness, our method produces the correct result as shown in the re-rendered image.

Evaluation on real data. We use the Matterport3D [6] dataset to evaluate our method on real captured scenes obtained through 3D reconstruction. The scene was parameterized using the segmentation provided in the dataset. Due to imperfections in the data, such as missing geometry and inaccurate surface normals, it is more challenging to perform an accurate light transport simulation. Nevertheless, our method produces impressive results for the given input. After the optimization, the optimized light direction

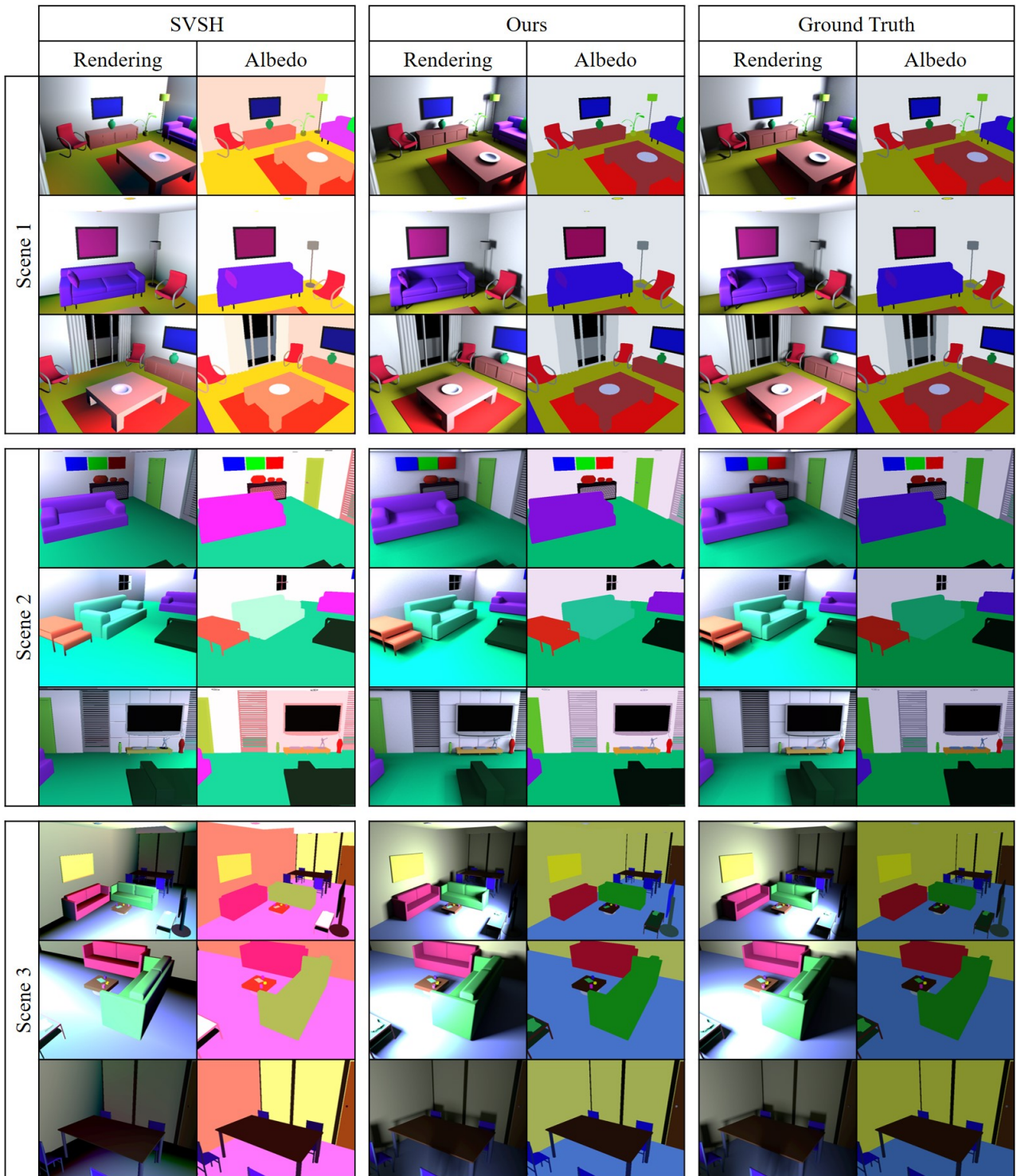


Figure 5: Evaluation on synthetic scenes. Three scenes have been rendered from different views with both direct and indirect lighting (right). An approximation of the albedo lighting with spatially-varying spherical harmonics is shown (left). Our method is able to detect the light source even though it was not observed in any of the views (middle). Notice that we are able to reproduce sharp lighting changes and shadows correctly. The albedo is also closer to the ground truth albedo.

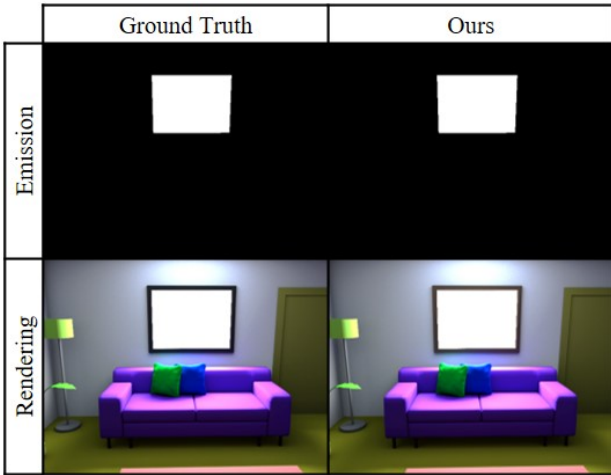


Figure 6: Inverse Path Tracing is able to correctly detect the light emitting object (top). The ground truth rendering and our estimate is shown on the bottom. Note that this view was not used during optimization.

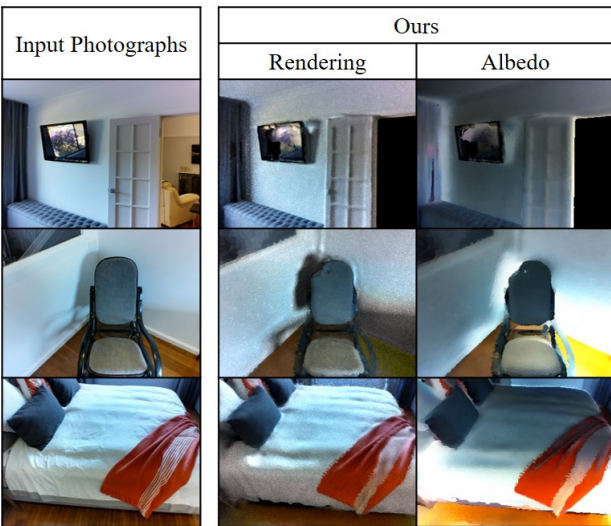


Figure 7: We can resolve object textures by optimizing for the unknown parameters per triangle. Higher resolution textures can be obtained by further subdividing the geometry.

matches the captured light direction and the rendered result closely matches the photograph. Fig. 11 shows a comparison to the SVSH method.

The albedo of real-world objects varies across its surface. Inverse Path Tracing is able to compute an object’s average albedo by employing knowledge of the scene segmentation. To reproduce fine texture, we refine the method to optimize for each individual triangle of the scene with adaptive subdivision where necessary. This is demonstrated in Fig. 7.

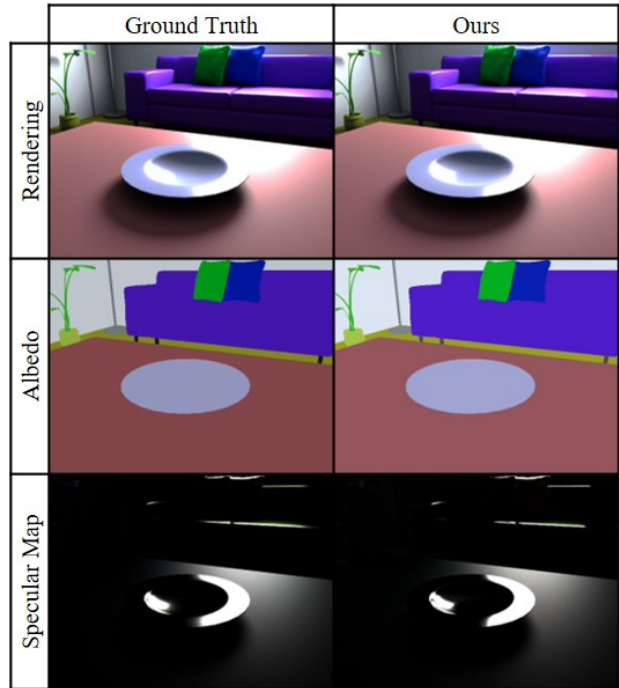


Figure 8: Inverse Path Tracing is agnostic to the underlying BRDF; *e.g.*, here, in a specular case, we are able to correctly estimate both the albedo and the roughness of the objects. The ground truth rendering and our estimate is shown on top, the albedo in the middle and the specular map on the bottom.

Optimizer Ablation. There are several ways to reduce the variance of our optimizer. One obvious way is to use more samples to estimate the pixel color and the derivatives, but this also results in slower iterations. Fig. 9 shows that the method does not converge if only a single path is used. A general recommendation is to use between 2^7 and 2^{10} depending on the scene complexity and number of unknowns.

Another important aspect of our optimizer is the sample distribution for pixel color and derivatives estimation. Our tests in Fig. 10 show that minimal variance can be achieved by using one sample to estimate the derivatives and the remaining samples in the available computational budget to estimate the pixel color.

Limitations. Inverse Path Tracing assumes that high-quality geometry is available. However, imperfections in the recovered geometry can have big impact on the quality of material estimation as shown in Fig. 11. Our method also does not compensate for the distortions in the captured input images. Most cameras, however, produce artifacts such as lens flare, motion blur or radial distortion. Our method can potentially account for these imperfections by simulating the corresponding effects and optimize not only for the ma-

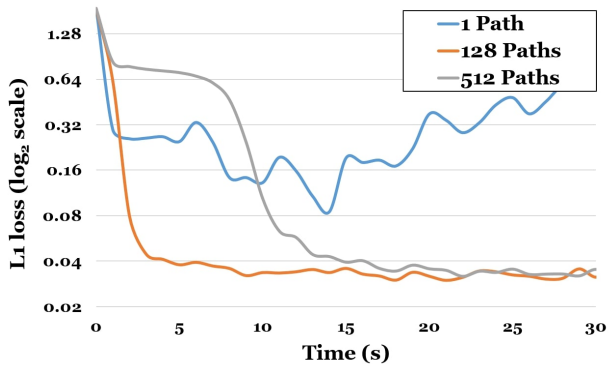


Figure 9: Convergence with respect to the number of paths used to estimate the pixel color. If this is set too low, the algorithm will fail.

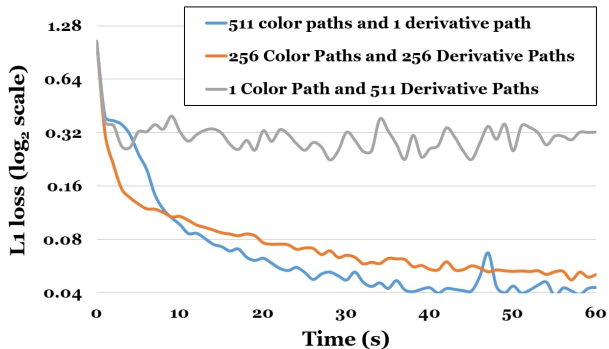


Figure 10: Convergence with respect to distributing the available path samples budget between pixel color and derivatives. It is best to keep the number of paths high for pixel color estimation and low for derivative estimation.

Method	Scene 1	Scene 2	Scene 3
SVSH Rendering Loss	0.052	0.048	0.093
Our Rendering Loss	0.006	0.010	0.003
SVSH Albedo Loss	0.052	0.037	0.048
Our Albedo Loss	0.002	0.009	0.010

Table 1: Quantitative evaluation for synthetic data. We measure the L1 loss with respect to the rendering error and the estimated albedo parameters. Note that our approach achieves a significantly lower error on both metrics.

terial parameters, but also for the camera parameters, which we leave for future work.

6. Conclusion

We present Inverse Path Tracing, a novel approach for joint lighting and material estimation in 3D scenes. We

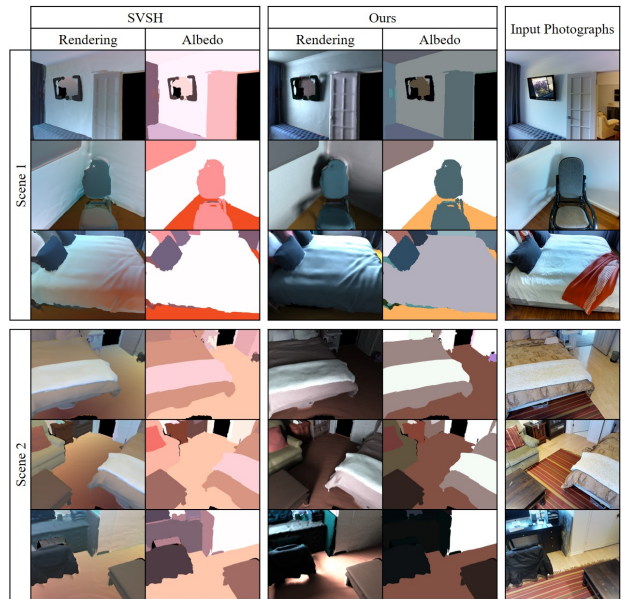


Figure 11: Evaluation on real scenes: (right) input is 3D scanned geometry and photographs. We employ object instance segmentation to estimate the emission and the average albedo of every object in the scene. Our method is able to optimize for the illumination and shadows. Other methods usually do not take occlusions into account and fail to model shadows correctly. Views 1 and 2 of Scene 2 show that if the light emitters are not present in the input geometry, our method gives an incorrect estimation.

demonstrate that our differentiable Monte Carlo renderer can be efficiently integrated in a nested stochastic gradient descent optimization. In our results, we achieve significantly higher accuracy than existing approaches. High-fidelity reconstruction of materials and illumination is an important step for a wide range of applications such as virtual and augmented reality scenarios. Overall, we believe that this is a flexible optimization framework for computer vision that is extensible to various scenarios, noise factors, and other imperfections of the computer vision pipeline. We hope to inspire future work along these lines, for instance, by incorporating more complex BRDF models, joint geometric refinement and completion, and further stochastic regularizations and variance reduction techniques.

Acknowledgements

This work is funded by Facebook Reality Labs. We also thank the TUM-IAS Rudolf Mößbauer Fellowship (Focus Group Visual Computing) for their support. We would also like to thank Angela Dai for the video voice over and Abhimitra Meka for the LIME comparison.

References

- [1] Jonathan T Barron and Jitendra Malik. Shape, illumination, and reflectance from shading. *Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, 2015. 1, 2
- [2] Harry Barrow, J Tenenbaum, A Hanson, and E Riseman. Recovering intrinsic scene characteristics. *Comput. Vis. Syst.*, 2:3–26, 1978. 2
- [3] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, pages 187–194, 1999. 2
- [4] Nicolas Bonneel, Balazs Kovacs, Sylvain Paris, and Kavita Bala. Intrinsic decompositions for image editing. *Computer Graphics Forum (Eurographics State of the Art Reports)*, 36(2), 2017. 2
- [5] Brent Burley and Walt Disney Animation Studios. Physically-based shading at disney. 4
- [6] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Habber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017. 5
- [7] Chengqian Che, Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. Inverse transport networks. *arXiv preprint arXiv:1809.10820*, 2018. 2
- [8] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (TOG)*, 36(4):76a, 2017. 1
- [9] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. *SIGGRAPH*, pages 145–156, 2000. 2
- [10] Valentin Deschaintre, Miika Aittala, Fredo Durand, George Drettakis, and Adrien Bousseau. Single-image SVBRDF capture with a rendering-aware deep network. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 37(4):128:1–128:15, 2018. 2
- [11] Yue Dong, Guojun Chen, Pieter Peers, Jiawan Zhang, and Xin Tong. Appearance-from-motion: Recovering spatially varying surface reflectance under unknown lighting. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 33(6):193:1–193:12, 2014. 2
- [12] Ioannis Gkioulekas, Anat Levin, and Todd Zickler. An evaluation of computational imaging techniques for heterogeneous inverse scattering. In *European Conference on Computer Vision*, pages 685–701, 2016. 2
- [13] Ioannis Gkioulekas, Shuang Zhao, Kavita Bala, Todd Zickler, and Anat Levin. Inverse volume rendering with material dictionaries. *ACM Trans. Graph.*, 32(6):162:1–162:13, nov 2013. 2
- [14] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011. 1
- [15] James T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 20(4):143–150, Aug. 1986. 3
- [16] Kaizhang Kang, Zimin Chen, Jiaping Wang, Kun Zhou, and Hongzhi Wu. Efficient reflectance capture using an autoencoder. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 37(4):127:1–127:10, 2018. 2
- [17] Mike Kasper, Nima Keivan, Gabe Sibley, and Christoffer R. Heckman. Light source estimation with analytical path-tracing. *CoRR*, abs/1701.04101, 2017. 2
- [18] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3D mesh renderer. In *Computer Vision and Pattern Recognition*, pages 3907–3916, 2018. 2
- [19] Kihwan Kim, Jinwei Gu, Stephen Tyree, Pavlo Molchanov, Matthias Niessner, and Jan Kautz. A lightweight approach for on-the-fly reflectance estimation, Oct 2017. 2
- [20] Diederick P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 5
- [21] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6):222:1–222:11, 2018. 2
- [22] Matthew M. Loper and Michael J. Black. OpenDR: An approximate differentiable renderer. In *European Conference on Computer Vision*, volume 8695, pages 154–169, sep 2014. 2
- [23] 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 *International Conference on Computer Vision (ICCV)*, Venice, Italy, October 2017. 2, 5
- [24] Stephen Robert Marschner. *Inverse Rendering for Computer Graphics*. PhD thesis, 1998. 2
- [25] Abhimitra Meka, Maxim Maximov, Michael Zollhoefer, Avishek Chatterjee, Hans-Peter Seidel, Christian Richardt, and Christian Theobalt. Lime: Live intrinsic material estimation. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2
- [26] Abhimitra Meka, Michael Zollhoefer, Christian Richardt, and Christian Theobalt. Live intrinsic video. *ACM Transactions on Graphics (Proceedings SIGGRAPH)*, 35(4), 2016. 1
- [27] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011. 1
- [28] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):169, 2013. 1
- [29] Gustavo Patow and Xavier Pueyo. A survey of inverse rendering problems. *Computer Graphics Forum*, 22(4):663–687, 2003. 2

- [30] Ravi Ramamoorthi and Pat Hanrahan. A signal-processing framework for inverse rendering. *SIGGRAPH*, pages 117–128, 2001. [2](#)
- [31] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2387–2395, 2016. [2](#)
- [32] Eric Veach. *Robust Monte Carlo Methods for Light Transport Simulation*. PhD thesis, Stanford, CA, USA, 1998. AAI9837162. [3](#)
- [33] Thomas Whelan, Renato F Salas-Moreno, Ben Glocker, Andrew J Davison, and Stefan Leutenegger. Elasticfusion: Real-time dense slam and light source estimation. *The International Journal of Robotics Research*, 35(14):1697–1716, 2016. [1](#)
- [34] Chenglei Wu, Michael Zollhöfer, Matthias Nießner, Marc Stamminger, Shahram Izadi, and Christian Theobalt. Real-time shading-based refinement for consumer depth cameras. *ACM Transactions on Graphics (TOG)*, 33(6), 2014. [1](#)
- [35] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *SIGGRAPH*, pages 215–224, 1999. [2](#)
- [36] Edward Zhang, Michael F. Cohen, and Brian Curless. Emptying, refurbishing, and relighting indoor spaces. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 35(6), 2016. [1](#), [2](#)
- [37] Michael Zollhöfer, Angela Dai, Matthias Innmann, Chenglei Wu, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Shading-based refinement on volumetric signed distance functions. *ACM Transactions on Graphics (TOG)*, 34(4), 2015. [1](#)

C.2 Neural RGB-D Surface Reconstruction

Copyright Notice

©2022 IEEE. Reprinted, with permission, from
Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner and Justus
Thies

Neural RGB-D Surface Reconstruction

*IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June
2022*

DOI: 10.1109/CVPR52688.2022.00619

In accordance with the IEEE Thesis/Dissertation Reuse Permissions, we include the accepted version of the original publication [15] in the following.

Neural RGB-D Surface Reconstruction

Dejan Azinović¹ Ricardo Martin-Brualla² Dan B Goldman² Matthias Nießner¹ Justus Thies^{1,3}

¹Technical University of Munich ²Google Research ³Max Planck Institute for Intelligent Systems

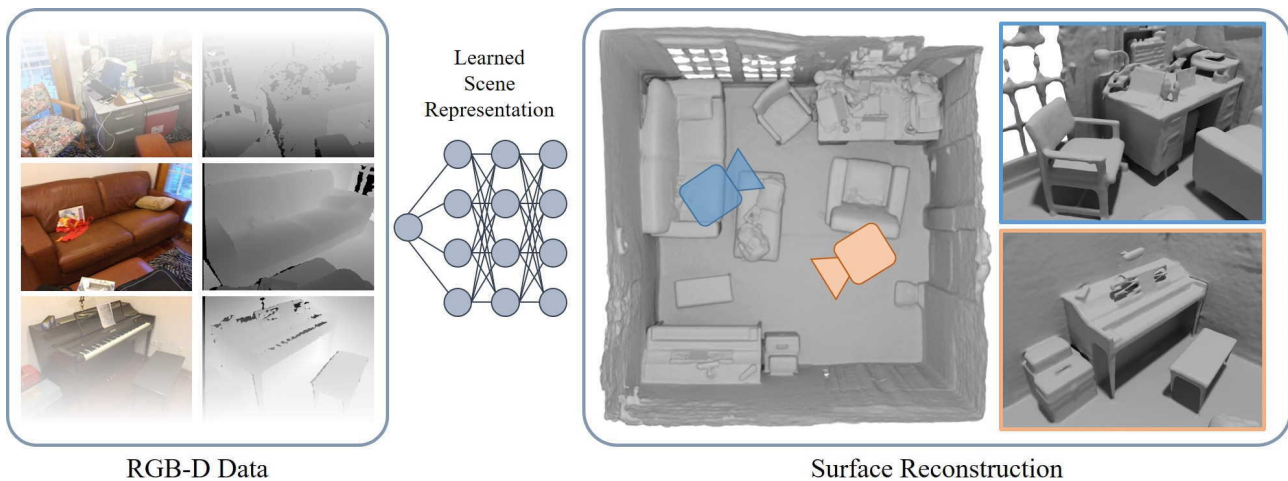


Figure 1. Our method obtains a high-quality 3D reconstruction from an RGB-D input sequence by training a multi-layer perceptron. The core idea is to reformulate the neural radiance field definition in NeRF [48], and replace it with a differentiable rendering formulation based on signed distance fields which is specifically tailored to geometry reconstruction.

Abstract

Obtaining high-quality 3D reconstructions of room-scale scenes is of paramount importance for upcoming applications in AR or VR. These range from mixed reality applications for teleconferencing, virtual measuring, virtual room planning, to robotic applications. While current volume-based view synthesis methods that use neural radiance fields (NeRFs) show promising results in reproducing the appearance of an object or scene, they do not reconstruct an actual surface. The volumetric representation of the surface based on densities leads to artifacts when a surface is extracted using Marching Cubes, since during optimization, densities are accumulated along the ray and are not used at a single sample point in isolation. Instead of this volumetric representation of the surface, we propose to represent the surface using an implicit function (truncated signed distance function). We show how to incorporate this representation in the NeRF framework, and extend it to use depth measurements from a commodity RGB-D sensor, such as a Kinect. In addition, we propose a pose and camera refinement technique which improves the overall reconstruction quality.

In contrast to concurrent work on integrating depth priors in NeRF which concentrates on novel view synthesis, our approach is able to reconstruct high-quality, metric 3D reconstructions.

1. Introduction

Research on neural networks for scene representations and image synthesis has made impressive progress in recent years [73]. Methods that learn volumetric representations [42, 48] from color images captured by a smartphone camera can be employed to synthesize near photo-realistic images from novel viewpoints. While the focus of these methods lies on the reproduction of color images, they are not able to reconstruct metric and clean (noise-free) meshes. To overcome these limitations, we show that there is a significant advantage in taking additional range measurements from consumer-level depth cameras into account. Inexpensive depth cameras are broadly accessible and are also built into modern smartphones. While classical reconstruction methods [9, 33, 53] that purely rely on depth measurements struggle with the limitations of physical sen-

sors (noise, limited range, transparent objects, etc.), a neural radiance field-based reconstruction formulation allows to also leverage the dense color information. Methods like BundleFusion [14] take advantage of color observations to compute sparse SIFT [44] features for re-localization and refinement of camera poses (loop closure). For the actual geometry reconstruction (volumetric fusion), only the depth maps are taken into account. Missing depth measurements in these maps, lead to holes and incomplete geometry in the reconstruction. This limitation is also shared by learned surface reconstruction methods that only rely on the range data [47, 67]. In contrast, our method is able to reconstruct geometry in regions where only color information is available. Specifically, we adapt the neural radiance field (NeRF) formulation of Mildenhall et al. [48] to learn a truncated signed distance field (TSDF), while still being able to leverage differentiable volumetric integration for color reproduction. To compensate for noisy initial camera poses which we compute based on the depth measurements, we jointly optimize our scene representation network with the camera poses. The implicit function represented by the scene representation network allows us to predict signed distance values at arbitrary points in space which is used to extract a mesh using Marching Cubes.

Concurrent work that incorporates depth measurements in NeRF focuses on novel view synthesis [16, 49, 81], and uses the depth prior to restrict the volumetric rendering to near-surface regions [49, 81] or adds an additional constraint on the depth prediction of NeRF [16]. NeuS [76] is also a concurrent work on novel view synthesis which uses a signed distance function to represent the geometry, but takes only RGB images as input, and thus fails to reconstruct the geometry of featureless surfaces, like white walls. In contrast, our method aims for high-quality 3D reconstructions of room-scale scenes using an implicit surface representation and direct SDF-based losses on the input depth maps. Comparisons to state-of-the-art scene reconstruction methods show that our approach improves the quality of geometry reconstructions both qualitatively and quantitatively.

In summary, we propose an RGB-D based scene reconstruction method that leverages both dense color and depth observations. It is based on an effective incorporation of depth measurements into the optimization of a neural radiance field using a signed distance-based surface representation to store the scene geometry. It is able to reconstruct geometry detail that is observed by the color images, but not visible in the depth maps. In addition, our pose and camera refinement technique is able to compensate for misalignments in the input data, resulting in state-of-the-art reconstruction quality which we demonstrate on synthetic as well as on real data from ScanNet [12].

2. Related Work

Our approach reconstructs geometry from a sequence of RGB-D frames, leveraging both dense color and depth information. It is related to classical fusion-based 3D reconstruction methods [9, 14, 50, 53, 92], learned 3D reconstruction [7, 15, 47, 58, 79], as well as to recent coordinate-based scene representation models [48, 69, 74].

Classical 3D Reconstruction. There exists a wide range of methods for RGB and RGB-D based 3D reconstruction that are not based on deep learning. Reconstructing objects and scenes can be done using passive stereo systems that rely on stereo matching from two or multiple color views [29, 62], Structure-from-Motion [63], or SLAM-based [20, 21, 23] methods. These approaches may use disjoint representations, like oriented patches [25], volumes [38], or meshes [32] to reconstruct the scene or object. Zollhöfer et al. [92] review the 3D reconstruction methods that rely on range data from RGB-D cameras like the Kinect. Most of these methods are based on [9], where multiple depth measurements are fused using a signed distance function (SDF) which is stored in a uniform 3D grid. E.g., KinectFusion [50] combines such representation with real-time tracking to reconstruct objects and small scenes in real-time. To handle large scenes Nießner et al. [53] propose a memory-efficient storage of the SDF grid using spatial hashing. To handle the loop closure problem when scanning large-scale scenes, bundle adjustment can be used to refine the camera poses [14]. In addition, several regularization techniques have been proposed to handle outliers during reconstruction [19, 61, 88].

Deep Learning for 3D Reconstruction. To reduce artifacts from classical reconstruction methods, a series of methods was proposed that use learned spatial priors to predict depth maps from color images [24, 28, 39], to learn multi-view stereo using 3D CNNs on voxel grids [34, 68, 82], or multi-plane images [22], to reduce the influence of noisy depth values [79], to complete incomplete scans [13, 15], to learn image features for SLAM [2, 10, 90] or feature fusion [4, 71, 80], to predict normals [89], or to predict objects or parts of a room from single images [11, 18, 27, 51, 75]. Most recently coordinate-based models have become popular [74]. These models use a scene representation that is based on a deep neural network with fully connected layers, i.e., a multi-layer perceptron (MLP) [73, 74]. As input the MLP takes a 3D location in the model space and outputs for example, occupancy [7, 47, 52, 55, 58–60], density [48], radiance [48], color [54], or the signed distance to the surface [56, 83, 84]. Scene Representation Networks [69] combine such a representation with a learned renderer which is inspired by classical sphere tracing, to reconstruct objects from single RGB images. Instead, Mildenhall et al. [48] propose a method that represents a scene as a neural ra-

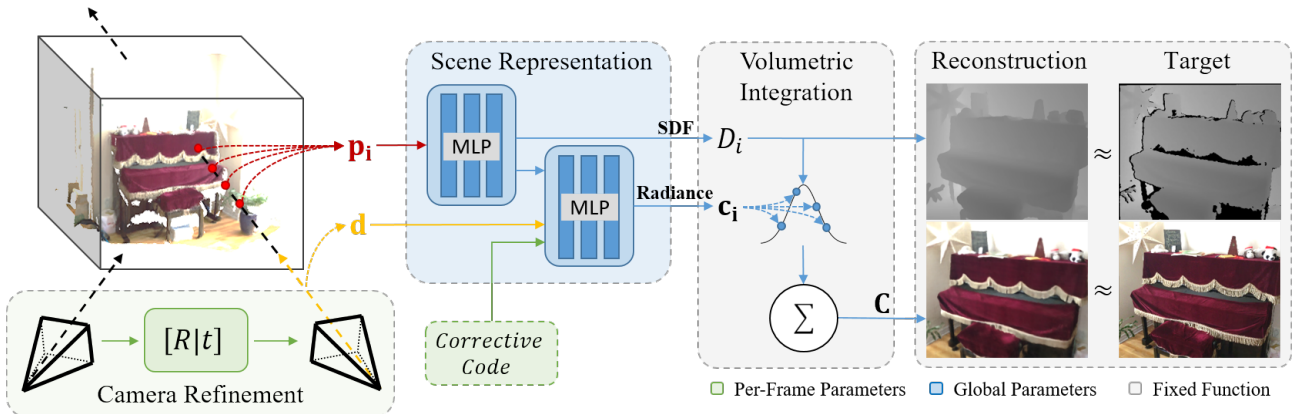


Figure 2. Differentiable volumetric rendering is used to reconstruct a scene that has been captured using an RGB-D camera. The scene is represented using multi-layer perceptrons (MLPs), encoding a signed distance value D_i and a viewpoint-dependent radiance value c_i per point p_i . We perform volumetric rendering by integrating the radiance along a ray, weighing the samples as a function of their signed distance D_i and their visibility. We also learn a per-frame latent corrective code to account for exposure or white balance changes throughout the capture, which is passed to the radiance MLP alongside the ray direction \mathbf{d} . We optimize the scene representation’s MLPs, together with the per-frame corrective codes, the input camera poses, and an image-plane deformation field (not shown) by computing losses for the signed distance D_i of the samples, and the final integrated color \mathbf{C} with respect to the input depth and color views.

diance field (NeRF) using a coordinate-based model, and a classical, fixed volumetric rendering formulation [46]. Based on this representation, they show impressive novel view synthesis results, while only requiring color input images with corresponding camera poses and intrinsics. Besides the volumetric image formation, a key component of the NeRF technique is a positional encoding layer, that uses sinusoidal functions to improve the learning properties of the MLP. In follow-up work, alternatives to the positional encoding were proposed, such as Fourier features [72] or sinusoidal activation layers [67]. NeRF has been extended to handle in-the-wild data with different lighting and occluders [45], dynamic scenes [40, 57], avatars [26], and adapted for generative modeling [5, 66] and image-based rendering [77, 87]. Others have focused on resectioning a camera given a learned NeRF [85], and optimizing for the camera poses while learning a NeRF [41, 78].

In our work, we take advantage of the volumetric rendering of NeRF and propose the usage of a hybrid scene representation that consists of an implicit surface representation (SDF) and a volumetric radiance field. We incorporate depth measurements in this formulation to achieve robust and metric 3D reconstructions. In addition, we propose a camera refinement scheme to further improve the quality of the reconstruction. In contrast to NeRF which uses a density based volumetric representation of the scene, our implicit surface representation leads to high quality geometry estimates of entire scenes.

Concurrent Work. In concurrent work, Wang et al. [76] present NeuS which uses an implicit surface representation to improve novel view synthesis of NeRF. Wei et al. [81]

propose a multi-view stereo approach to estimate dense depth maps which they use to constrain the sampling region when optimizing a NeRF. Similarly, Neff et al. [49] restrict the volumetric rendering to near surface regions. Additional constraints on the depth predictions of NeRF were proposed by Deng et al. [16]. In contrast to these, our method focuses on accurate 3D reconstructions of room-scale scenes, with explicit incorporation of depth measurements using an implicit surface representation.

3. Method

We propose an optimization-based approach for geometry reconstruction from an RGB-D sequence of a consumer-level camera (e.g., a Microsoft Kinect). We leverage both the N color frames \mathcal{I}_i as well as the corresponding aligned depth frames \mathcal{D}_i to optimize a coordinate-based scene representation network. Specifically, our hybrid scene representation consists of an implicit surface representation based on a truncated signed distance function (TSDF) and a volumetric representation for the radiance. As illustrated in Fig. 2, we use differentiable volumetric integration of the radiance values [46] to compute color images from this representation. Besides the scene representation network, we optimize for the camera poses and intrinsics. We initialize the camera poses \mathcal{T}_i using BundleFusion [14]. At evaluation time, we use Marching Cubes [43] to extract a triangle mesh from the optimized implicit scene representation.

3.1. Hybrid Scene Representation

Our method is built upon a hybrid scene representation which combines an implicit surface representation with a

volumetric appearance representation. Specifically, we implement this representation using a multi-layer perceptron (MLP) which can be evaluated at arbitrary positions \mathbf{p}_i in space to compute a truncated signed distance value D_i and view-dependent radiance value \mathbf{c}_i . As a conditioning to the MLP, we use a sinusoidal positional encoding $\gamma(\cdot)$ [48] to encode the 3D query point \mathbf{p}_i and the viewing direction \mathbf{d} .

Inspired by the recent success of volumetric integration in neural rendering [48], we render color as a weighted sum of radiance values along a ray. Instead of computing the weights as probabilities of light reflecting at a given sample point based on the density of the medium [48], we compute weights directly from signed distance values as the product of two sigmoid functions:

$$w_i = \sigma\left(\frac{D_i}{tr}\right) \cdot \sigma\left(-\frac{D_i}{tr}\right), \quad (1)$$

where tr is the truncation distance. This bell-shaped function has its peak at the surface, i.e., at the zero-crossing of the signed distance values. A similar formulation is used in concurrent work [76], since this function produces unbiased estimates of the signed distance field. The truncation distance tr directly controls how quickly the weights fall to zero as the distance from the surface increases. To account for the possibility of multiple intersections, weights of samples beyond the first truncation region are set to zero. The color along a specific ray is approximated as a weighted sum of the K sampled colors:

$$\mathbf{C} = \frac{1}{\sum_{i=0}^{K-1} w_i} \sum_{i=0}^{K-1} w_i \cdot \mathbf{c}_i. \quad (2)$$

This scheme gives the highest integration weight to the point on the surface, while points farther away from the surface have lower weights. Although such an approach is not derived from a physically-based rendering model, as is the case with volumetric integration over density values, it represents an elegant way to render color in a signed distance field in a differentiable manner, and we show that it helps deduce depth values through a photometric loss (see Sec. 4). In particular, this approach allows us to predict hard boundaries between occupied and free space which results in high-quality 3D reconstructions of the surface. In contrast, density-based models [48] can introduce semi-transparent matter in front of the actual surface to represent view-dependent effects when integrated along a ray. This leads to noisy reconstructions and artifacts in free space, as can be seen in Sec. 4.

Network Architecture Our hybrid scene representation network is composed of two MLPs which represent the shape and radiance, as depicted in Fig. 2. The shape MLP takes the encoding of a queried 3D point $\gamma(\mathbf{p})$ as input and

outputs the truncated signed distance D_i to the nearest surface. The task of the second MLP is to produce the surface radiance for a given encoded view direction $\gamma(\mathbf{d})$ and an intermediate feature output of the shape MLP. The view vector conditioning allows our method to deal with view-dependent effects like specular highlights, which would otherwise have to be modeled by deforming the geometry. Since color data is often subject to varying exposure or white-balance, we learn a per-frame latent corrective code vector as additional input to the radiance MLP [45].

Pose and Camera Refinement The camera poses \mathcal{T}_i , represented with Euler angles and a translation vector for every frame, are initialized with BundleFusion [14] and refined during the optimization. Inspired by [91], an additional image-plane deformation field in form of a 6-layer ReLU MLP is added as a residual to the pixel location before unprojecting into a 3D ray to account for possible distortions in the input images or inaccuracies of the intrinsic camera parameters. Note that this correction field is the same for every frame. During optimization, camera rays are first shifted with the 2D vector retrieved from the deformation field, before being transformed to world space using the camera pose \mathcal{T}_i .

3.2. Optimization

We optimize our scene representation network by randomly sampling a batch of P_b pixels from the input dataset of color and depth images. For each pixel p in the batch, a ray is generated using its corresponding camera pose and S_p sample points are generated on the ray. Our global objective function $\mathcal{L}(\mathcal{P})$ is minimized w.r.t. the unknown parameters \mathcal{P} (the network parameters Θ and the camera poses \mathcal{T}_i) over all B input batches and is defined as:

$$\mathcal{L}(\mathcal{P}) = \sum_{b=0}^{B-1} \lambda_1 \mathcal{L}_{rgb}^b(\mathcal{P}) + \lambda_2 \mathcal{L}_{fs}^b(\mathcal{P}) + \lambda_3 \mathcal{L}_{tr}^b(\mathcal{P}). \quad (3)$$

$\mathcal{L}_{rgb}^b(\mathcal{P})$ measures the squared difference between the observed pixel colors \hat{C}_p and predicted pixel colors C_p of the b -th batch of rays:

$$\mathcal{L}_{rgb}^b(\mathcal{P}) = \frac{1}{|P_b|} \sum_{p \in P_b} (C_p - \hat{C}_p)^2. \quad (4)$$

\mathcal{L}_{fs}^b is a ‘free-space’ objective, which forces the MLP to predict a value of tr for samples $s \in S_p^{fs}$ which lie between the camera origin and the truncation region of a surface:

$$\mathcal{L}_{fs}^b(\mathcal{P}) = \frac{1}{|P_b|} \sum_{p \in P_b} \frac{1}{|S_p^{fs}|} \sum_{s \in S_p^{fs}} (D_s - tr)^2. \quad (5)$$

For samples within the truncation region ($s \in S_p^{tr}$), we apply $\mathcal{L}_{tr}^b(\mathcal{P})$, the signed distance objective of samples close

to the surface:

$$\mathcal{L}_{tr}^b(\mathcal{P}) = \frac{1}{P_b} \sum_{p \in P_b} \frac{1}{|S_p^{tr}|} \sum_{s \in S_p^{tr}} (D_s - \hat{D}_s)^2, \quad (6)$$

where D_s is the predicted signed distance of sample s , and \hat{D}_s the signed distance observed by the depth sensor, along the optical axis. In our experiments, we use a truncation distance $tr = 5$ cm, and scale the scene so that the truncation region maps to $[-1, 1]$ (positive in front of the surface, negative behind).

The S_p sample points on the ray are generated in two steps. In the first step S'_c sample points are generated on the ray using stratified sampling. Evaluating the MLP on these S'_c sample points allows us to get a coarse estimate for the ray depth by explicitly searching for the zero-crossing in the predicted signed distance values. In the second step, another S'_f sample points are generated around the zero-crossing and a second forward pass of the MLP is performed with these additional samples. The output of the MLP is concatenated to the output from the first step and color is integrated using all $S'_c + S'_f$ samples, before computing the objective loss. It is important that the sampling rate in the first step is high enough to produce samples within the truncation region of the signed distance field, otherwise the zero-crossing may be missed.

We implement our method in Tensorflow using the ADAM optimizer [36] with a learning rate of 5×10^{-4} and set the loss weights to $\lambda_1 = 0.1$, $\lambda_2 = 10$ and $\lambda_3 = 6 \times 10^3$. We run all of our experiments for 2×10^5 iterations, where in each iteration we compute the gradient w.r.t. $|P_b| = 1024$ randomly chosen rays. We set the number of S'_f samples to 16. S'_c is chosen such that there is on average one sample for every 1.5 cm of the ray length. The ray length itself needs to be greater than the largest distance in the scene that is to be reconstructed and ranges from 4 to 8 meters in our scenes.

4. Results

In the following, we evaluate our method on real, as well as on synthetic data. For the shown results, we use Marching Cubes [43] with a spatial resolution of 1 cm to extract a mesh from the reconstructed signed distance function.

Results on real data. We test our method on the ScanNet dataset [12] which provides RGB-D sequences of room-scale scenes. The data has been captured with a StructureIO camera which provides quality similar to that of a Kinect v1. The depth measurements are noisy and often miss structures like chair legs or other thin geometry. To this end our method proposes the additional usage of a dense color reconstruction loss, since regions that are missed by the range sensor are often captured by the color camera. To compensate for the exposure and white balancing of the used camera, our approach learns a per-frame latent code as proposed

Method	C- ℓ_1 ↓	IoU ↑	NC ↑	F-score ↑
BundleFusion	0.062	0.594	0.892	0.805
RoutedFusion	0.057	0.615	0.864	0.838
COLMAP + Poisson	0.057	0.619	0.901	0.839
Conv. Occ. Nets	0.077	0.461	0.849	0.643
SIREN	0.060	0.603	0.893	0.816
NeRF + Depth	0.065	0.550	0.768	0.782
Ours (w/o pose)	0.049	0.655	0.908	0.868
Ours	0.044	0.747	0.918	0.924

Table 1. Reconstruction results on a dataset of 10 synthetic scenes. The Chamfer ℓ_1 distance, normal consistency and the F-score [37] are computed between point clouds sampled with a density of 1 point per cm^2 , using a threshold of 5 cm for the F-score. We voxelize the mesh to compute the intersection-over-union (IoU) between the predictions and ground truth.

in [45]. In Fig. 3, we compare our method to the original ScanNet BundleFusion reconstructions which often suffer from severe camera pose misalignment. Our approach jointly optimizes for the scene representation network as well as the camera poses, leading to substantially reduced misalignment artifacts in the reconstructed geometry.

Quantitative evaluation. We perform a quantitative evaluation of our method on a dataset of 10 synthetic scenes for which the ground truth geometry and camera trajectory are known. Note that the ground truth camera trajectory is only used for the rendering and evaluation, and not for the reconstruction. For each frame, we render a photo-realistic image using Blender [8, 17]. We apply noise and artifacts, similar to those of a real depth sensor [1, 3, 30, 31]. On this data, we compare our technique to several state-of-the-art methods that use either depth input only, or both color and depth data to reconstruct geometry (see Tab. 1).

BundleFusion. BundleFusion [14] uses the color and depth input to reconstruct the scene. It is a classical depth fusion approach [9] which compensates misalignments using a global bundle adjustment approach.

RoutedFusion. RoutedFusion [79] uses a routing network which filters sensor-specific noise and outliers from depth maps and computes pixel-wise confidence values, which are used by a fusion network to produce the final SDF. It takes the depth maps and camera poses as input.

COLMAP with screened Poisson surface reconstruction. We obtain camera poses using COLMAP [63–65] and use these to back-project depth maps into world space. We obtain a mesh by applying screened Poisson surface reconstruction [35] on the resulting point cloud.

Convolutional Occupancy Networks. We accumulate the point clouds from the depth maps using BundleFusion poses and evaluate the pre-trained convolutional occupancy networks model [58] provided by the authors (which has been used on similar data [6]).

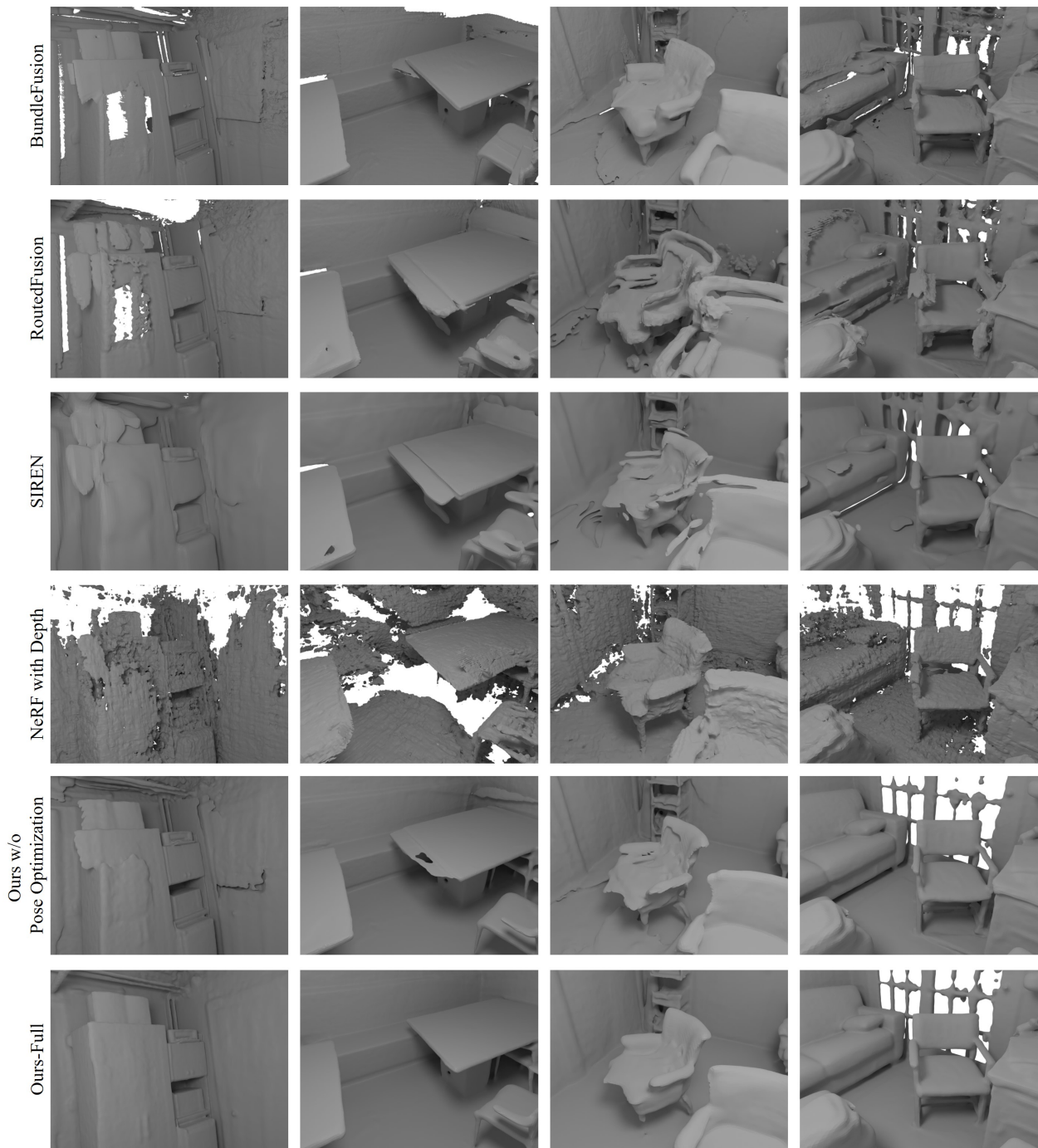


Figure 3. We compare our model without pose optimization and our full model with both the pose optimization and image-plane deformation field to BundleFusion, RoutedFusion, SIREN and a NeRF optimized with depth supervision in scenes 2, 5, 12, and 50 of the ScanNet dataset. Our model without pose optimization recovers smoother meshes than the density-based NeRF model, but still suffers from misalignment artifacts. These are solved by our full model to recover a clean reconstruction.

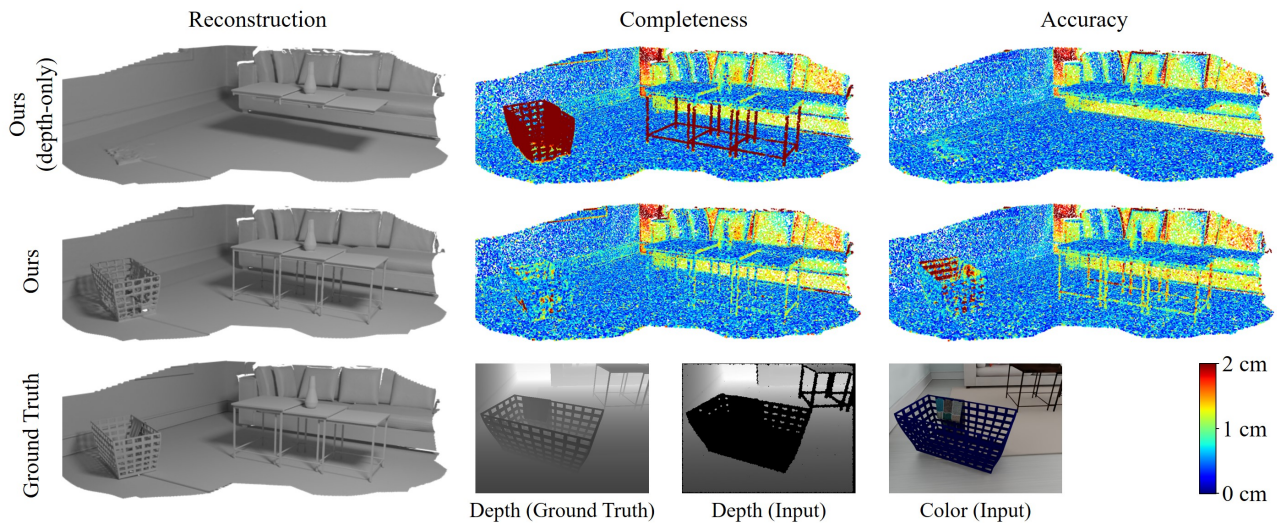


Figure 4. Accuracy shows how close ground truth points are to predicted points, while completeness shows how close predicted points are to ground truth points. Geometry reconstructed purely through the photometric loss has slightly lower accuracy than geometry for which depth observations were also available. Furthermore, the accuracy and completeness drop in distant areas, which had less multi-view constraints and more noise in the depth measurements.

SIREN. We optimize a SIREN [67] per scene using the back-projected point cloud data. The ICL-NUIM [31] scene on which the method was originally tested, is also included in our synthetic dataset.

NeRF with an additional depth loss. NeRF [48] proposes using the expected ray termination distance as a way to visualize the depth of the scene. In our baseline, we add an additional loss to NeRF, where this depth value is compared to the input depth using an L2 loss. Note that this baseline still uses NeRF’s density field to represent geometry.

As can be seen in Tab. 1, our approach with camera refinement results in the lowest Chamfer distance, and the highest IoU, normal consistency (mean of the dot product of the ground truth and predicted normals), and F-score [37]. Especially, the comparison to the density-based NeRF with an additional depth constraint shows the benefit of our proposed hybrid scene representation.

Ablation studies. We conduct ablation studies to justify our choice of network architecture and training parameters. In Fig. 3, we show the difference between a volumetric representation (density field, ‘NeRF with Depth’) to an implicit surface representation (signed distance field, ‘Ours-Full’) on real data from ScanNet [12]. While representing scenes with a density field works great for color integration, extracting the geometry itself is a challenging problem. Although small variations in density may not affect the integrated color much, they cause visible noise in the extracted geometry and produce floating artifacts in free space. These artifacts can be reduced by choosing a different iso-level for geometry extraction with Marching Cubes, but this leads to less complete reconstructions. In contrast, a signed distance

field models a smooth boundary between occupied and free space, and we show that it can be faithfully represented by an MLP. However, the reconstruction quality is still limited by the provided camera poses, as can be seen in Fig. 3 (e.g., the cabinet in the left column). Optimizing for pose corrections further improves the quality of our reconstructions.

Effect of the photometric term. A fundamental component of our method is the use of a photometric term to infer depth values which are missing from camera measurements. We analyze the effect of this term on the synthetic scene in Fig. 4, where we simulate missing geometry of the table legs and the meshed basket. In the figure, we visualize the completeness and accuracy. In contrast to a model without the photometric term, our method is still able to reconstruct the missing geometry leveraging the RGB observations.

For our full approach, we also separately evaluate the reconstruction quality of geometry where depth measurements were available and where they were missing. Regions that relied only on color have a somewhat worse average accuracy of 11 mm, compared to 8 mm for regions that had access to depth measurements. We refer the reader to the supplemental material for more details and a qualitative comparison on real data.

Effect of pose refinement. We show that initial camera pose estimates can be further improved by jointly optimizing for the rotation and translation parameters of the cameras which are initialized with BundleFusion [14]. We quantitatively evaluate this on all scenes in our synthetic dataset. An aggregate of the positional and rotational errors of different methods is presented in Tab. 2. A detailed per-

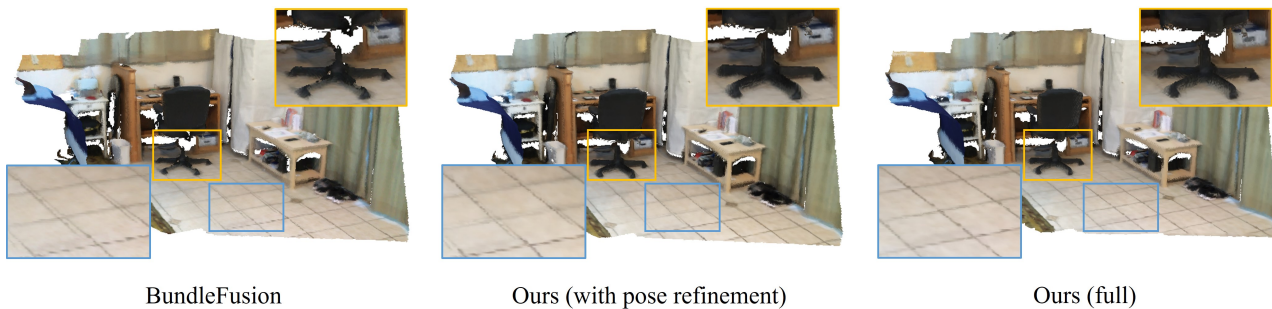


Figure 5. Our method improves the camera alignment over the baseline, as visible in the tiles of the floor. The additional image-plane distortion correction results in straight and aligned edges in the reconstruction.

Method	Pos. error (meters) ↓	Rot. error (degrees) ↓
BundleFusion	0.033	0.571
COLMAP	0.038	0.692
Ours	0.021	0.144

Table 2. Based on our synthetic dataset, we evaluate the average positional and rotational errors of the estimated camera poses. Our method is able to further increase the pose estimation accuracy compared to its BundleFusion initialization.

scene breakdown is given in the supplemental material. In Tab. 1 and Fig. 3, we show that optimizing camera poses reduces geometry misalignment artifacts and improves the overall reconstruction, both quantitatively and qualitatively.

Effect of the image-plane deformation field. To evaluate the effect of the pixel-space deformation field, we initialize the camera with an incorrect focal length and optimize our model with and without the deformation field. Tab. 3 shows that the deformation field mitigates this inaccuracy in the camera’s intrinsic parameters which leads to significantly better reconstruction results compared to the model that does not use the deformation field. Fig. 5 showcases the effects of our camera pose and image-plane deformation field [14]. Blurry frames and sparse features lead to systematic camera pose errors in BundleFusion. Our method improves these camera poses and the camera distortion model, and, thus, is able to better align scene features, resulting in higher reconstruction quality.

Limitations and future work. Similar to other methods that are based on a scene representation which uses a scene-specific MLP, our method runs offline (around 9 hours for 2×10^5 iterations using an NVIDIA RTX 3090). Recent methods that utilize voxel grids to optimize a radiance field [70, 86] have shown significantly faster convergence compared to earlier MLP-based methods and we believe that they would also be good candidates for improving our method. Nonetheless, our proposed method offers a high-quality scene reconstruction which outperforms online fusion approaches. Another limitation is the global MLP

Method	$C-\ell_1$ ↓	IoU ↑	NC ↑	F-score ↑
Ours (w/o IPDF)	0.061	0.266	0.886	0.406
Ours (w/ IPDF)	0.031	0.609	0.911	0.904

Table 3. Ablation of the image-plane deformation field (IPDF) which compensates image space distortions and incorrect intrinsic parameters. The experiment is based on a synthetic scene, where we assume an incorrect focal length of 570 instead of 554.26 (GT).

which stores the entire scene information which comes at the cost of missing high-frequency local detail in very large scenes. Approaches like IF-Nets [7] or Convolutional Occupancy Networks [58] benefit from locally-conditioned MLPs and can be integrated in future work. Finally, our method was designed to handle only opaque surfaces.

5. Conclusion

We have presented a new method for 3D surface reconstruction from RGB-D sequences by introducing a hybrid scene representation that is based on an implicit surface function and a volumetric representation of radiance. This allows us to efficiently incorporate depth observations, while still benefiting from the differentiable volumetric rendering of the original neural radiance field formulation. As a result, we obtain high-quality surface reconstructions, outperforming traditional and learned RGB-D fusion methods. Overall, we believe our work is a stepping stone towards leveraging the success of implicit, differentiable representations for 3D surface reconstruction.

Acknowledgements

This work was supported by a Google Gift Grant, a TUM-IAS Rudolf Mößbauer Fellowship, an NVidia Professorship Award, the ERC Starting Grant Scan2CAD (804724), and the German Research Foundation (DFG) Grant Making Machine Learning on Static and Dynamic 3D Data Practical. We would also like to thank Angela Dai for the video voice-over.

References

- [1] Jonathan T. Barron and Jitendra Malik. Intrinsic scene properties from a single rgb-d image. *CVPR*, 2013. 5
- [2] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J. Davison. Codeslam — learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2
- [3] Jeannette Bohg, Javier Romero, Alexander Herzog, and Stefan Schaal. Robot arm pose estimation through pixel-wise part classification. *ICRA*, 2014. 5
- [4] Aljaž Božič, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. Transformerfusion: Monocular rgb scene reconstruction using transformers. *Proc. Neural Information Processing Systems (NeurIPS)*, 2021. 2
- [5] Eric Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *arXiv*, 2020. 3
- [6] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017. 5
- [7] Julian Chibane, Thiemo Alldieck, and Gerard Pons-Moll. Implicit functions in feature space for 3d shape reconstruction and completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2020. 2, 8
- [8] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 5
- [9] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '96*, page 303–312, New York, NY, USA, 1996. Association for Computing Machinery. 1, 2, 5
- [10] J Czarnowski, T Laidlow, R Clark, and AJ Davison. Deepfactors: Real-time probabilistic dense monocular slam. *IEEE Robotics and Automation Letters*, 5:721–728, 2020. 2
- [11] Manuel Dahnert, Ji Hou, , Matthias Nießner, and Angela Dai. Panoptic 3D scene reconstruction from a single RGB image. 2021. 2
- [12] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, *IEEE*, 2017. 2, 5, 7
- [13] Angela Dai, Christian Diller, and Matthias Nießner. Sg-nn: Sparse generative neural networks for self-supervised scene completion of rgb-d scans. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, *IEEE*, 2020. 2
- [14] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (TOG)*, 36(4):76a, 2017. 2, 3, 4, 5, 7, 8
- [15] Angela Dai, Yawar Siddiqui, Justus Thies, Julien Valentin, and Matthias Nießner. Spsg: Self-supervised photometric scene generation from rgb-d scans. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, *IEEE*, 2021. 2
- [16] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. *arXiv preprint arXiv:2107.02791*, 2021. 2, 3
- [17] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc. *arXiv preprint arXiv:1911.01911*, 2019. 5
- [18] Maximilian Denninger and Rudolph Triebel. 3d scene reconstruction from a single viewport. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 2
- [19] Wei Dong, Qiuyuan Wang, Xin Wang, and Hongbin Zha. PSDF fusion: Probabilistic signed distance function for on-the-fly 3d data fusion and scene reconstruction. *CoRR*, abs/1807.11034, 2018. 2
- [20] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision (ECCV)*, September 2014. 2
- [21] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *2013 IEEE International Conference on Computer Vision*, pages 1449–1456, 2013. 2
- [22] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world’s imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5515–5524, 2016. 2
- [23] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, 2014. 2
- [24] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018. 2
- [25] Y. Furukawa and J. Ponce. Accurate, dense, and robust multi-view stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, 2010. 2
- [26] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 3
- [27] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. *CoRR*, abs/1906.02739, 2019. 2
- [28] Clément Godard, Oisín Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017. 2
- [29] M. Goesele, N. Snavely, B. Curless, H. Hoppe, and S. M. Seitz. Multi-view stereo for community photo collections.

- In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007. 2
- [30] Ankur Handa. Simulating kinect noise for the icl-nuim dataset. <https://github.com/ankurhanda/simkinect>. Accessed: 2021-11-15. 5
- [31] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. *ICRA*, 2014. 5, 7
- [32] Vu Hoang Hiep, Renaud Keriven, Patrick Labatut, and Jean-Philippe Pons. Towards high-resolution large-scale multi-view stereo. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1430–1437. IEEE, 2009. 2
- [33] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011. 1
- [34] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. *arXiv preprint arXiv:1708.05375*, 2017. 2
- [35] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3), July 2013. 5
- [36] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 5
- [37] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), July 2017. 5, 7
- [38] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International journal of computer vision*, 38(3):199–218, 2000. 2
- [39] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth international conference on 3D vision (3DV)*, pages 239–248. IEEE, 2016. 2
- [40] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *arXiv preprint arXiv:2011.13084*, 2020. 3
- [41] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 3
- [42] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes. *ACM Transactions on Graphics*, 38(4):1–14, Jul 2019. 1
- [43] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '87*, page 163–169, New York, NY, USA, 1987. Association for Computing Machinery. 3, 5
- [44] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999. 2
- [45] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *CVPR*, 2021. 3, 4, 5
- [46] N. Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. 3
- [47] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2
- [48] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2, 4, 7
- [49] T. Neff, P. Stadlbauer, M. Parger, A. Kurz, J. H. Mueller, C. R.A. Chaitanya, A. Kaplanyan, and M. Steinberger. DONeRF: Towards Real-Time Rendering of Compact Neural Radiance Fields using Depth Oracle Networks. *Computer Graphics Forum*, 40(4):45–59, 2021. 2, 3
- [50] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011. 2
- [51] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, Jian Chang, and Jian Jun Zhang. Total3DUnderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. pages 55–64, 2020. 2
- [52] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2
- [53] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (TOG)*, 2013. 1, 2
- [54] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *International Conference on Computer Vision*, Oct. 2019. 2
- [55] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *International Conference on Computer Vision (ICCV)*, 2021. 2
- [56] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2
- [57] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo

- Martin Brualla. Deformable neural radiance fields. *arXiv preprint arXiv:2011.12948*, 2020. 3
- [58] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *European Conference on Computer Vision (ECCV)*, Cham, Aug. 2020. Springer International Publishing. 2, 5, 8
- [59] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. *arXiv preprint arXiv:1905.05172*, 2019. 2
- [60] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020. 2
- [61] Nikolay Savinov, Christian Häne, L’ubor Ladický, and Marc Pollefeys. Semantic 3d reconstruction with continuous regularization and ray potentials using a visibility consistency constraint. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5460–5469, 2016. 2
- [62] D. Scharstein, R. Szeliski, and R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, pages 131–140, 2001. 2
- [63] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2, 5
- [64] Johannes Lutz Schönberger, True Price, Torsten Sattler, Jan-Michael Frahm, and Marc Pollefeys. A vote-and-verify strategy for fast spatial verification in image retrieval. In *Asian Conference on Computer Vision (ACCV)*, 2016. 5
- [65] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 5
- [66] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3
- [67] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *arXiv*, 2020. 2, 3, 7
- [68] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019. 2
- [69] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Advances in Neural Information Processing Systems*, 2019. 2
- [70] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. *arXiv preprint arXiv:2111.11215*, 2021. 8
- [71] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. NeuralRecon: Real-time coherent 3D reconstruction from monocular video. *CVPR*, 2021. 2
- [72] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020. 3
- [73] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. In *Computer Graphics Forum*, volume 39, pages 701–727. Wiley Online Library, 2020. 1, 2
- [74] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, Yifan Wang, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, Tomas Simon, Christian Theobalt, Matthias Niessner, Jonathan T. Barron, Gordon Wetzstein, Michael Zollhofer, and Vladislav Golyanik. Advances in neural rendering, 2021. 2
- [75] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018. 2
- [76] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 2, 3, 4
- [77] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. *CVPR*, 2021. 3
- [78] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf-: Neural radiance fields without known camera parameters, 2021. 3
- [79] Silvan Weder, Johannes L. Schönberger, Marc Pollefeys, and Martin R. Oswald. Routedfusion: Learning real-time depth map fusion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2, 5
- [80] Silvan Weder, Johannes L. Schönberger, Marc Pollefeys, and Martin R. Oswald. Neurfusion: Online depth fusion in latent space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3162–3172, June 2021. 2
- [81] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *ICCV*, 2021. 2, 3
- [82] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 767–783, 2018. 2
- [83] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34, 2021. 2

- [84] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems*, 33, 2020. [2](#)
- [85] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. 2020. [3](#)
- [86] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. *arXiv preprint arXiv:2112.05131*, 2021. [8](#)
- [87] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. [3](#)
- [88] Christopher Zach, Thomas Pock, and Horst Bischof. A globally optimal algorithm for robust $tv-l^1$ range image integration. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007. [2](#)
- [89] Yinda Zhang and Thomas Funkhouser. Deep depth completion of a single rgb-d image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 175–185, 2018. [2](#)
- [90] Shuaifeng Zhi, Michael Bloesch, Stefan Leutenegger, and Andrew J. Davison. Scenecode: Monocular dense semantic reconstruction using learned encoded scene representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. [2](#)
- [91] Qian-Yi Zhou and Vladlen Koltun. Color map optimization for 3d reconstruction with consumer depth cameras. 33(4), July 2014. [4](#)
- [92] M. Zollhöfer, P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb. State of the Art on 3D Reconstruction with RGB-D Cameras. *Computer Graphics Forum (Eurographics State of the Art Reports 2018)*, 37(2), 2018. [2](#)

C.3 High-Res Facial Appearance Capture from Polarized Smartphone Images

Copyright Notice

©2023 IEEE. Reprinted, with permission, from
Dejan Azinović, Olivier Maury, Christophe Hery, Matthias Nießner and Justus Thies
High-Res Facial Appearance Capture from Polarized Smartphone Images
IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2023
DOI: 10.1109/CVPR52729.2023.01615

In accordance with the IEEE Thesis/Dissertation Reuse Permissions, we include the accepted version of the original publication [16] in the following.

High-Res Facial Appearance Capture from Polarized Smartphone Images

Dejan Azinović¹ Olivier Maury² Christophe Hery² Matthias Nießner¹ Justus Thies³

¹Technical University of Munich ²Meta Reality Labs ³Max Planck Institute for Intelligent Systems

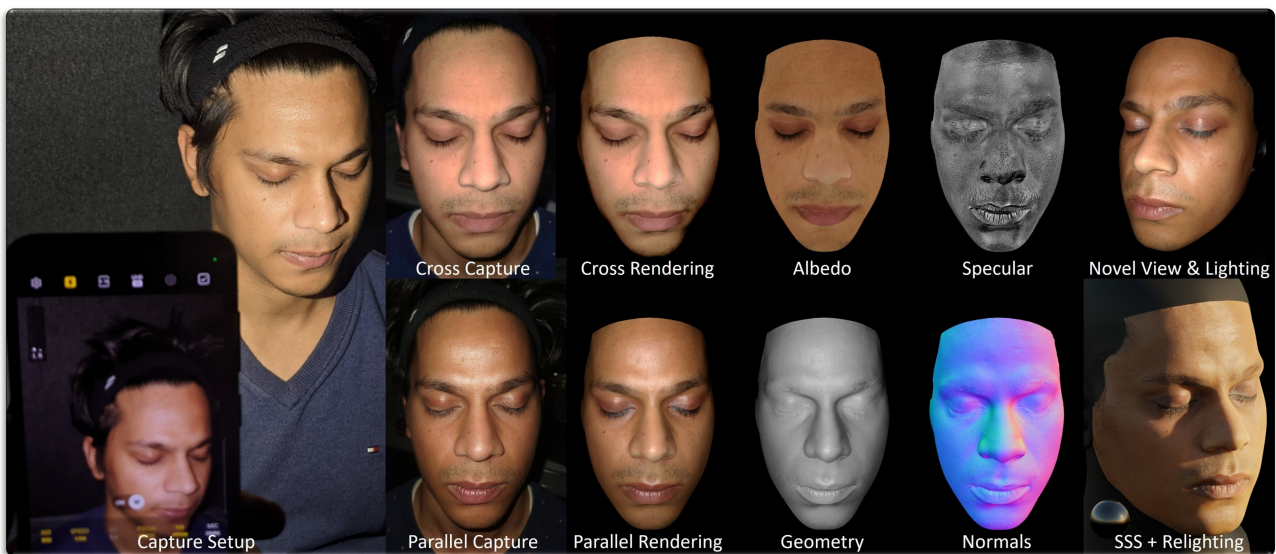


Figure 1. Our method obtains high-resolution skin textures from two RGB input sequences captured with polarization foils attached to a smartphone. The core idea is to separate the skin’s diffuse and specular response by capturing one cross-polarized and one parallel-polarized sequence. We recover an accurate geometry with multi-view stereo, fit a parametric head model, and employ a differentiable rendering strategy to recover 4K diffuse albedo, specular gain and normal maps. These can be used with off-the-shelf rendering software, such as Blender, to produce photo-realistic images from novel views, under novel illumination and with subsurface scattering (SSS).

Abstract

We propose a novel method for high-quality facial texture reconstruction from RGB images using a novel capturing routine based on a single smartphone which we equip with an inexpensive polarization foil. Specifically, we turn the flashlight into a polarized light source and add a polarization filter on top of the camera. Leveraging this setup, we capture the face of a subject with cross-polarized and parallel-polarized light. For each subject, we record two short sequences in a dark environment under flash illumination with different light polarization using the modified smartphone. Based on these observations, we reconstruct an explicit surface mesh of the face using structure from motion. We then exploit the camera and light collocation within a differentiable renderer to optimize the facial textures using an analysis-by-synthesis approach. Our

All data has been captured at the Technical University of Munich.

method optimizes for high-resolution normal textures, diffuse albedo, and specular albedo using a coarse-to-fine optimization scheme. We show that the optimized textures can be used in a standard rendering pipeline to synthesize high-quality photo-realistic 3D digital humans in novel environments.

1. Introduction

In recent years, we have seen tremendous advances in the development of virtual and mixed reality devices. At the same time, the commercial availability of such hardware has led to a massive interest in the creation of ‘digital human’ assets and photo-realistic renderings of human faces. In particular, the democratization to commodity hardware would open up significant potential for asset creation in video games, other home entertainment applications, or immersive teleconferencing systems. However, rendering a

human face realistically in a virtual environment from arbitrary viewpoints with changing lighting conditions is an extremely difficult problem. It involves an accurate reconstruction of the face geometry and skin textures, such as the diffuse albedo, specular gain, or skin roughness. Traditionally, this problem has been approached by recording data in expensive and carefully calibrated light stage capture setups, under expert supervision. We seek to simplify this capture process to allow individuals to reconstruct their own faces, while keeping the quality degradation compared to a light stage to a minimum.

The disentanglement of geometry and material of human faces is an extremely ill-posed problem. Current solutions involve a capture setup with multiple cameras and light sources, with millimeter-accurate calibration. A common approach to disentangling face skin surface from subsurface response is the use of polarization filters [9] in tandem with such expensive capture setups. Given such a carefully calibrated capture setting, one can use differentiable rendering to estimate the individual skin parameters in an analysis-by-synthesis approach. While these methods do produce visually impressive results, they are limited to high-budget production studios.

In this paper, we propose a capture setup consisting of only a smartphone and inexpensive polarization foils, which can be attached to the camera lens and flashlight. Inspired by light stage capture setups, a user captures two sequences of their face, one with perpendicular filter alignment, and one with parallel alignment. This allows for a two-stage optimization, where we first reconstruct a high-resolution diffuse albedo texture of a user's face from the cross-polarized capture, followed by recovery of the specular albedo, normal map, and roughness from the parallel-polarized views. Data is captured in a dark room to avoid requiring pre-computation of an environment map. In addition to visually compelling novel view synthesis and relighting results, our method produces editable textures and face geometry.

In summary, the key contributions of our project are:

- We propose a commodity capture setup that combines a smartphone's camera and flashlight with polarization foils. The polarization allows us to separate diffuse from specular parts, and to reconstruct the user's face textures, such as diffuse albedo, specular albedo and normal maps.
- Our proposed capture setting with the co-located camera and light enables separation of skin properties from illumination, which is of key importance for realistic rendering of faces.
- We propose a coarse-to-fine optimization strategy with mip-mapping, which increases sharpness of the reconstructed appearance textures.

2. Related Work

High-fidelity face appearance capture and reconstruction has received significant attention in the entertainment industry for creating digital humans and more recently in the AR/VR community for generating realistic avatars. In our context, facial appearance reconstruction means recovering a set of high-resolution albedo, specular (gain and roughness) and normal maps. Over the years, physically-based skin scattering models have become ever more sophisticated [6, 26, 53]; however, their input texture quality remains the single most important factor to photo-realism.

Polarization. For some time, polarization has been used to separate specular from diffuse [35, 39, 51]. These techniques rely on the fact that single bounce specular reflection does not alter the polarization state of incoming light. Riviere et al. [41] propose an approach to reconstruct reflectance in uncontrolled lighting, using the inherent polarization of natural illumination, using the inherent polarization of natural illumination. Nogue et al. [38] recover SVBRDF maps of planar objects with near-field display illumination, exploiting Brewster angle properties. Deschaintre et al. [10] use polarization to estimate the shape and SVBRDF of an object with normal, diffuse, specular, roughness and depth maps from a single view. Dave et al. [8] propose a similar approach for multi-view data. In MoRF [48], a studio setup with polarization is used to reconstruct relightable neural radiance fields of a face.

Lightstage capture systems. In their foundational work, Debevec et al. [9] introduced the Lightstage system to capture human face reflectance using a dome equipped with controlled lights, separating the diffuse from the specular component using polarization filters. Follow-up work reconstructs high-resolution normal maps using photometric stereo [52], compensates for motion during the capture [50] and expands the captured area [19].

The proposed capture studios didn't come without limitations, as the lighting environment needed to be tightly controlled, the lighting patterns involved took a relatively long time, and the polarization filters were challenging to set up for multiple cameras and lights. Fyffe et al. [14–17] proposed the use of color gradients and spectral multiplexing to reduce capture time. With the objective of designing a more practical system, Kampouris et al. [24] demonstrate that binary gradients are sufficient for separating diffuse from specular without polarization. Lattas et al. [28] use an array of monitors or tablets for a practical binary gradients capture studio. In line with this thread of research, Gotardo et al. [20] present a multi-view setup for dynamic facial texture acquisition without the need for polarized illumination. Riviere et al. [40] build a similar lightweight system reintroducing polarization without active illumination, and modeling subsurface scattering. This effort was refined to include global illumination and polarization modeling [54]. The

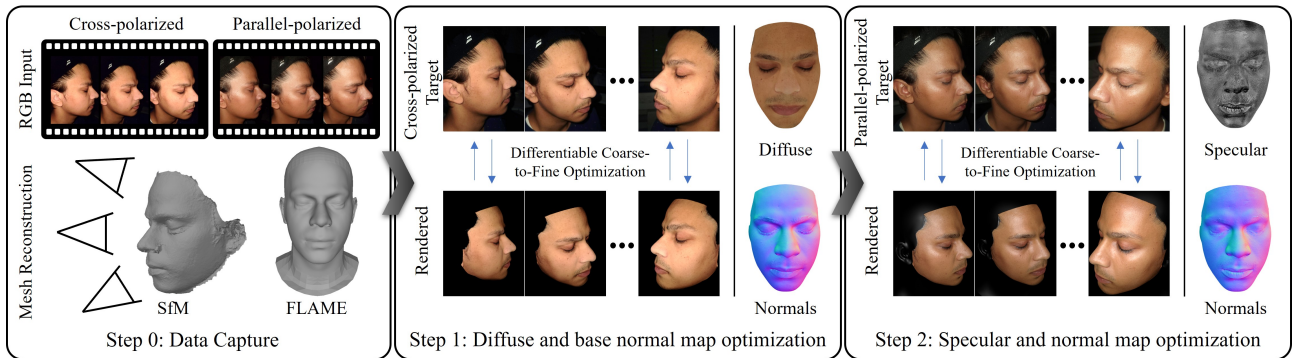


Figure 2. Our optimization has three steps: In step 0, we capture data with a handheld smartphone which is equipped with polarization foils (on the camera, as well as on the flashlight; see Figure 3). We reconstruct the facial geometry and estimate camera poses based on all captured images using structure-from-motion and multi-view stereo. To ensure consistent texture parameterization across different subjects, we non-rigidly fit a FLAME mesh to the scan. In a subsequent photometric optimization step (step 1), we estimate a high-resolution diffuse texture of the skin from the cross-polarized data, as well as an initial normal map. The reconstructed geometry, diffuse and normal map are used as input for step 2 of the optimization. Using the parallel-polarized sequence, we estimate the specular gain and final normal map in a second photometric optimization. In addition, a global skin roughness value is optimized in this step.

proposed solutions deliver impressive visual results, but require expensive and difficult to use hardware. We propose a solution for high-resolution facial texture reconstruction using commodity devices, such as smartphones.

Differentiable rendering. Recent progress in differentiable rendering [4, 47, 56, 57] has led to the development of mature frameworks [23, 27, 37] and a number of methods that try to jointly estimate appearance and lighting [36]. For an overview of differentiable rendering techniques, see [25]. Luan et al. [33] use a co-located camera and light setup to reconstruct shape and material, relying on a differentiable renderer [57] to produce unbiased gradients for shape estimation on an explicit mesh. With the same co-located setup, Zhang et al. [58] improve results by using a hybrid volume radiance field and neural SDFs for the shape estimation. While using a similar capture configuration to our work, the previous techniques focus on shape reconstruction, while we can lean on an accurate prior for the basis of our face shape. Furthermore, by using polarization, we can properly decouple diffuse from specular textures.

Dib et al. [11–13] propose the estimation of face skin textures by modelling the illumination with a virtual light stage, and using a differentiable ray tracer [32]. The method fits a parametric face mask to the observed images and is able to handle self-shadowing, but complex lighting environments can have an impact on separation of lighting and material. Wang et al. [49] propose a capture setup with the sun as the main light source. A FLAME [31] model is fit to the observed data, after which geometry and material are jointly refined using an analysis-by-synthesis approach. As with other methods in uncontrolled lighting, separation of individual textures remains a challenge.

Deep learning-based approaches. A wide range of work proposes learning a neural network from large collections of high-quality light stage data, and subsequently applying the model to new data [5, 22, 29, 30, 34, 42, 55]. Zhang et al. [59] propose learning a neural light transport model from uv-space light and view direction information. At test time, the model generalizes to novel views and lighting. Several other works propose learning neural rendering models, either from single-view [18, 21, 44, 60] or multi-view [45, 46] data, for a range of different applications.

3. Method

We propose a two-step analysis-by-synthesis approach for the estimation of high resolution face textures, as depicted in Figure 2. The user captures two video sequences and a series of photographs of their face under linear-polarized point light illumination using a smartphone. The first sequence has the polarization filters oriented in a perpendicular fashion, i.e., the filter covering the camera lens is perpendicular to the filter covering the smartphone’s flashlight. In accordance with existing literature, we denote this sequence as the cross-polarized sequence. The second video sequence has parallel oriented filters and will be referred to as the parallel-polarized sequence.

We use structure-from-motion and multiview-stereo on all captured frames jointly to compute the camera alignment and reconstruct coarse geometry in form of a triangle mesh. We then non-rigidly fit the FLAME model [31] to the scan and use it as our base geometry model. This fitting helps us avoid noise from the multiview-stereo and provides a consistent UV-parameterization for all subjects. Based on this geometry, we recover the diffuse albedo texture of the

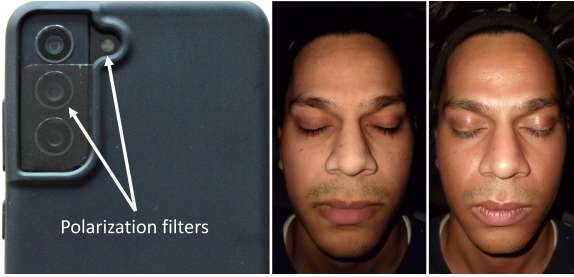


Figure 3. Left to right: smartphone equipped with polarization filters, cross-polarized image (perpendicular filter orientation) and parallel-polarized image (parallel filter orientation).

subject using the cross-polarized data and photo-metric optimization. While keeping the diffuse albedo fixed, we estimate the remaining textures based on the parallel-polarized data. Note that we reconstruct textures using only the photographs, as these capture more detail than the video frames. For the geometry reconstruction, we use all captured data, as we found that this leads to more robust results compared to only using a small set of photographs.

3.1. Capturing Polarized Data with a Smartphone

We capture one cross-polarized and one parallel-polarized video sequence with a smartphone in a dark room, with the smartphone’s flashlight as the only source of illumination. Such a capture setup has the advantage of not requiring optimization of the scene lighting, leading to better separation of appearance and shading. We assume that the flashlight is co-located with the camera lens and that its color is white. We capture a color-checker under both filter orientations to color-calibrate both sequences. This is important, since the filters introduce wavelength-dependent attenuation which tints the color of the light. We use an affine color calibration scheme to compute the corresponding color correction matrix only once, and apply it to all subsequent sequences. Furthermore, since an arbitrary smartphone’s flashlight does not behave like an ideal point light (e.g., due to occlusion by the phone’s cover along grazing directions), we pre-compute a per-pixel light attenuation map, that is multiplied with the final rendered images during optimization. To this end, we put markers on a flat white surface and record a cross-polarized sequence of the surface. We form an optimization problem with the unknowns being the surface’s diffuse texture and the per-pixel light attenuation map. The map is then kept fixed for all future face texture optimizations. We refer to the supplemental material for more detail on this calibration step.

We ensure that all captures have consistent and fixed camera settings: focal length, exposure time and white balance. We capture at 4K resolution and 30fps and select the sharpest frame from every 10-frames window, using variance of the Laplacian as the sharpness metric. In addition to

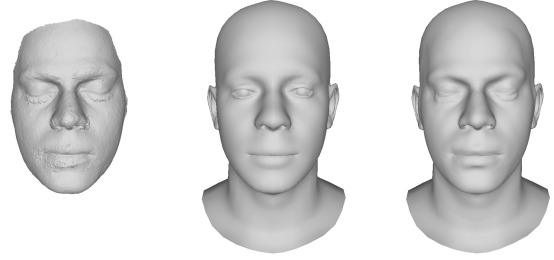


Figure 4. Geometry reconstruction for subject from Figure 3. From left to right: reconstruction via structure from motion, fitted FLAME [31] mesh, ICP-based refinement of the mesh.

the video data, we capture a set of cross-polarized and a set of parallel-polarized photographs to obtain higher-quality data. Since the flash is much brighter for photographs than for videos, we capture the photographs with shorter exposure and lower ISO to roughly match the brightness of the video frames. The entire capture takes about five minutes.

3.2. Geometry Reconstruction

We use Agisoft Metashape [1] on all frames jointly to obtain an initial mesh reconstruction. We provide Metashape with face masks estimated by [61], to make the reconstruction more robust to rigid motion of the head. We then fit the FLAME model [31] to the scanned geometry, first by optimizing the shape parameters of the FLAME face space, and then by an ICP-based as-rigid-as-possible deformation approach (see Figure 4). For the non-rigid deformation, we subdivide the triangles of the face region, to obtain detailed geometry. The resulting mesh is used as the base mesh for the subsequent texture optimizations.

3.3. Rendering Equation & BRDF

We model the skin with a spatially-varying bidirectional reflectance distribution function (SV-BRDF). Assuming a point light source \mathbf{l} in a dark environment, the rendering equation that defines the outgoing radiance $L_o(\mathbf{x}, \omega)$, at point \mathbf{x} with normal \mathbf{n}^\top in direction ω , has the following simplified form:

$$L_o(\mathbf{x}, \omega) = \frac{f(\mathbf{x}, \omega)(\mathbf{n}^\top \omega)L_i(\mathbf{x}, \omega)}{|\mathbf{x} - \mathbf{l}|_2^2}. \quad (1)$$

Here, we make use of the fact that the light direction aligns with the view direction, *i.e.*, $\omega_i = \omega_o = \omega$. The BRDF $f(\mathbf{x}, \omega)$ has a diffuse component f_d , and a specular component f_s . We use the Cook-Torrance [7] BRDF for our specular term:

$$f_s(\mathbf{x}, \omega) = k_s(\mathbf{x}) \frac{D(\omega, \mathbf{n}^\top, \alpha)G(\mathbf{n}, \omega)F(\mathbf{n}, \omega)}{4(\mathbf{n}^\top \omega)(\mathbf{n}^\top \omega)}, \quad (2)$$

with k_s being the spatially-varying specular gain and α a global roughness blend factor for the Blinn-Phong distri-

bution term D of the 2-lobe mix (D_{12} and D_{48}) suggested by [40]. G denotes the geometry term of the Cook-Torrance BRDF model. We use Shlick’s approximation [43] for the Fresnel term F :

$$F(\mathbf{n}, \omega) = F_0 + (1 - F_0)(1 - \mathbf{n}^\top \omega)^5. \quad (3)$$

To model the skin’s diffuse response, we implement the BRDF model proposed by Ashikhmin and Shirley [2, 3], that accounts for the fact that a portion of the light has already scattered before penetrating the skin surface:

$$f_d(\mathbf{x}, \omega) = \frac{28k_d(\mathbf{x})}{23\pi} (1 - F_0) \left(1 - \left(1 - \frac{\mathbf{n}^\top \omega}{2}\right)^5\right)^2, \quad (4)$$

where $F_0 = 0.04$ is the reflectance of the skin at normal incidence. Indirect light bouncing from the capture environment and on the captured face itself might have a significant contribution to pixel intensity at grazing angles, so we also add a Fresnel-modulated ambient term to our BRDF f :

$$f_a(\mathbf{x}, \omega) = k_a(\mathbf{x}) \left(1 - \left(1 - F_0\right) \left(1 - \left(1 - \frac{\mathbf{n}^\top \omega}{2}\right)^5\right)^2\right), \quad (5)$$

with an ambient map k_a which is regularized to be smooth via a total variation loss and close to zero.

Note that using a diffuse scattering model for the optimization is compatible with state-of-the-art physically-based subsurface scattering skin shading [6, 53], as shown in Figure 1. Production-ready subsurface scattering models typically include an albedo inversion stage, which takes a diffuse albedo as input, and converts it to extinction coefficients for the volume rendering random walk.

3.4. Optimization

The objective of the photometric optimization step is to minimize the difference between rendered images \hat{I} and color-corrected target images I :

$$\mathcal{L}(\hat{I}, I) = \left| W \cdot (\hat{I} - I) \right|, \quad (6)$$

with $\hat{I} = \mathcal{M} \cdot L_o$, where \mathcal{M} is the pre-computed light attenuation map, that accounts for uneven light distribution in different directions. We apply a per-pixel loss weight W based on the respective mip level and the angle between viewing direction and normal $\mathbf{n}^\top \omega$ to improve sharpness. Specifically, to ensure that distant or grazing angle observations do not blur the resulting textures, for each pixel that is projected from the target image to texture space, we calculate which mip level l would need to be looked up in classical forward rendering. W is set to $(\mathbf{n}^\top \omega)(1 - l)$ if the pixel corresponds to a mip level below 1, and zero otherwise.

We optimize $\mathcal{L}(\hat{I}, I)$ in two steps, using a coarse-to-fine optimization strategy in each. In the first step, we only use the cross-polarized images to optimize the spatially-varying

Method	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
NLT [59]	31.51	0.96	0.11
NextFace [11]	22.85	0.89	0.31
Ours	32.37	0.96	0.10

Table 1. We compare our method to NLT and NextFace on validation frames over 10 different subjects.

diffuse albedo texture $k_d(\mathbf{x})$ and an initial tangent-space normal map $n(\mathbf{x})$, while assuming $f_s(\cdot) = 0$ for the specular term. In the second step, we fix the diffuse texture and optimize for specular gain $k_s(\mathbf{x})$, specular roughness α , and the final normal map $n(\mathbf{x})$. To account for potentially different light attenuation in the cross and parallel-polarized filter settings, we also optimize per-channel scaling factors for the diffuse texture. The optimization is performed entirely in texture space. In each step, we employ a four-level coarse-to-fine optimization strategy, starting with a texture resolution of 512×512 , and increasing the size by a factor of two after convergence of each level, up to the final resolution of 4096×4096 .

We implement our optimization framework in PyTorch, using nvdiffrast [27] as our differentiable renderer. We optimize on batches of 4 images, using Adam with an initial learning rate $lr_0 = 10^{-3}$ for all parameters at the beginning of every coarse-to-fine step, and updating it to $lr = lr_0 \cdot 10^{-0.001t}$ in every iteration t . We scale the FLAME mesh to unit size and set the light intensity to 10. The total optimization time is about 90 minutes.

4. Results

In this section, we present texture reconstruction and rendering results on several subjects. Figure 5 shows the texture reconstruction on several actors of different ethnicity. Our method is able to reconstruct pore-level detail in the diffuse, specular and normal maps. Further, we evaluate the quality of our reconstructed textures by rendering the mesh from novel views and under novel illumination. Figure 6 shows that our method faithfully reconstructs the skin’s appearance under novel views and lighting.

Comparison to state of the art. We perform both a qualitative and quantitative evaluation of our method and compare to state-of-the-art methods for relighting and texture reconstruction. During optimization, we hold out a validation frame on which we compute image metrics.

Neural Light Transport. Neural Light Transport [59] is a deep learning-based method that takes as input pre-computed diffuse base, light-cosine and view-cosine uv-space maps. The diffuse base is computed as the average of all observations. The cosine maps contain per-texel cosines of the angles between the normal vector and the light or view vector. Based on these inputs, as well as nearest neighbor observations, a neural network learns to predict the final

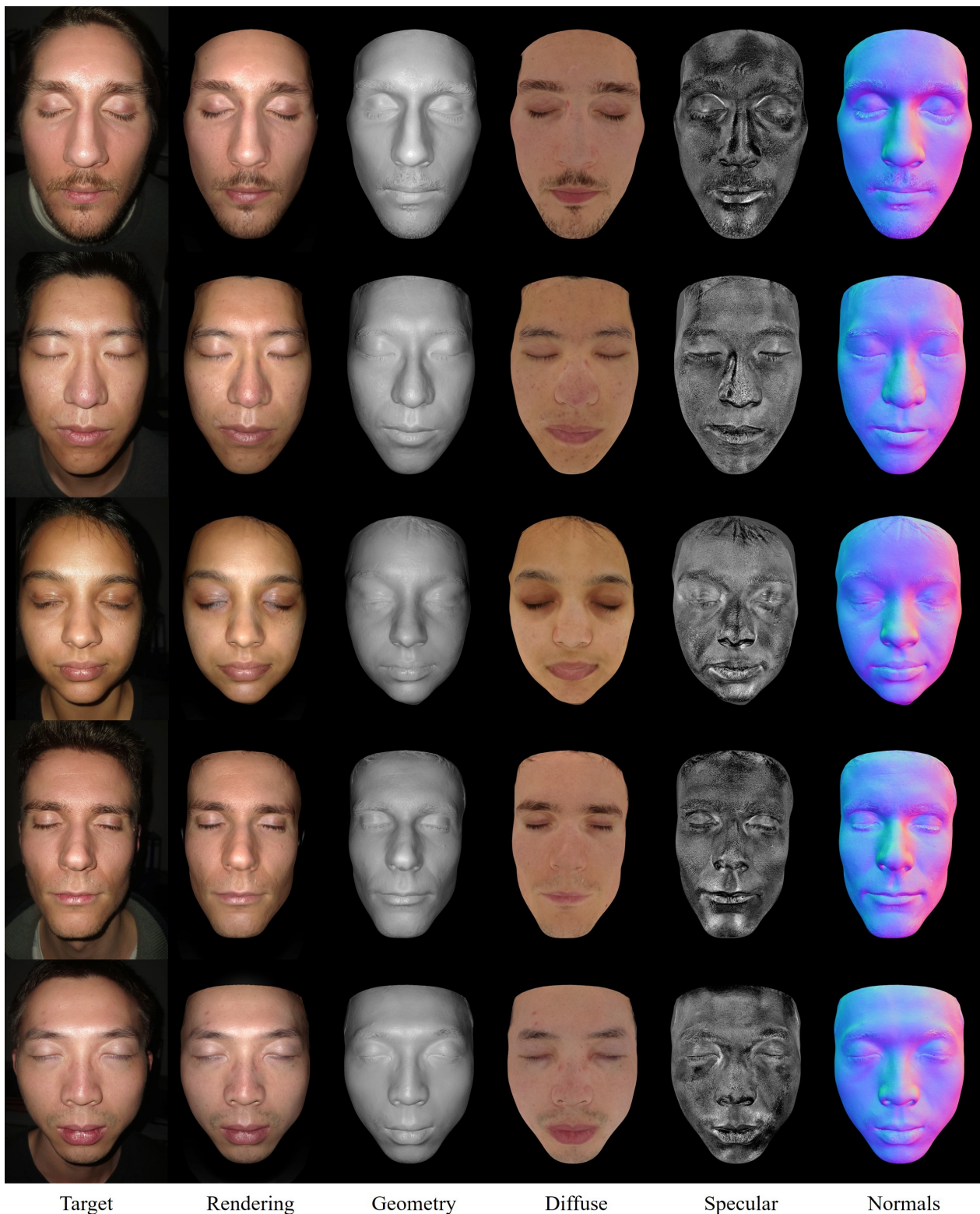


Figure 5. We show skin texture reconstructions of several actors of different skin type. The rendered images closely match the reference target images, and we achieve good separation of diffuse and specular textures.

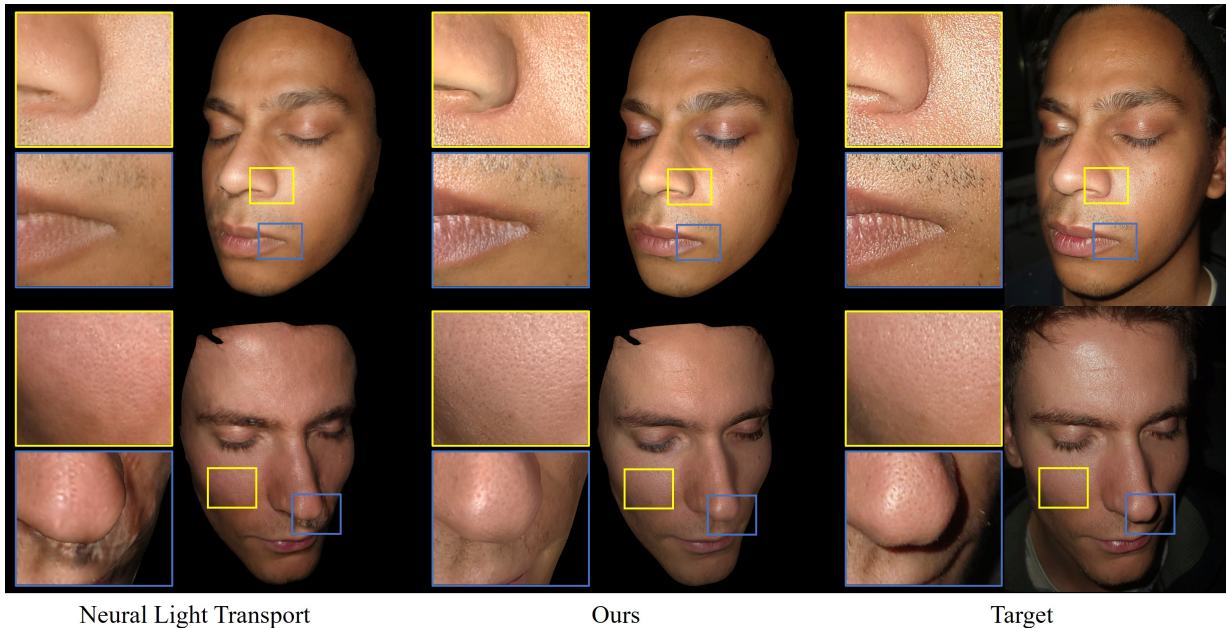


Figure 6. We evaluate on a validation frame from a novel viewpoint and with novel lighting that was held out during the optimization. As visible in the crop regions, our method is able to synthesize sharper texture details and specular highlights compared to NLT [59].

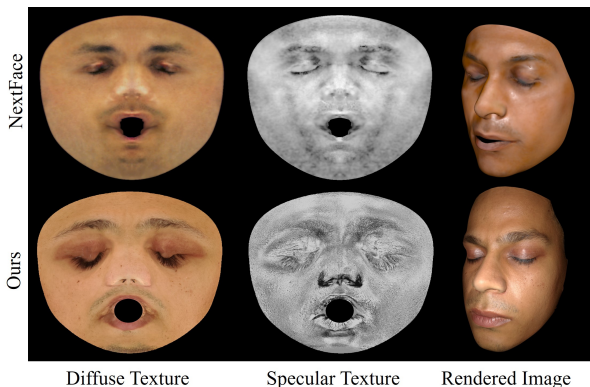


Figure 7. Comparison to NextFace [11] in terms of reconstructed appearance textures.

shaded image. Since the method does not take light intensity and falloff into account, we optimize the rendered validation image’s brightness to match the target as closely as possible, before computing the rendering error.

NextFace. NextFace [11–13] first fits a morphable face model to the input frames, then estimates the face shape, pose, lighting, statistical diffuse and specular albedos by minimizing a photo-consistency loss between the target image and a ray traced estimate. In a final step, the statistical albedos are refined on a per-texel basis. We conducted several experiments with different illumination conditions and number of frames, including an experiment on our data for which we replaced the spherical harmonics lighting representation with a small area light, modelling our flashlight.

As shown in Table 1, our approach achieves favorable image metrics. Figure 6 compares our method to NLT on novel lighting and viewpoint. NLT closely matches the target by using nearby camera views, but specular highlights are often blurry, and the low number of training views results in the model producing artifacts in shadowed areas. We obtained the best NextFace results in an experiment with uniform illumination using three frames that cover the whole face region. As shown in Figure 7, inaccuracies in the face model fitting lead to somewhat blurry textures. This issue is exacerbated by adding more frames. Using fewer frames degraded the separation of the diffuse and specular textures. Our method is able to overcome these issues by accurately fitting a geometric model to the input data and by using polarization to separate the individual textures.

Ablation Studies. We conduct ablation studies to justify our choice of capture setup and training parameters. In Figure 8, we show that accounting for the direction-dependent light attenuation of a smartphone’s flashlight leads to an overall lower error in the re-rendered images. In the same figure, we also show the importance of accounting for the Fresnel effect when reconstructing the diffuse texture. A purely Lambertian BRDF will not be able to model the skin’s diffuse response at all angles. In Figure 9, we show that optimizing textures without cross-polarization will leak specular information into the diffuse texture.

Coarse-to-fine optimization and mipmapping. Pixels of the target images have different footprints in uv-space, depending on distance and angle between camera and surface.

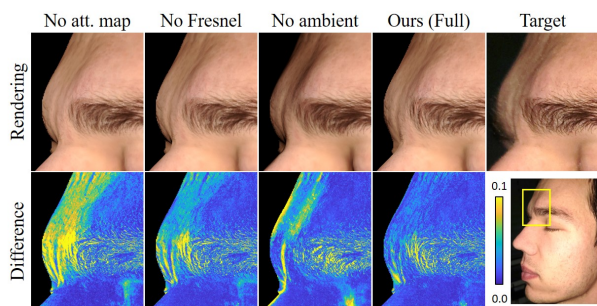


Figure 8. Ignoring the angle-dependent flashlight attenuation, the Fresnel effect, or the ambient light leads to an incorrect reconstruction, that can no longer reproduce the shading from all views. We account for these effects to closely match the target data.

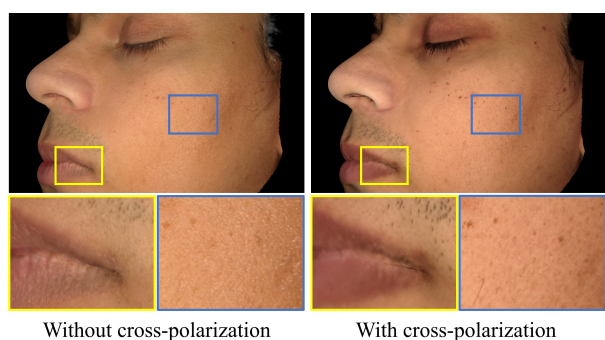


Figure 9. We compare joint optimization of all textures to our full approach on a purely diffuse render. Optimizing jointly leaks specular and normal map information into the diffuse texture.

Weighting the loss of each pixel equally leads to blur in the reconstruction. Optimizing coarse-to-fine, where at each resolution we use only pixels with the corresponding uv-space footprint, helps us reconstruct additional detail in the textures. Figure 10 shows a comparison between our full approach and a direct optimization of the highest resolution texture. We additionally show the decrease in quality when optimizing only on video frames (w/o photographs).

Runtime and memory consumption. Including 1 hour spent on MVS, our method needs about 2.5h to reconstruct a face. Photo-metric skin texture reconstruction takes about 90 minutes on an Nvidia RTX A6000. We reconstruct facial geometry with Metashape using an average of 420 video frames and 70 photographs. At a texture resolution of 4096×4096 and target image resolution of 3840×2160 , the photo-metric optimization requires 30GB of GPU memory. In comparison, NLT takes about 10h and NextFace about 6h given the same number of frames.

Discussion & Limitations. Our method reconstructs high-quality face textures with a low-cost capture routine. However, it is restricted to static expressions, i.e., it does not handle dynamically changing face geometry and textures. An avenue for future research is the reconstruction of dy-

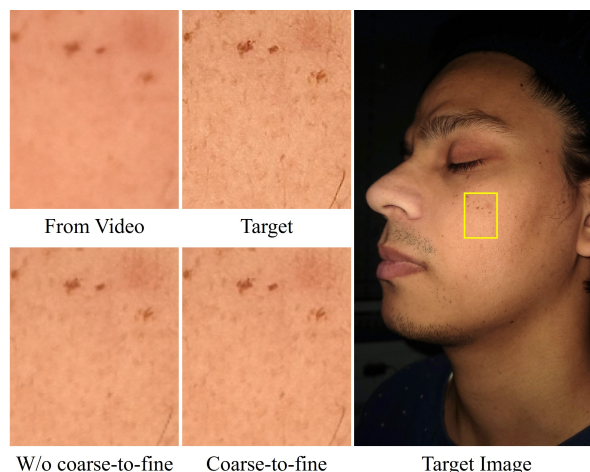


Figure 10. To increase sharpness, we optimize from photographs, instead of video frames. Using only pixels of the appropriate mip level in a coarse-to-fine approach further enhances results.

amic expressions by fitting a parametric model with consistent mesh topology to each frame, and optimizing over the entire non-rigid sequence. Our method does not explicitly handle global illumination. A differentiable path tracer could potentially improve results in the concavities of the eye region. As we assume a static face with closed mouth and closed eyes, we only recover the skin area of a face. Eyes, mouth interior and hair are a subject of future work.

5. Conclusion

We have presented a practical and inexpensive method of capturing high-resolution textures of a person’s face by coupling commodity smartphones and polarization foils. The co-location of the camera lens and light source allows us to reduce the problem complexity and separate material from shading information. As a result, we obtain high-resolution textures of the skin area of the human face. We believe that polarization is a powerful tool for material recovery in the real world, and future smartphones could benefit from including filters directly in the hardware. Overall, we believe that our work is a stepping stone towards democratizing the creation of digital human face assets by making it more accessible to smaller production studios or individual users.

Acknowledgements

Dejan Azinović’s contribution was supported by the ERC Starting Grant Scan2CAD (804724), the German Research Foundation (DFG) Grant “Making Machine Learning on Static and Dynamic 3D Data Practical”, and the German Research Foundation (DFG) Research Unit “Learning and Simulation in Visual Computing”. We would also like to thank Angela Dai for the video voice over and Simon Giebenhain for help with the FLAME fitting.

References

- [1] Agisoft. *Agisoft Metashape Professional (Version 1.8.4)*. Agisoft, 2022. 4
- [2] Michael Ashikhmin and Peter Shirley. An anisotropic phong light reflection model. *University of Utah Computer Science Technical Report*, 2000. 5
- [3] Michael Ashikhmin and Peter Shirley. An anisotropic phong brdf model. *Journal of Graphics Tools* 5 (2), 25–32, 2002. 5
- [4] Dejan Azinovic, Tzu-Mao Li, Anton Kaplanyan, and Matthias Niessner. Inverse path tracing for joint material and lighting estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3
- [5] Sai Bi, Stephen Lombardi, Shunsuke Saito, Tomas Simon, Shih-En Wei, Kevyn Mcphail, Ravi Ramamoorthi, Yaser Sheikh, and Jason Saragih. Deep relightable appearance models for animatable faces. *ACM Trans. Graph.*, 40(4), jul 2021. 3
- [6] Matt Jen-Yuan Chiang, Peter Kutz, and Brent Burley. Practical and controllable subsurface scattering for production path tracing. In *ACM SIGGRAPH 2016 Talks*, pages 1–2. 2016. 2, 5
- [7] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Trans. Graph.*, 1(1):7–24, jan 1982. 4
- [8] Akshat Dave, Yongyi Zhao, and Ashok Veeraraghavan. Pandora: Polarization-aided neural decomposition of radiance. *arXiv preprint arXiv:2203.13458*, 2022. 2
- [9] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 145–156, 2000. 2
- [10] Valentin Deschaintre, Yiming Lin, and Abhijeet Ghosh. Deep polarization imaging for 3d shape and svbrdf acquisition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021. 2
- [11] Abdallah Dib, Junghyun Ahn, Cedric Thebault, Philippe-Henri Gosselin, and Louis Chevallier. S2f2: Self-supervised high fidelity face reconstruction from monocular image. *arXiv preprint arXiv:2203.07732*, 2022. 3, 5, 7
- [12] Abdallah Dib, Gaurav Bharaj, Junghyun Ahn, Cédric Thébault, Philippe Gosselin, Marco Romeo, and Louis Chevallier. Practical face reconstruction via differentiable ray tracing. In *Computer Graphics Forum*, volume 40, pages 153–164. Wiley Online Library, 2021. 3, 7
- [13] Abdallah Dib, Cedric Thebault, Junghyun Ahn, Philippe-Henri Gosselin, Christian Theobalt, and Louis Chevallier. Towards high fidelity monocular face reconstruction with rich reflectance using self-supervised learning and ray tracing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12819–12829, 2021. 3, 7
- [14] Graham Fyffe. Cosine lobe based relighting from gradient illumination photographs. In *SIGGRAPH'09: Posters*, pages 1–1. 2009. 2
- [15] Graham Fyffe. Single-shot photometric stereo by spectral multiplexing. In *ACM SIGGRAPH ASIA 2010 Sketches*, pages 1–2. 2010. 2
- [16] Graham Fyffe and Paul Debevec. Single-shot reflectance measurement from polarized color gradient illumination. In *2015 IEEE International Conference on Computational Photography (ICCP)*, pages 1–10. IEEE, 2015. 2
- [17] Graham Fyffe, Paul Graham, Borom Tunwattanapong, Abhijeet Ghosh, and Paul Debevec. Near-instant capture of high-resolution facial geometry and reflectance. In *Computer Graphics Forum*, volume 35, pages 353–363. Wiley Online Library, 2016. 2
- [18] Guy Gafni, Justus Thies, Michael Zollhöfer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8649–8658, June 2021. 3
- [19] Abhijeet Ghosh, Graham Fyffe, Borom Tunwattanapong, Jay Busch, Xueming Yu, and Paul Debevec. Multiview face capture using polarized spherical gradient illumination. *ACM Trans. Graph.*, 30(6):1–10, dec 2011. 2
- [20] Paulo Gotardo, Jérémy Riviere, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. Practical dynamic facial appearance modeling and acquisition. *ACM Trans. Graph.*, 37(6), dec 2018. 2
- [21] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. *arXiv preprint arXiv:2112.01554*, 2021. 3
- [22] Loc Huynh, Weikai Chen, Shunsuke Saito, Jun Xing, Koki Nagano, Andrew Jones, Paul Debevec, and Hao Li. Mesoscopic facial geometry inference using deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8407–8416, 2018. 3
- [23] Wenzel Jakob, Sébastien Speierer, Nicolas Roussel, and Dario Vicini. Dr. jit: a just-in-time compiler for differentiable rendering. *ACM Transactions on Graphics (TOG)*, 41(4):1–19, 2022. 3
- [24] Christos Kampouris, Stefanos Zafeiriou, and Abhijeet Ghosh. Diffuse-specular separation using binary spherical gradient illumination. In *EGSR (EI&I)*, pages 1–10, 2018. 2
- [25] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*, 2020. 3
- [26] Oliver Klehm, Fabrice Rousselle, Marios Papas, Derek Bradley, Christophe Hery, Bernd Bickel, Wojciech Jarosz, and Thabo Beeler. Recent advances in facial appearance capture. In *Computer Graphics Forum*, volume 34, pages 709–733. Wiley Online Library, 2015. 2
- [27] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020. 3, 5
- [28] Alexandros Lattas, Yiming Lin, Jayanth Kannan, Ekin Ozturk, Luca Filipi, Giuseppe Claudio Guarnera, Gaurav Chawla, and Abhijeet Ghosh. Practical and scalable desktop-based high-quality facial capture. 2022. 2
- [29] Alexandros Lattas, Stylianos Moschoglou, Baris Gecer, Stylianos Ploumpis, Vasileios Triantafyllou, Abhijeet

- Ghosh, and Stefanos Zafeiriou. Avatarme: Realistically renderable 3d facial reconstruction” in-the-wild”. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 760–769, 2020. 3
- [30] Jiaman Li, Zhengfei Kuang, Yajie Zhao, Mingming He, Karl Bladin, and Hao Li. Dynamic facial asset and rig generation from a single scan. *ACM Trans. Graph.*, 39(6), nov 2020. 3
- [31] Tianye Li, Timo Bolkart, Michael J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 3, 4
- [32] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, 37(6):222:1–222:11, 2018. 3
- [33] Fujun Luan, Shuang Zhao, Kavita Bala, and Zhao Dong. Unified shape and svbrdf recovery using differentiable monte carlo rendering. In *Computer Graphics Forum*, volume 40, pages 101–113. Wiley Online Library, 2021. 3
- [34] Abhimitra Meka, Christian Haene, Rohit Pandey, Michael Zollhöfer, Sean Fanello, Graham Fyffe, Adarsh Kowdle, Xueming Yu, Jay Busch, Jason Dourgarian, et al. Deep reflectance fields: high-quality facial reflectance field inference from color gradient illumination. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019. 3
- [35] Volker Müller. Polarization-based separation of diffuse and specular surface-reflection. In *Mustererkennung 1995*, pages 202–209. Springer, 1995. 2
- [36] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting triangular 3d models, materials, and lighting from images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8280–8290, 2022. 3
- [37] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)*, 38(6):1–17, 2019. 3
- [38] Emilie Nogue, Yiming Lin, and Abhijeet Ghosh. Polarization-imaging Surface Reflectometry using Near-field Display. In Abhijeet Ghosh and Li-Yi Wei, editors, *Eurographics Symposium on Rendering*. The Eurographics Association, 2022. 2
- [39] Stefan Rahmann and Nikos Canterakis. Reconstruction of specular surfaces using polarization imaging. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I. IEEE, 2001. 2
- [40] Jérémy Riviere, Paulo Gotardo, Derek Bradley, Abhijeet Ghosh, and Thabo Beeler. Single-shot high-quality facial geometry and skin appearance capture. *ACM Trans. Graph.*, 39(4), jul 2020. 2, 5
- [41] Jérémy Riviere, Ilya Reshetouski, Luka Filipi, and Abhijeet Ghosh. Polarization imaging reflectometry in the wild. *ACM Transactions on Graphics (TOG)*, 36(6):1–14, 2017. 2
- [42] Shunsuke Saito, Lingyu Wei, Liwen Hu, Koki Nagano, and Hao Li. Photorealistic facial texture inference using deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 3
- [43] Christophe Schlick. An inexpensive brdf model for physically-based rendering. *Computer Graphics Forum*, 13(3):233–246, 1994. 5
- [44] Soumyadip Sengupta, Brian Curless, Ira Kemelmacher-Shlizerman, and Steven M. Seitz. A light stage on every desk. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2420–2429, October 2021. 3
- [45] Artem Sevastopolsky, Savva Ignatiev, Gonzalo Ferrer, Evgeny Burnaev, and Victor Lempitsky. Relightable 3d head portraits from a smartphone video. *arXiv preprint arXiv:2012.09963*, 2020. 3
- [46] Tiancheng Sun, Kai-En Lin, Sai Bi, Zexiang Xu, and Ravi Ramamoorthi. Nelf: Neural light-transport field for portrait view synthesis and relighting. In *Eurographics Symposium on Rendering*, 2021. 3
- [47] Delio Vicini, Sébastien Speierer, and Wenzel Jakob. Path replay backpropagation: differentiating light paths using constant memory and linear time. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 3
- [48] Daoye Wang, Prashanth Chandran, Gaspard Zoss, Derek Bradley, and Paulo Gotardo. Morf: Morphable radiance fields for multiview neural head modeling. In *ACM SIGGRAPH 2022 Conference Proceedings, SIGGRAPH '22*, New York, NY, USA, 2022. Association for Computing Machinery. 2
- [49] Yifan Wang, Aleksander Holynski, Xiuming Zhang, and Xuaner Cecilia Zhang. Sunstage: Portrait reconstruction and relighting using the sun as a light stage. *arXiv preprint arXiv:2204.03648*, 2022. 3
- [50] Cyrus A Wilson, Abhijeet Ghosh, Pieter Peers, Jen-Yuan Chiang, Jay Busch, and Paul Debevec. Temporal upsampling of performance geometry using photometric alignment. *ACM Transactions on Graphics (TOG)*, 29(2):1–11, 2010. 2
- [51] Lawrence B Wolff and Terrance E Boult. Constraining object features using a polarization reflectance model. *Phys. Based Vis. Princ. Pract. Radiom.*, 1:167, 1993. 2
- [52] Robert J Woodham. Photometric method for determining surface orientation from multiple images. *Optical engineering*, 19(1):139–144, 1980. 2
- [53] Magnus Wrenninge, Ryusuke Villemin, and Christophe Hery. Path traced subsurface scattering using anisotropic phase functions and non-exponential free flights. Technical report, Tech. Rep. 17-07, Pixar. <https://graphics.pixar.com/library/...>, 2017. 2, 5
- [54] Yingyan Xu, Jérémy Riviere, Gaspard Zoss, Prashanth Chandran, Derek Bradley, and Paulo Gotardo. Improved lighting models for facial appearance capture. 2022. 2
- [55] Shugo Yamaguchi, Shunsuke Saito, Koki Nagano, Yajie Zhao, Weikai Chen, Kyle Olszewski, Shigeo Morishima, and Hao Li. High-fidelity facial reflectance and geometry inference from an unconstrained image. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 3
- [56] Tizian Zeltner, Sébastien Speierer, Iliyan Georgiev, and Wenzel Jakob. Monte carlo estimators for differential light

- transport. *ACM Transactions on Graphics (TOG)*, 40(4):1–16, 2021. 3
- [57] Cheng Zhang, Zihan Yu, and Shuang Zhao. Path-space differentiable rendering of participating media. *ACM Transactions on Graphics (TOG)*, 40(4):1–15, 2021. 3
- [58] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5565–5574, 2022. 3
- [59] Xiuming Zhang, Sean Fanello, Yun-Ta Tsai, Tiancheng Sun, Tianfan Xue, Rohit Pandey, Sergio Orts-Escolano, Philip Davidson, Christoph Rhemann, Paul Debevec, Jonathan T. Barron, Ravi Ramamoorthi, and William T. Freeman. Neural light transport for relighting and view synthesis. *ACM Trans. Graph.*, 40(1), jan 2021. 3, 5, 7
- [60] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C. Bühler, Xu Chen, Michael J. Black, and Otmar Hilliges. I M avatar: Implicit morphable head avatars from videos. In *IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 13545–13555, June 2022. 3
- [61] Yinglin Zheng, Hao Yang, Ting Zhang, Jianmin Bao, Dongdong Chen, Yangyu Huang, Lu Yuan, Dong Chen, Ming Zeng, and Fang Wen. General facial representation learning in a visual-linguistic manner. *arXiv preprint arXiv:2112.03109*, 2021. 4

Acronyms

1D, 2D, 3D, ...	n spatial dimentionions.
Adam	Adaptive Moment Estimation.
AR	Augmented Reality.
ARAP	As-Rigid-As-Possible.
BRDF	Bidirectional Reflectance Distribution Function.
BSDF	Bidirectional Scattering Distribution Function.
BSSRDF	Bidirectional Scattering Surface Reflectance Distribution Function.
BTDF	Bidirectional Transmittance Distribution Function.
CMOS	Complementary metal-oxide-semiconductor.
CNN	Convolutional Neural Network.
CPU	Central Processing Unit.
GPU	Graphics Processing Unit.
ICP	Iterative Closest Point.
IoU	Intersection-over-Union.
IR	Infra-red.
ISO	International Organization for Standardization.
MIS	Multiple Importance Sampling.
MLP	Multi-Layer Perception.
MSE	Mean Squared Error.
MVS	Multi-View Stereo.
NC	Normal Consistency.
NeRF	Neural Radiance Field.
OpenGL	Open Graphics Library.
ReLU	Rectified Linear Unit.
RGB	Red, Green, Blue.

Appendix

RGB-D	Red, Green, Blue, Depth.
SDF	Signed Distance Field.
SfM	Structure from Motion.
SGD	Stochastic Gradient Descent.
SH	Spherical Harmonics.
SIFT	Scale-invariant Feature Transform.
SLAM	Simultaneous Localization and Mapping.
SSS	Subsurface Scattering.
SVBRDF	Spatially-varying Bidirectional Reflectance Distribution Function.
SVSH	Spatially-varying Spherical Harmonics.
TSDF	Truncated Signed Distance Field.
UDF	Unsigned Distance Field.
VR	Virtual Reality.

List of Tables

3.1	Quantitative evaluation for synthetic data. We measure the L1 loss with respect to the rendering error and the estimated albedo parameters. Note that our approach achieves a significantly lower error on both metrics. . .	47
3.2	We compare the relative error between the estimated diffuse albedo for two objects. We outperform LIME even though our method is not restricted to the estimation of only a single material at a time.	52
4.1	Reconstruction results on a dataset of 10 synthetic scenes. The Chamfer ℓ_1 distance, normal consistency and the F-score [28] are computed between point clouds sampled with a density of 1 point per cm^2 , using a threshold of 5 cm for the F-score. We voxelize the mesh to compute the intersection-over-union (IoU) between the predictions and ground truth.	64
4.2	Based on our synthetic dataset, we evaluate the average positional and rotational errors of the estimated camera poses. Our method is able to further increase the pose estimation accuracy compared to its BundleFusion initialization.	66
4.3	Ablation of the image-plane deformation field (IPDF) which compensates image space distortions and incorrect intrinsic parameters. The experiment is based on a synthetic scene, where we assume an incorrect focal length of 570 instead of 554.26 (GT).	67
4.4	We list the number of samples S'_c and the ray length in meters that were used to reconstruct each of the ScanNet scenes and the synthetic scenes. Note that these settings are dependent on the scene size.	69
4.5	Source and license information of the used data.	70
4.6	Detailed reconstruction results for Figure 4 from the main paper. Our method reconstructs geometry visible only in color images, leading to significantly better reconstruction results in scenes with geometry which is not captured by the depth sensor.	71
4.7	Impact of the truncation region width on reconstruction quality.	72
4.8	We compare the quality of our reconstruction on several synthetic scenes for which ground truth data is available. The Chamfer ℓ_1 distance, normal consistency and the F-score [28] are computed between point clouds sampled with a density of 1 point per cm^2 . We use a threshold of 5 cm for the F-score. We further voxelize each mesh to compute the intersection-over-union (IoU) between the predictions and ground truth.	75

Appendix

- 4.9 We compare the quality of our reconstruction on several synthetic scenes for which ground truth data is available. The Chamfer ℓ_1 distance, normal consistency and the F-score [28] are computed between point clouds sampled with a density of 1 point per cm^2 . We use a threshold of 5 cm for the F-score. We further voxelize each mesh to compute the intersection-over-union (IoU) between the predictions and ground truth. 76
- 5.1 We compare our method to NLT and NextFace on validation frames over 10 different subjects. 94

List of Figures

2.1	Pinhole camera model. The image plane, i.e., image sensor, is located at a distance f away from the aperture, whose location corresponds to the principle point (c_x, c_y) on the image plane. The 3D scene is projected through the aperture onto the image plane. In case of a virtual camera (e.g., in a computer graphics rendering pipeline), the image plane is defined to be at distance f in front of the aperture, as opposed to behind the aperture. In that case, the projected image is no longer flipped w.r.t. the 3D scene.	10
2.2	The Microsoft Azure Kinect is an RGB-D camera that consists of an RGB sensor, IR emitter and a depth sensor that can detect the reflected IR light.	12
2.3	Example RGB and depth frame from the ScanNet [1] dataset. The RGB and depth cameras have already been calibrated so that each pixel from the RGB frame corresponds to the depth pixel at the same image coordinates in the depth frame. Furthermore, the captured raw images suffer from lens distortion. In this figure, the images have been undistorted (as visible by the undistortion artifacts on the border of both images), making it possible to use the pinhole camera model. The raw depth data is noisy and has missing regions, primarily in very dark regions, on thing structures and along object edges.	13
2.4	Triangle mesh of the Stanford bunny.	14
2.5	Point cloud of a synthetic scene. Back-projecting values from a depth map gives us positions in 3D space.	15
2.6	Low-resolution occupancy grid of the Stanford bunny.	16
2.7	Visualization of the sphere tracing algorithm. Starting from the camera origin, we march along a ray towards the nearest surface. At each step, the marching distance is equal to the distance to the nearest surface. The marching stops once the distance falls below a certain threshold.	17
2.8	Reconstructed room from the ScanNet [1] dataset.	19
2.9	The feature matching stage tries to match distinctive features from multiple images. These matches are subsequently used to compute the position of the features in 3D space.	20
2.10	Screened Poisson Surface Reconstruction [17] reconstructs a mesh from the point cloud produced by COLMAP.	21
2.11	An image of the ‘whiteroom’ synthetic scene rendered in Blender [21]. . .	23
2.12	Visualization of the full OpenGL rasterization pipeline [31]. Programmable stages of the pipeline are depicted in blue. Stages with a dashed line border are optional.	24

2.13	Illustration of the path tracing algorithm. A ray is cast from the camera into the scene. At the intersection with the scene, a bounce occurs. The direction of the bounced ray is sampled based on the material (BRDF sampling) at the intersection point. This is repeated until the ray either hits the light sources, leaves the scene bounds (in which case an environment illumination map can be sampled) or the maximum number of bounces is reached. To increase performance, the light source can also be directly sampled and a <i>shadow ray</i> cast towards its position.	25
2.14	NeRF pipeline as visualized in the original paper [8]. A number of points is sampled on rays that are cast through each pixel in a set of input images (a). The position of the points, along with the corresponding ray direction, is passed to the MLP F_θ to obtain per-sample RGB color and density σ values (b). Based on this output, a per-pixel color value is computed using volume rendering (c). Finally, a loss between the rendered and input image is calculated (d) and gradients propagated back to update the weights of F_θ	29
3.1	Our Inverse Path Tracing algorithm takes as input a 3D scene and up to several RGB images (left), and estimates material as well as the lighting parameters of the scene. The main contribution of our approach is the formulation of an end-to-end differentiable inverse Monte Carlo renderer which is utilized in a nested stochastic gradient descent optimization. . .	34
3.2	Inserting virtual objects in real 3D scenes; the estimated lighting and material parameters of our approach enable convincing image compositing in AR settings.	35
3.3	Overview of our pipeline. Given (a) a set of input photos from different views, along with (b) an accurate geometry scan and proper segmentation, we reconstruct the material properties and illumination of the scene, by iteratively (c) rendering the scene with path tracing, and (d) backpropagating to the material and illumination parameters in order to update them. After numerous iterations, we obtain the (e) reconstructed material and illumination.	37
3.4	Methods based on spherical harmonics have difficulties handling sharp shadows or lighting changes due to the distant illumination assumption. A physically based method, such as Inverse Path Tracing, correctly reproduces these effects.	40
3.5	Evaluation on synthetic scenes. Three scenes have been rendered from different views with both direct and indirect lighting (right). An approximation of the albedo lighting with spatially-varying spherical harmonics is shown (left). Our method is able to detect the light source even though it was not observed in any of the views (middle). Notice that we are able to reproduce sharp lighting changes and shadows correctly. The albedo is also closer to the ground truth albedo.	42

3.6	Inverse Path Tracing is able to correctly detect the light emitting object (top). The ground truth rendering and our estimate is shown on the bottom. Note that this view was not used during optimization.	43
3.7	We can resolve object textures by optimizing for the unknown parameters per triangle. Higher resolution textures can be obtained by further subdividing the geometry.	44
3.8	Inverse Path Tracing is agnostic to the underlying BRDF; e.g., here, in a specular case, we are able to correctly estimate both the albedo and the roughness of the objects. The ground truth rendering and our estimate is shown on top, the albedo in the middle and the specular map on the bottom.	45
3.9	Convergence with respect to the number of paths used to estimate the pixel color. If this is set too low, the algorithm will fail.	46
3.10	Convergence with respect to distributing the available path samples budget between pixel color and derivatives. It is best to keep the number of paths high for pixel color estimation and low for derivative estimation. . .	46
3.11	Evaluation on real scenes: (right) input is 3D scanned geometry and photographs. We employ object instance segmentation to estimate the emission and the average albedo of every object in the scene. Our method is able to optimize for the illumination and shadows. Other methods usually do not take occlusions into account and fail to model shadows correctly. Views 1 and 2 of Scene 2 show that if the light emitters are not present in the input geometry, our method gives an incorrect estimation.	48
3.12	Mixed-reality setting: we insert two new 3D objects (chairs) into an existing 3D scene. Our goal is to find a consistent lighting between the existing and newly-inserted content. In the middle column, we show a naive compositing approach; on the right the results of our approach. The naive approach does not take the 3D scene and light transport into consideration, and fails to photo-realistically render the chair.	49
3.13	Convergence with respect to the batch size: in this experiment, we assume the same compute/time budget for all experiments (x -axis), but we use different distributions of rays within each batch; i.e., we try different batch sizes.	49
3.14	Use of Multiple Importance Sampling during path tracing significantly improves the convergence rate.	50
3.15	A scene rendered with 10 bounces of light is given as input to our algorithm. We estimate emission and material parameters by using one, two, and three bounces during optimization. Two bounces are enough to capture most of the diffuse indirect illumination in the scene.	51
3.16	Results of our approach on synthetic scenes with textured objects. Our optimization is able to recover the scene lighting in addition to high-resolution surface texture material parameters.	51

3.17 Examples from Matterport3D [2] (real-world RGB-D scanning data) where we reconstruct emission parameters, as well as high-resolution surface texture material parameters. We are able to reconstruct fine texture detail by subdividing the geometry mesh and optimizing on individual triangle parameters. Since not all light sources are present in the reconstructed geometry, some inaccuracies are introduced into our material reconstruction. Albedo in shadow regions can be overestimated to compensate for missing illumination (visible behind the chair in Scene 1), specular effects can be baked into the albedo (reflection of flowers on the TV) or color may be projected onto the incorrect geometry (part of the chair is missing, so its color is projected onto the floor and wall). 52

4.1 Our method obtains a high-quality 3D reconstruction from an RGB-D input sequence by training a multi-layer perceptron. The core idea is to reformulate the neural radiance field definition in NeRF [8], and replace it with a differentiable rendering formulation based on signed distance fields which is specifically tailored to geometry reconstruction. 56

4.2 Differentiable volumetric rendering is used to reconstruct a scene that has been captured using an RGB-D camera. The scene is represented using multi-layer perceptrons (MLPs), encoding a signed distance value D_i and a viewpoint-dependent radiance value \mathbf{c}_i per point \mathbf{p}_i . We perform volumetric rendering by integrating the radiance along a ray, weighing the samples as a function of their signed distance D_i and their visibility. We also learn a per-frame latent corrective code to account for exposure or white balance changes throughout the capture, which is passed to the radiance MLP alongside the ray direction \mathbf{d} . We optimize the scene representation’s MLPs, together with the per-frame corrective codes, the input camera poses, and an image-plane deformation field (not shown) by computing losses for the signed distance D_i of the samples, and the final integrated color \mathbf{C} with respect to the input depth and color views. . . . 57

4.3 We compare our model without pose optimization and our full model with both the pose optimization and image-plane deformation field to BundleFusion, RoutedFusion, SIREN and a NeRF optimized with depth supervision in scenes 2, 5, 12, and 50 of the ScanNet dataset. Our model without pose optimization recovers smoother meshes than the density-based NeRF model, but still suffers from misalignment artifacts. These are solved by our full model to recover a clean reconstruction. 63

4.4	Accuracy shows how close ground truth points are to predicted points, while completeness shows how close predicted points are to ground truth points. Geometry reconstructed purely through the photometric loss has slightly lower accuracy than geometry for which depth observations were also available. Furthermore, the accuracy and completeness drop in distant areas, which had less multi-view constraints and more noise in the depth measurements.	65
4.5	Our method improves the camera alignment over the baseline, as visible in the tiles of the floor. The additional image-plane distortion correction results in straight and aligned edges in the reconstruction.	67
4.6	Our method obtains a high-quality 3D reconstruction from an RGB-D input sequence by training a multi-layer perceptron. In comparison to state-of-the-art methods like BundleFusion [6] or the theoretical NeRF [8] with additional depth constraints, our approach results in cleaner and more complete reconstructions. As can be seen, the pose optimization of our approach is key to resolving misalignment artifacts.	68
4.7	The photometric energy term encourages correct depth prediction in areas where the depth sensor did not capture any depth measurements.	71
4.8	We test the robustness of our method by removing frames from the dataset used for optimization. Our method achieves good reconstruction results using as few as 13 frames.	77
4.9	We test the robustness of our reconstructions to noise in the initial camera position and direction. Our method achieves good results even in the presence of significant noise. At $\sigma = 10$ cm, some of the cameras intersect geometry, degrading the reconstruction quality.	77
4.10	We test the robustness of our pose refinement to noise in the initial camera position and direction. The rotation error has been scaled by a factor of 10 for better visibility. Our method is able to correct poses even in the presence of significant noise. At $\sigma = 10$ cm, some of the cameras start intersecting geometry, making refinement impossible.	78
4.11	Reconstruction quality with varying batch size.	78
4.12	Comparison between NeuS and our method on the ‘morning apartment’ scene.	78
4.13	We compare the color synthesis of BundleFusion and NeRF-style methods. NeRF without any depth constraints shows severe fogging when rendering an image from a novel view. This gets resolved after adding depth constraints to the optimization. BundleFusion produces the sharpest results, but suffers from incorrect view-dependent effects and misalignment artifacts. Our method produces results similar to NeRF with a depth constraint. A combination of classic and NeRF-style methods may yield both high-quality geometry and high-quality view synthesis and we encourage further research in this direction.	79

4.14 We show a qualitative comparison of synthetic scene reconstructions obtained using our method and several baseline methods. The BundleFusion reconstruction is incomplete in some regions, screened Poisson and SIREN attempt to fit noise in the depth data, while the NeRF reconstruction suffers from noise in the density field. Our method manages to fill in gaps in geometry, while maintaining the smoothness of classic fusion approaches. 80

4.15 We show a qualitative comparison of synthetic scene reconstructions obtained using our method and several baseline methods. The BundleFusion reconstruction is incomplete in some regions, screened Poisson and SIREN attempt to fit noise in the depth data, while the NeRF reconstruction suffers from noise in the density field. Our method manages to fill in gaps in geometry, while maintaining the smoothness of classic fusion approaches. 81

5.1 Our method obtains high-resolution skin textures from two RGB input sequences captured with polarization foils attached to a smartphone. The core idea is to separate the skin’s diffuse and specular response by capturing one cross-polarized and one parallel-polarized sequence. We recover an accurate geometry with multi-view stereo, fit a parametric head model, and employ a differentiable rendering strategy to recover 4K diffuse albedo, specular gain and normal maps. These can be used with off-the-shelf rendering software, such as Blender, to produce photo-realistic images from novel views, under novel illumination and with subsurface scattering (SSS). 84

5.2 Our optimization has three steps: In step 0, we capture data with a handheld smartphone which is equipped with polarization foils (on the camera, as well as on the flashlight; see Figure 5.3). We reconstruct the facial geometry and estimate camera poses based on all captured images using structure-from-motion and multi-view stereo. To ensure consistent texture parameterization across different subjects, we non-rigidly fit a FLAME mesh to the scan. In a subsequent photometric optimization step (step 1), we estimate a high-resolution diffuse texture of the skin from the cross-polarized data, as well as an initial normal map. The reconstructed geometry, diffuse and normal map are used as input for step 2 of the optimization. Using the parallel-polarized sequence, we estimate the specular gain and final normal map in a second photometric optimization. In addition, a global skin roughness value is optimized in this step. 86

5.3 Left to right: smartphone equipped with polarization filters, cross-polarized image (perpendicular filter orientation) and parallel-polarized image (parallel filter orientation). 88

5.4 Geometry reconstruction for subject from Figure 5.3. From left to right: reconstruction via structure from motion, fitted FLAME [191] mesh, ICP-based refinement of the mesh. 89

5.5	We show skin texture reconstructions of several actors of different skin type. The rendered images closely match the reference target images, and we achieve good separation of diffuse and specular textures.	92
5.6	We evaluate on a validation frame from a novel viewpoint and with novel lighting that was held out during the optimization. As visible in the crop regions, our method is able to synthesize sharper texture details and specular highlights compared to NLT [199].	93
5.7	Comparison to NextFace [189] in terms of reconstructed appearance textures.	93
5.8	Ignoring the angle-dependent flashlight attenuation, the Fresnel effect, or the ambient light leads to an incorrect reconstruction, that can no longer reproduce the shading from all views. We account for these effects to closely match the target data.	95
5.9	We compare joint optimization of all textures to our full approach on a purely diffuse render. Optimizing jointly leaks specular and normal map information into the diffuse texture.	95
5.10	To increase sharpness, we optimize from photographs, instead of video frames. Using only pixels of the appropriate mip level in a coarse-to-fine approach further enhances results.	96
5.11	To calibrate the light of the smartphone, we record a cross-polarized sequence of a white planar surface with markers for tracking. We fit a UV-parameterized plane to the data and optimize for a light attenuation map which we use for all experiments.	98
5.12	We found that the polarization filters introduce a color shift depending on the polarization direction. To this end, we perform a color calibration with a Macbeth colorchecker board which we capture in both scenarios (cross-, and parallel-polarized). We use an affine color correction to match both captures, and apply this transformation to recordings of all subjects.	98
5.13	Distribution of cross-polarized (red) and parallel-polarized (blue) views.	99