Lights, Camera, Render!
Neural Fields for Structured Lighting

Aarrushi Shandilya

CMU-RI-TR-23-35

July 20, 2023

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

Here's to magic and resilience.

Abstract

3D scene reconstruction from 2D image supervision alone is an under-constrained problem. Recent neural rendering frameworks have made great strides in learning 3D scene representations to enable novel view synthesis, but they struggle to reconstruct geometry of low-texture regions or from sparse views. The prevalence of active depth sensors in common devices (*e.g.*, iPhone, Kinect, RealSense) has stimulated the use of depth-supervised neural models to accurately disambiguate the scene's geometry. However, the depth processed from these sensors can be prone to error, or even fail outright. Instead, a more principled approach is to explicitly model the raw structured light images themselves. In this work, we present an image formation model and optimization procedure that combines the advantages of neural radiance fields and structured light imaging. Our proposed approach enables the estimation of high-fidelity depth maps from sparse views, including for objects with complex material properties (*e.g.*, partially-transparent surfaces). Additionally, the raw structured light images confer useful radiometric cues, which enable predicting surface normals and decomposing scene appearance in terms of a direct, indirect, and ambient component. We evaluate our framework quantitatively and qualitatively on a range of real and synthetic scenes, and decompose scenes into their constituent components for novel views.

Acknowledgments

First and foremost, I am eternally grateful to my advisor Matthew O'Toole for his exceptional guidance and encouragement. This work would not have been possible without him. Thank you for giving me an opportunity to learn and grow under your wing – I have been deeply inspired by your great zeal and ideas.

I want to thank Benjamin Attal for his persistent insights on experiments and implementations, especially during critical times. I am also thankful for my collaboration with James Tompkin and Christian Richardt – our discussions have equipped me with a deeper understanding of the many technical subtleties of this work.

Additionally, I want to acknowledge the faculty members at the Robotics Institute, especially Ioannis Gkioulekas, Srinivasa Narasimhan, Shubham Tulsiani and Kris Kitani. The fundamentals they taught so ingeniously as a part of their courses form the foundation of this work. Thanks to my committee members, Shubham Tulsiani and Kangle Deng for their interest in this work and for taking the time to review it.

I couldn't be prouder to be part of the RI student fraternity, which brings forth such a positive sense of collective. A huge shout out to Winnie Kuang, Eric Schneider, Mateo Guaman Castro, Nikhil Bakshi, David Russell, Ananya Rao, Daphne Chen, Seth Karten, Sarvesh Patil, Harry Freeman and many more cohort members for being the best cheerleaders on this journey. Thank you for your patience and friendship, hugs and impromptu walks, assignment sessions, potlucks, and dance parties! I also want to acknowledge my lab mates – Dorian Chan and Sally Chen for providing a stimulating work environment and celebrating every small debug together.

Last but not least, I would like to thank my parents for their love, care and sacrifice through the years and my brother Arnav for being my eternal confidant.

Funding

We acknowledge funding support from NSF IIS-2008464 and Meta. James Tompkin thanks support from NSF CAREER-2144956 and Cognex.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Notation

Symbol	Units	Description
$\mathbf{x}, \mathbf{x}_c, \mathbf{x}_p$		A point $\in \mathbb{R}^3$, camera center, projector center.
$\boldsymbol{\omega}$, $\boldsymbol{\omega}_i$, $\boldsymbol{\omega}_o$		A unit vector $\in \mathbb{R}^3$, incoming direction, outgoing direction.
$\mathbf{n}(\mathbf{x})$		A unit normal $\in \mathbb{R}^3$ perpendicular to a surface at point $\mathbf{x}$.
$L(\mathbf{x}, \boldsymbol{\omega})$	$\mathrm{W} \cdot \mathrm{sr}^{-1} \cdot \mathrm{m}^{-2}$	Radiance measured at point $\mathbf{x}$ in direction $\boldsymbol{\omega}$.
$L_i(\mathbf{x}, \boldsymbol{\omega}_i)$	$\mathrm{W} \cdot \mathrm{sr}^{-1} \cdot \mathrm{m}^{-2}$	Incident radiance to a point $\mathbf{x}$ from a direction $\boldsymbol{\omega}_i$.
$L_o(\mathbf{x}, \boldsymbol{\omega}_o)$	$\mathrm{W} \cdot \mathrm{sr}^{-1} \cdot \mathrm{m}^{-2}$	Outgoing radiance from a point $\mathbf{x}$ in a direction $\boldsymbol{\omega}_o$.
$\sigma(\mathbf{x})$	m^{-1}	Density function at a point.
$T(\mathbf{x}, \mathbf{x}')$	*unitless*	Transmittance function, *i.e.*, accumulated density.
$f(\mathbf{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$	sr^{-1}	Bidirectional reflectance distribution function (BRDF).
$f_r(\mathbf{x}, \boldsymbol{\omega}_o)$	sr^{-1}	(Retro-)reflectance function given by $f(\mathbf{x}, \boldsymbol{\omega}_o, \boldsymbol{\omega}_o)$.

Chapter 1

Introduction

1.1 Motivation

3D scene reconstruction from 2D images is central to many use cases in photogrammetry, robotics, cinematic visual effects and digital preservation. Classical approaches like structure-from-motion [32] struggle in textureless regions, where finding correspondences between images is hard. Recently, neural rendering techniques like Neural Radiance Fields (NeRF) [24] and other variants [36] have enabled learning of compact 3D representations in a simple and fast manner. However, these can also struggle to reconstruct geometry in scenes with low-texture regions or from few input views.

Many depth cameras alleviate these issues by introducing their own lighting into the scene [13, 30]. For example, active stereo systems (*e.g.*, Intel RealSense [19]) use a projector to illuminate the scene with an (often unknown) light pattern, which adds texture to help solve the stereo correspondence problem. Coded structured light systems use known light patterns to solve correspondences using as few as one camera viewpoint. Such depth-sensing devices are found in many smartphones and tablets [3, 19, 47], and unlock new VR and AR applications. However, these sensors can also fail to reliably estimate depth, especially in cases where light misbehaves [14, 28], *e.g.*, due to light traveling many different paths before reaching a particular camera pixel.

1.2 Proposed Approach

In this thesis, we propose a volumetric image formation model and corresponding optimization procedure designed to synthesize structured light images under a known projection pattern. Given a set of raw structured light and ambient-only images captured from different viewpoints, our proposed framework retrieves a 3D representation of the scene through a neural volume rendering procedure [24], enabling novel view synthesis. Beyond recovering the geometry of challenging scenes (*e.g.*, scenes containing translucent objects or textureless surfaces), our image formation model takes advantage of additional radiometric cues present in the raw structured light images to solve for normals and separate the scene's appearance into direct, indirect, and ambient components. Figure 1.1 visualizes the different representations of geometry and appearance recovered using our proposed framework.

The main contributions of this thesis can be summarized as:

- A physically-based neural volume rendering model for multi-view structured light imaging, incorporating shading cues that inform normals and the separation of direct and indirect components for novel views.
- An implementation on a widely-available commercial system, an Intel RealSense camera [19], leading to reliable depth reconstruction performance using sparse views when compared to baseline approaches and the original RealSense depth.
- A demonstration that our model allows us to tackle new problems with structured light cameras, such as recovering geometry through partially transparent surfaces and through fine meshes.

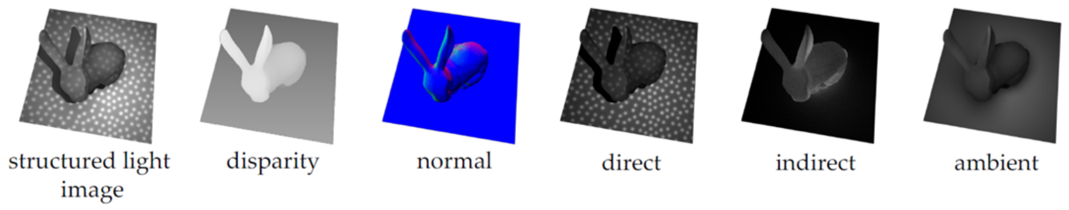


Figure 1.1: **Visualizing scene decomposition.** The proposed method decomposes a novel-view structured light image into its geometric (disparity, normal) and appearance (ambient, direct, indirect) components.

Chapter 2

Background

In this chapter, we discuss the relevant background for modeling and optimization of the 3D world using 2D captures. We first consider the high level flow of a neural inverse rendering pipeline and then dive into one such method - NeRF [24], which forms the base model for our proposed approach. Then, we discuss some fitting directions taken by prior works to enhance on this model. This includes depth supervision methods, specifically the advantages and challenges associated with the use of active depth sensing in common devices. Additionally, we highlight some important ideas and works related to active illumination, followed by a brief on structured lighting.

2.1 Neural Inverse Rendering with NeRF

Rendering is the process of converting a 3D model of the world containing information about its geometry, material, appearance, lighting and camera viewpoint into a 2D image. The scene representation could be explicit (*e.g.*, meshes, voxel grids, point clouds), implicit (*e.g.*, signed distance fields [10], radiance fields [24]), or even hybrid [35]. The domain of inverse rendering or inverse graphics pertains to reverting this process *i.e.*, retrieving 3D information of the scene from its 2D images. However, directly reverting this image formation model is not trivial. In most cases, there exists no analytical solution to this inversion process and mapping from 2D to 3D is a one-to-many problem.

Recent advances in neural rendering [36] combine generative machine learning with physical knowledge from computer graphics by incorporating differentiable rendering into network training. Hence, the gradients from the generated 2D image's reconstruction loss can be back-propagated in a differentiable manner to learn the corresponding 3D representation. This process can be repeated for images from different view points to resolve ambiguities for a particular scene. Such a neural inverse rendering paradigm enables photo-realistic rendering of images from novel view points and more degrees of freedom for scene manipulation.

An overview of this pipeline using Neural Radiance Fields (NeRF [24]) is shown in fig. 2.1. The core technical components of the NeRF model are described next.

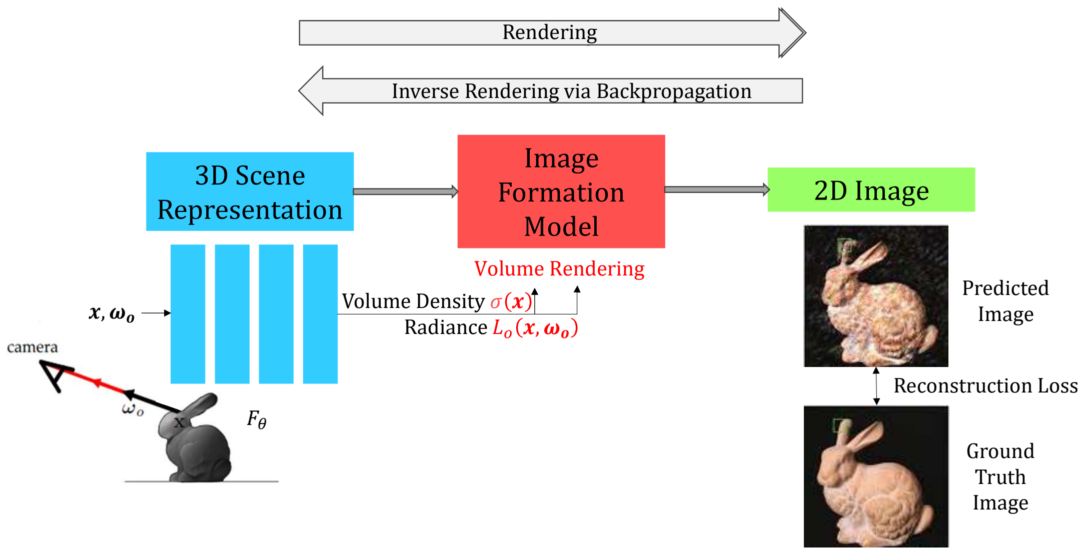


Figure 2.1: **Overview of a NeRF-based neural inverse rendering pipeline.**

2.1.1 3D Scene Representation

For a static scene, NeRF maps a 3D location $\mathbf{x}$ and a 2D viewing direction $\boldsymbol{\omega}_o$ (normalized unit direction) to a volume density $\sigma(\mathbf{x})$ and an outgoing radiance value $L_o(\mathbf{x}, \boldsymbol{\omega}_o)$ using the learnt weights of an MLP (Multi-Layer Perceptron). Hence, the network $F_{\boldsymbol{\theta}}$: $(\mathbf{x}, \boldsymbol{\omega}_o) \rightarrow (\sigma(\mathbf{x}), L_o(\mathbf{x}, \boldsymbol{\omega}_o))$ represents a continuous 5D radiance field. The volume density describes the point's opacity *i.e.*, amount of light occluded (ranging from 0 for transparent objects and higher values for opaque objects). Since it remains constant with respect to camera viewpoint, it is a function of location only. The outgoing radiance, however, depends on both the location and the viewing direction in order to model view-dependent effects, *e.g.*, specular highlights.

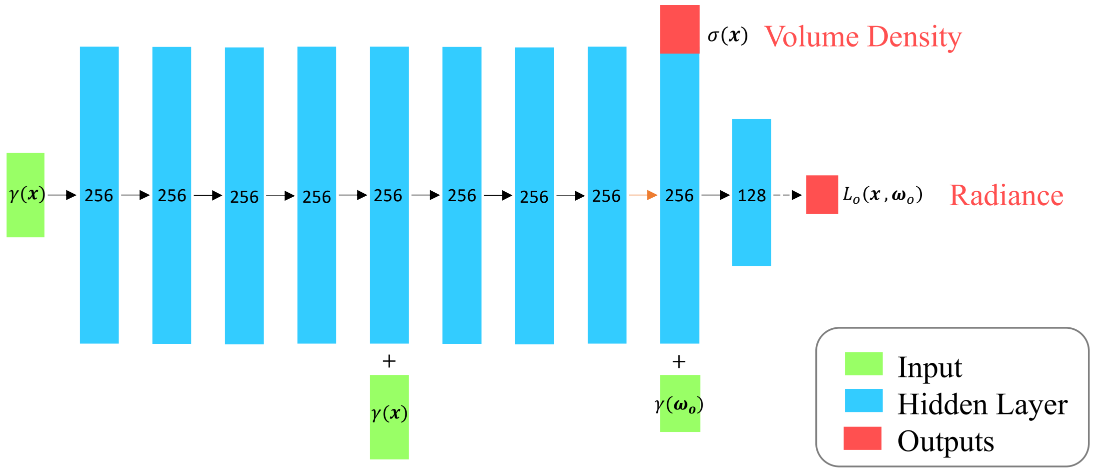

$\sigma(\boldsymbol{x})$ Volume Density
$\gamma(\boldsymbol{x})$
256
256
256
256
256
256
256
256
256
128
$L_o(\boldsymbol{x}, \boldsymbol{\omega}_o)$ Radiance
+
$\gamma(\boldsymbol{x})$
+
$\gamma(\boldsymbol{\omega}_o)$
Input
Hidden Layer
Outputs

Figure 2.2: **NeRF's fully-connected network architecture.**

The MLP architecture is depicted in fig. 2.2. The encoded input location $\gamma(\mathbf{x})$ is passed through 8 fully-connected, 256-channel layers, including a skip connection of this input at the fifth layer. Then, an additional layer combines: (i) an output channel for volume density; (ii) a 256-dimensional hidden layer concatenated with the encoded input viewing direction $\gamma(\boldsymbol{\omega}_o)$. After another 128-channel fully-connected layer, a final output layer produces the outgoing radiance. All layers are activated by ReLU, except for the output radiance which uses sigmoid activation. The positional encoding $\gamma : \mathbb{R} \to \mathbb{R}^{2L}$ helps capture high frequency details within a small neighbourhood of input coordinates by projecting them onto a high dimensional input space. It is defined as:

$$\gamma(p) = (\sin(2^0\pi p), \cos(2^0\pi p), \sin(2^1\pi p), \cos(2^1\pi p), ... \sin(2^{(L-1)}\pi p), \cos(2^{(L-1)}\pi p)) \tag{2.1}$$

2.1.2 Image Formation Model

The above representation can be used to render images of a scene within a bounding volume from different view points using a simple ray marching and volume rendering procedure.

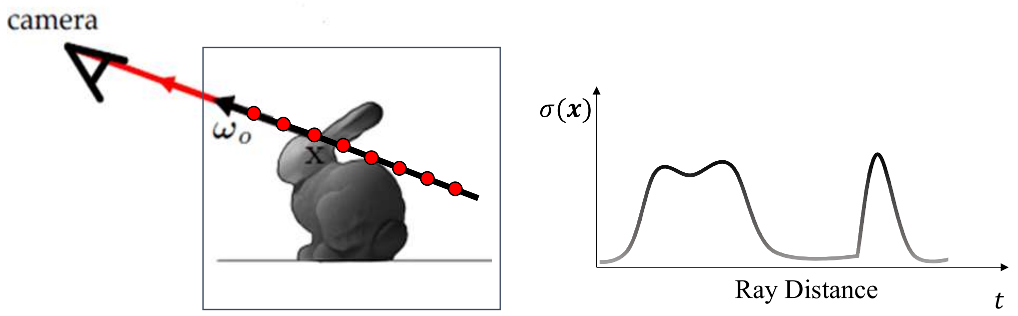


Figure 2.3: **Ray marching.** (a) Tracing a ray from the camera center $\mathbf{x}_c$ through the scene in direction $\boldsymbol{\omega}_o$ and with sampled points $\mathbf{x} = \mathbf{x}_c + \boldsymbol{\omega}_o t$. (b) Visualization of volume density σ along a ray, where the sampled points are determined by the parameter t.

Ray marching. Given a calibrated camera, rays are cast from its center through each pixel location onto the scene. Transformation from the image to world coordinates is given by:

$$\mathbf{x}_c = T; \quad \boldsymbol{\omega}_o = R^T K^{-1} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}, \tag{2.2}$$

where $\mathbf{x}_c$ is the ray origin or camera center in world coordinates and $\boldsymbol{\omega}_o$ is the unit ray direction in world coordinates, T and R are the translation and rotational pose respectively, K is the intrinsic matrix, and $[u, v, 1]^T$ is the homogeneous pixel coordinate. For each 3D point $\mathbf{x} = \mathbf{x}_c + \boldsymbol{\omega}_o t$ sampled along a ray, the NeRF MLP is queried for its volume density and directional outgoing radiance. This procedure is visualized in fig. 2.3. NeRF uses a hierarchical sampling approach, inspired from prior volume rendering work [20].

Volume rendering. The total radiance observed at a pixel can be rendered as the accumulated radiance of its ray using direct volume rendering [16, 23]. Let us attribute a pixel (u, v) by it's corresponding ray $(\mathbf{x}_c, \boldsymbol{\omega}_o)$ for a camera view. Consider a camera ray passing through a non-homogeneous continuous medium, marked by the bounded box in fig. 2.3a. The total intensity $L(\mathbf{x}_c, \boldsymbol{\omega}_o)$ at a pixel is the integrated contribution of infinitesimally small segments at each point sampled along the ray :

$$L(\mathbf{x}_c, \boldsymbol{\omega}_o) = \int_{t_\mathrm{n}}^{t_\mathrm{f}} T(\mathbf{x}_c, \mathbf{x})\sigma(\mathbf{x})L_o(\mathbf{x}, \boldsymbol{\omega}_o)\ \mathrm{d}t. \tag{2.3}$$

Here, $\mathrm{d}t$ is the size of the segment at a sampled 3D location $\mathbf{x}{=}\mathbf{x}_c{+}\boldsymbol{\omega}_o t$. The near (t_n) and far (t_f) bounds restrict the ray sampling within a bounded volume. The intensity or radiance values can be scalar for a grayscale image or vectors of multiple wavelength bands (*e.g.*, red, green, and blue) for a color image.

The transmittance function $T(\mathbf{x}_c, \mathbf{x})$ represents the proportion of light that travels from $\mathbf{x}$ to $\mathbf{x}_c$. It is computed as:

$$T(\mathbf{x}_c, \mathbf{x}) = \exp\left(-\int_{t_\mathrm{n}}^{t} \sigma(\mathbf{x}_c - \boldsymbol{\omega}_o s)\ \mathrm{d}s\right). \tag{2.4}$$

Assuming homogeneous medium within an infinitesimally small segment around a sample point $\mathbf{x}_k$, the integral in eq. (2.3) is evaluated using quadrature [23, 24] :

$$L(\mathbf{x}_c, \boldsymbol{\omega}_o) = \sum_k w_k L_o(\mathbf{x}_k, \boldsymbol{\omega}_o), \tag{2.5}$$

where

$$w_k = \hat{T}(\mathbf{x}_c, \mathbf{x}_k)(1 - \exp\left(-\sigma(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k)\right)), \tag{2.6}$$

$$\hat{T}(\mathbf{x}_c, \mathbf{x}_k) = \prod_{j<k} \exp\left(-\sigma(\mathbf{x}_j)(\mathbf{x}_{j+1} - \mathbf{x}_j)\right). \tag{2.7}$$

2.1.3 Photometric Loss

The training objective is formed by Mean Squared Error (MSE) between the pixel intensities of the ground truth image and the accumulated ray radiance via predicted volume density and radiance. This objective is trained for randomly sampled pixel batches across multi-view images.

$$\mathcal{L} = \sum_{\boldsymbol{\omega}_o} \left\| L(\mathbf{x}_c, \boldsymbol{\omega}_o) - \hat{L}(\mathbf{x}_c, \boldsymbol{\omega}_o) \right\|^2, \tag{2.8}$$

where L is the predicted intensity from eq. (2.3) and $\hat{L}$ is the ground truth intensity.

2.1.4 Limitations

Decomposing 2D images into 3D scene attributes is an under-constrained problem and hence, NeRF usually requires large number of training views. Like passive stereo, it can fail to establish correspondence in textureless regions, leading to inaccurate disambiguation of geometry and appearance in these cases (*e.g.*, learning a plausible cloudy geometry for a flat wall).

2.2 Depth Supervision

The dependency on large number of training views for better disentanglement of geometry can be overcome via depth consistency priors [27], semantic priors [15], or depth supervision from traditional multi-view stereo algorithms [11, 29, 39]—although the processed depth can inherit the limitations of traditional passive stereo approaches. For instance, fig. 2.4 shows a point cloud reconstruction for a scene using a Structure from Motion (SfM) algorithm [32]. It lacks features in textureless areas, and hence depth supervision using this point cloud will be sparse in terms of spanning the entire scene.

Past works also combine depth supervision from active illumination sensors for sparse-view reconstruction [5, 9, 11, 34, 49]. Such sensors have been prevalent in common devices (fig. 2.5), *e.g.*, Intel Realsense [19], Apple iPhone [3], Microsoft Kinect [47]. As shown in fig. 2.6a, an Intel Realsense D435 camera projects a high frequency dot pattern onto a scene to retrieve denser depth even in textureless areas. However, such sensors use limited or simplified image formation models to process depth, which can fail to track the multi-path illumination behaviour. To demonstrate this behaviour, let's introduce a partially-transparent mesh in between the camera

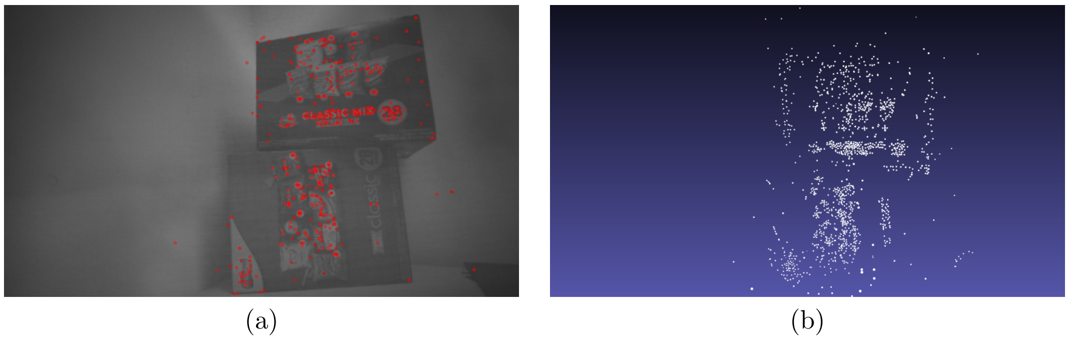


Figure 2.4: **Sparse depth from SfM.** (a) Features detected in a single view. (b) Point cloud reconstruction using multiple views.

and the boxes in the scene, leading to complete breakdown of the RealSense depth (fig. 2.6b). It fails to establish multi-view correspondences when light is both reflected back directly from the mesh and transmitted through the mesh. In contrast, modelling the physical image formation process of the raw images from a structured light sensor can take better advantage of the benefits of both structured light and volumetric reconstruction.

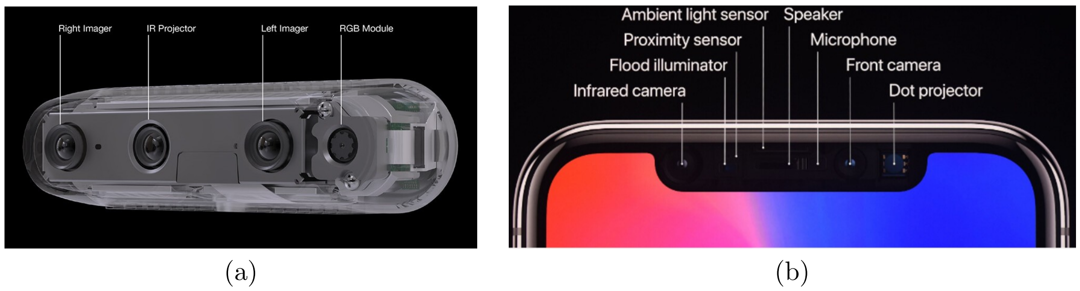


Figure 2.5: **Common devices with active depth sensors.** (a) Intel RealSense D435 camera. (b) Apple iPhoneX with TrueDepth camera.

2.3 Active Illumination

Active illumination methods refer to explicit modelling and control of illumination sources in the scene. These methods can recover scene properties such as reflectance and shape with much greater reliability compared to the ambient lighting-only case. One such classical technique is photometric stereo [40], where the scene is illuminated

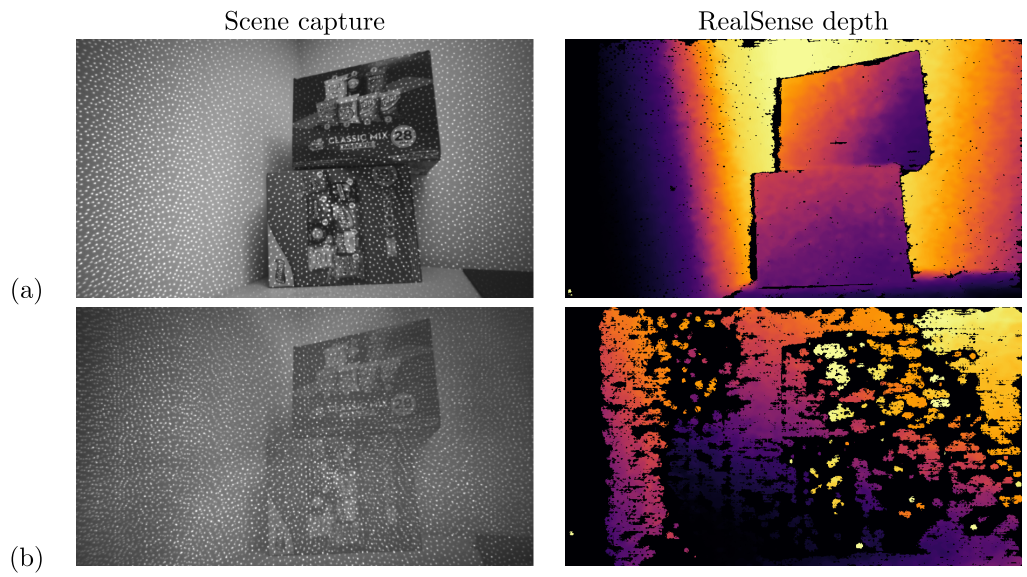


Figure 2.6: **Depth processed from a RealSense camera.** (a) Original scene (no mesh). (b) Scene with a translucent mesh added in front of the box.

with different directional light sources to recover the surface orientation for each view.

Similarly, modeling illumination within a volume rendering image formation model can offer several benefits in addition to improving reconstruction quality. For example, Bi et al. [6] and Zhang et al. [43] combine neural volume rendering with a flash light source collocated with the camera to recover depth, normals, and scene reflectance. Flash light offers near-field illumination and usually requires disabling or masking out ambient lighting in the scene. Various works make use of slightly more complicated illumination conditions in the form of point light sources at several different positions [18, 33, 45, 48]. Since these use illumination with limited spatial variation, they rely on large number of captures or light configurations. Other works leverage environment map lighting that is optimized alongside the neural volume [8, 17, 21, 22, 41, 46]. While some works account for global illumination [33, 48], they either limit themselves to two-bounce global illumination or else do not demonstrate direct-indirect separation in real-world settings. Modeling global light transport with respect to the introduced light source is an important aspect of active illumination methods. This includes direct light *i.e.*, light reaching camera after single bounce from the scene and indirect light which accounts for multiple bounces, *e.g.*, inter-reflections, sub-surface scattering. Using the proposed approach, we show that we can achieve accurate scene reconstruction and intrinsic decomposition (including direct-indirect separation) on real-world scenes. Our approach is similar in spirit to prior works that use flood illumination [6] or time-of-flight sensors [4], though ours focuses on using a structured light system.

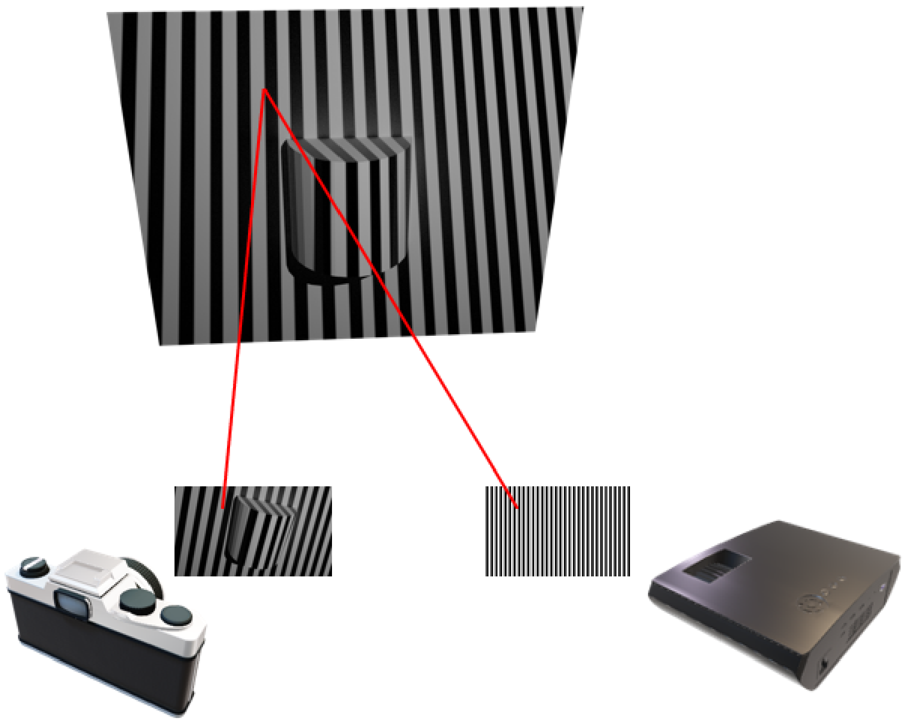


Figure 2.7: **Structured lighting.** Pixel correspondences between the projected pattern and the captured camera image are triangulated to compute scene depth.

2.3.1 Structured Lighting

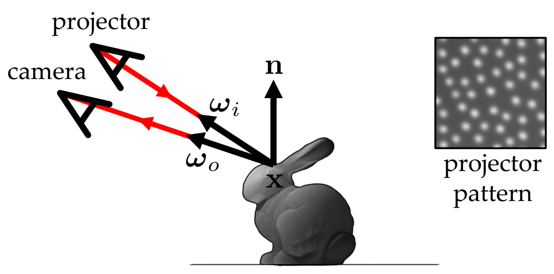
Structured lighting is an active illumination technique to acquire 3D geometry by projecting an illumination pattern or a set of patterns onto a scene while simultaneously using a camera to capture images. For a known pattern, it is possible to find pixel correspondences between the camera and projector and recover depth via triangulation [13]. These correspondences are more robust than the spatial template matching approach used in passive stereo. Salvi et al. [30] discuss the trade-offs involved in designing illumination patterns and decoding strategies for structured light depth sensing. Our proposed method does not require decoding of multiple illumination patterns. Instead, we work with a single high frequency pattern and use the calibrated camera-projector system to determine which projector pixel illuminates the 3D points on a camera ray. This known illumination is modeled within the volume rendering integral and helps with better disambiguation of geometry along with further scene decomposition. The high frequency illumination in the projected patterns helps reconstruct accurate geometry with sparser views compared to other active illumination methods, *e.g.*, using point lights. The proposed system is detailed in the next chapter.

Chapter 3

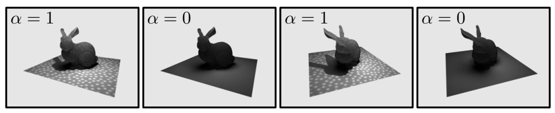
Neural Fields for Structured Lighting

3.1 The Structured Light System

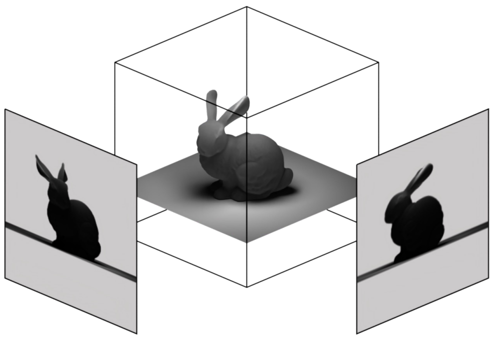
Consider using a projector-camera system (*e.g.*, the Intel RealSense [19]) to capture measurements of a scene from multiple viewpoints, where the projection pattern is known. In this scenario, the projector produces stroboscopic illumination, *i.e.*, it is turned *"on"* for even frames and *"off"* for odd frames. When the projector is *on*, it actively illuminates a scene with the known fixed pattern, and the camera measures the scene's radiometric response to both the projector's illumination and all other ambient light sources in the environment. When the projector is *off*, the camera only measures the ambient light. Given these measurements, our proposed volume rendering framework reconstructs the depths and normals corresponding to scene geometry, as well as the direct, indirect, and ambient light transport components that contribute to scene appearance (see overview in fig. 3.1).



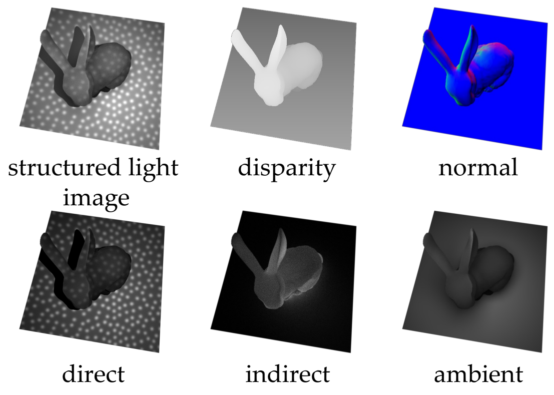
(a) acquisition setup



(b) input images



(c) reconstructed volume



(d) output images

Figure 3.1: **Overview of the proposed structured light reconstruction procedure. (a)** The acquisition setup consists of a single camera and projector illuminating the scene with a fixed projection pattern. **(b)** The projector strobes the illumination as the setup moves around the scene, producing an image sequence where the pattern alternates between on and off. **(c)** The proposed volume reconstruction problem recovers the appearance and shape of the scene. **(d)** The constituent components that make up appearance and shape can then be synthesized for novel views.

To understand how, first consider modeling the light incident at a point $\mathbf{x}$ in the scene with a function $L_i^\alpha(\mathbf{x}, \omega_i)$:

$$L_i^\alpha(\mathbf{x}, \omega_i) = \underbrace{L_i^{\text{ambient}}}_{\text{passive}} + \alpha \underbrace{\left(L_i^{\text{direct}} + L_i^{\text{indirect}}\right)}_{\text{active}}, \tag{3.1}$$

where $\alpha \in \{0, 1\}$ accounts for whether the projector is *"off"* or *"on"*. The incident light from the projector can either take a direct path to an object's surface (L_i^{direct}) or reach the surface indirectly, by reflecting off of other scene points first (L_i^{indirect}). The ambient term, L_i^{ambient}, represents the incident light (both direct and indirect) from all sources other than the projector.

Given these incident light sources, the outgoing radiance at the point can be calculated using the rendering equation:

$$L_o^\alpha(\mathbf{x}, \boldsymbol{\omega}_o) = \int_{\Omega(\mathbf{x})} f(\mathbf{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) L_i^\alpha(\mathbf{x}, \boldsymbol{\omega}_i)(\mathbf{n}(\mathbf{x}) \cdot \boldsymbol{\omega}_i)\, \mathrm{d}\boldsymbol{\omega}_i, \tag{3.2}$$

where the domain $\Omega(\mathbf{x})$ represents the hemisphere of incident light directions at point $\mathbf{x}$. Here, the bidirectional reflectance distribution function (BRDF), $f(\mathbf{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o)$, defines the proportion of incoming light scattered in the outgoing direction. When combined with eq. (3.1), we can further decompose the rendering equation as follows:

$$L_o^\alpha(\mathbf{x}, \boldsymbol{\omega}_o) = L_o^{\text{ambient}} + \alpha(L_o^{\text{direct}} + L_o^{\text{indirect}}), \tag{3.3}$$

where each component of outgoing radiance represents the result of evaluating the integral with respect to the corresponding incident radiance. Note that both active components, L_o^{direct} and L_o^{indirect} depend on the pose of the projector, and contribute to the total outgoing radiance only when the projector is *"on"* *i.e.*, $\alpha = 1$.

3.2 Neural Scene Representation

To model the environment, a neural network $F_{\boldsymbol{\theta}}$ takes as input a 3D position $\mathbf{x}$ and a unit viewing direction $\boldsymbol{\omega}_o$, and outputs $(\sigma(\mathbf{x}), \mathbf{n}(\mathbf{x}), f_r(\mathbf{x}, \boldsymbol{\omega}_o), L_o^{\text{ambient}}(\mathbf{x}, \boldsymbol{\omega}_o), L_o^{\text{indirect}}(\mathbf{x}, \boldsymbol{\omega}_o))$ that capture the scene's geometric and radiometric properties. The MLP architecture is same as described in section 2.1.1. Only the output layers are modified to account for the additional predicted outputs, as shown in fig. 3.2. This neural representation can be used to render scenes from different viewpoints by tracing rays through the volume and computing the radiometric response at each point sampled along the ray as described next.

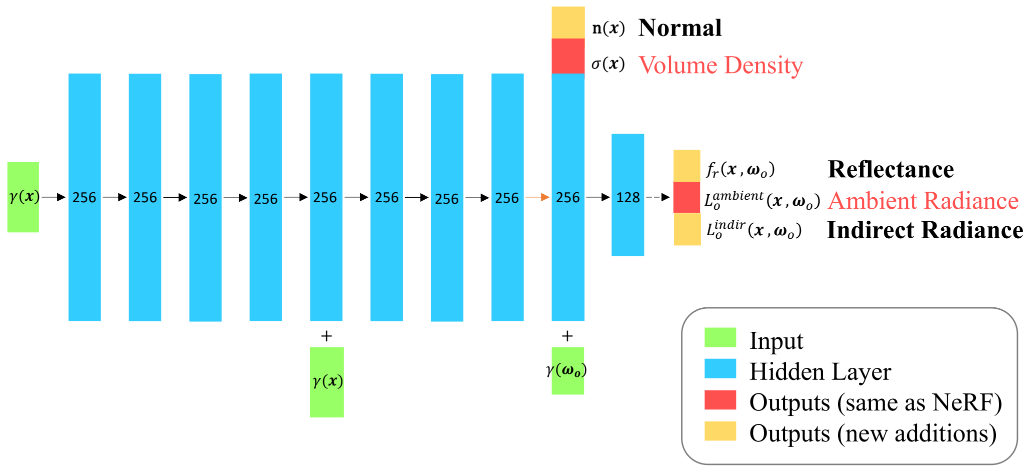

$\gamma(x)$
256
256
256
256
256
256
256
256
256
128
$\gamma(x)$
$\gamma(\omega_o)$
$n(x)$ Normal
$\sigma(x)$ Volume Density
$f_r(x, \omega_o)$ Reflectance
$L_o^{ambient}(x, \omega_o)$ Ambient Radiance
$L_o^{indir}(x, \omega_o)$ Indirect Radiance
🟩 Input
🟦 Hidden Layer
🟥 Outputs (same as NeRF)
🟨 Outputs (new additions)

Figure 3.2: **Neural network architecture for the proposed scene representation.** Output layer additions on top of NeRF's base model.

3.3 Neural Volume Rendering

Given a camera's pose and intrinsic parameters, rays are cast from the camera's optical center $\mathbf{x}_c$ through each pixel in direction $\boldsymbol{\omega}_o$. Volume density, normals, reflectance, and outgoing radiance functions (ambient and indirect) are queried at a set of 3D points along the ray, and the total radiance is accumulated at the camera pixel by computing the following integral:

$$L^{\alpha}(\mathbf{x}_c, \boldsymbol{\omega}_o) = \int_{t_\mathrm{n}}^{t_\mathrm{f}} T(\mathbf{x}_c, \mathbf{x})\sigma(\mathbf{x}) L_o^{\alpha}(\mathbf{x}, \boldsymbol{\omega}_o)\ \mathrm{d}t, \tag{3.4}$$

This is same as the volume rendering integral in eq. (2.3), by replacing the radiance terms of a general camera system with the components of the stroboscopic structured light system as described above. Here, the transmittance term $T(\mathbf{x}_c, \mathbf{x})$ is same as given by eq. (2.4).

As discussed in eq. (3.3), in this work, we independently model three different sources of illumination:
(i) the neural network directly outputs the ambient term, as done by previous works [24];
(ii) we provide a physics-based model for the direct term using the predicted normals and reflectance terms, and
(iii) we propose to optimize a term that approximates the indirect component.

The Ambient Component As described in section 2.1.2, we model the accumulated ambient radiance for a pixel using the predicted outgoing ambient radiance $L_o^{ambient}(\mathbf{x}, \boldsymbol{\omega}_o)$ and volume density $\sigma(\mathbf{x})$:

$$L^{ambient}(\mathbf{x}_c, \boldsymbol{\omega}_o) = \int_{t_\mathrm{n}}^{t_\mathrm{f}} T(\mathbf{x}_c, \mathbf{x})\sigma(\mathbf{x}) L_o^{ambient}(\mathbf{x}, \boldsymbol{\omega}_o)\ \mathrm{d}t. \tag{3.5}$$

The Direct Component. L_o^{direct} models the single-bounce light transport component, where light travels from the projector to a single scene point and back to the camera.

For a projector's origin positioned at $\mathbf{x}_p$, the direct lighting incident at a particular point $\mathbf{x}$ is given by the following function:

$$L_i^{\mathrm{direct}}(\mathbf{x}, \boldsymbol{\omega}_i) = \frac{P(\mathbf{x})}{\|\mathbf{x}_p - \mathbf{x}\|^2} T(\mathbf{x}_p, \mathbf{x}) \delta(\boldsymbol{\omega}_p - \boldsymbol{\omega}_i), \tag{3.6}$$

where

$$\boldsymbol{\omega}_p = \frac{\mathbf{x}_p - \mathbf{x}}{\|\mathbf{x}_p - \mathbf{x}\|}. \tag{3.7}$$

The function $P(\mathbf{x})$ queries the intensity of the projector pixel illuminating point $\mathbf{x}$, identified through perspective projection, as the projector has similar geometric properties to a camera; note that the output depends on the pose of the projector. The $1/\|\mathbf{x}_p - \mathbf{x}\|^2$ term models the inverse square light fall-off. The transmission function $T(\mathbf{x}_p, \mathbf{x})$ determines the proportion of light transmitted between points $\mathbf{x}_p$ and $\mathbf{x}$. Finally, the Dirac distribution $\delta(\cdot)$ ensures that the lighting comes from a single direction based on the projector's position.

When combined with the rendering equation as in eq. (3.2), we obtain:

$$L_o^{\text{direct}}(\mathbf{x}, \boldsymbol{\omega}_o) = \frac{f(\mathbf{x}, \boldsymbol{\omega}_p, \boldsymbol{\omega}_o)}{\|\mathbf{x}_p - \mathbf{x}\|^2} P(\mathbf{x}) T(\mathbf{x}_p, \mathbf{x})(\mathbf{n}(\mathbf{x}) \cdot \boldsymbol{\omega}_p). \tag{3.8}$$

This expression is non-trivial to evaluate for two reasons. First, this requires knowledge of the full BRDF at every point in space. Second, this requires evaluating the projector ray's transmission function, which would be a computational bottleneck in neural volume rendering due to additional sampling and network queries for incident projector rays.

To help, we make two assumptions: (i) the projector light casts no shadows, *i.e.*, $T(\mathbf{x}_p, \mathbf{x}) = 1$; and (ii) the BRDF can be approximated with the reflectance function $f_r(\mathbf{x}, \boldsymbol{\omega}_o) = f(\mathbf{x}, \boldsymbol{\omega}_o, \boldsymbol{\omega}_o)$, representing the ratio of light reflected in the direction of the illumination source. This holds approximately for small-baseline projector-camera systems, provided that the distance to the scene is sufficiently large. Note that another intuitive way to bypass the projector ray transmittance computation is to assume it is same as the camera ray transmittance given the small baseline, although we did not observe any significant difference in results with $T(\mathbf{x}_p, \mathbf{x}) = T(\mathbf{x}_c, \mathbf{x})$ versus $T(\mathbf{x}_p, \mathbf{x}) = 1$ on our data.

Hence, the simplified expression for the contribution of direct light to accumulated pixel radiance is given as follows:

$$L^{direct}(\mathbf{x}_c, \boldsymbol{\omega}_o) = \int_{t_n}^{t_f} \frac{T(\mathbf{x}_c, \mathbf{x})}{\|\mathbf{x}_p - \mathbf{x}\|^2} \sigma(\mathbf{x}) f_r(\mathbf{x}, \boldsymbol{\omega}_o) P(\mathbf{x}) (\mathbf{n}(\mathbf{x}) \cdot \boldsymbol{\omega}_p) \, \mathrm{d}t. \tag{3.9}$$

The Indirect Component. As a byproduct of our framework, it is possible to recover an approximation of the indirect component for scenes. L_o^{indirect} models the component of light that misbehaves (*e.g.*, bounces around a scene multiple times). However, the global nature of the indirect channel makes it non-trivial to model accurately. This is because, in a volume rendering framework, the indirect component at any given 3D point $\mathbf{x}$ would also depends on 6D pose of the projector-camera system—making it far too challenging to model and reconstruct explicitly.

Our ability to separately recover the direct and indirect components of a scene is based on the work by Nayar et al. [26]. The key idea is to illuminate a scene with a high-frequency pattern and observe the response at a point $\mathbf{x}$ to *different* illumination conditions, *e.g.*, the result of moving the structured light pattern across the scene. Provided that the indirect component is smooth relative to this illumination pattern, the indirect component at a point in the scene stays more or less constant with respect to small perturbations to the global position of the structured light pattern. Therefore, to approximate the indirect component, we propose using a function $L_o^{\text{indirect}}(\mathbf{x}, \boldsymbol{\omega}_o)$ which *only* takes as input the scene point $\mathbf{x}$ and viewing direction $\boldsymbol{\omega}_o$. This predicted indirect component can be integrated using the same volume rendering procedure to compute the indirect component contribution towards the accumulated pixel radiance:

$$L^{indirect}(\mathbf{x}_c, \boldsymbol{\omega}_o) = \int_{t_\text{n}}^{t_\text{f}} T(\mathbf{x}_c, \mathbf{x})\sigma(\mathbf{x}) L_o^{indirect}(\mathbf{x}, \boldsymbol{\omega}_o)\ \mathrm{d}t \tag{3.10}$$

3.4 Neural Volume Optimization

Photometric Loss We optimize the neural volume framework using both ambient-only measurements $\hat{L}^0(\mathbf{x}_c, \omega_o)$ and structured light measurements $\hat{L}^1(\mathbf{x}_c, \omega_o)$ using calibrated camera-projector poses. The calibration procedure is described in section 4.3.3.

We build our framework on top of NeRF-PyTorch [42], which we extend to output normal $\mathbf{n}(\mathbf{x})$, reflectance $f_r(\mathbf{x}, \boldsymbol{\omega}_o)$, and indirect radiance $L_o^{\text{indirect}}(\mathbf{x}, \boldsymbol{\omega}_o)$. We train coarse and fine networks using an ambient photometric loss:

$$\mathcal{L}^{\text{ambient}} = \sum_{\boldsymbol{\omega}_o} \left\| L^0(\mathbf{x}_c, \boldsymbol{\omega}_o) - \hat{L}^0(\mathbf{x}_c, \boldsymbol{\omega}_o) \right\|^2, \tag{3.11}$$

and a structured light photometric loss:

$$\mathcal{L}^{\text{SL}} = \sum_{\boldsymbol{\omega}_o} \left\| L^1(\mathbf{x}_c, \boldsymbol{\omega}_o) - \hat{L}^1(\mathbf{x}_c, \boldsymbol{\omega}_o) \right\|^2, \tag{3.12}$$

where L^0 and L^1 are the predicted ambient and structured light terms from eq. (3.4).

Because the contribution of indirect radiance is often weaker than the other components, we scale the network output corresponding to indirect radiance by 0.1 to implicitly bias the resultant indirect radiance towards a small value. For scenes with minimal indirect component, we simply omit this channel.

Normals Loss We follow the approach proposed by Verbin et al. [37] to penalize the gap between the predicted normals and the density gradient normals:

$$\mathcal{L}_p^{\text{normal}} = \sum_k w_k \left\| \mathbf{n}_k - \hat{\mathbf{n}}_k \right\|^2, \tag{3.13}$$

where $\mathbf{n}_k$ are the predicted normals, $\hat{\mathbf{n}}_k = {}^{\nabla\sigma(\mathbf{x})}/_{||\sigma(\mathbf{x})||}$ are the analytical or density gradient normals. We also use their proposed regularization term to penalize back-facing normals:

$$\mathcal{L}_o^{\text{normal}} = \sum_k w_k \cdot \max(0, \mathbf{n}_k \cdot \boldsymbol{\omega}_o)^2. \tag{3.14}$$

w_k are the weights associated with each sampled point as mentioned in eq. (2.6). Note that in our implementation of the normals loss terms, we compute weighted mean over samples instead weighted sum over samples for each ray. This effectively reduces the weight on the normals loss terms in the total loss.

Total Loss The total loss is as follows:

$$\mathcal{L}^{\text{total}} = \mathcal{L}^{\text{ambient}} + \lambda_1 \mathcal{L}^{\text{SL}} + \lambda_2 \mathcal{L}_p^{\text{normal}} + \lambda_3 \mathcal{L}_o^{\text{normal}}. \tag{3.15}$$

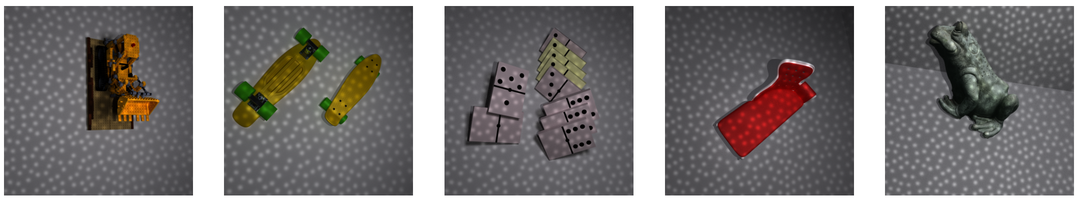
During training, we gradually decay the weight λ_1 to a small value. This ensures that the optimization procedure can initially make use of the structured light images to recover geometry (especially of low-texture regions), while the ambient loss dominates in later iterations to allow for better detail reconstruction. In practice, we alternate between optimizing the $\mathcal{L}^{\text{ambient}}$ and $\mathcal{L}^{\text{SL}}$ objectives, as the poses (and thus rays) for the structured light and ambient images are different. For all our experiments, we train our model for 100K iterations and use the same learning rate decay and optimizer as in NeRF [24]. Training with our method takes 4-6 hours on an NVIDIA 3090 RTX GPU (24GB RAM).

Chapter 4

Data and Calibration

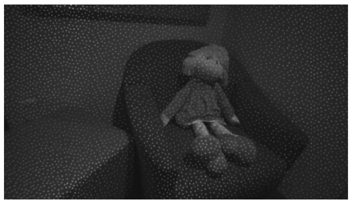
4.1 Synthetic Data

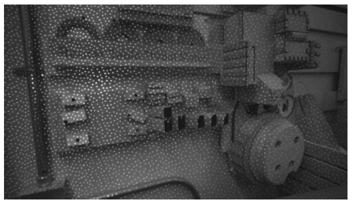
In Blender [7], we form a structured light system using a projector plugin [31]. The scenes include objects from NeRF's Blender dataset [24] and open source 3D models [1] (license information in table A.1) placed on a textureless plane. Our synthetic dataset consists of 5 scenes (fig. 4.1) - *lego, skateboard, domino, sofa* and *frog*. Keeping the relative camera-projector pose fixed, the structured light rig is perturbed to generate random views. For each view, we render 400×400 images with the projector illumination turned *on* and *off*. When the projector is *on*, it illuminates the scene with a high frequency dot pattern. We disable any post processing (*e.g.*, tonemapping) to avoid unknown non-linearities in the image formation model. Global illumination and shadows are kept enabled. Ground truth extrinsics and intrinsics are directly retrieved from Blender without any additional calibration.

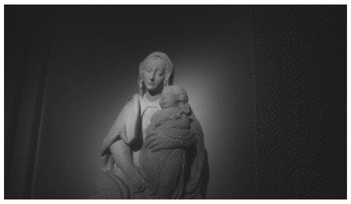


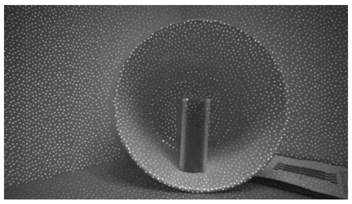
(a) *lego* (b) *skateboard* (c) *domino* (d) *sofa* (e) *frog*

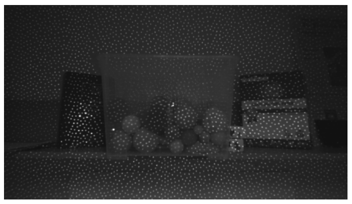
Figure 4.1: **Synthetic dataset scenes.** Sample views of structured light images.


(a) *doll*


(b) *woodshop*


(c) *sculpture*


(a) *candle*


(b) *translucent box*

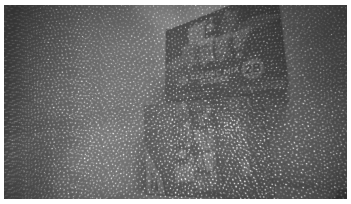

(c) *mesh box*

Figure 4.2: **Real dataset scenes.** Sample views of structured light images.

4.2 Real Data

We use an Intel RealSense D435 system [19] with a built-in infrared dot projector and stereoscopic depth cameras to capture real data. Although the device has three cameras, we choose to use only one monochromatic camera for our experiments, but our framework can easily be extended to account for all three cameras. While streaming data, the device strobes the illumination to capture a set of frames when the projector is *on* and *off* alternately. We assume that the projector is rigidly fixed to the camera and that the camera and projector move together around a scene to illuminate and capture scenes. We capture 6 scenes (fig. 4.2) - *doll, woodshop, sculpture, candle, translucent box* and *mesh box.* All images are captured at a resolution 848 $\times$ 480.

4.3 Calibration

In this section, we detail our calibration routine to recover the intrinsic and extrinsic parameters of the infrared stereo cameras and the projector, and also recover the projector's structured light dot pattern. Our image formation model uses these calibration parameters to synthesize the structured light images of the scene.

4.3.1 Camera Calibration

The first step of the calibration procedure is to compute the intrinsic matrices $(\mathbf{K}_1, \mathbf{K}_2)$, distortion $(\mathbf{d}_1, \mathbf{d}_2)$ and rectification $(\mathbf{r}_1, \mathbf{r}_2)$ parameters, relative extrinsics $(\mathbf{R}, \mathbf{T})$ and projection matrices $(\mathbf{P}_1, \mathbf{P}_2)$ for the monochromatic stereo cameras. Streaming both the cameras (with projector *off*) at a resolution of 848×480 pixels, we capture images of a $7{\times}8$ planar checkerboard in a variety of poses (fig. 4.3). Using standard OpenCV functions and calibration flow [2], we compute these parameters as outlined fig. 4.4 and in pseudocode algorithm 1.

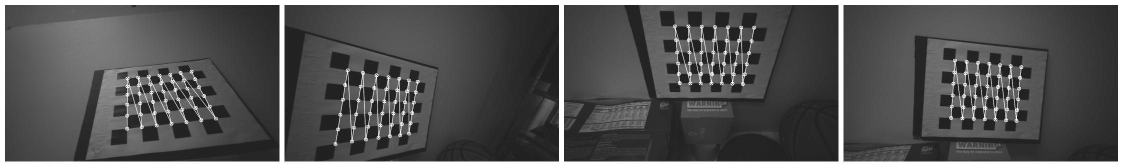


Figure 4.3: **Sample checkerboard captures with detected corners.** 100+ checkerboard poses are used to span both cameras' field of view at different depths and orientations to accurately calibrate for the camera parameters.

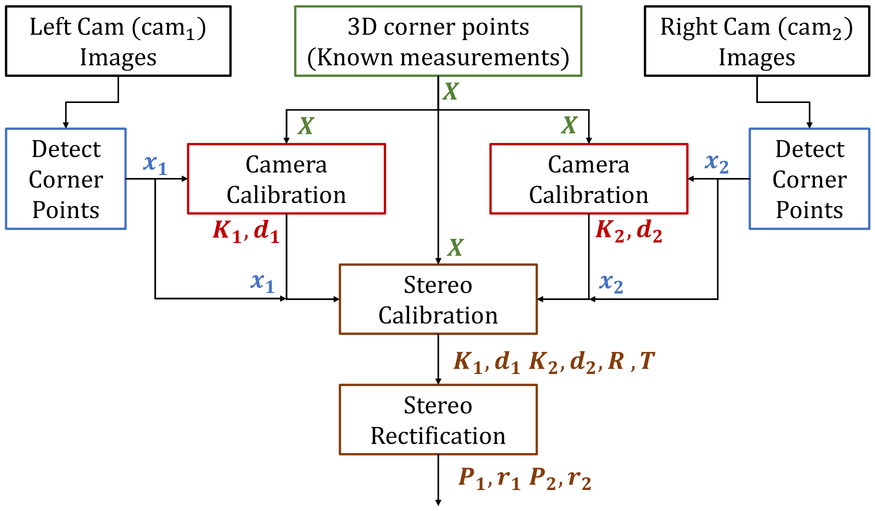


Figure 4.4: **Camera calibration flow using OpenCV.** Corner point correspondences between the world frame and camera images are used to recover camera parameters via optimization function calls of Camera Calibration, Stereo Calibration and Stereo Rectification.

Algorithm 1 Camera Calibration using standard OpenCV functions

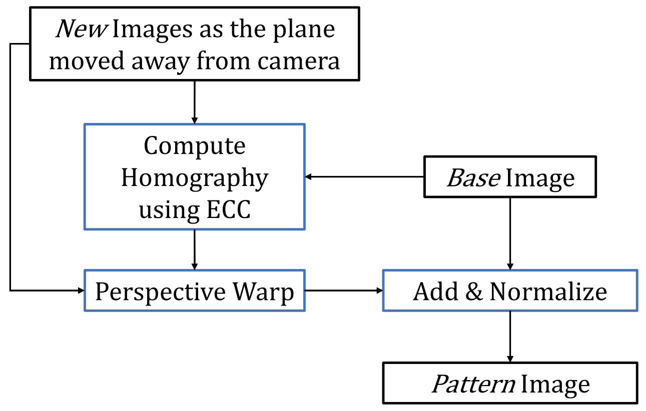
```
$\mathbf{x}_1 \leftarrow$ findChessboardCorners ($cam_1$ images)
$\mathbf{x}_2 \leftarrow$ findChessboardCorners ($cam_2$ images)
$\mathbf{X} \leftarrow$ 3D world coordinates for checkerboard corners
$\mathbf{K}_1, \mathbf{d}_1 \leftarrow$ calibrateCamera$(\mathbf{X}, \mathbf{x}_1)$
$\mathbf{K}_2, \mathbf{d}_2 \leftarrow$ calibrateCamera$(\mathbf{X}, \mathbf{x}_2)$
$\mathbf{K}_1, \mathbf{d}_1, \mathbf{K}_2, \mathbf{d}_2, \mathbf{R}, \mathbf{T} \leftarrow$ stereoCalibrate$(\mathbf{X}, \mathbf{x}_1, \mathbf{x}_2, \mathbf{K}_1, \mathbf{d}_1, \mathbf{K}_2, \mathbf{d}_2)$
$\mathbf{P}_1, \mathbf{P}_2, \mathbf{r}_1, \mathbf{r}_2 \leftarrow$ stereoRectify$(\mathbf{K}_1, \mathbf{d}_1, \mathbf{K}_2, \mathbf{d}_2, \mathbf{R}, \mathbf{T})$
```

4.3.2 Projector Calibration

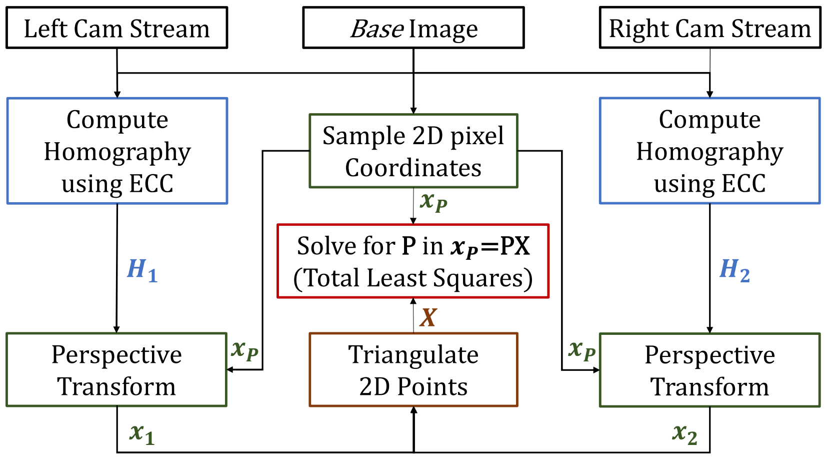
The next step is to calibrate the projector itself. The stream for both cameras is enabled (with projector *on*) at a resolution of 848×480 pixels. We capture the pattern projected onto a white plane as it is moved at different depths, and kept roughly parallel to the camera plane. We assume the scene contains no ambient illumination (*i.e.*, the images are captured in a dark room).

Since the camera is imaging a planar surface, the structured light pattern formed on the sensor is related to all other frames through a homography transform. The key idea is to compute homography between any two images of the pattern using an image alignment technique such as Enhanced Correlation Coefficient (ECC) Maximization [12]. This homography can then be used to warp one image onto another or transform the pixel coordinates of an image to corresponding coordinates in the other image.

The calibration flow is depicted in fig. 4.5 and the pseudocode in algorithm 2. Assigning the first frame to be the *base* image, we estimate its homography with respect to all other captured frames from both cameras. Using this transform, each image is warped onto the base image to formulate the complete *pattern* image (fig. 4.6) with some additional normalization. Similarly, the homographies are used to transform the 2D image coordinates sampled in the *pattern* image to corresponding coordinates in the left and right camera images. These 2D coordinates from both camera images are then triangulated to 3D world coordinates using the calibrated camera projection matrices (obtained in section 4.3.1). Repeating this for all images, we get a set of correspondences between the 2D projector *pattern* coordinates and 3D *world* coordinates, which allows us to solve for the projector's projection matrix using Singular Value Decomposition (SVD) for Total Least Squares. Note that this procedure does not account for effects like projector defocus. Additionally, we postprocess the pattern to remove any contribution from the fall-off in pattern intensities as we account for this factor explicitly in our image formation model.


New Images as the plane moved away from camera
Compute Homography using ECC
Base Image
Perspective Warp
Add & Normalize
Pattern Image

(a) Calibrating Projector Pattern



(b) Calibrating Projection Matrix

Figure 4.5: **Projector calibration flow.** (a) Warping each frame onto a *base* image using an estimated Homography formulates the *pattern* image. (b) Correspondences between the projector's *base* image pixels and the triangulated world coordinates are used to solve for the projector's projection matrix.

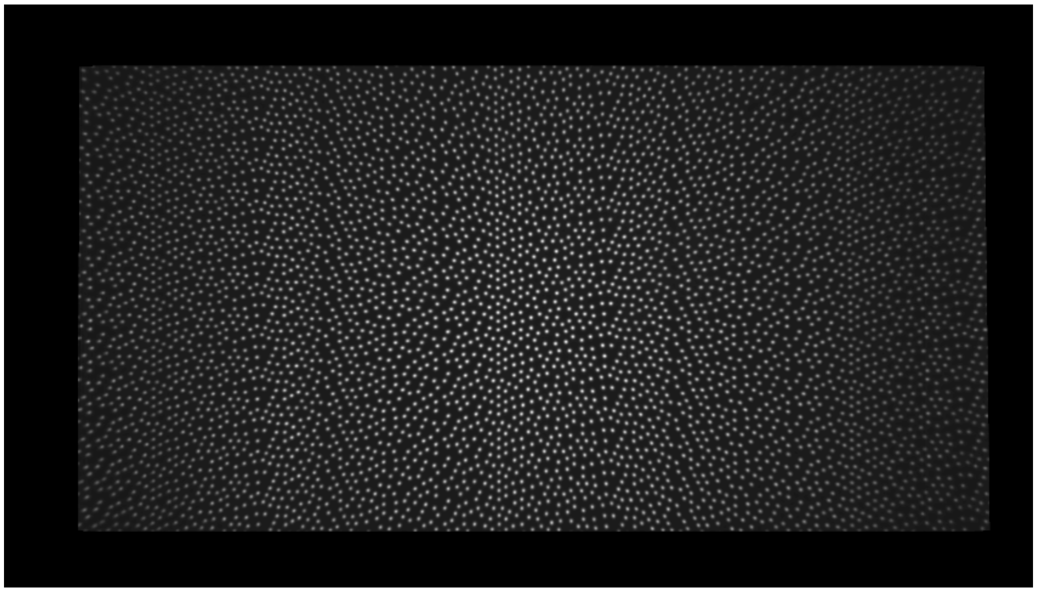


Figure 4.6: **Calibrated projector pattern for Intel RealSense D435.**

4.3.3 Pose Optimization

For each scene, we also use the stereo camera system to compute poses through COLMAP [32] for frames where the projector is *off*. As the baseline between cameras is known, there is no scale ambiguity associated with the poses. Note that the stereo camera pair is used only for calibration, however all experiments demonstrated in chapter 5 use the captures from a single camera only.

During the stroboscopic streaming of the camera, there can be significant motion between the projector *on* and *off* images. Hence, the poses for *off* images obtained via COLMAP cannot be used directly for the *on* images. To calibrate for the different poses, we enable a one-shot pose-optimization flow as a pre-processing step. This allows us to leverage large number of views for accurate pose recovery irrespective of sparse-view training during our experiments. Representing the poses as twists, we initialize the *on* image poses same as the *off* image poses from COLMAP. Then, the *on* image poses across all captures of a scene are optimized using the total photometric loss of eq. (3.15) for a total of 160K iterations. For optimizing these poses, we use Adam optimizer with an initial learning rate of 0.001, which is decayed by a rate of 0.1 over the first 100K iterations. For the volumetric scene model, we use the same optimizer and learning rate as in NeRF [24], while the weight λ_2 is decayed from 1 to 0.001 over the first 100K iterations. Prediction and loss on normals is disabled during this optimization process *i.e.*, $\lambda_2 = \lambda_3 = 0$ in eq. (3.15) to speed-up the calibration. Going forward, we use these per-scene calibrated poses for all our experiments without any further need of pose optimization during training or testing.

Algorithm 2 Projector Calibration using standard OpenCV functions

```
$stream_1, stream_2 \leftarrow$ load stream from both cameras as grayscale images
$stream_1, mask_1, stream_2, mask_2 \leftarrow$ undistort, rectify images using $\mathbf{d}_1, \mathbf{d}_2, \mathbf{r}_1, \mathbf{r}_2$
$base \leftarrow$ first frame from $stream_1$
$pattern,\ mask \leftarrow 0$
$\mathbf{x}_p \leftarrow 0$ (2D $pattern$ coordinates)
$\mathbf{X} \leftarrow 0$ (3D $world$ coordinates)
$\mathbf{H}_1 \leftarrow \mathbf{I}_{2\times3}$
$\mathbf{H}_2 \leftarrow \begin{bmatrix} 1 & -0.005 & 0 \\ 0 & 1 & -48.9 \end{bmatrix}$
for all $\{img_1, img_2\} \in \{stream_1, stream_2\}$ do
    $\mathbf{H}_1 \leftarrow$ findTransformECC($base, img_1, \mathbf{H}_1, mask_1$)
    $rimg_1 \leftarrow$ warpPerspective($img_1, \mathbf{H}_1^{-1}$)
    $rmask_1 \leftarrow$ warpPerspective($mask_1, \mathbf{H}_1^{-1}$)
    $pattern \leftarrow pattern + rimg_1 * rmask_1$
    $mask \leftarrow mask + rmask_1$

    $\mathbf{H}_2 \leftarrow$ findTransformECC($base, img_2, \mathbf{H}_2, mask_2$)
    $rimg_2 \leftarrow$ warpPerspective($img_2, \mathbf{H}_2^{-1}$)
    $rmask_2 \leftarrow$ warpPerspective($mask_2, \mathbf{H}_2^{-1}$)
    $pattern \leftarrow pattern + rimg_2 * rmask_2$
    $mask \leftarrow mask + rmask_2$

    $\{u, v\} \leftarrow$ sample image coordinates on the $pattern$ image
    $\{u_1, v_1\} \leftarrow$ perspectiveTransform($\{u, v\}, \mathbf{H}_1$)
    $\{u_2, v_2\} \leftarrow$ perspectiveTransform($\{u, v\}, \mathbf{H}_2$)
    $\{X, Y, Z\} \leftarrow$ triangulatePoints($\{u_1, v_1\}, \mathbf{P}_1, \{u_2, v_2\}, \mathbf{P}_2$)
    $\mathbf{x}_p \leftarrow$ append $\{u, v\}$ points
    $\mathbf{X} \leftarrow$ append $\{X, Y, Z\}$ points
end for
$pattern = 0.5 \times (pattern/mask)$
Projector's $\mathbf{P} \leftarrow$ Total Least Squares using the point correspondences $\mathbf{x}_p$ and $\mathbf{X}$
```

Chapter 5

Experiments and Results

5.1 Novel View Synthesis from Sparse Views

For real scenes, we test the novel-view reconstruction and disparity map of our method trained with 2, 4 and 8 views. Since the goal is to perform accurate novel-view synthesis for the ambient images, we decay the initial structured lighting objective's weight $\lambda_1 = 1$ by a rate of 10^{-1} over the first 40K iterations for the 2-view case, and by 10^{-3} for the 4- and 8-view cases. We compare our model's performance with the following baseline approaches:

NeRF (no depth supervision): We train NeRF [24] with the photometric loss of eq. (3.11) on ambient images only.

NeRF + sparse depth: We train DS-NeRF [11] using COLMAP's sparse point cloud. This adds a depth supervision loss with a weight of 0.1 on top of the photometric loss for its fine network.

NeRF + dense depth: Similar to DS-NeRF [11], we add depth supervision to the fine network of NeRF using the dense RealSense depth maps. We mask out the unresolved regions in the RealSense depth maps to avoid supervising with an unreliable signal. Since the RealSense's active depth sensing comes into play when the projector is *on*, the depth supervision objective is trained for the poses capturing structured light images only. For ambient images, only photometric loss is used as depth from passive stereo is noisier and incomplete. The depth loss weight is decayed at the same rate as λ_1 in our method *i.e.*, by 10^{-1} for 2 views, and 10^{-3} for 4 and 8 views. However, the initial depth loss weight is set to 0.1 (as opposed to 1 for our method) to approximately account for the depth range normalization.

We perform this comparative study on 50 to 100 held-out views each for four real scenes: *woodshop*, *doll*, *sculpture*, and *translucent box*. For all the cases, we train using a ray batch size of 2048, 32 uniform samples, and 64 importance samples for 100k iterations. For the *translucent box* scene, we use 64 uniform samples across all methods. For these comparisons, we drop the explicit prediction of normals and the indirect component in our model, and focus on the analysis of novel-view and geometry reconstruction quality.

We report the quantitative analysis for novel-view synthesis in table 5.1 using PSNR, SSIM [38], and LPIPS [44] metrics. Quantitatively, the explicit depth supervision methods (sparse depth, dense depth) and raw structured light supervision (ours); all produce comparable high-quality representations of the scene, as expected. Per-scene metrics breakdown can be found in appendix A.2.4.

Figures 5.1 and 5.2 present qualitative results for the 2-view case on the *doll* and *woodshop* scenes. The 4-view results can be found in appendix A.2.3. Qualitatively, it is clear that both NeRF and sparse depth supervision struggle to accurately capture the scene geometry, especially for regions with less features, *e.g.*, walls of the *doll* scene. They recover cloudy geometry in such textureless regions and are unable to interpolate well with sparse views. Provided the scenes contain relatively simple geometry such that the RealSense depth is reliable, both dense depth supervision and our proposed method produce comparable results. Additionally, our method depicts cleaner, sharper depth and fewer artifacts near object edges and scene boundaries.

Table 5.1: **Quantitative analysis on real data:** Novel-view synthesis from sparse-views. '2-v' denotes two views.

	PSNR ▲			SSIM ▲			LPIPS ▼		
Method	2-v	4-v	8-v	2-v	4-v	8-v	2-v	4-v	8-v
NeRF	27.53	33.40	38.31	0.886	0.936	0.965	0.399	0.307	0.231
+ sparse depth	**36.74**	**41.22**	**42.38**	0.969	**0.988**	**0.990**	0.214	**0.158**	**0.148**
+ dense depth	36.61	40.74	42.06	**0.971**	0.986	0.988	**0.207**	0.173	0.165
Ours	34.89	40.97	42.02	0.959	0.986	0.989	0.218	0.167	0.163

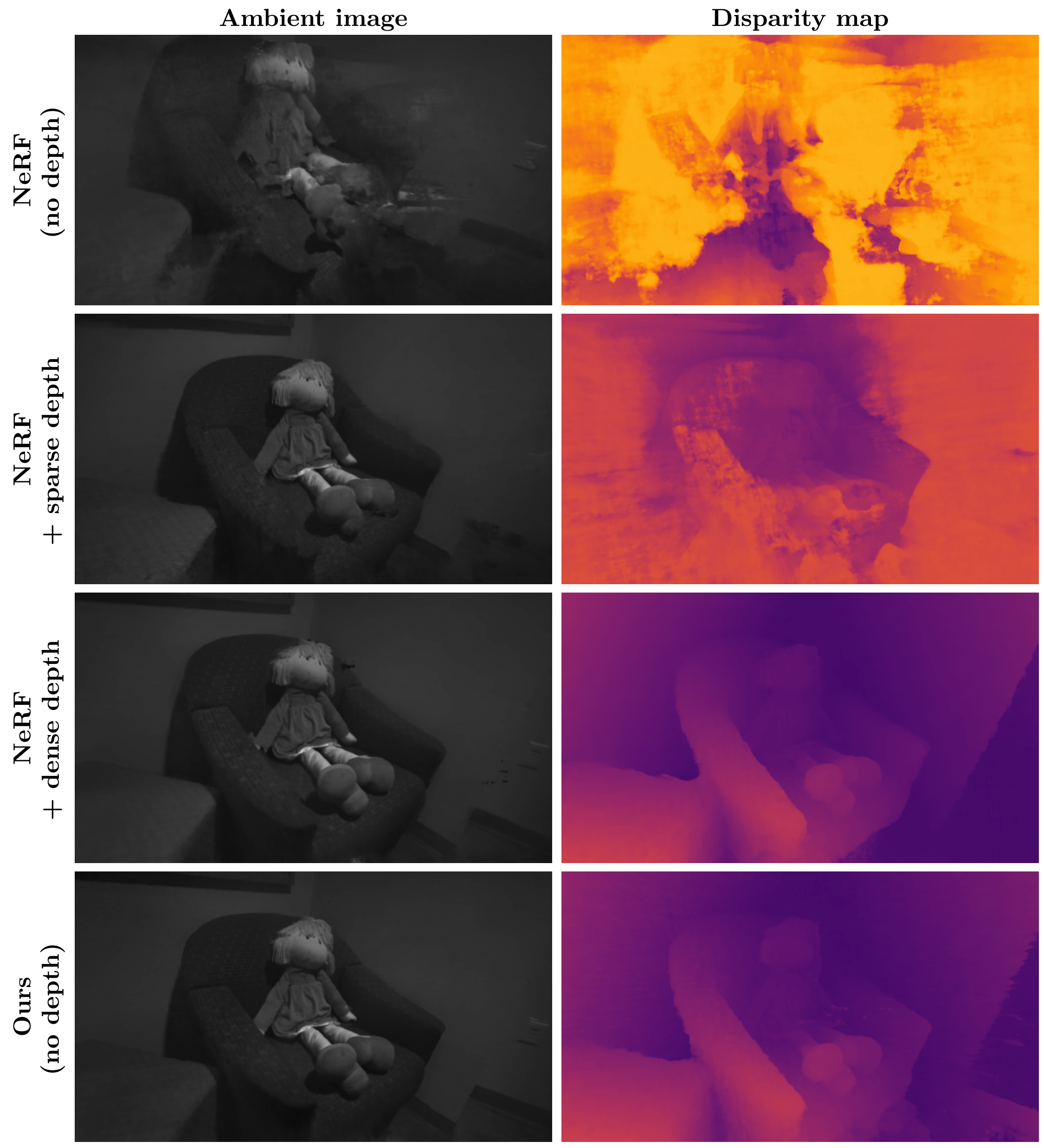


Figure 5.1: **Novel-view reconstruction for the *doll* scene:** Trained with 2 views.

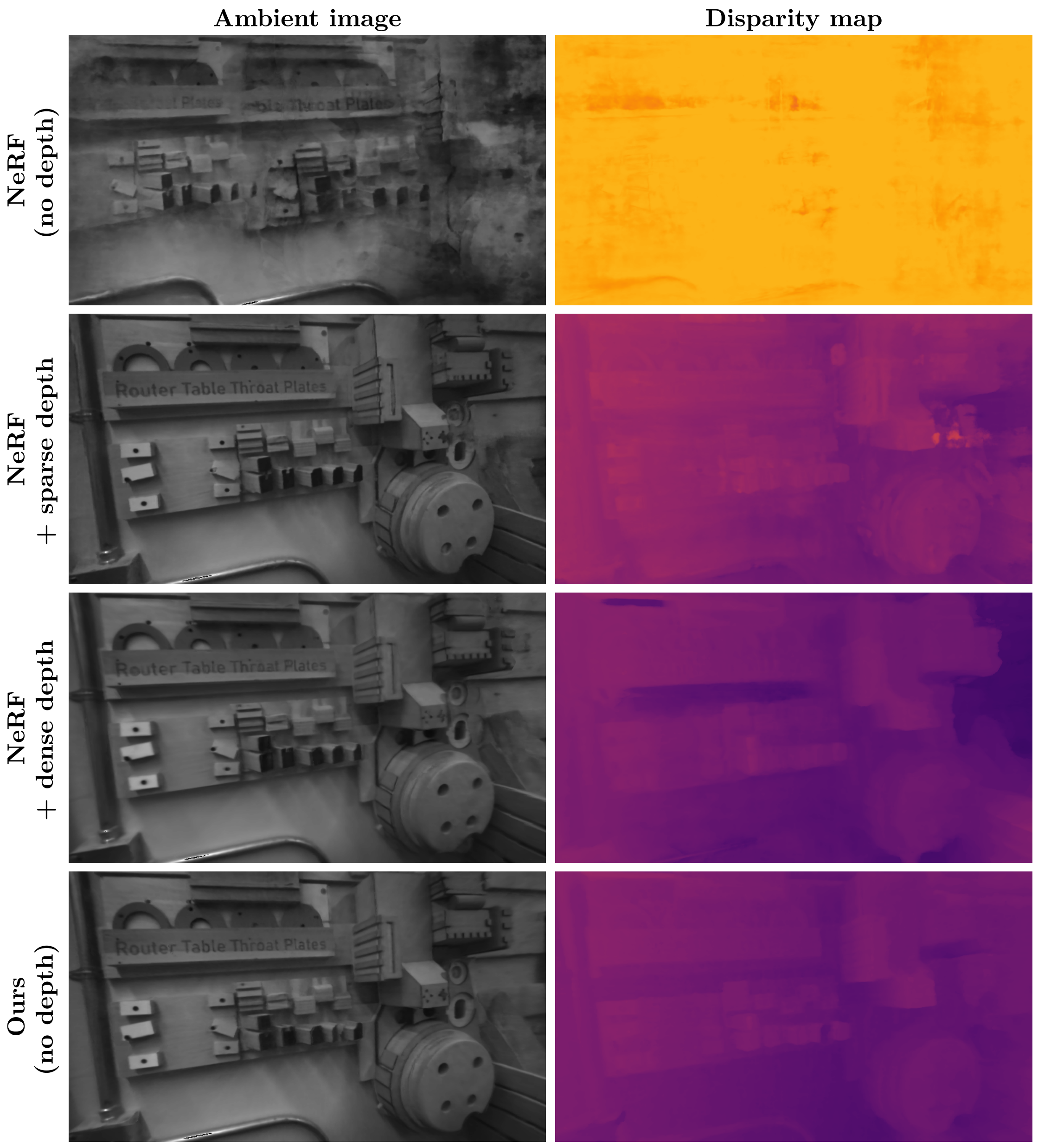


Figure 5.2: **Novel-view reconstruction for the *woodshop* scene:** Trained with 2 views.

5.2 Reconstructing Translucent Objects

Previous section showed that depth sensors can be reliable for supervising scene reconstruction with simple objects. However, there are scenarios where structured light sensors completely fail to capture depth. For example, when imaging translucent objects, multiple depth planes contribute light to a sensor (fig. 5.3b). As a result, recovering a single depth map for such scenes is fundamentally ill-defined.

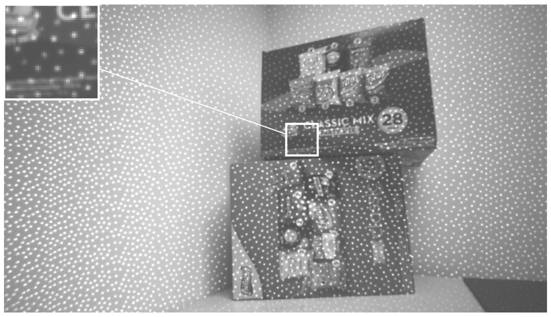
We perform a qualitative comparison of our approach to the depth from Intel RealSense on scenes containing partially transparent objects. In particular, we capture an additional *mesh box* scene containing a plastic translucent mesh placed in front of a table containing a stack of boxes. In fig. 5.3, we show that the Intel RealSense fails to accurately capture the scene's geometry due to the ambiguous correspondences caused by multiple direct reflections. In contrast, our approach has the ability to model the contribution of direct illumination from *multiple points* along the path of a ray, allowing us to capture both the geometry of the mesh and the objects behind it.

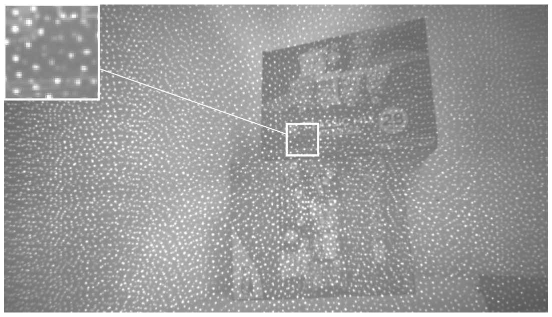
We further provide a qualitative comparison of our approach to 'NeRF + dense depth' in fig. 5.4 on the *translucent box* scene containing a partially transparent plastic box. Here, we visualize the rendering weights w_k (eq. (2.6)) along a ray passing through the plastic box, demonstrating that our method more accurately recovers the geometry of all surfaces (translucent and opaque) along the path of a ray.

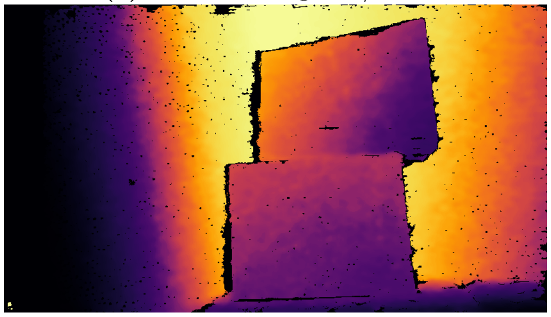
5.3 Predicting Normals

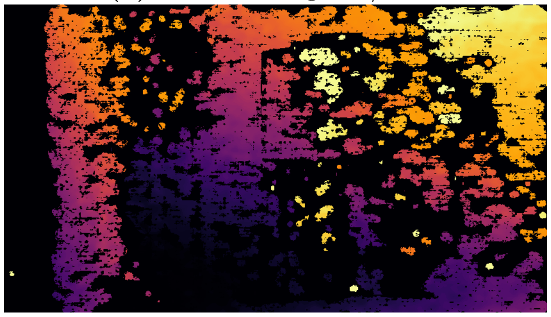
A unique advantage of working with active illumination methods such as modelling the raw structured light images is retrieval of surface normals via shading cues. In this section, we provide quantitative and qualitative assessments of our proposed method in simulation.

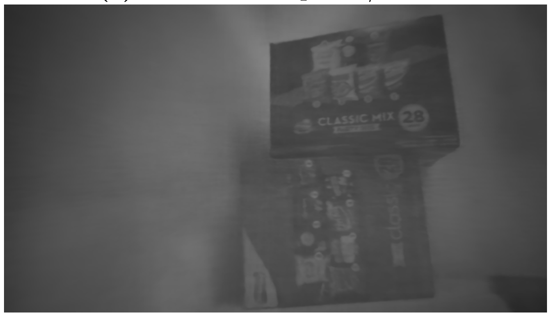
We test the reconstruction of novel views, disparity, and analytical normal maps for the following methods that are representative of different levels of illumination modelling: (i) our structured lighting-based method which models high frequency illumination projected onto the scene, (ii) replacing the structured light (dot) pattern with flood illumination (to mimic the setup by Bi et al. [6]) *i.e.*, a less spatially-


(a) Structured light w/o mesh


(b) Structured light w/ mesh


(c) RealSense Depth w/o mesh


(d) RealSense Depth w/ mesh


(e) Our reconstructed view (behind the mesh)

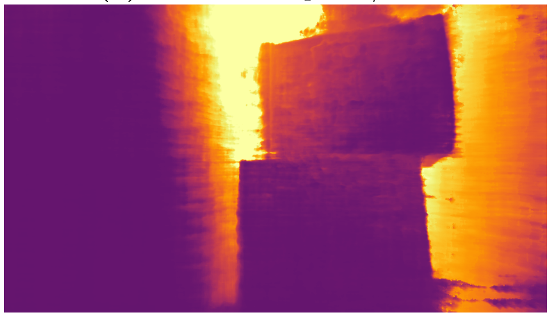

(f) Our depth (behind the mesh)

Figure 5.3: **Reconstructing multiple surfaces for the *mesh box* scene. (a–b)** Structured light images captured by the Intel RealSense without and with a large plastic mesh placed in between the camera and the scene. Note that the partially-transparent mesh obscures the scene and effectively creates a second "copy" of the structured light pattern, leading to ambiguous correspondences (see insets). **(c–d)** We show the RealSense depth for the scene without and with the mesh. Note that the RealSense completely fails to predict reasonable depth in the latter case. **(e–f)** We show a rendered view from our approach, as well as a depth map produced by filtering out the geometry of the mesh. Our method accurately reconstructs the geometry of the background, while the RealSense fails to do so.

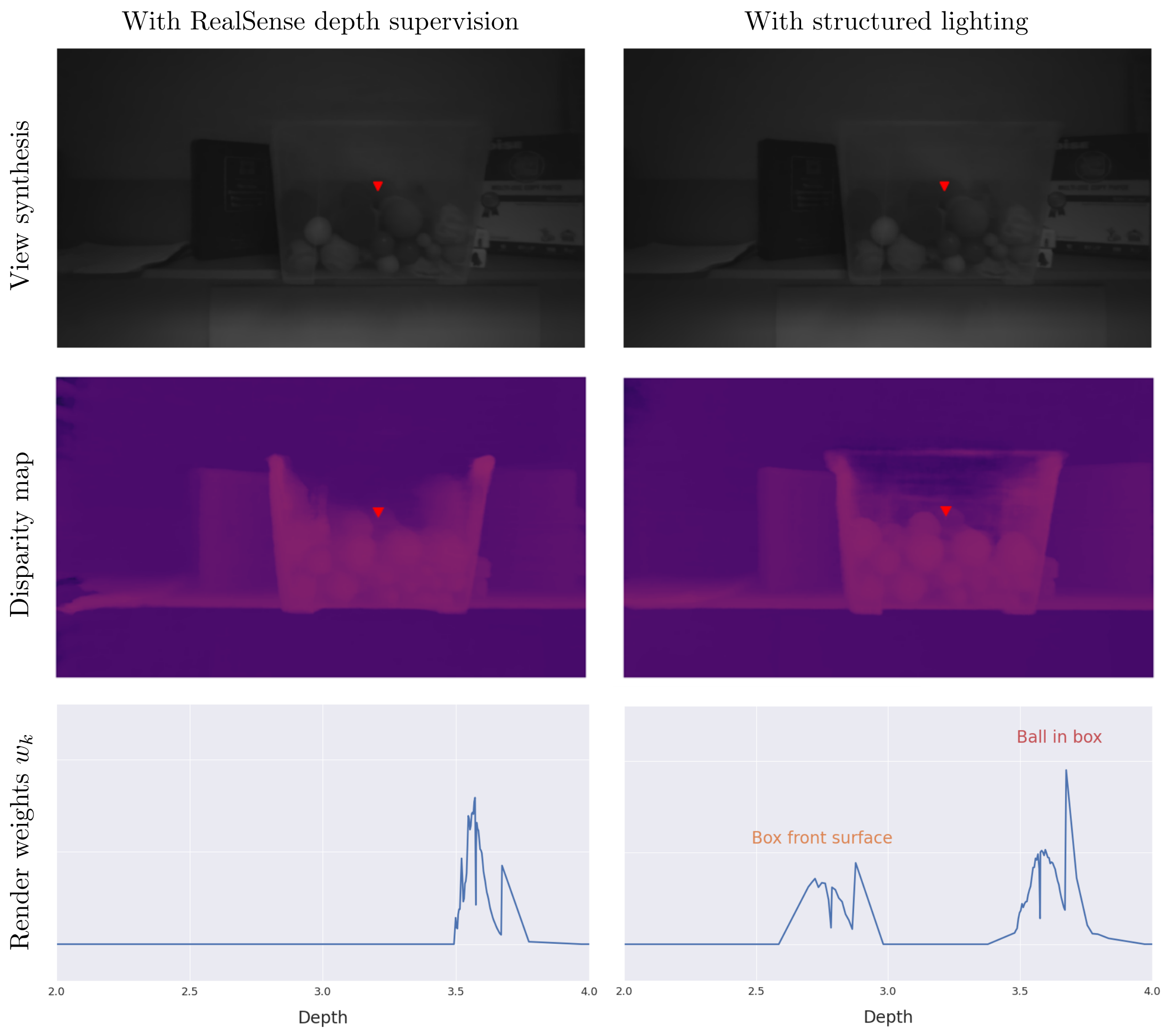


Figure 5.4: **Reconstructing multiple surfaces for the *translucent box* scene.** In this example, we show that our method recovers the geometry of a partially-transparent plastic container, while NeRF with dense depth supervision fails to do so. The red marker indicates in the top two rows the pixel chosen for rendering weight visualization. In particular, as shown in the bottom row, *(Right)* our method recovers the depth of both front surface of the container (first peak), as well as a ball placed within the container (second peak), *(Left)* while NeRF only recovers the depth of the ball.

varying illumination, (ii) the base NeRF model with ambient-only lighting without any explicit introduction or modelling of illumination.

For both the dot pattern and flood illumination, we decay the structured light loss weight λ_1 from 1 to 0.05 over the first 40K iterations, and set the normal prediction and normal orientation loss weights to $\lambda_2 = 3 \times 10^{-4}$ and $\lambda_3 = 0.1$, respectively. All methods are trained for 100K iterations with a 1024 ray batch size, 64 uniform samples and 128 importance samples. We train using 25 views and test on 50 held-out views for 4 scenes. We omit the indirect component of our image formation model here to focus on normal recovery.

For quantitative analysis in table 5.2, we compute PSNR, SSIM [38] and LPIPS [44] on the reconstructed images, MSE (mean squared error) on the depth map, and MAE (mean angular error in degrees) for the normal maps. Per-scene metrics breakdown can be found in appendix A.2.2. Figure 5.5 shows the qualitative analysis for two scenes - *lego* and *skateboard*. More qualitative results can be found in appendix A.2.1. Both quantitatively and qualitatively, the use of a structured light dot pattern significantly outperforms the case of single intensity flood illumination or no active illumination (base NeRF) in terms of normal and depth fidelity.

Explicit prediction of normals is useful as the MLP is capable of smoother interpolation of normals compared to the noisy gradients of volume density. As shown in fig. 5.6, explicitly predicting normals, modelling them into the shading term along with the normals loss eq. (3.13) between the analytical and predicted normals helps improve the quality of normals, making them less noisy while capturing detail.

Table 5.2: **Quantitative analysis on synthetic data:** Structured light reduces depth and normal error significantly while maintaining novel-view synthesis quality.

Method	PSNR ▲	SSIM ▲	LPIPS ▼	Depth MSE ▼	Normals MAE °▼
NeRF	**44.98**	**0.985**	**0.368**	0.615	24.34
Flood Light	43.51	0.982	0.375	0.786	8.76
Structured Light	44.13	0.984	0.370	**0.013**	**2.84**

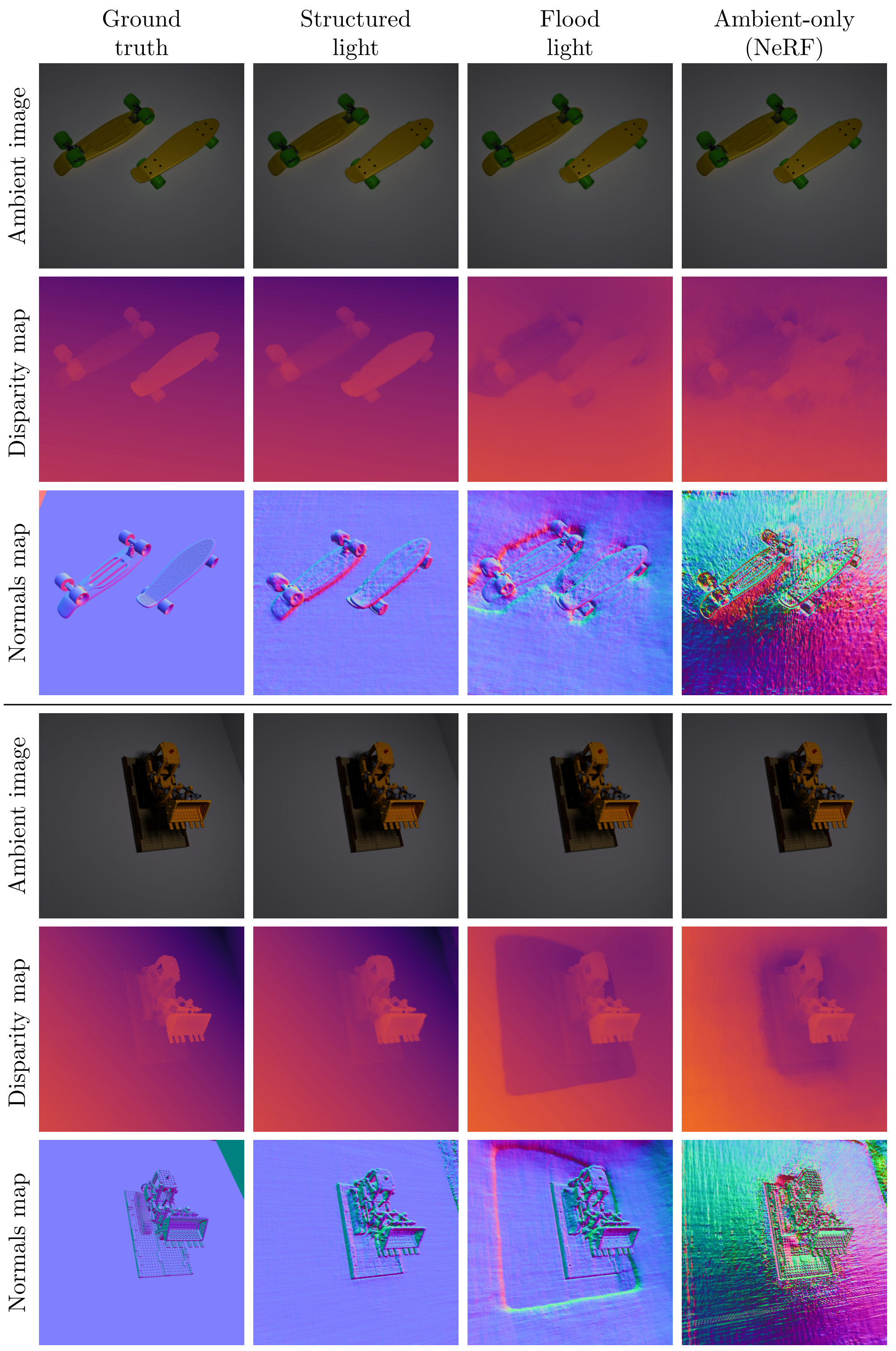


Figure 5.5: **Qualitative analysis on synthetic data:** Novel view synthesis, disparity and analytical normal maps for *lego* and *skateboard* scenes.

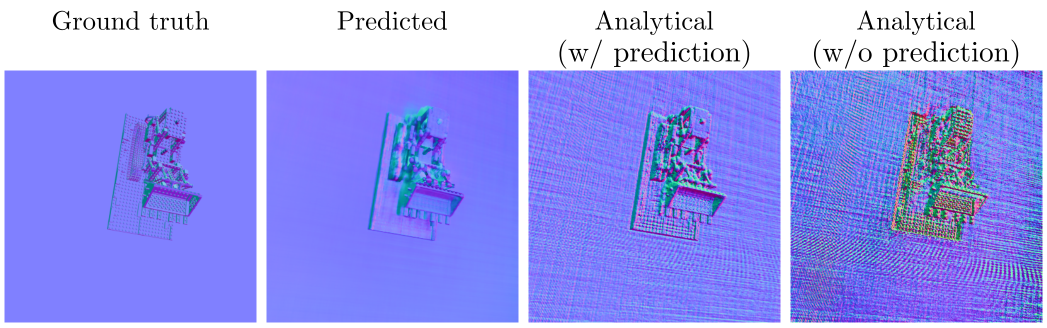


Figure 5.6: **Impact of explicitly predicting, modelling and penalizing normals.** Normals map visualization for a test view from the *lego* scene.

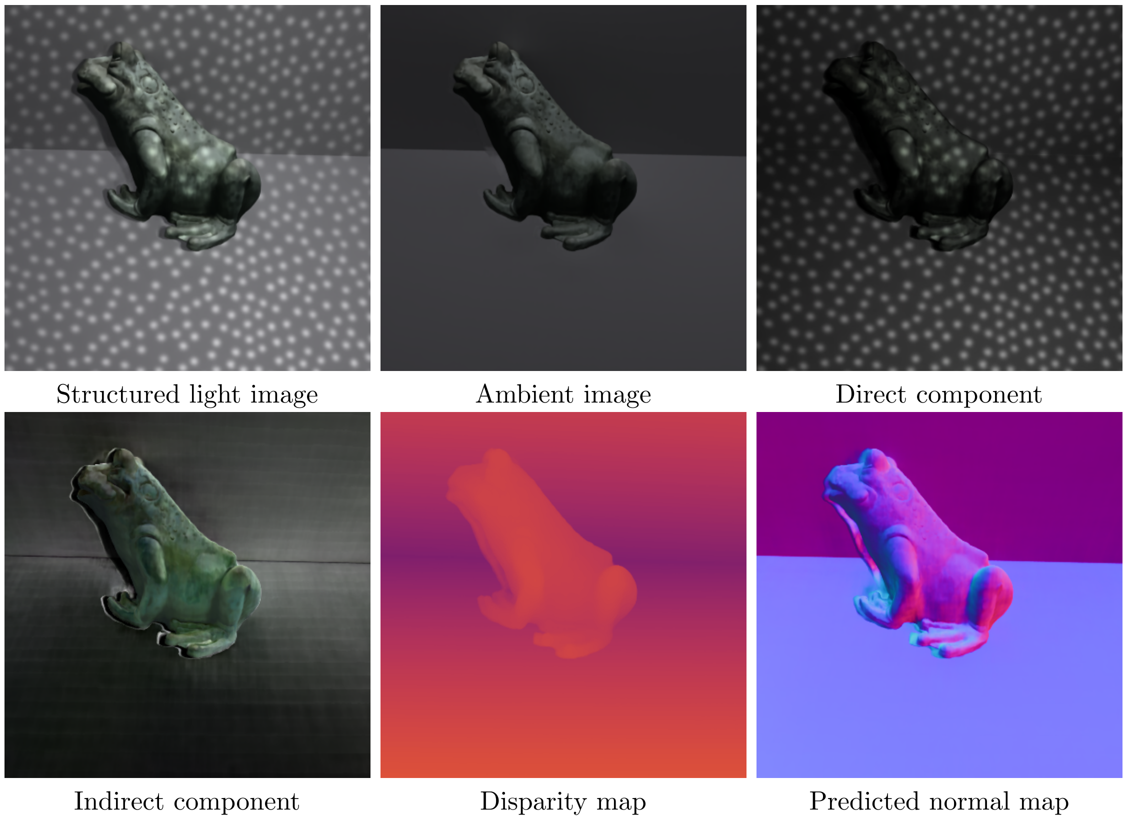
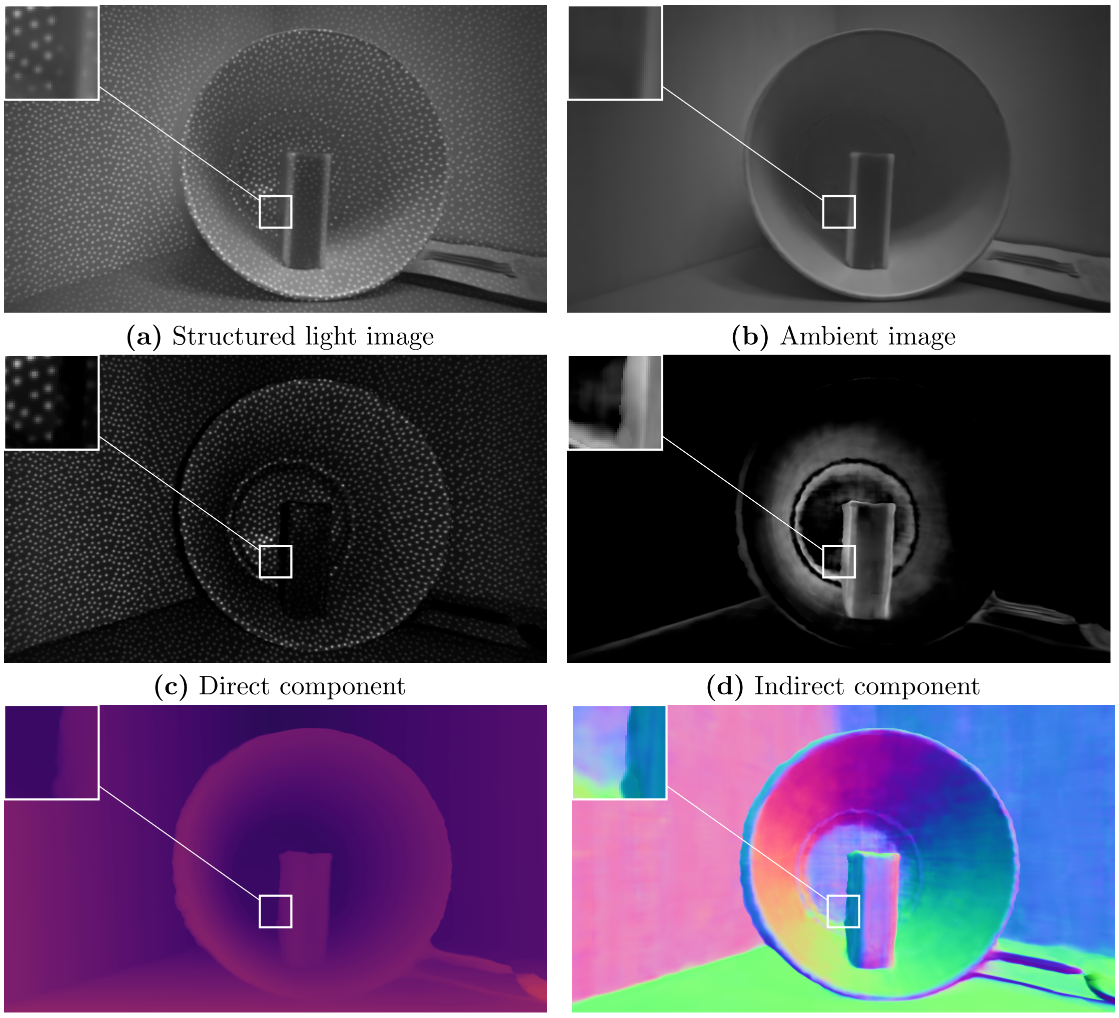


Figure 5.7: **Novel-view scene decomposition on a synthetic scene.** Our method enables the synthesis of the ambient, direct, and indirect appearance along with accurate estimation of the normals and disparity map for novel views by physically-based modeling of scenes exhibiting complex light interactions like sub-surface scattering and inter-reflections.



(a) Structured light image **(b)** Ambient image

(c) Direct component **(d)** Indirect component

(e) Disparity map **(f)** Predicted normal map

Figure 5.8: **Scene decomposition of a real scene from a novel viewpoint.** Our proposed framework uses the raw measurements from a single infrared camera on an Intel RealSense to generate a volumetric representation of the scene, and synthesizes images **(a–f)** from a novel camera viewpoint, showing the different representations of shape and appearance recovered using our proposed framework.

5.4 Decomposing Scene Appearance

Finally, we showcase the effectiveness of modeling the direct and indirect radiance, and produce a complete decomposition of all components using our framework. This includes the ambient, direct, and indirect components of the scene appearance, and the disparity and normals associated with scene geometry.

We first construct a synthetic scene with a frog object, whose skin exhibits subsurface scattering. The frog is positioned close to the intersection of two planes, which also introduces diffuse inter-reflections. Using the same hyperparameters as described in section 5.3, we show the ability to synthesize scenes from a novel viewpoint, and decompose the scene into its constituent components in fig. 5.7.

To demonstrate this in practice, we capture a real scene in fig. 5.8, consisting of a translucent candle placed within a large concave object. This scene exhibits strong subsurface scattering and inter-reflections. We train our model using 8 views, a 2048 batch size, 32 uniform samples, and 64 importance samples for 100K iteration. The structured light weight λ_1 is decayed to 0.1 over 40K iterations, and the normal prediction and orientation loss weights are set to $\lambda_2 = 0.001$ and $\lambda_3 = 0.1$, respectively. The result is a decomposition of all components of the scene, accurately capturing the presence of indirect light within this scene.

Chapter 6

Conclusions

In this paper, we proposed a neural volume rendering framework for multi-view structured lighting. This framework recovers accurate geometry and synthesizes novel views by modelling the image formation process for a commodity Intel RealSense structured light system. We demonstrated that our physically-based framework provides a more principled approach to recovering scene geometry, enabling it to account for challenging scenes that contain partially transparent objects. Moreover, our ability to model the raw structured light images further enables our method to recover accurate surface normals, and to separate direct and indirect components.

6.1 Limitations

As with existing structured light systems, our method can be confused by objects that produce complex light transport effects. For example, recovering scene geometry in the presence of mirrors and refractive objects is a notoriously difficult problem [14, 28].

When imaging outdoors under bright ambient lighting or imaging objects placed far away, the illumination from the projector may be too weak to detect. In such cases, however, our proposed method is expected to fail gracefully and have similar performance to NeRF [24], because the ambient photometric loss dominates the structured light photometric loss.

While the overall quality of novel-view synthesis of our method is similar to other methods, our framework is unable to accurately disambiguate geometry for edges near projector shadows.

Additionally, our current illumination pattern calibration does not account for effects like projector defocus which can cause some inconsistencies between the ground truth and modeled illumination.

6.2 Future Work

Looking forward, we believe that this framework can be extended to make use of all three cameras on RealSense devices, and potentially even support scanning scenes with multiple such devices in tandem. Using a sparse set of such easily accessible devices could significantly reduce the complexity of 3D acquisition setups that currently rely on large-scale lights and cameras.

Another future enhancement is to explicitly model illumination visibility to improve reconstruction in shadow areas and enable photorealistic relighting. Optical techniques can also be employed to further mitigate the effect of shadows from images [25]. Additionally, the proposed model can be extended to incorporate material estimation and editable geometric representations, *e.g.*, meshes, that are compatible with standard 3D rendering setups. With the prevalence of active depth sensors in millions of consumer devices, such a framework could enable several downstream applications for AR/VR using just a mobile phone.

To conclude, with further enhancements, the proposed method could take us one step closer to facilitating simplified content creation and machine vision pipelines.

Appendix A

Appendix

A.1 Blender Scenes: License Information

Out of the 5 Blender scenes in our dataset, 4 were created using open source 3D models available at [1]. Table A.1 consolidates the license information for the same.

Table A.1: **License information for open-source Blender scenes.**

Scene	License	Author	Link
Sofa	CC-0	Darilon	https://blendswap.com/blend/30053
Frog	CC-0	craggle	https://blendswap.com/blend/30092
Skateboard	CC-0	MattMump	https://blendswap.com/blend/4859
Domino	CC-0	alepx	https://blendswap.com/blend/6584

A.2 Additional Results

A.2.1 Qualitative Results on Synthetic Data

Figure A.1 demonstrates visualizations on additional scenes - *sofa* and *domino*, for the experiments conducted in section 5.3.

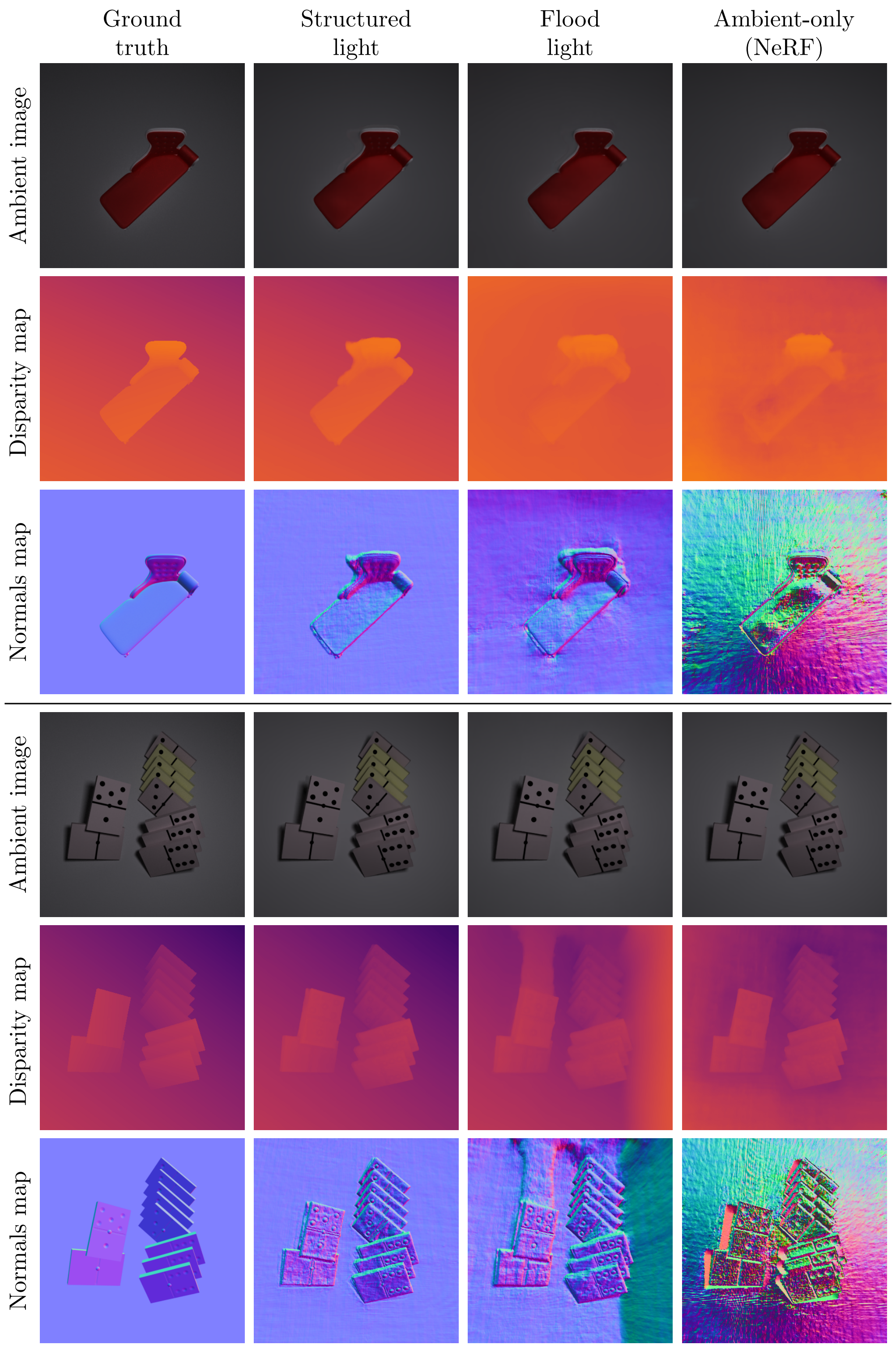


Figure A.1: **Qualitative analysis on synthetic data:** Novel view synthesis, disparity and analytical normal maps for *sofa* and *domino* scenes.

A.2.2 Quantitative Results on Synthetic Data

Table A.2 shows the per-scene breakdown of metrics for synthetic scenes. Here, some scene names are abbreviated as - *skateboard (sb)* and *domino (dom)*. These experiments use the same hyperparameters and settings as defined in section 5.3.

Table A.2: **Scene-wise quantitative analysis on synthetic data.**

	Method	PSNR ▲	SSIM ▲	LPIPS ▼	Depth MSE ▼	Normals MAE °▼
lego	NeRF	**43.62**	**0.984**	**0.407**	0.915	23.90
	Flood Light	41.47	0.979	0.418	0.574	7.95
	Structured Light	42.62	0.982	0.411	**0.008**	**2.61**
sb	NeRF	**45.49**	**0.985**	**0.323**	0.409	24.62
	Flood Light	44.32	0.983	0.327	0.632	7.30
	Structured Light	44.68	0.984	0.325	**0.004**	**2.71**
sofa	NeRF	**46.22**	**0.986**	**0.389**	0.846	23.33
	Flood Light	45.17	0.985	0.395	1.096	7.28
	Structured Light	45.59	**0.986**	**0.389**	**0.033**	**2.40**
dom	NeRF	**44.58**	**0.984**	**0.353**	0.291	25.51
	Flood Light	43.09	0.981	0.358	0.841	12.50
	Structured Light	43.65	0.983	0.355	**0.008**	**3.61**

A.2.3 Qualitative Results for Sparse-view Reconstruction

Appendix A.2.3 visualize the 4-view reconstruction results for the *doll* and *woodshop* scenes, as discussed in section 5.1.

A.2.4 Quantitative Results for Sparse-view Reconstruction

Tables A.3 to A.6 show the sparse-view reconstruction metrics for real scenes. These experiments use the same hyperparameters and settings as defined in section 5.1.

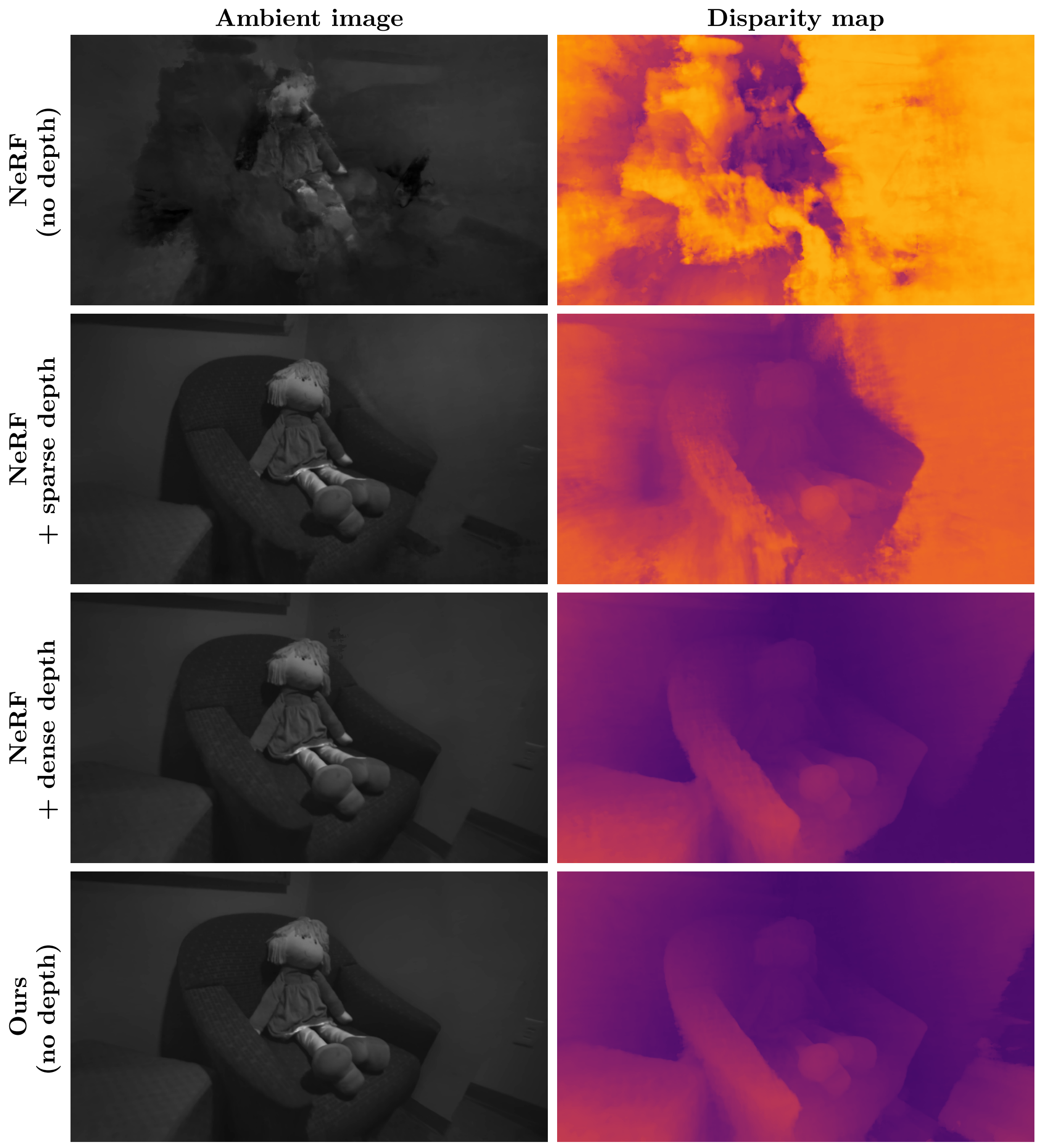


Figure A.2: **Novel-view reconstruction for the *doll* scene:** Trained with 4 views.

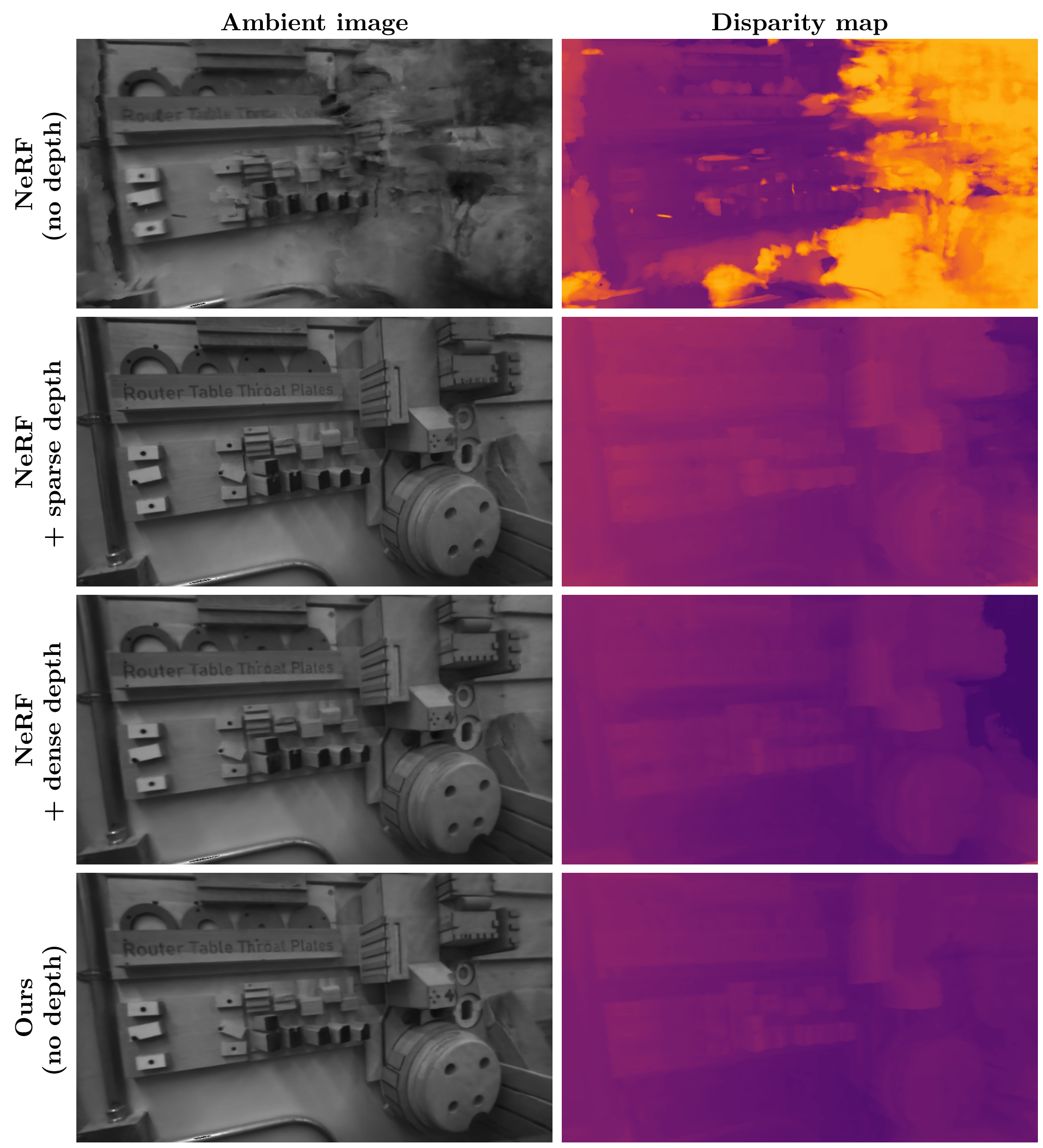


Figure A.3: **Novel-view reconstruction for the *woodshop* scene:** Trained with 4 views.

Table A.3: **Quantitative analysis on *doll* scene.**

	PSNR ▲			SSIM ▲			LPIPS ▼		
Method	2-v	4-v	8-v	2-v	4-v	8-v	2-v	4-v	8-v
NeRF	29.51	32.14	39.37	0.925	0.938	0.980	0.381	0.369	0.261
+ sparse depth	39.01	39.88	41.21	0.977	0.983	0.986	0.254	0.225	0.211
+ dense depth	39.77	41.54	**41.87**	0.983	0.987	**0.988**	0.212	0.202	**0.199**
Ours	**39.90**	**41.85**	41.74	**0.984**	**0.988**	**0.988**	**0.204**	**0.197**	0.200

Table A.4: **Quantitative analysis on *woodshop* scene.**

	PSNR ▲			SSIM ▲			LPIPS ▼		
Method	2-v	4-v	8-v	2-v	4-v	8-v	2-v	4-v	8-v
NeRF	24.14	27.50	31.67	0.778	0.860	0.917	0.480	0.400	0.269
+ sparse depth	36.92	**38.45**	**38.39**	0.969	**0.984**	**0.986**	**0.144**	**0.115**	**0.101**
+ dense depth	37.19	37.64	38.17	0.970	0.979	0.983	0.163	0.149	0.140
Ours	**37.60**	37.65	38.37	**0.973**	0.977	0.982	0.147	0.146	0.138

Table A.5: **Quantitative analysis on *sculpture* scene.**

	PSNR ▲			SSIM ▲			LPIPS ▼		
Method	2-v	4-v	8-v	2-v	4-v	8-v	2-v	4-v	8-v
NeRF	25.68	38.39	43.83	0.899	0.984	0.991	0.344	0.161	0.141
+ sparse depth	32.54	**43.15**	**44.42**	0.950	**0.992**	**0.993**	0.226	**0.125**	**0.117**
+ dense depth	**33.17**	41.30	43.30	**0.958**	0.987	0.991	**0.202**	0.146	0.142
Ours	27.17	40.96	43.15	0.911	0.988	0.991	0.277	0.145	0.141

Table A.6: **Quantitative analysis on *translucent box* scene.**

	PSNR ▲			SSIM ▲			LPIPS ▼		
Method	2-v	4-v	8-v	2-v	4-v	8-v	2-v	4-v	8-v
NeRF	30.78	35.57	38.39	0.943	0.964	0.971	0.392	0.298	0.252
+ sparse depth	**38.50**	**43.41**	**45.51**	**0.979**	**0.991**	**0.993**	**0.233**	**0.166**	**0.162**
+ dense depth	36.31	42.47	44.89	0.972	0.989	0.992	0.252	0.194	0.177
Ours	34.90	43.40	44.82	0.967	**0.991**	**0.993**	0.244	0.179	0.171

Bibliography

[1] Blendswap 3d models. https://blendswap.com/. 4.1, A.1

[2] Opencv camera calibration. https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html. 4.3.1

[3] Apple Inc. Face ID advanced technology. https://support.apple.com/en-us/HT208108, 2022. 1.1, 2.2

[4] Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O'Toole. TöRF: Time-of-flight radiance fields for dynamic scene view synthesis. In *NeurIPS*, 2021. 2.3

[5] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural RGB-D surface reconstruction. In *CVPR*, 2022. 2.2

[6] Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. arXiv:2008.03824, 2020. 2.3, 5.3

[7] Blender Online Community. *Blender – a 3D modelling and rendering package*. Blender Foundation, 2023. URL https://www.blender.org/. 4.1

[8] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T Barron, Ce Liu, and Hendrik Lensch. NeRD: Neural reflectance decomposition from image collections. In *ICCV*, 2021. 2.3

[9] Ilya Chugunov, Yuxuan Zhang, Zhihao Xia, Xuaner Zhang, Jiawen Chen, and Felix Heide. The implicit values of a good hand shake: Handheld multi-frame neural depth refinement. In *CVPR*, 2022. 2.2

[10] 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*, pages 303–312, 1996. 2.1

[11] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer views and faster training for free. In *CVPR*, 2022. 2.2, 2.2, 5.1

[12] Georgios D Evangelidis and Emmanouil Z Psarakis. Parametric image alignment using enhanced correlation coefficient maximization. *IEEE transactions on*

pattern analysis and machine intelligence, 30(10):1858–1865, 2008. 4.3.2

[13] Jason Geng. Structured-light 3D surface imaging: a tutorial. *Adv. Opt. Photon.*, 3 (2):128–160, Jun 2011. doi: 10.1364/AOP.3.000128. URL `https://opg.optica.org/aop/abstract.cfm?URI=aop-3-2-128`. 1.1, 2.3.1

[14] Mohit Gupta, Amit Agrawal, Ashok Veeraraghavan, and Srinivasa G Narasimhan. Structured light 3d scanning in the presence of global illumination. In *CVPR 2011*, pages 713–720. IEEE, 2011. 1.1, 6.1

[15] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting NeRF on a diet: Semantically consistent few-shot view synthesis. In *ICCV*, 2021. 2.2

[16] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH computer graphics*, 18(3):165–174, 1984. 2.1.2

[17] Berk Kaya, Suryansh Kumar, Carlos Oliveira, Vittorio Ferrari, and Luc Van Gool. Uncalibrated neural inverse rendering for photometric stereo of general surfaces. In *CVPR*, 2021. 2.3

[18] Berk Kaya, Suryansh Kumar, Francesco Sarno, Vittorio Ferrari, and Luc Van Gool. Neural radiance fields approach to deep multi-view photometric stereo. In *WACV*, 2022. 2.3

[19] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel RealSense stereoscopic depth cameras. In *CVPR Workshops*, 2017. 1.1, 1.2, 2.2, 3.1, 4.2

[20] Marc Levoy. Efficient ray tracing of volume data. *ACM Transactions on Graphics (TOG)*, 9(3):245–261, 1990. 2.1.2

[21] Junxuan Li and Hongdong Li. Self-calibrating photometric stereo by neural inverse rendering. In *ECCV*, 2022. 2.3

[22] Linjie Lyu, Ayush Tewari, Thomas Leimkühler, Marc Habermann, and Christian Theobalt. Neural radiance transfer fields for relightable novel-view synthesis with global illumination. In *ECCV*, 2022. 2.3

[23] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. 2.1.2, 2.1.2

[24] 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. *Communications of the ACM*, 65(1):99–106, 2021. 1.1, 1.2, 2, 2.1, 2.1.2, 3.3, 3.4, 4.1, 4.3.3, 5.1, 6.1

[25] Shree K Nayar and Mohit Gupta. Diffuse structured light. In *2012 IEEE international conference on computational photography (ICCP)*, pages 1–11. IEEE, 2012. 6.2

[26] Shree K Nayar, Gurunandan Krishnan, Michael D Grossberg, and Ramesh

Raskar. Fast separation of direct and global components of a scene using high frequency illumination. In *SIGGRAPH*, pages 935–944, 2006. 3.3

[27] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. RegNeRF: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022. 2.2

[28] Matthew O'Toole, John Mather, and Kiriakos N Kutulakos. 3D shape and indirect appearance by structured light transport. In *CVPR*, 2014. 1.1, 6.1

[29] Barbara Roessle, Jonathan T Barron, Ben Mildenhall, Pratul P Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *CVPR*, 2022. 2.2

[30] Joaquim Salvi, Jordi Pagès, and Joan Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4):827–849, 2004. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2003.10.002. URL `https://www.sciencedirect.com/science/article/pii/S0031320303003303`. Agent Based Computer Vision. 1.1, 2.3.1

[31] Jonas Schell. Projector add-on for Blender. `https://github.com/Ocupe/Projectors`, 2023. 4.1

[32] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1.1, 2.2, 4.3.3

[33] Pratul P Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T Barron. Nerv: Neural reflectance and visibility fields for relighting and view synthesis. In *CVPR*, 2021. 2.3

[34] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. iMAP: Implicit mapping and positioning in real-time. In *ICCV*, 2021. 2.2

[35] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022. 2.1

[36] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, W Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in neural rendering. In *Computer Graphics Forum*, volume 41, pages 703–735, 2022. 1.1, 2.1

[37] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. In *CVPR*, 2022. 3.4

[38] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861. 5.1, 5.3

[39] 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. 2.2

[40] Robert J Woodham. Photometric method for determining surface orientation from multiple images. *Optical engineering*, 19(1):139–144, 1980. 2.3

[41] Wenqi Yang, Guanying Chen, Chaofeng Chen, Zhenfang Chen, and Kwan-Yee K Wong. PS-NeRF: Neural inverse rendering for multi-view photometric stereo. In *ECCV*, 2022. 2.3

[42] Lin Yen-Chen. Nerf-pytorch. `https://github.com/yenchenlin/nerf-pytorch/`, 2020. 3.4

[43] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. IRON: Inverse rendering by optimizing neural SDFs and materials from photometric images. In *CVPR*, 2022. 2.3

[44] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. doi: 10.1109/CVPR.2018.00068. 5.1, 5.3

[45] Xiuming Zhang, Sean Fanello, Yun-Ta Tsai, Tiancheng Sun, Tianfan Xue, Rohit Pandey, Sergio Orts-Escolano, Philip Davidson, Christoph Rhemann, Paul Debevec, et al. Neural light transport for relighting and view synthesis. *ACM Transactions on Graphics*, 40(1):1–17, 2021. 2.3

[46] Xiuming Zhang, Pratul P Srinivasan, Boyang Deng, Paul Debevec, William T Freeman, and Jonathan T Barron. NeRFactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics*, 40(6):1–18, 2021. 2.3

[47] Zhengyou Zhang. Microsoft Kinect sensor and its effect. *IEEE MultiMedia*, 19 (2):4–10, 2012. doi: 10.1109/MMUL.2012.24. 1.1, 2.2

[48] Quan Zheng, Gurprit Singh, and Hans-Peter Seidel. Neural relightable participating media rendering. In *NeurIPS*, volume 34, 2021. 2.3

[49] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. NICE-SLAM: Neural implicit scalable encoding for SLAM. In *CVPR*, 2022. 2.2