Casual Scene Capture and Editing for AR/VR Applications

Thesis submitted in partial fulfillment
of the requirements for the degree of

Masters of Science
in
Computer Science and Engineering by Research

by

Pulkit Gera
20171035
pulkit.gera@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
September, 2022

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled ‘Casual Scene Capture and Editing for AR/VR Applications‘ by Pulkit Gera, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. P.J. Narayanan
Date	Co-Adviser: Prof. Jean-François Lalonde

To, my late grandmother

Acknowledgments

This master's thesis and related publications resulted from many people who supported, nurtured, scolded, and, most importantly, believed in me over the last three years.

Firstly, I want to thank my family for their undying support and being my biggest cheerleaders. The last two years were incredibly tough due to the pandemic, and despite that, they kept me pushing, and this thesis would not have been possible without their support. No matter how tough and desperate the situation was, they always calmed me down and cared for me. They cheered my name when I won and cheered louder when I lost.

I want to thank my advisor, Prof P.J. Narayanan, for believing in me and constantly pushing me to break the glass ceiling. More than his guidance on my work, his philosophy towards life inspired me to look at the larger picture and run towards excellence instead of success. I hope to write papers as elegantly as he does someday. I want to thank my co-advisor, Prof. Jean-François Lalonde, who has played a highly influential role in my research pursuits. Working with him gave me new insights and pushed me to another level. My internship and subsequent project with him taught me how to formulate new ideas and pursue them systematically. I am indebted to having such exceptional advisors who guided me through some fascinating work. I hope to continue collaborating with them in the future.

A big part of my journey can be solely attributed to Aakash K.T, who taught me how to walk. Without his constant guidance, this dissertation would not have been possible. He taught me how to read papers, conduct experiments, present them well, and write crisp papers. More than a mentor, he was a patient friend willing to go over simple concepts a hundredth time, even if he was super busy. Also, a massive shoutout to Dhawal for always providing a critical point of view and being super supportive.

I can't stress how significant a role my friends played during this journey. Manan, Anush, Anchit, Neel, and the "Babbar Shers" always backed me and lent a helping hand. Life at IIIT was always stressful and uncertain, but the "Babbar Shers" ensured that it was full of laughter and fun.

Lastly, I like to thank myself. There were many times when I felt inadequate and unworthy of even being a student here. So many occasions when I felt scared of charting into unknown territory. I like to thank my past self for keeping going and never giving up.

"If we keep holding onto yesterday, what are we going to be tomorrow?" - Haikyuu.

Abstract

Augmented Reality and Virtual Reality(AR/VR) applications can become far more widespread if they can photo-realistically capture our surroundings and modify them in different ways. It could include editing the scene's lighting, changing the objects' material, or augmenting virtual objects onto the scene. There has been a significant amount of work done in this domain. However, most of these works capture the data in a controlled setting consisting of expensive setups such as light stages. These methods are impractical and cannot scale. Thus, we must design solutions that capture scenes casually from off-the-shelf devices commonly available to the public. Further, the user should be able to interact with the captured scenes and modify these scenes in exciting directions, such as editing the material or augmenting new objects into the scene.

In this thesis, we study how we can produce novel views of a casually captured scene and modify them in interesting ways. First, we present a neural rendering framework for simultaneous novel view synthesis and appearance editing of a casually captured scene using off-the-shelf smartphone cameras under known illumination. Existing approaches cannot perform novel view synthesis and edit the materials of the scene objects. We propose a method to explicitly disentangle appearance from lighting while estimating radiance and learn an independent lighting estimation of the scene. This allows us to generalize arbitrary changes in the scene's materials while performing novel view synthesis. We demonstrate our results on synthetic and real scenes.

Next, we present PanoHDR-NeRF, a neural representation of an indoor scene's high dynamic range (HDR) radiance field that can be captured casually without elaborate setups or complex capture protocols. First, a user captures a low dynamic range (LDR) omnidirectional video of the scene by freely waving an off-the-shelf camera around the scene. Then, an LDR2HDR network converts the captured LDR frames to HDR, which are subsequently used to train a modified NeRF++ model. The resulting PanoHDR-NeRF representation can synthesize full HDR panoramas from any location in the scene. We also show that the HDR images produced by PanoHDR-NeRF can synthesize correct lighting effects, enabling the augmentation of indoor scenes with synthetic objects that are lit correctly.

Through these works, we demonstrate how we can casually capture scenes for AR/VR applications that the user can further edit.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Augmented reality (AR) systems overlay spatially registered augmentation onto the physical world. On the other hand, virtual reality (VR) systems create a simulated environment that immerses the user. While AR augments the real-world environment, VR completely replaces it. AR/VR applications have wide-ranging use cases in different domains such as education, healthcare, tourism and navigation. These applications require photorealistic reconstruction of the real world and should allow users to modify it in interesting ways. However, capturing and reconstructing 3D scenes requires elaborate setups such as light stages. These methods are impractical and cannot scale. For these applications to become mainstream, it is crucial that anyone is able to easily capture these 3D scenes using off the shelf devices without any complex equipment. Further, they should be able to interact with the captured scene and modify it in interesting ways. In this thesis, we study how using casually captured images from off-the-shelf devices, we can create photorealistic reconstructions for AR/VR applications and further modify them in interesting ways. We present a method that facilitates novel view synthesis of a casually captured bounded scene from off-the-shelf smartphone cameras and edit its appearance under known illumination. This has wide-ranging uses for AR applications. VR applications, on the other hand, demand 360°frames for the user to observe all viewpoints within the desired direction from the given position. We study how to render novel omnidirectional views of a casually captured indoor scene from off-the-shelf commercial cameras. Further, we capture the high dynamic range (HDR) radiance field of the indoor scene. This enables us to augment the scenes with virtual objects with correct lighting effects.

1.1 Scene Capture

Several approaches have been taken to capture and reconstruct 3D scenes. Active methods reconstruct the 3D scene by taking depth maps as input. LiDAR(Light Detection and Ranging) [54] determines the depth by shooting a laser towards a surface and measuring the time for the light to be reflected to the sensor. It is widely used in geology, geography, etc. Similarly, Time of Flight (TOF) cameras determine depth by emitting modulated infrared light and measuring the time for it to be reflected by

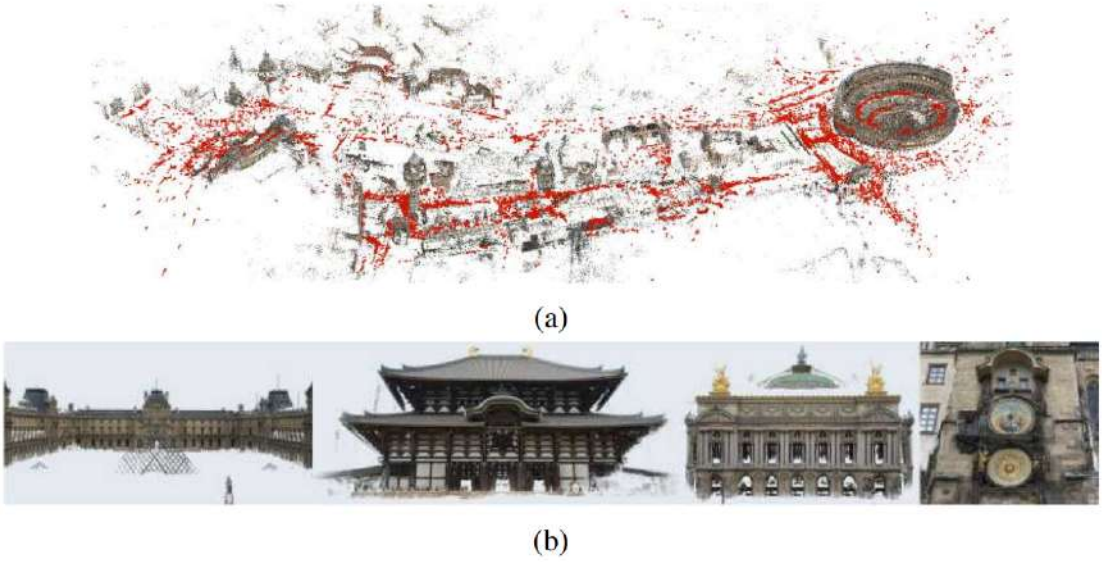


Figure 1.1 Examples of large-scale reconstructions using COLMAP (structure from motion). (a) Sparse Model of Central Rome created using 21,000 images by Schönberger *et al*. [85]. (b) Dense models of several landmarks were created similarly by Schönberger *et al*. [86]

the objects in the scene. [23]. Kinect cameras [116] generate depth maps in real-time using a structured light technique. Once we capture depth maps, they are then used to reconstruct a point cloud or mesh representation of the scene. Levoy *et al*. [45] leveraged high-quality DSLR cameras and Cyberware laser stripe scanners to reconstruct Michaelangelo's David. Although these methods reconstruct the scene accurately, they are costly, require elaborate setups, and tend to suffer from low-resolution noise and missing parts.

Passive methods utilize the reflected radiance measured by the sensor to infer the 3D structure of the scene. The sensor outputs a set of digital images or video, which is used to reconstruct a 3D scene. Structure from Motion (SfM) is a surface-based approach that establishes the relation between multiple images of the scene, retrieves their camera parameters, and reconstructs an image-based point cloud. Pollefeys *et al*. [74] demonstrated how a 3D scene could be reconstructed from a set of images using the same cameras. Snavely *et al*. [93] further demonstrated how we could reconstruct scenes from images captured in the wild from the internet.

Neural rendering allows a compact representation of scenes, and rendering can be learned from existing observations by utilizing neural networks. The objective is to generate photo-realistic imagery in a controllable way that enables novel view synthesis, deformation of the scene, relighting, and compositing. In this thesis, we study how neural rendering methods can perform various tasks from casually captured images of a scene.

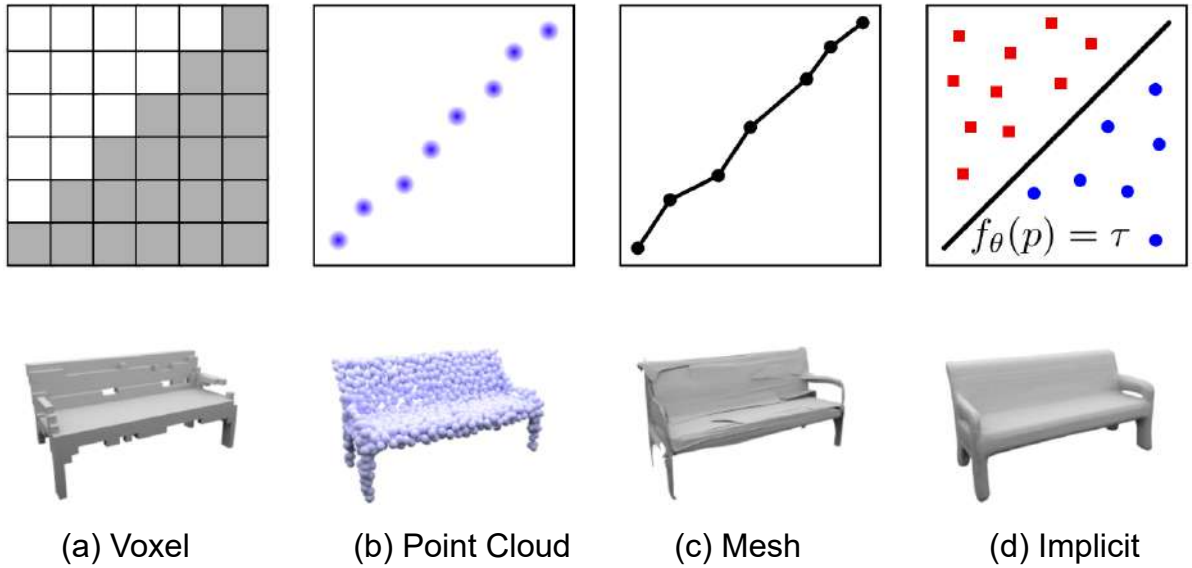


Figure 1.2 Different 3D representations. Above shown is the discretization of space, and below is the scene represented by a) Voxel, b) Point Cloud, c) Mesh, and d) Implicit. Note: Unlike other representations, we can represent continuous boundaries with implicit functions. Adapted from Mescheder *et al*. [61]

1.2 3D Scene Representation

Once the scenes are captured, there are multiple ways to represent a reconstructed 3D scene. In this section, we briefly discuss them and the advantages and disadvantages of each.

1.2.1 Mesh

Mesh represents a 3D shape as a set of vertices and connecting faces. It is widely used in computer graphics to represent complex 3D objects compactly. The graphics pipeline and GPUs are optimized to process and rasterize billions of triangles per second. The vertex positions and attributes can be optimized to match the ground truth image with the help of a differentiable renderer. The color of a pixel is determined by first assigning a triangle to it and then taking a weighted average of the colors of its vertices.

Similarly, the material and normal vectors at a pixel are typically expressed as a weighted sum of the material and normal vectors defined at the corresponding triangle's vertices. However, reconstructing 3D meshes from multi-view images requires knowledge of the topology beforehand. Further, it requires handcrafting gradient calculations at the edge boundaries, which may lead to incorrect reconstructions. Finally, compared to other methods, it is computationally too expensive.

1.2.2 Point Cloud

Point cloud is a set of elements of Euclidean space. It can be used to represent a continuous surface by a set of discrete points and volumes. In addition, they can store additional attributes such as normals or colors. They are popular due to their ability to represent a large variety of topologies with minimal memory footprint. A point cloud is rendered by projecting a 3D point onto the screen. Pixel color is estimated by taking a weighted average of each 3D point projected on the pixel. However, pint cloud-based approaches face size ambiguity since mapping 3D point to pixel naively leads to sparse images. Although they are a natural choice for many differentiable rendering methods, they fail to capture dense surface information. Further, they cannot take occlusion into account while calculating color.

1.2.3 Voxel

Voxel is a unit cube representation of a 3D space. It can be parameterized as an N-dimensional vector that stores different attributes of the 3D scene, such as occupancy information, transparency, etc. Volume attributes can be accessed at any point within the voxel grid using trilinear interpolation. In addition, we can also represent shapes as the shortest distance from the center of each voxel to the object's surface. Each voxel cell is assigned a distance function and given a sign which is called a Signed Distance Function (SDF). To render a scene, we shoot a ray from a pixel, and all voxels located along the ray are considered. Voxel-based approaches are easy to use but require excessive memory and parameter usage. Their applicability is limited to small scenes and low resolutions.

1.2.4 Implicit Representation

Implicit representations encode space as the output of a function. This enables us to represent continuous boundaries as we can derive the surface by querying the function at different points in space. Neural implicit representations aim to encode the entire scene in a neural network. Formerly the scene geometry information at a point P^i_{xyz} is described as the output of the neural network f_θ. Compared to voxel-based methods, the memory footprint of the scene is constant and thus can be used to represent large scenes with high-frequency details. The network can represent the scene in different ways, such as storing whether a particular point is inside or outside the object [61]. Alternatively the surface of an object can be defined as the set of points P^i_{xyz} which satisfy $f_\theta(P^i_{xyz}) = 0$. This is also called the level-set method. Analogous to SDF, the network outputs the distance of a point from the surface with the signs indicating whether it is inside or outside the surface [72]. A broad range of topologies can be represented at virtually infinite resolution and low memory cost. However, finding a level set is tricky. It also requires querying and aggregating data for many points, making it computationally expensive.

1.3 Novel view synthesis

From a set of fixed images of a scene, novel view synthesis is defined as generating images from novel camera poses. Some challenges include representing the 3D scene, modeling view-dependent effects, and inferring occluded/unseen regions. We discuss some of the approaches in this section.

1.3.1 Traditional Image Based Rendering

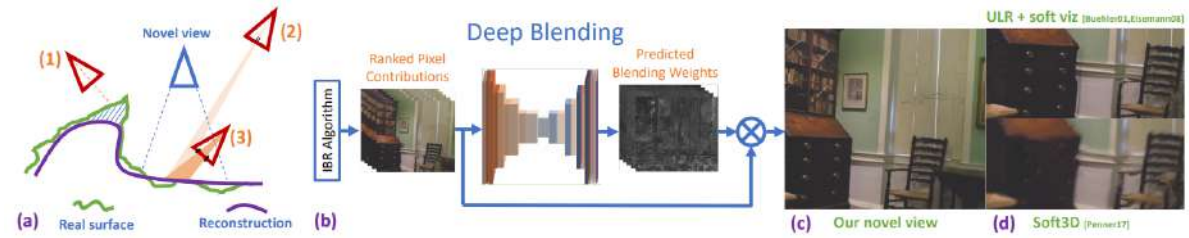
Classical rendering renders 2D images from 3D contents, whereas image-based rendering (IBR) aims to generate images by transforming the given set of images. Generally, the transformation involves warping and compositing the source images. Traditional IBR techniques generate novel viewpoint images based on proxy geometry and estimated camera poses. Blending methods require capturing multiple views which are warped into the target view. However, if large parts of the scene are not visible or the scene contains view-dependent effects, IBR methods create ghosting-like artifacts and holes.

1.3.2 Neural Image-Based Rendering

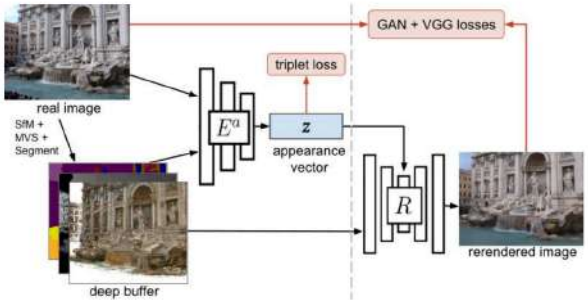
Neural Image-Based Rendering class of approaches either augment the hand-crafted parts of the pipeline or replace them altogether. These approaches use a network to perform blending and consider corrections needed to estimate view-dependent effects. DeepBlending [31] learns a network to blend the projected source images to generate the target image. Thies *et al.* [98] overfit a network over a scene and predict images from novel viewpoints with view-dependent effects. It produces a diffuse-only image and removes specular highlights from input images. Xu *et al.* [106] take six images as input captured under point illumination in a cone of 60°and perform novel view synthesis within the cone. They construct a plane sweeping volume and capture the scene's light transport, which is used to create novel viewpoint images with view-dependent effects. It outputs both scene depth and appearance. FVS [80] and SVS [81] leverage multi-view stereo and create a coarse geometric scaffold. The scaffold helps in creating a proxy depth map for the target view. The network reprojects features from source images onto the scaffold to synthesize a new view.

1.3.3 Neural Re-Rendering

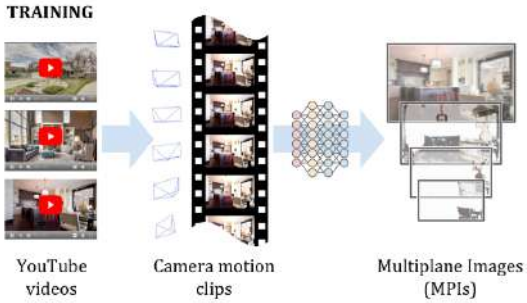
Neural Re-Rendering combines classical 3D representations and renderers with deep neural networks to create photorealistic novel views. Instead of relying on source views to render target views, they rely entirely on the network to synthesize novel views. Meshry *et al.* [62] take multimodal input such as the rendered deep buffer (containing depth and color) along with an appearance code vector to output realistic novel views of the scene. Deferred Neural Rendering (DNR) [97] aims to represent and render a scene entirely using learned components. It recovers neural textures analogous to classical texture maps and learns a neural renderer to render novel views from neural texture. We leverage this idea to perform appearance editing along with novel view synthesis for diffuse scenes in chapter 3.



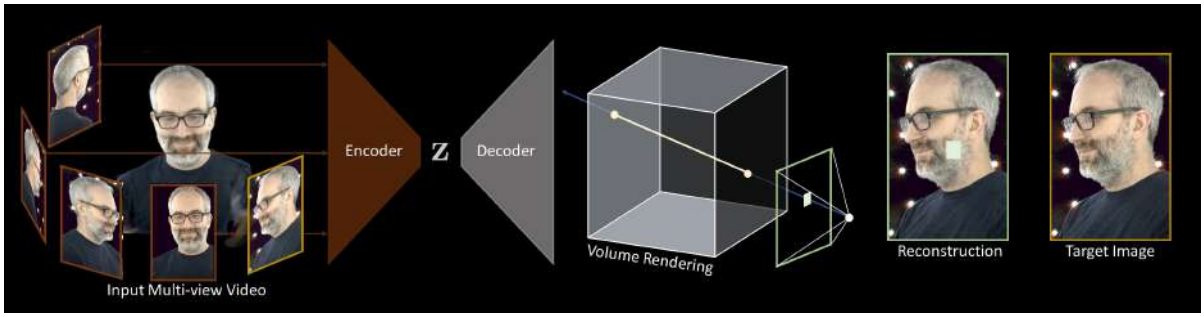
(a) Deep Blending



(b) Neural Rerendering in the wild



(c) Stereo magnification using MPI



(d) Neural Volumes

Figure 1.3 Different approaches for novel view synthesis. (a) Hedman *et al.* [31] learns blending weights and composites source images. (b) Meshry *et al.* [62] takes a deep buffer as input and renders a novel image from the network. (c) Zhou *et al.* [119] learns an MPI representation and uses it to extrapolate novel views. (d) Lombardi *et al.* [53] learns a voxel representation of the scene.

1.3.4 Multi Plane Images

Multiplane images (MPI) are a set of fronto-parallel planes at fixed depths from a reference camera coordinate frame. Each plane captures the scene appearance at the corresponding depth and encodes an RGB image with an alpha map which is used to extrapolate the views. Zhou *et al.* [118] recovered the occluded parts of the scene, and the alpha map also helped estimate reflective objects. MPI can be rendered efficiently and is used in real-time applications.

1.3.5 Voxel Based Approach

Although learned, unstructured neural representations improve upon hand-crafted scene representations, they do not consider the 3D structure of the scene. This makes them highly dependent on the training data, and the scope of view synthesis is relatively limited. Instead, we utilize voxel grids as our 3D representation and combine it with neural rendering approaches. DeepVoxels [88] combines a neural renderer utilizing projective geometry priors with a voxel gird of learned embedded features to generate novel views. Neural Volumes [53] learns a volumetric representation for a scene by optimizing a 3D CNN-based network. Plenoxels [83] on the other hand, create a sparse voxel grid with density and spherical harmonic coefficients at each voxel. Instead of a network, we utilize volumetric rendering techniques to render new views. Unlike MPI-based methods, these methods are not skewed to one particular viewing direction and can render the encoded scene from any direction.

1.3.6 Implicit functions based Approaches

Voxel-based approaches consume much memory and cannot encode a scene with large resolution. SRN [90] encoded the scene using a multilayer perceptron (MLP) and leveraged a sphere-tracing-based renderer to reconstruct scenes. DVR [69] utilized a surface rendering approach. They overfit on a single scene which enabled encoding of more complex appearance and geometry.

1.3.6.1 Neural Radiance Fields

Neural Radiance Fields (NeRF) [65] represents a scene as an implicit continuous 5D function that maps every points with the position (x, y, z) and the viewing direction (θ, ϕ) to the their corresponding transmitted radiance $\mathbf{c}$ and volume density σ. Through every pixel, we pass a ray $\mathbf{r} = \mathbf{o} + t\mathbf{d}$ where $\mathbf{o}$ is the camera origin and $\mathbf{d}$ is the viewing direction. We sample s points along the ray between the t_{near} and t_{far}. The expected pixel color along the ray passing is defined as

$$\hat{C}(\mathbf{r}) = \int_{t_{\text{near}}}^{t_{\text{far}}} T(s)\sigma(\mathbf{r}(s))\mathbf{c}(\mathbf{r}(s), \mathbf{d})ds \tag{1.1}$$

$$T(s) = \exp\left(-\int_{t_{\text{near}}}^{t_{\text{far}}} \sigma(\mathbf{r}(p))dp\right) \tag{1.2}$$

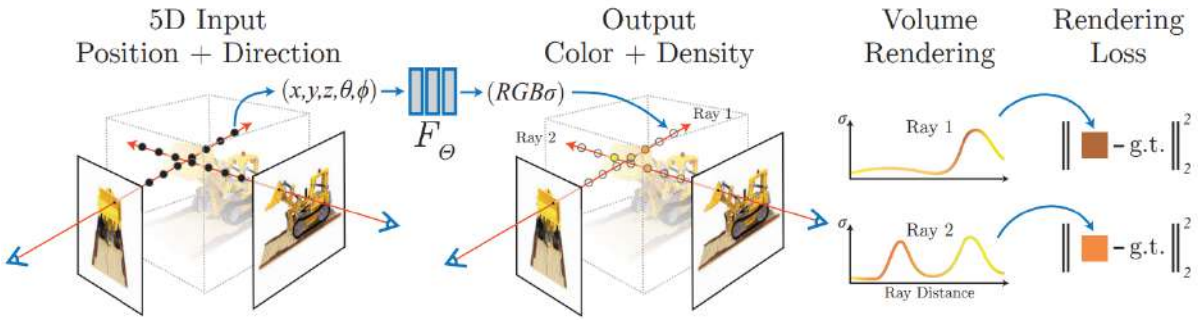


Figure 1.4 Mildenhall *et al*. [65] encodes the scene within an MLP. They sample points on a camera ray, and for each point, query the color and volume density from the MLP. They are then composited using volume rendering techniques to render the pixel color and density.

$T(s)$ is the accumulated transmittance along the ray. To learn the scene, NeRF uses the loss between predicted color $\hat{C}$ and ground truth C as

$$\ell_{\text{nerf}} = \sum_{r \in \mathcal{R}(\mathbf{P})} \|\hat{C}(\mathbf{r}) - C(\mathbf{r})\|^2 \tag{1.3}$$

where $\mathcal{R}(\mathbf{P})$ is the set of camera rays at point $\mathbf{P}$. NeRF optimizes two models: coarse and fine. The densities procured from the coarse model guide the sampling for the fine model. In addition, NeRF applies positional encoding γ that maps the input i.e $\mathbf{x}$ and $\mathbf{d}$ to a higher dimension and enables the network to learn higher frequencies of data.

$$\gamma^k : \mathbf{p} \longrightarrow (sin(2^0\mathbf{p}), cos(2^0\mathbf{p}), sin(2^1\mathbf{p}), cos(2^1\mathbf{p}), ..., sin(2^k\mathbf{p}), cos(2^k\mathbf{p})) \tag{1.4}$$

where k is a hyper-parameter specifying the dimensionality of Fourier feature vector. NeRF is able to model view-dependent effects and has a very small memory footprint.

1.3.6.2 Unbounded Neural Radiance Fields

The NeRF formulation assumes that the entire scene fits into a bounded volume, which does not hold for large indoor scenes. We can either fit the entire scene into the bounded volume and lose detail due to limited sampling or only fit a small part of the scene in detail and lose detail in the remaining part.

NeRF++ [114] proposes dividing the scene space into foreground and background. We partition the scene into two volumes, an inner unit sphere containing the foreground and the remaining volume ,containing the background. These two volumes are modelled with two separate NeRFs. To render the color for a ray, they are inferred separately and then combined.The parameterization for the foreground is the same as the original NeRF. For the background, a 3D point $(x, y, z), r = \sqrt{x^2 + y^2 + z^2} > 1$ is parameterized by the quadruple $(x', y', z', 1/r)$ where (x', y', z') is a unit vector representing a direction on the sphere and $0 < 1/r < 1$ is the inverse radius (disparity) along this direction specifying the point

$r \cdot (x', y', z')$ outside the sphere. This bounds of this reparameterization change from infinity in euclidean space to $x', y', z' \in [-1, 1], 1/r \in [0, 1]$ leading to numeric stability. We modify this representation to represent unbounded indoor scenes from a set of omnidirectional images in chapter 4.

1.3.7 Omnidirectional Images

Omnidirectional cameras are crucial for various applications in robotics [84, 7],computer vision [60, 26] and virtual reality due to their wide viewing angle. To create highly immersive virtual reality (VR) content, 360°videos are required, allowing the users to observe all viewpoints from the position where the video is recorded. While a regular camera has a field of view ranging from a few degrees to 180°, omnidirectional images can cover the full 360°along the equator of the sphere except the top and bottom of the sphere.

Novel view synthesis from omnidirectional images is pretty limited so far. Huang *et al.* [34] synthesize novel views from a reconstructed point cloud and achieve real-time video playback on VR headsets. Analogous to MPIs, some work [48, 2] design a layered scene representation. However, these works either require elaborate capture setups or strong priors like the room layout [105] to work.

Some flavors of NeRF [33, 27] attempt to perform novel view synthesis. However, they require depths as input and are limited to translation motion only.

1.4 Appearance Modelling

In this section, we study how given the lighting and materials of the scene, the color of a point on a surface is calculated. These fundamentals help us understand how we can edit the scene's appearance.

1.4.1 Light Transport

Ray Tracing estimates the color of a pixel by evaluating the scattering or reflection equation. It describes how the outgoing light distribution from a point is transformed from the incident light distribution based on the scattering properties at the surface. The light transport equation (LTE), a more general form of the equation, is defined as [39]:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_\Omega f_r(x, \omega_o, \omega_i) L(t(x, \omega_i), -\omega_i) |cos\theta_i| d\omega_i, \tag{1.5}$$

where L_o is the computed radiance from a 3D point x in the direction ω_o towards the camera, L_e is the emitted radiance from point x towards ω_o. f is the Bidirectional Reflectance Distribution Function (BRDF) which characterizes the surface properties (material proeprties) which define the appearance of the points of the scene. L is the incoming radiance from another point $z = t(x, \omega_i)$ in the scene towards direction $-\omega_i$. The function $t(p, \omega_i)$ computes the first surface point intersected by a ray from x in the direction ω_i. $cos\theta$ is the cosine foreshortening factor on the angle θ_i between different directions ω_i

on the upper hemisphere Ω and the surface normal n at x. We integrate them over the hemisphere of directions Ω around x.

Since L is on the both sides of the equation, we cannot analytically solve 1.5. This means that we can recursively replace L with an integral of the same form as 1.5 and end up with an infinite-dimensional integral. We solve this using Monte Carlo (MC techniques) to evaluate the equation. Thus the equation can be rewritten as:

$$L_o(x, \omega_o) \approx L_e(x, \omega_o) + \frac{1}{N} \sum_{j=1}^{N} \frac{f(x, \omega_o, \omega_j) L(t(x, \omega_i), -\omega_j) |cos\theta_j|}{p(\omega_j)} \tag{1.6}$$

where N independent samples of ω_j are drawn according to the probability distribution $p(\omega_j))$. The quality of rendering is dependent on N, and as it approaches infinity, the estimation converges to the true value of integral.

1.4.2 Precomputed Radiance Transfer

Pre-computed radiance transfer(PRT) frameworks [91] aims at representing, storing, and rendering the light transport of a scene under varying illumination. The work of Ben-Artzi *et al.* [6] re-formulates PRT to allow real-time appearance editing. PRT assumes that all objects in the scene are non-emitters and that the light sources are infinitely distant. This makes the incoming light direction independent of the position of point x. Further, it assumes that all surfaces in the scene are Lambertian reflectors. These assumptions simplify the rendering eq. (1.5), and we can rewrite it as:

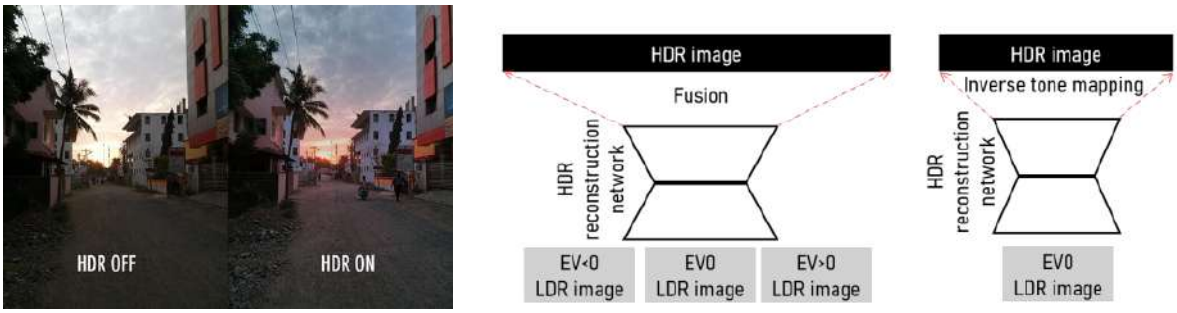
$$L_o(x, \omega_o) = \int_{\Omega} L(\omega_i) T(x, \omega_o, \omega_i) d\omega_i \tag{1.7}$$

where T is the transfer function. In PRT, the lighting function and the transfer functions are independent and thus can be estimated separately and combined later to produce final result.

However, Eq. (1.7) still does not have an analytical solution. We choose an angular basis of continuous functions - Spherical Harmonics(SH) [92] for instance and perform all light transport operations in that domain. The idea is that instead of solving the integral in eq. (1.7), we can take the dot product of the two functions in that basis domain [92]. We preprocess the transfer matrices, which encode the transformation from distant light to local incoming light in basis coordinates. They are then computed at every scene vertices and stored. At run-time, the environment lighting and the materials are projected into SH, and all light transport is then computed trivially via matrix multiplications in the SH domain [92]. Instead of lighting, we aim to edit the appearance of a scene and use similar decomposition to achieve that as shown in chapter 3.

1.5 Radiance Capture

High dynamic range (HDR) imaging is a technique that allows the capture of a higher range of intensities compared to traditional imaging techniques. The objective is to accurately represent a wide range



(a) LDR vs HDR capture

(b) Fusion vs Inverse Tone Mapping

Figure 1.5 (a) Difference between an LDR and HDR capture from a Xiaomi smartphone. HDR image is far richer in detail. (b) Comparison between the two approaches for estimating HDR images from LDR captures. Adapted from Wang *et al.* [101].

of intensity levels captured in real scenes, ranging from sunlight to shadows. Real-world lighting can be captured, stored, transmitted, and fully used in different applications without linearizing the signal or clamping the range [3]. This has wide applications, especially in AR/VR, as it allows immersive exploration and realistic scene augmentation.

Capturing HDR images requires special HDR cameras, which are prohibitively expensive and complex for general users.

Alternatively, Debevec [12] pioneered *image-based lighting* which involves photographing a chrome sphere—called a *light probe*—at different exposures and merging them into a single, high dynamic range (HDR) image [14]. HDR light probing was later extended to use wide-angle lenses or 360° cameras and is at the heart of the lighting capture necessary to achieve special effects in movies today[1]. Because of the close proximity between light sources and objects in the scene, the HDR radiance field of a typical indoor scene varies rapidly: lighting near a window is vastly different from the center of the room. Accurately capturing radiance indoors involves moving the apparatus and repeating the operation many times, limiting scalability. Approximations such as reprojecting the measured radiance onto a proxy 3D model [15] could be used.

In addition to the above mentioned approaches, a standard method is reconstructing HDR images from low dynamic range (LDR) images that off-the-shelf cameras can easily capture. Two classes of methods are used to estimate HDR images from a set of LDR images. The first is to fuse multiple LDR images of the same scene at different exposure times, similar to what Debevec [14] proposed. The second approach is to estimate HDR content from a single-exposure image.

The LDR camera sets an exposure time Δt and relies on camera response dunction (CRF) f_{CRF} to map the irradiance E of scenes to LDR image x.

$$x = f_{CRF}(E\Delta t) \tag{1.8}$$

[1] https://www.fxguide.com/fxfeatured/the-definitive-weta-digital-guide-to-ibl/.

Now the objective is to learn a mapping function M with parameters θ that maps a set of multi-exposure LDR images $X = x_1, x_2, ...x_n$ to an HDR image y where n is the number of LDR images. HDR estimation restores scene irradiance using feature learning from LDR images. The reconstructed HDR images must satisfy three criteria: high contrast ratio, high bit depth, and preserved details. Multi-exposure LDR images can be mapped to the irradiance domain using deep neural networks (DNN) based on the inverse CRF and exposure time from eq. (1.8). Alternatively, a DNN can be trained to align and merge dynamic information from LDR images. However, these methods suffer from ghosting artifacts, and acquiring multiple exposure images for dynamic scenes and viewpoints is also challenging. On the other hand, estimating HDR images from a single-exposure LDR image is much simpler where an inverse tone-mapping network is learned to generate an HDR image. However, the HDR images recover a limited dynamic range compared to the previous class of methods.

1.6 Contribution

In this thesis, we study how we can produce novel views of a casually captured scene and modify them in a few interesting ways. We present a neural rendering framework for simultaneous view synthesis and appearance editing of a scene with known environmental illumination captured casually using a mobile camera. We perform fast view generation using a disentangled representation of the appearance and local irradiance function, which are learnt from the scene. We use a neural network to *represent* and *render* the scene, beyond merely representing it.

Next we present PanoHDR-NeRF, a novel neural representation of the full HDR radiance field of an indoor scene that can be captured casually. HDR radiance at any point in the scene can be produced from PanoHDR-NeRF subsequently. Our method does not require any special equipment or complicated capture protocols. It accepts as input a video sequence captured by freely moving a commercial 360°camera around the scene. As output, it produces the HDR radiance at any given location in the scene. To evaluate our proposed method, we capture a set of six different indoor scenes, which we augment with a set of ground truth HDR light probes at each scene. Our experiments demonstrate that, despite the simplicity of the capture procedure, PanoHDR-NeRF can accurately predict HDR radiance in a variety of challenging conditions. Our approach can render 360° HDR light probes, which can be used to provide correct lighting effects when the scene is augmented with virtual objects.

To summarize, the following are the key contributions of this thesis:

- We present a neural rendering framework for simultaneous view synthesis and appearance editing of a scene with known environmental illumination captured casually using a mobile camera. (Chapter 3)

- We present PanoHDR-NeRF, a novel neural representation of the full HDR radiance field of an indoor scene that can be captured casually. (Chapter 4)

1.7 Thesis Layout

This thesis comprises of 5 main chapters. Chapter 1 presents the problem of capture of scenes for AR/VR applications. We discuss the previous approaches to capturing and representing 3D scenes. We dive deeper into various paradigms for performing novel view synthesis. Further we study how appearance is modeled and how pre-computed radiance transfer enables real-time rendering. We discuss various approaches used for capturing radiance. Finally we summarize the key contributions of the thesis.

Chapter 2 gives an overview of the prior works performing novel view synthesis, estimating HDR images from a set of LDR images and capturing HDR radiance.

Chapter 3 presents a neural rendering framework for simultaneous novel view synthesis and appearance editing of a causally captured scene from off-the-shelf smartphone cameras under known illumination.

Chapter 4 presents PanoHDR-NeRF, a neural representation of an indoor scene's high dynamic range (HDR) radiance that can be captured casually without elaborate setups or complex capture protocols.

Chapter 5 concludes this thesis by providing closing remarks and motivation for future works.

Chapter 2

Related Works

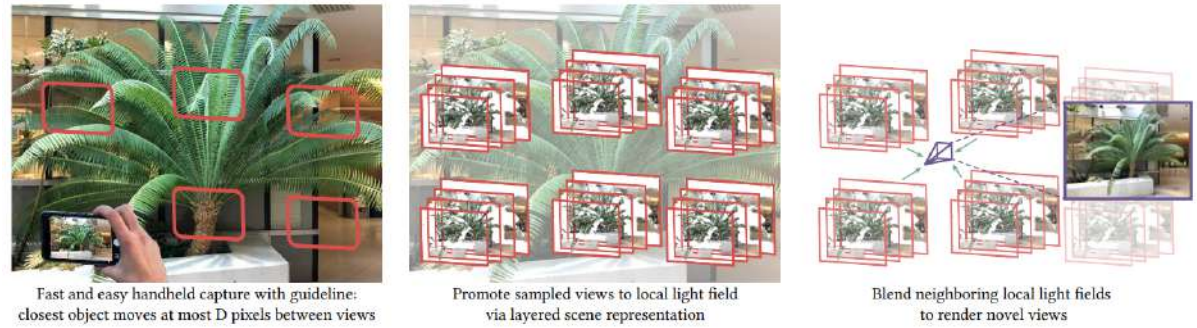
In recent years, capturing scenes from casually captured images and editing them in different ways has become an active field of research in computer vision. Sec. 2.1 discusses some recent works performing novel view synthesis from perspective and omnidirectional images. In sec. 2.2 we look closely at DNR [97] and derivative works which form the basis for our work presented in chapter 3. We discuss NeRF and its various variants in sec. 2.3 and look at some concurrent works that try to achieve the same objectives. We take a detailed look into different ways of HDR image estimation in sec. 2.4. Finally, we study works that aim to capture HDR radiance from a scene and discuss their limitations with respect to our work presented in chapter 4.

2.1 Novel View Synthesis

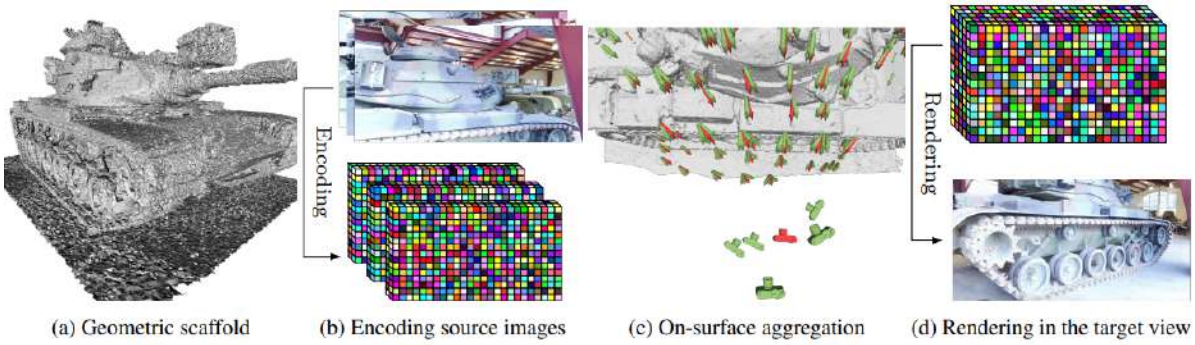
Image-based rendering utilizes 2D images to generate a novel representation of the 3D world.Classical methods reconstruct an explicit 3D model of the scene [15, 24, 32]. Recent works utilize off-the-shelf SfM techniques to generate a coarse geometry and use neural approaches to render photorealistic novel views. Hedman *et al*. [30] designed a generalized network that infers the blending weights of the source images for compositing in the target space. Meshry *et al*. [62] presents a multi-modal pipeline that takes a rendered deep buffer and an appearance code vector as input and renders photorealistic novel views of the images. The buffers are generated from images in the wild, and the network learns an implicit embedding of appearance, representing the time of the day, weather conditions, etc.

Breaking the scene into Multi-Plane Image (MPI) representation to render novel views by blending has been tried. DeepView [18] learns visualizes light fields under novel views. Zhou *et al*. [119] takes stereo images from a Youtube video and learns a network that predicts an MPI. The inferred MPI generates a range of novel views of the scene that extend beyond the input baseline. LLFF [64] takes an irregular grid of views as input and constructs a local light field via an MPI representation. Novel views can be rendered by blending adjacent local light fields.

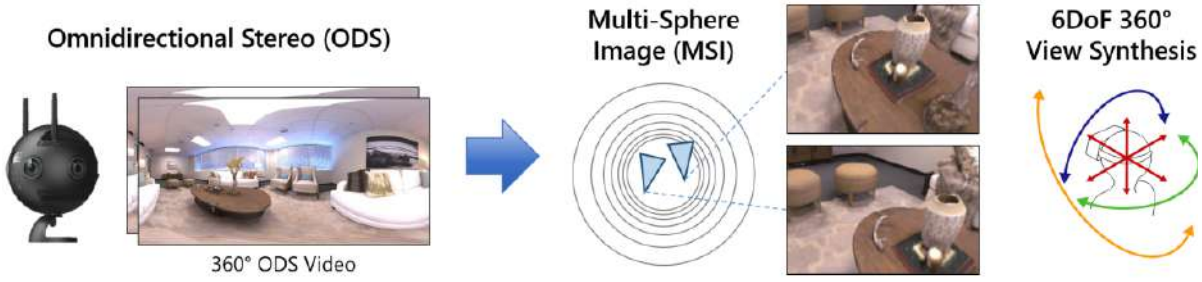
DeepVoxels [89] learn a voxel-based volumetric representation of the scene using Gated Recurrent Units (GRUs) [11]. Free View Synthesis [80] maps the encoded features from the source images into



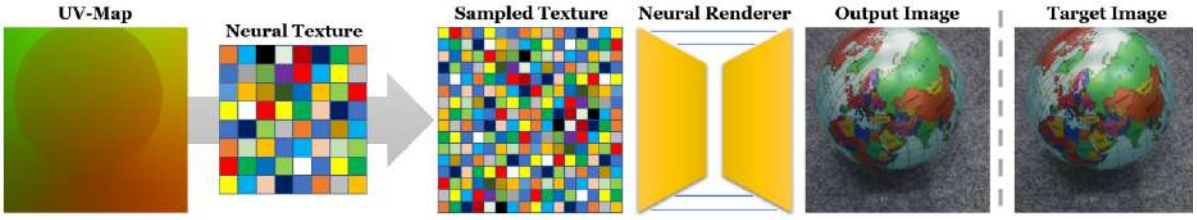
(1) Local Light Fusion



(2) Stable View Synthesis



(3) MatryODShka



(4) Deferred Neural Rendering

Figure 2.1 Recent works performing novel view synthesis (1) Mildenhall *et al.* [64] constructs a local light field via an MPI representation. (2) Riegler *et al.* [81] maps encoded features onto reconstructed geometry and aggregates them for a novel view (3) Attal *et al.* [2] learns an MSI representation from an RGB-D panorama and uses it to extrapolate novel views. (4) Thies *et al.* [97] recovers neural textures and learns a neural renderer to render novel views.

the target view and blends them via a neural network. Stable View Synthesis [81] does the same by mapping encoded image feature vectors onto the reconstructed geometry and aggregating these features for a novel view.

Novel view synthesis from omnidirectional images is pretty limited so far. Huang *et al*. [34] synthesize novel views from a reconstructed point cloud and achieve real-time video playback on VR headsets. Analogous to MPIs, Serrano *et al*. [87] designed a layered scene representation that facilitates parallax and real-time playback of 360° video. Attal *et al*. [48, 2] create 6-DoF renderings via Multi-Depth Panorama(MDP) and multi-sphere images. However, it requires an elaborate setup and does not accommodate free viewpoint synthesis. Xu *et al*. [105] estimate the entire indoor scene from a single image using a CNN but need room layout priors and depth that are challenging to obtain for real-world scenes.

2.2 Deferred Neural Rendering

Deferred Neural Rendering (DNR) [97] demonstrates this by performing view synthesis and face re-enactment with only images and an approximate 3D reconstruction. It recovers neural textures analogous to classical texture maps and learns a neural renderer to render novel views from neural texture. DNR achieves novel view synthesis by keeping the captured scene's geometry, appearance, and lighting constant. Deferred Neural Lighting (DNL) [19] extended this to allow relighting by projecting learned neural textures onto a rough proxy geometry using a scene-dependent neural rendering network for relighting. Relightable Neural Rendering (RNR) [10] extends the DNR framework to allow lighting changes with arbitrary environment lighting using a learned light transport function. The network regresses on the lighting, object intrinsics, and light transport function rather than directly translating deep features to appearance. In this work, however, we extend the DNR framework to allow appearance editing under known illumination.

2.3 Neural Radiance Fields

Neural Radiance Fields (NeRF) [65] represents a static scene as a 5D continuous function by encoding it within a neural network. To perform view synthesis, we pass rays through each pixel and sample points on them. For each point, we query color and volume density from the MLP, which are then combined using volume rendering techniques. NeRF++ [114] models an unbounded scene by splitting it into foreground/background, learning each separately. Mip-NeRF [4] replaces rays with anti-aliased conical frustums for speed and accuracy. Mip-NeRF 360 [5] extends this for unbounded real scenes. Mega-NeRF [99] divides a large-scale area into smaller clusters and models, each with a small MLP. Each of these clusters is processed parallelly to render novel views. Block-NeRF [95] models the entire city by splitting it into smaller blocks and training a NeRF for each block. The results are combined from the different NeRFs trained for a novel viewpoint. BaRF [47], iNeRF [49] and SC-NeRF [37] also estimate accurate camera poses given a reasonable initialization. KiloNeRF [79], FastNeRF [20],

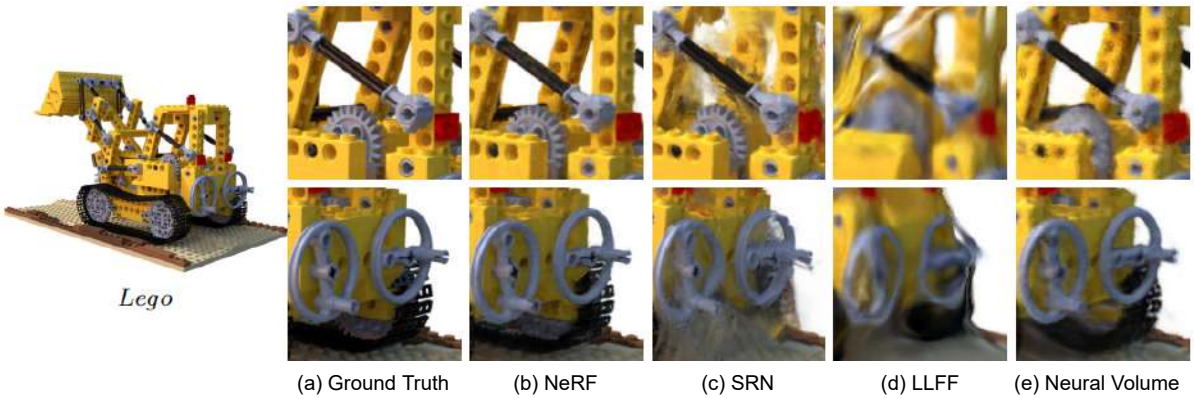


Figure 2.2 Qualitative comparison between different novel view synthesis methods on a test scene. NeRF is able to recover the fine geometry and appearance details whereas other methods fail to do so.

PlenOctree [109] demonstrate how to enable real-time rendering of NeRFs without sacrificing quality. PlenOxels [83] and Instant-NGP [67] massively reduce the training time by using a different representation for encoding scenes instead of naively using an MLP. NeRF-W [59] reconstructs scenes from images captured in the wild. It models appearance and uncertainty in separate embeddings and thus can easily interpolate in appearance without affecting 3D geometry. NeRV [9] adds the relighting and material editing capability by adding a second "visibility" neural network to support one-bounce indirect illumination and environment lighting. NeRD [8] is an explicit decomposition model for shape, reflectance, and illumination within a NeRF-like coordinate-based neural representation framework. NeRFactor [115] learns an embedding-based neural field of BRDF parameters as a prior on the material parameter space. PhySG [113] proposed a method to relight and edit materials of a scene and recover the geometry as an SDF. Their method, however, works for only specular objects and fails when the geometry is too complex or the object is diffuse. Our work (Chap:3), on the other hand, represents and renders the scene using learnable components, thereby lifting the limitation of classic rendering.

NeRF with HDR images has been explored in two works. NeRF in the dark [63] train directly on linear RAW images with a higher effective dynamic range (14-16 bits compared to more typical 8-bit cameras). We show this is not high enough to accurately capture radiance in most indoor environments. Huang *et al*. [35] recovers a HDR radiance field from a set of LDR images captured at alternating exposures. Capturing multiple exposure LDR requires a more elaborate capture setup. In contrast, our method (Chap:4) accurately models the full dynamic range from a single input video.

Hsu *et al*. [33] present a method to synthesize panoramas from a single RGB-D panorama. However, they can only render novel views on a straight-line path. OmniNeRF [25] synthesizes novel fish-eye projection images and incorporates spherical sampling to improve the quality of results.

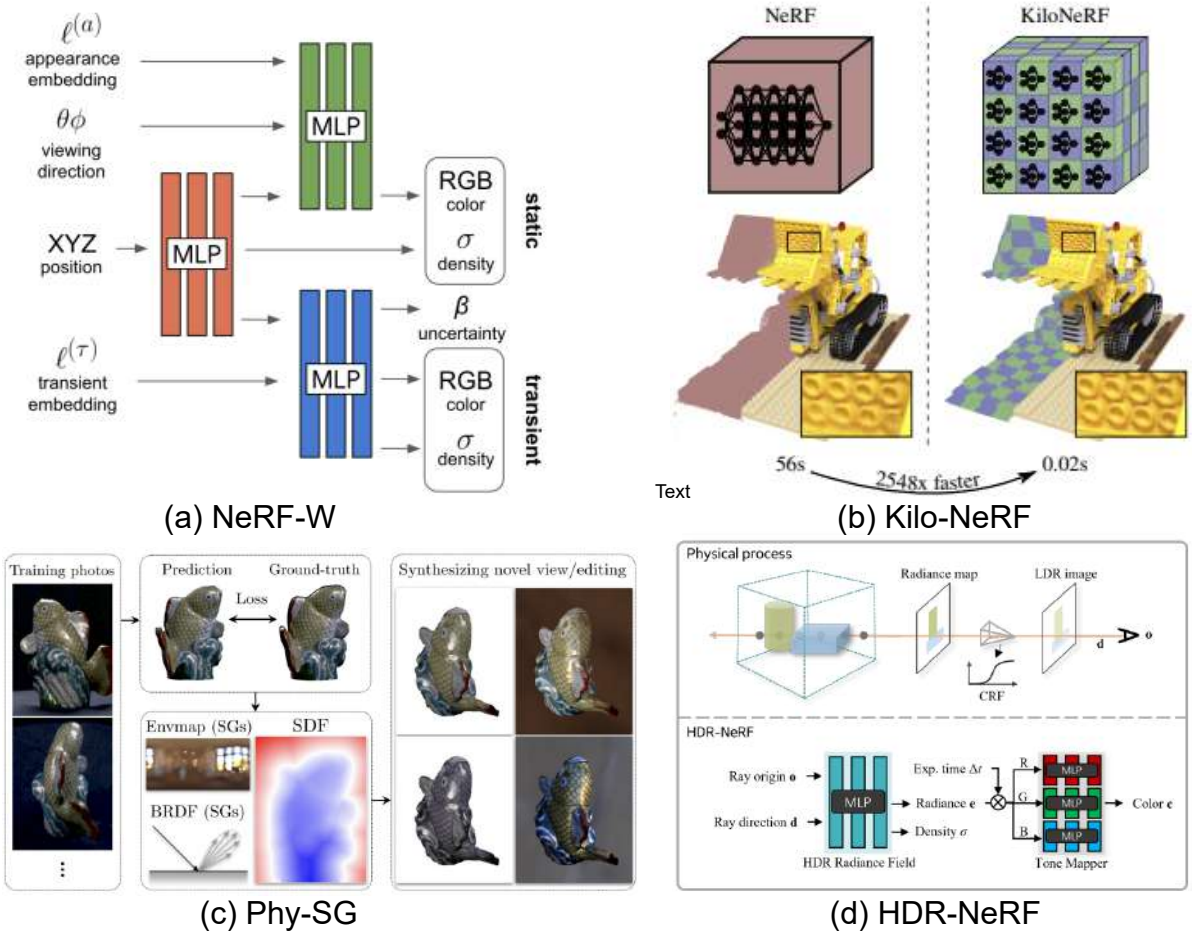


Figure 2.3 Different variants of NeRF: (a) Martin-Brualla *et al*. [59] models appearance and uncertainty separately. (b) Reiser *et al*. [79] improves the speed significantly by training 100 small NeRFs. (c) Zhang *et al*. [113] uses sphere tracing and performs relighting and material edits for glossy objects. (d) Huang *et al*. [35] models the HDR radiance field from a set of LDR images captured at varying exposures.

2.4 HDR estimation

From a set of LDR images, there are two methods for estimating HDR images. The first requires a set of LDR images of the same scene captured at multiple exposures. An HDR image can be reconstructed from learned networks, which are trained to align and merge dynamic information from the LDR images. Alternatively, we can also estimate HDR images from a single-exposure LDR image. Here the learned network recovers missing information in the under and over-saturated areas of an LDR image.

2.4.1 Multi-Exposure Image Fusion

Debevec *et al.* [14] demonstrated recovery of HDR maps from LDR images captured at multiple exposures. Given a response function, they fuse multiple LDR images into a single HDR map. The pixel values of the HDR map are proportional to the radiance values in the scene. However, for a dynamic scene, the input LDR images must first be aligned. The quality of the alignment is usually reflected in the reconstructed HDR images. Kalantari *et al.* [40] presents a learning-based approach that uses optical flow to align low and high-exposure images to a medium-exposure image. Prabhakar *et al.* [75] proposed the alignment of images using FlowNet [16], a DL-based optical flow network, which produced significantly fewer artifacts.

Optical flow-based approaches could not align LDR images with large-scale foreground motion. Wu *et al.* [103] presented a framework consisting of three encoders, a merger network, and a decoder. The encoders encoded the LDR images into a latent space, merged, and decoded to recover the HDR image. Niu *et al.* [71] proposed multiscale LDR encoders which extracted visual features at different scales. They then fused these features at different scales via a residual network. Yan *et al.* [107] introduced attention modules in their encoding phase. This helped in excluding artifacts caused by misalignment and saturation. ADNet [52] proposed the alignment with a pyramid, cascading, and deformable module and adaptively fuses them with a spatial attention module. However, these approaches generate less realistic details in highly saturated regions.

DeepFuse [76] uses unsupervised learning to perform static multi-exposure image fusion. FusionDN [104] proposes an unsupervised and unified densely connected network for image fusion tasks. It obtains image-driven weights as the measurement of information preservation in the features of different LDR images. Similarity loss based on weights can thus be applied without ground truth HDR images.

Multi-view exposure methods suffer greatly from alignment and ghosting artifacts. They are sensitive to over-saturated regions, which often causes the loss of textual details. Further, obtaining cameras and equipment for scene capture is relatively expensive.

2.4.2 Inverse Tone-Mapping

The objective of inverse tone-mapping networks is to recover missing information in the under-and over-saturated areas of a single LDR image.

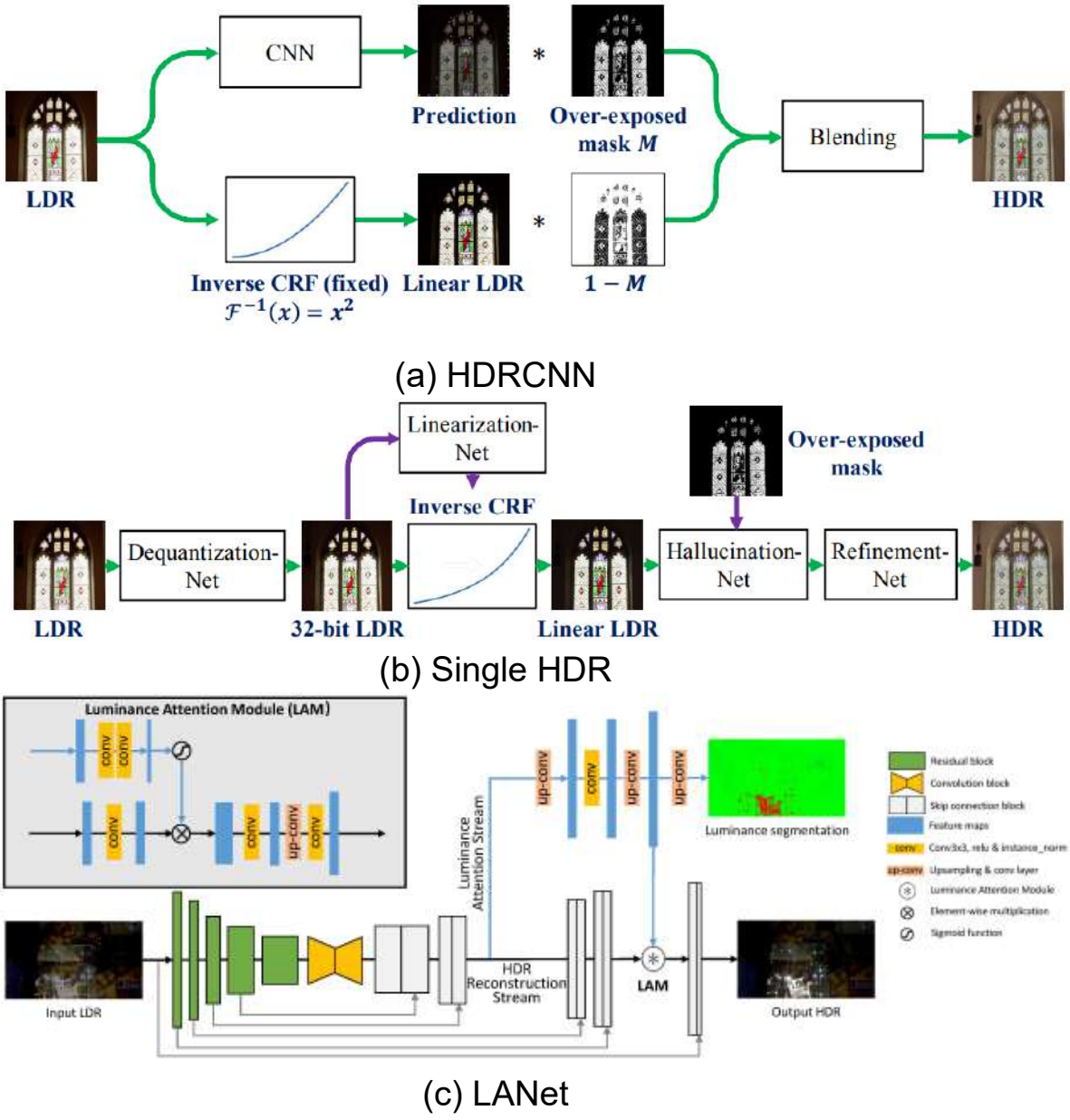


Figure 2.4 Recent works performing inverse tone-mapping (a) Li *et al.* [46] estimates HDR using a encoder-decoder architecture (b) Liu *et al.* [51] model the LDR image formation by (from right to left) dynamic range clipping, non-linear mapping, and quantization. They learn a network to invert each process (c) Yu *et al.* [110] is a multi-task network optimizing for an attention map spotting the overexposed regions and using that to estimate HDR radiance.

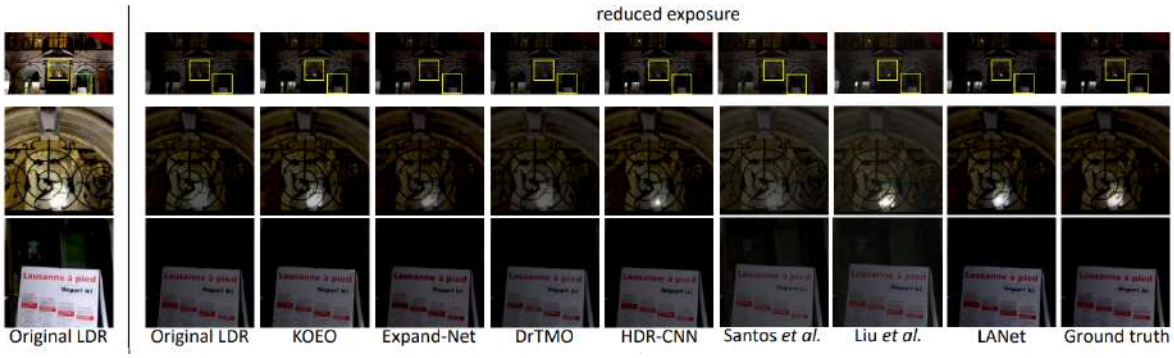


Figure 2.5 Qualitative comparison between different networks for over-exposed region recovery from LDR images. The comparison on predicted HDR is under the same dynamic range and an approximate exposure for best visual comparison. LANet performs the best among the given networks. Adapted from Yu*et al*. [110]

Many works have leveraged encoder-decoder network architectures. HDRCNN [46] estimates HDR image using a hybrid LDR encoder and decoder, which operate in the log domain. Santos *et al*. [82] argues that using the same convolution filters for well-exposed and saturated pixels gives way to checkerboard and halo artifacts. They masked out the saturated areas and introduced perceptual loss to counter these issues. ExpandNet [57] leverages a multiscale autoencoder that learns different features. These features are then merged to reconstruct an HDR image. Moriwarki *et al*. [66] argued that MSE loss causes blur effects and loss of semantic details in HDR images. Perceptual and adversarial loss were used to improve results. iTM-Net [43] proposed a cosine similarity loss, which distributed the pixel value of HDR images.

Inverse tone mapping networks suffer from a lack of quality training data and is an ill-posed problem. Endo *et al*. [17] proposed a framework with 2D encoders and 3D decoders with skip connections to generate bracketed LDR image stacks (up-exposed and down-exposed) from an HDR image using the camera response framework (CRF) database. Thus the HDR image is reconstructed from the bracketed LDR images. This method produced natural tones and improved results. SingleHDR proposed by Liu*et al*. [51] models the HDR-to-LDR formation into three sub-steps: dynamic range clipping, non-linear mapping with a CRF, and quantization. For each sub-step, a tailored network is learned to invert the process. The sub-networks are sequentially learned and then jointly fine-tuned in an end-to-end manner to reconstruct an HDR image. Hallucination Net, a sub-network of SingleHDR, recovers the over-saturated components of the image. It has shown better recovery of HDR intensities compared to bracketed LDR stacks methods.

So far, we have discussed methods for 2D images. A 3D panorama represents the physical world in a 3D wide-range view [94]. Lighting in a scene often varies spatially, making it more challenging to estimate 3D lighting for HDR imaging. Zhang *et al*. [112] presented an encoder-decoder-based

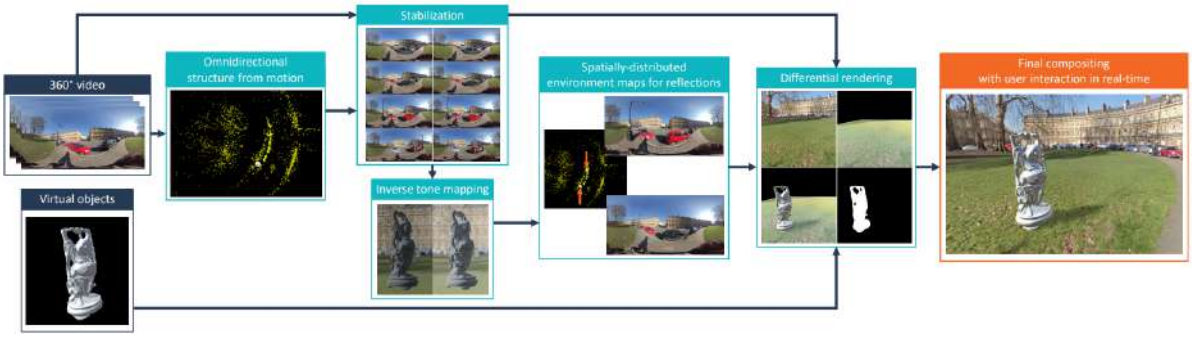


Figure 2.6 Tarko *et al*. [96] adds a virtual object to a captured omnidirectional video with accurate shading in real-time. However, it cannot perform novel view synthesis.

network to estimate HDR panorama image from an LDR panorama image in outdoor scenes. Gardner *et al*. [21] focused on indoor panorama HDR imaging. From an RGBD image, it estimated lighting as a set of discrete 3D lights with photometric and geometric parameters.

Luminance Attention Network (LANet) [110] is designed as a multi-task network with two streams. Luminance attention stream estimates a spatially-weighted attention map of the overexposed regions in the LDR image. The HDR reconstruction stream combines the attention map with the network output to estimate the HDR image. They also proposed an extension dubbed "panoLANet" for HDR panorama reconstruction. We build on the LANet architecture and augment it with an additional rendering loss. This combination outperforms alternatives in predicting indoor environments' high dynamic range radiance.

2.5 HDR scene capture

The objective is to capture a scene with accurate radiance. This enables us to augment the scene with virtual objects which are shaded accurately. Realistic relighting virtual objects in real scenes brings augmented reality closer to more people. Zhang *et al*. [111] recovers the indoor scene parameters like light sources, materials, and geometry from an RGBD scan. Walton *et al*. [100] combine a depth sensor with a fisheye camera in a SLAM-based approach to recover geometry and lighting. Tarko *et al*. [96] render a 360°video with correctly shaded virtual objects in real-time. The HDR environment maps of the captured frames were recovered using a deep network [57] and rendered using Unity. Lechlek *et al*. [44] estimate HDR images from a set of unstructured images with varying exposures. Using the point cloud proxy obtained from COLMAP, they recover a depth map which helps them synthesize new viewpoints. Yang *et al*. [108] learn background and objects as separate NeRF models and combine them with a captured panoramic image into a single image. In contrast, we (Chap:4) simultaneously learn HDR and synthesize new viewpoints.

Chapter 3

Neural View Synthesis with Appearance Editing from Casually Captured Images

Figure 3.1 We propose a pipeline that can synthesize novel views and edit the material of a scene, from handheld images of the scene with known environment illumination. We show few training images on the left, novel views of the scene in the middle and four different material edits to the leaves and the pot of the *Plant* scene in the right.

3.1 Introduction

In this chapter, we present a method that achieves simultaneous view synthesis and appearance editing of a scene from unstructured captures under known distant illumination. We tackle appearance editing of a scene instead of relighting by extending the DNR framework [97]. Our pipeline is designed according to the following principles:

- We seek to capture an unstructured scene and its lighting using consumer mobile cameras. Our system only requires images of a scene and its environment illumination, which can be easily captured on today's mobile devices.
- We perform fast view generation using a disentangled representation of the appearance and local irradiance function, which are learned from the scene. We use a neural network to *represent* and *render* the scene, beyond merely representing it.

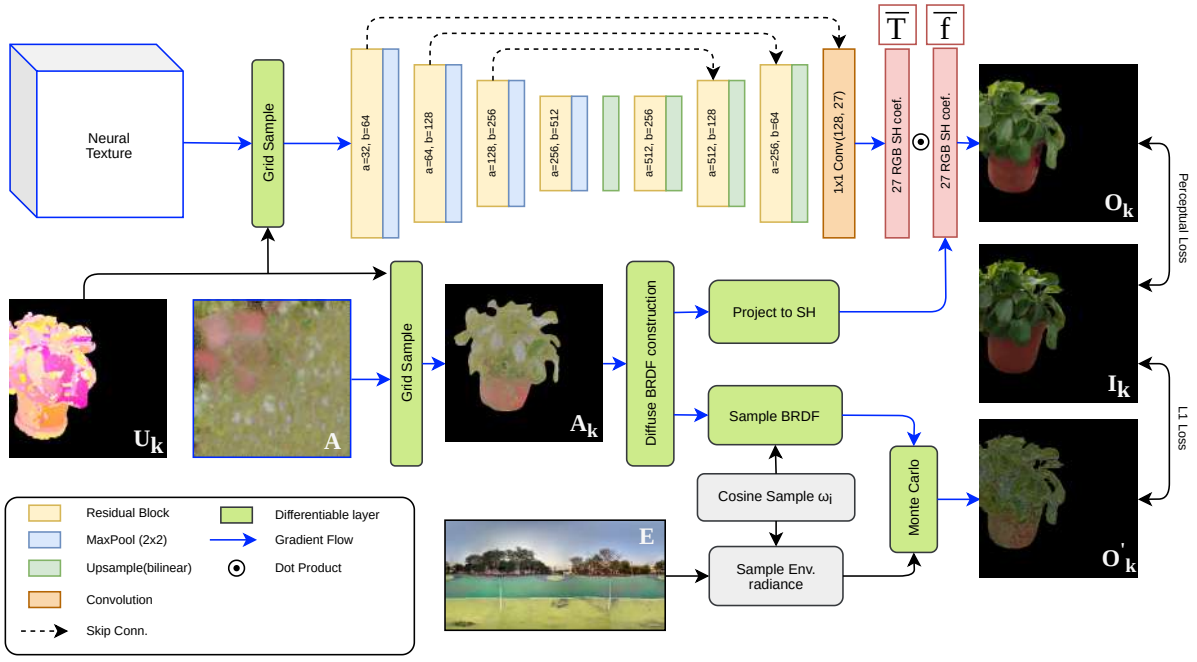


Figure 3.2 Overview of our pipeline. A high-dimensional neural texture is grid sampled according to the UV map U_k, which is then input to a neural network. This neural network outputs the SH coefficients of the LIF $\bar{T}$. Similarly, the albedo texture A is grid sampled according to U_k to obtain A'_k, which is then use to construct and project a diffuse BRDF to SH basis ($\bar{f}_r$). A dot product of the two (eq. (3.2)) produces the final image O_k. In the bottom, an approximate render O'_k (sec. 3.2.3) is generated using E and A_k. We take the perceptual loss between O_k and I_k and the ℓ_1 loss between O'_k and I_k (sec. 3.3.2)

We propose a formulation that disentangles the Bi-directional Reflectance Distribution Function (BRDF) and the captured object's local irradiance function (LIF). The BRDF and the LIF are estimated together, using a novel loss that ensures proper diffuse albedo estimation. This disentanglement allows the diffuse albedo to be edited or modified independently. The modified appearance is then combined with the learned LIF to produce the final image for a novel viewpoint. The contributions of this work are the following.

- A disentangled, appearance-independent, learned representation of the scene.
- A method to recover this representation from casually captured images of a scene.
- A novel pipeline to generate new views from this representation and edit material properties on synthetic and real scenes.

3.2 Method

Given the input data specified as $\{I_k, E\}_{k=1}^{N}$, consisting of a corpus of N images I_k of a scene, our aim is to render novel views of the scene and edit its appearance, assuming a fixed distant environment illumination E. We achieve this by disentangling the local irradiance and the BRDF. Fig. 3.2 shows the overview of our pipeline.

3.2.1 Preprocessing

We preprocess the input data and extract the camera poses p_k and object masks m_k for each image I_k along with a rough geometry G. We further generate a UV mapping of the geometry G, and store the UV coordinates for every pixel in all input images I_k as U_k (Fig 1). These per-view UV coordinates are useful for the view-dependent sampling of the neural texture (as done in DNR [97]) as well as sampling of the albedo texture. Typically, the recovered geometry G also contains vertex colors projected from I_k, which we use to initialize an albedo texture A. Our preprocessed input set is thus denoted as $\{I_k, E, p_k, m_k, U_k, G, A\}_{k=1}^{N}$.

3.2.2 Disentangling BRDF & local irradiance

For a pixel x of the input image I_k, the incoming radiance L is modeled by the rendering equation [38] as:

$$L(x) = \int_{\Omega} f_r(p_x, \omega_x, \omega_i) T(p_x, \omega_i) d\omega_i, \tag{3.1}$$

which is an integral over all directions on the hemisphere Ω of a product of two terms: the BRDF f_r, and the *Local Irradiance Function* (LIF) [6]. The LIF is defined as $T(p_x, \omega_i) = L_e(\omega_i) V(p_x, \omega_i)(\omega_i \cdot n)$ where L_e is the illumination, V is the visibility function, $\omega \cdot n$ is the cosine foreshortening factor, p_x is the 3D point corresponding to the 2D pixel x, ω_i is the incoming light direction, ω_x is the direction from p_x toward the camera. A different method of separating the rendering equation into the light transport function and the illumination has also been used for relighting [10]. The integral in eq. (3.1) can equivalently be evaluated as a dot product of the projections of f_r and T onto the Spherical Harmonic (SH) basis [36, 78, 10]:

$$L(x) = \bar{f}_r \circ \bar{T}, \tag{3.2}$$

where $\bar{f}_r$ and $\bar{T}$ are the SH coefficient vectors of the Bidirectional Reflectance Function (BRDF) and LIF respectively. Readers are referred to [36, 78, 91] for a detailed explanation of the integration of the product of two functions being evaluated as the dot product of their corresponding SH coefficients.

In fig. 3.2, the albedo texture A is grid sampled according to U_k to get a view-dependent albedo texture A'_k, which is used to construct a per-pixel diffuse BRDF. This BRDF is then projected to SH basis to obtain $\bar{f}_r$. For a single-pixel x in I_k, the BRDF f_r is a 2D slice of the full 5D BRDF, which can be readily projected to SH at run-time time. To generate the final image O_k, a per-pixel dot product

of $\bar{f}_r$ with the LIF $\bar{T}$ from the neural network is performed. The grid sampling, BRDF construction, and SH projections are all differentiable operations, thus allowing gradient flow back from O to A (blue path, fig. 3.2).

3.2.3 Estimating BRDF & learning the LIF

For training, we assume that the BRDF is diffuse, thus $f_r(p_x, \omega_x, \omega_i) = \frac{c}{\pi}$ where c is the colour to be estimated. As explained before, $\bar{f}_r$ is obtained by SH projecting a diffuse BRDF, which is itself constructed by grid sampling the albedo texture A. Since all operations that lead to $\bar{f}_r$ are differentiable, we can optimize for A during training.

The SH coefficients of the LIF are learned using a neural network, as shown in fig. 3.2. The neural texture and lif network architecture follow directly from DNR [97]. We use grid sampling to sample a high-dimensional Neural Texture based on the UV mapping U_k. This sampled neural texture is given as input to the network. The output of the network is an SH representation of the LIF $\bar{T}$ for each pixel. As mentioned in sec. 3.2.2, we take the dot product of $\bar{f}$ and $\bar{T}$. The dot product is performed in a per-pixel and per-color channel fashion, resulting in an output RGB image O_k, which is compared with the ground truth I_k for training.

A naive loss on the output RGB and ground truth alone results in a poor albedo texture. To better estimate the albedo, which disentangles lighting from the base color, we numerically integrate the environment map with the diffuse BRDF to obtain an approximate render and evaluate loss with the ground truth (fig. 3.2, bottom right). Specifically, we generate N ω_i directions and use them to sample the constructed diffuse BRDF and the environment map E. We then perform Monte Carlo (MC) integration to obtain an approximate rendered image O'_k:

$$O'(x) = \frac{1}{N} \sum_{i=0}^{N} \frac{f_r(p_x, \omega_x, \omega_i) E(\omega_i)}{p(\omega_i)}. \tag{3.3}$$

The notation is as defined before, and $p(\omega_i)$ is the PDF from which ω_i is sampled. We show that this is necessary to obtain a good albedo texture in sec. 3.4.2. The gradient flow in our pipeline is shown with blue arrows in fig. 3.2. During backpropagation, the albedo texture A is also optimized along with the neural network and corresponding neural texture.

3.3 Experimental Evaluation

In this section, we provide implementational details of our method (sec. 3.3.1). We further analyze and evaluate our method on synthetic and real scenes (sec. 3.3.3, sec. 3.3.4 resp.).

3.3.1 Implementation Details

3.3.1.1 Preprocessing

For real scenes, we obtain the environment illumination E using the Google Street View app on a consumer mobile phone. We use COLMAP [85] to estimate the camera poses p_k and obtain the rough geometry G by running Poisson surface reconstruction [41] on the dense point cloud. We do a minimal manual cleanup of the rough geometry and remove parts of it that are not the object of interest (eg. ground, background objects). The UV mapping for G is computed using Blender 3D, and the albedo texture A is initialized with vertex colors from COLMAP. Finally, the masks m_k are obtained using U2Net [77]. For synthetic scenes, we initialize the albedo texture A with $(0.5, 0.5, 0.5)$, and we use ground truth geometry, poses, and environment illumination.

3.3.1.2 BRDF SH coefficient computation

Spherical harmonics (SH) are the orthonormal basis functions $Y_{lm}(\theta, \phi)$ with $l \geq 0$ and $-l \leq m \leq m$. The real spherical harmonics are defined as:

$$Y_{lm}(\theta, \phi) = \sqrt{\frac{(2l+1)}{4\pi}\frac{(l-|m|)!}{(l+|m|)!}} P_l^{|m|}(cos\theta)t(|m|\phi), \tag{3.4}$$

where P_l^m are the associated Legendre polynomials and the trigonometric function $t(|m|\phi)$ is set to 1 when $m = 0$, given by $\sqrt{2}cos(m\phi)$ when $m > 0$ and $\sqrt{2}sin(m\phi)$ when $m < 0$. Readers are requested to refer to [36, 91] for more details.

Projection of a function $a(\omega)$ to SH basis gives a set of coefficients, which depend on the order of projection. These coefficients are obtained by integrating the function a with the corresponding SH basis functions. In practise, we use Monte Carlo (MC) to compute this integral.

$$a_{lm} = \int_\Omega a(\omega_i) Y_{lm}(\omega_i) d\omega_i \approx \frac{4\pi}{N} \sum_s a(\omega_i) Y_{lm}(\omega_i). \tag{3.5}$$

The integral is performed over the entire unit sphere Ω, and the MC samples are generated with a uniform probability distribution.

3.3.1.3 Network Details

Our network architecture is based on Deep Residual UNet [117, 29] following Gao *et al.* [19] and is implemented in PyTorch [73]. We follow a similar methodology as DNR [97] and use a four-level mipmap Laplacian pyramid for the 32-channel neural texture and set the top level's resolution to 512x512. We also multiply channels 3-12 of the neural texture with nine SH coefficients evaluated at per-pixel view direction. The output of our network represents the per-pixel LIF with a 27 channel volume corresponding to order two ($l = 2$) SH projection (nine coefficients for each colour channel).

We similarly project the BRDF to order two SH. We also pre-train a smaller binary mask network and post multiply the mask with the LIF and the BRDF SH representation as is done by Gao *et al*. [19]. The final image is obtained by a dot product of the LIF SH representation and the BRDF SH representation, which is then compared to the ground truth for training. We train our network using the Adam [42] optimizer with $lr = 1 \cdot e^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1 \cdot e^{-8}$. The network is trained for approximately 250 epochs.

3.3.2 Loss Functions

Mask Loss We apply the binary cross-entropy loss on the mask prediction, following Gao *et al*. [19]. As mentioned before, the mask network is pre-trained for all training views using ground truth masks from U2Net.

Rendering Loss. We use the perceptual loss paired with the ℓ_1 loss between O_k and I_k to preserve sharp details. Specifically, we use the feature reconstruction loss from a pre-trained VGG16 [50] network, which is given by :

$$\mathcal{L}^{j}_{feat}(y', y) = \frac{1}{C_j H_j W_j} \left\| \phi_j(y') - \phi_j(y) \right\|_2^2, \tag{3.6}$$

where ϕ_j is the activation of the *jth* convolutional layer with dimensions $C_j \times H_j \times W_j$ representing number of channels, width and height of the feature map, respectively. Here, y denotes the predicted output and y' is the ground truth. We use the *relu_3_3* (*j=relu_3_3*) [1] feature representation in our experiments. Thus, the rendering loss is given by:

$$\mathcal{L}_{rend} = 0.8\ell_1(O_k, I_k) + 0.2\mathcal{L}^{relu\_3\_3}_{feat}(O_k, I_k). \tag{3.7}$$

Albedo Loss We apply an additional ℓ_1 loss on O' and I_k to obtain a better estimate of the albedo texture:

$$\mathcal{L}_{albedo} = \frac{1}{N} \left\| I_k(x) - O'_k(x) \right\|_1, \tag{3.8}$$

where N is the number of image pixels.

Our total loss is an equally weighted combination of the above:

$$\mathcal{L} = \mathcal{L}_{rend} + \mathcal{L}_{albedo}. \tag{3.9}$$

3.3.3 Results on synthetic scenes

We first evaluate our network's results and analyze the learnt representation of the LIF on synthetic datasets. We use the *Kitty* synthetic scene from PhySG [113] and render our own scene, *Stanford Buddha*, using Mitsuba 2 [70]. Both scenes have the camera looking at the object with positions samples on a trajectory on the upper hemisphere and an environment map as the only light source. We use 100 frames as the training set and 100 as the test set, with a resolution of 512 $\times$ 512.

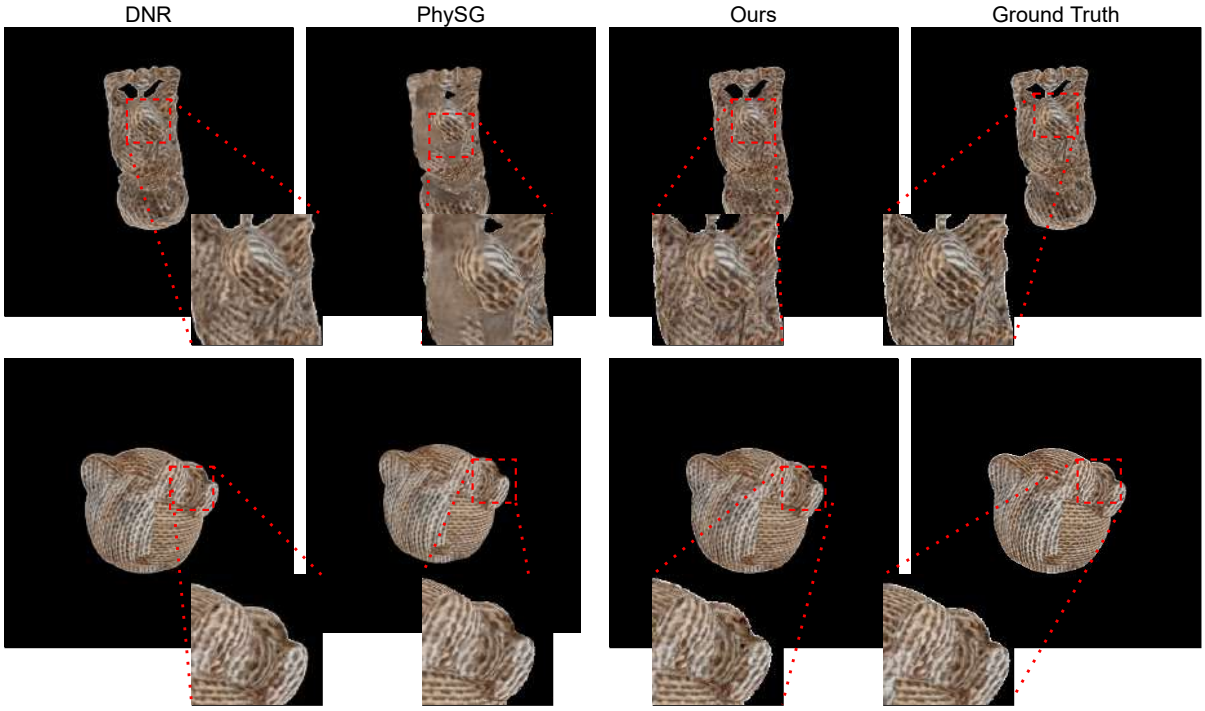


Figure 3.3 Results of novel view synthesis on synthetic data: Qualitative comparison of view synthesis results with previous methods and the ground truth. Our method performs at par with the others. Refer to tab. 3.1 for quantitative metrics.

	DNR [97]	PhySG [113]	Ours
Datasets	PSNR↑	PSNR↑	PSNR↑
Kitty	31.25	22.3.54	32.42
Buddha	30.53	19.87	31.04

Table 3.1 Quantitative metrics PSNR for two synthetic scenes compared to DNR and PhySG. The values are averaged over the test set.

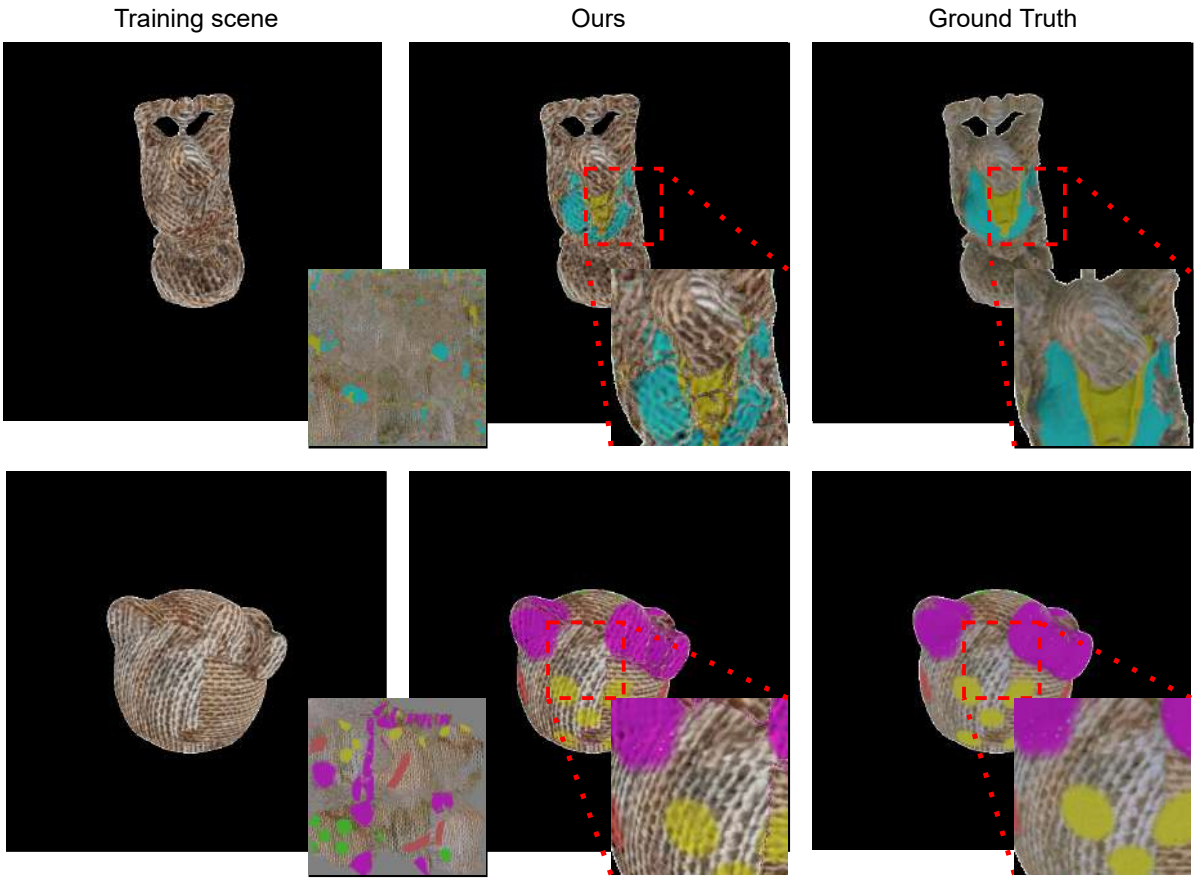


Figure 3.4 Diffuse colour edits: We edit the diffuse colour texture to a different one as shown in insets.

3.3.3.1 Novel View Synthesis

We perform a qualitative and quantitative comparison of view synthesis with DNR [97], and PhySG [113]. For a fair comparison, we use a constant white environment map for both scenes and initialize PhySG to that environment. Fig. 3.3 and tab. 3.1 shows the visual comparison and quantitative metrics (PSNR, SSIM) for the two scenes, respectively. Our network performance is at par with DNR and PhySG, both quantitatively and qualitatively. PhySG performs good for a scene having simple geometry (*Kitty*), but is unable to perform well for complex geometry (*Buddha*). We would like to emphasize here that our method can not only perform view synthesis but also edit the scene's appearance and also works for relatively complex geometry.

3.3.3.2 Appearance Editing.

We edit the diffuse color texture of both scenes and render the result using our pipeline, which is shown in fig. 3.4. We compare this to the ground truth, which is rendered using Mitsuba 2. Our method

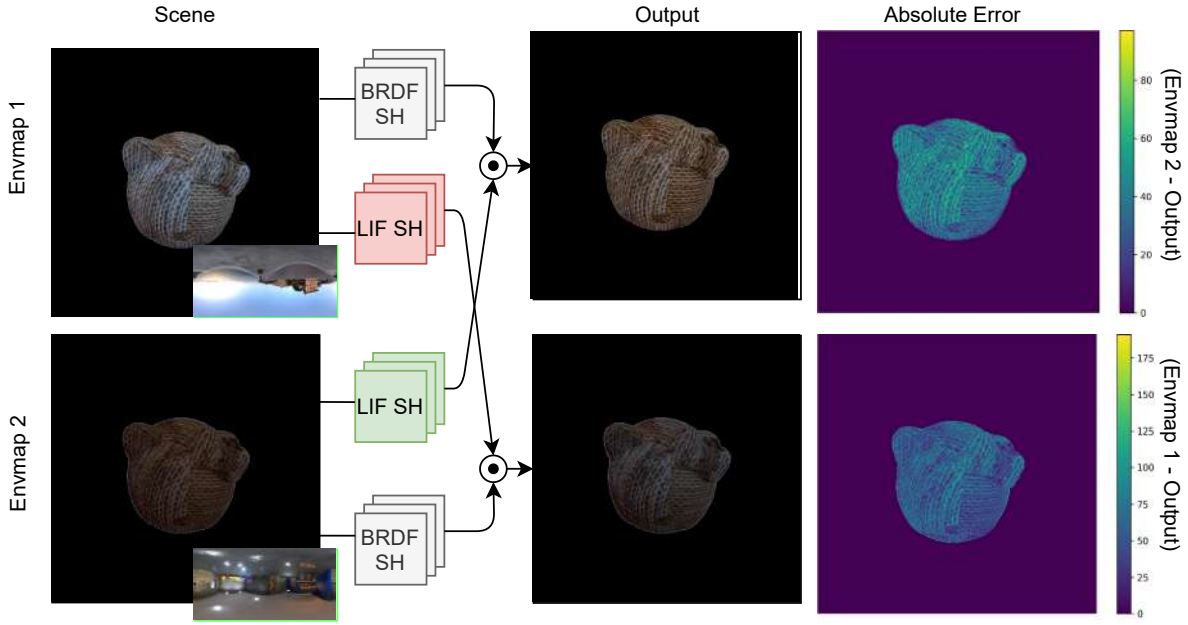


Figure 3.5 Interchanging the learnt LIF representation: Here, we train the *Kitty* with the same material (brown cloth) but different envmap lighting (green insets), and interchange the learnt LIF. The envmaps for the two scenes are shown in insets. The LIF interchange results being almost identical indicate that the LIF representation is accurate to a high degree. The error maps are between the ground truth lit scene and the LIF interchange output for the same envmap.

is directly able to handle such changes, thanks to the disentangled formulation and representation of the LIF.

3.3.3.3 Learnt LIF representation.

We now analyze the learnt representation of the LIF. For this, we render two differently lit variants of the *Kitty* scene with a brown cloth material (fig. 3.5). Denote the first variant as *Envmap 1 Kitty* and the second as *Envmap 2 Kitty*. We train two separate networks on both variants. From a novel viewpoint, we then render the *Envmap 1 Kitty* with the learnt LIF of *Envmap 2 Kitty* and vice-versa. Results are shown in fig. 3.5. The error map for the result of LIF interchange for *Envmap 1 Kitty* is computed using the absolute difference between the *Envmap 1 Kitty* regular and LIF changed output. Similarly, we compute the error map for *Envmap 2 Kitty*. As seen from fig. 3.5, LIF interchanged results very closely match the ground truth, which suggests that the LIF is practically independent of the underlying appearance.

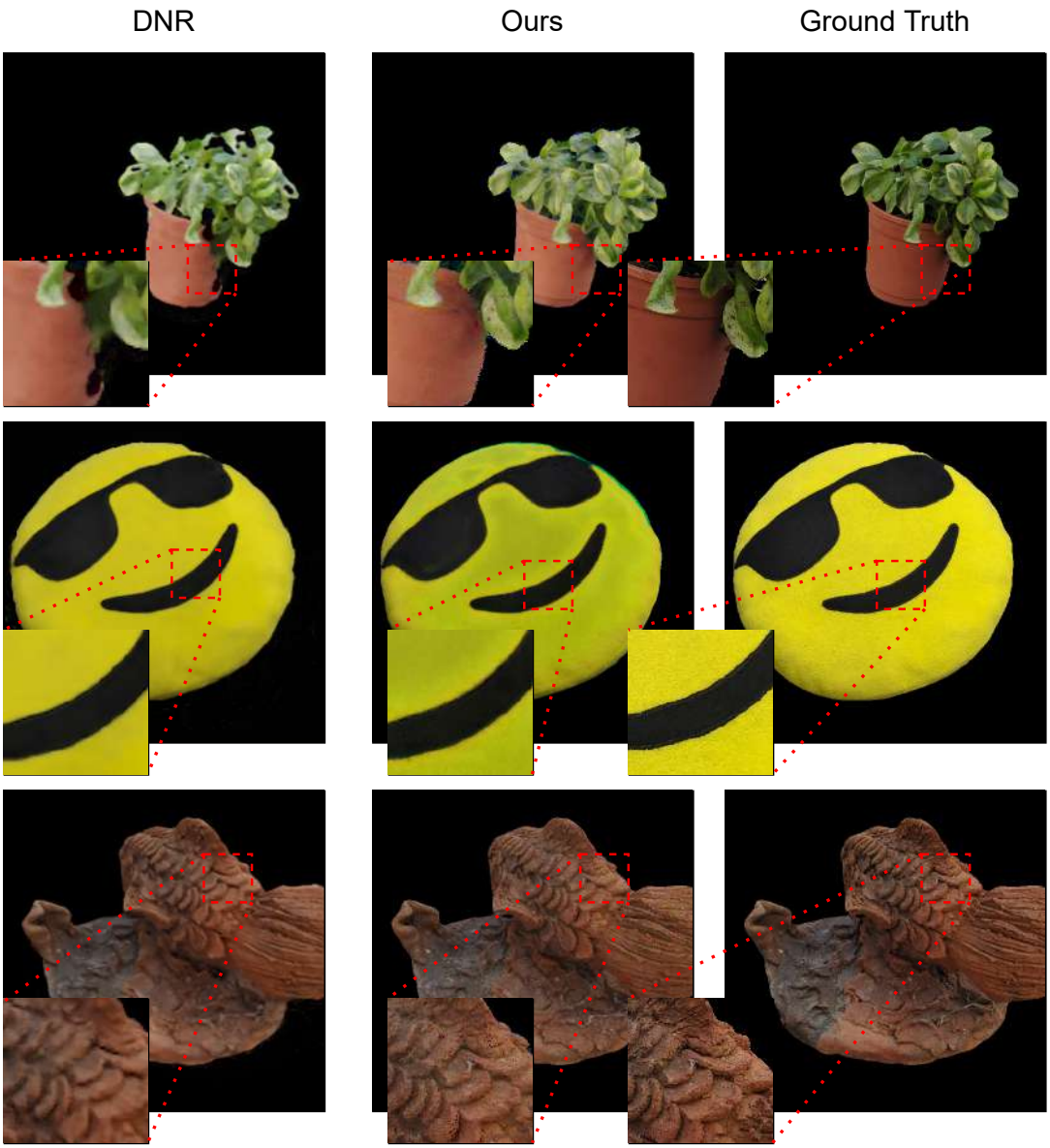


Figure 3.6 Results of novel view synthesis on real data: Qualitative comparison of view synthesis results with previous methods and the ground truth. Our method performs at-par with the others. Refer totab. 3.2 for quantitative metrics.

	DNR [97]		Ours	
Datasets	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Fish	26.59	0.76	26.54	0.79
Cushion	24.16	0.78	23.20	0.83
Plant	21.93	0.82	22.91	0.83

Table 3.2 Quantitative metrics for three real scenes, *Fish*, *Cushion* and *Plant*, compared to DNR. The values are averaged over the test set.

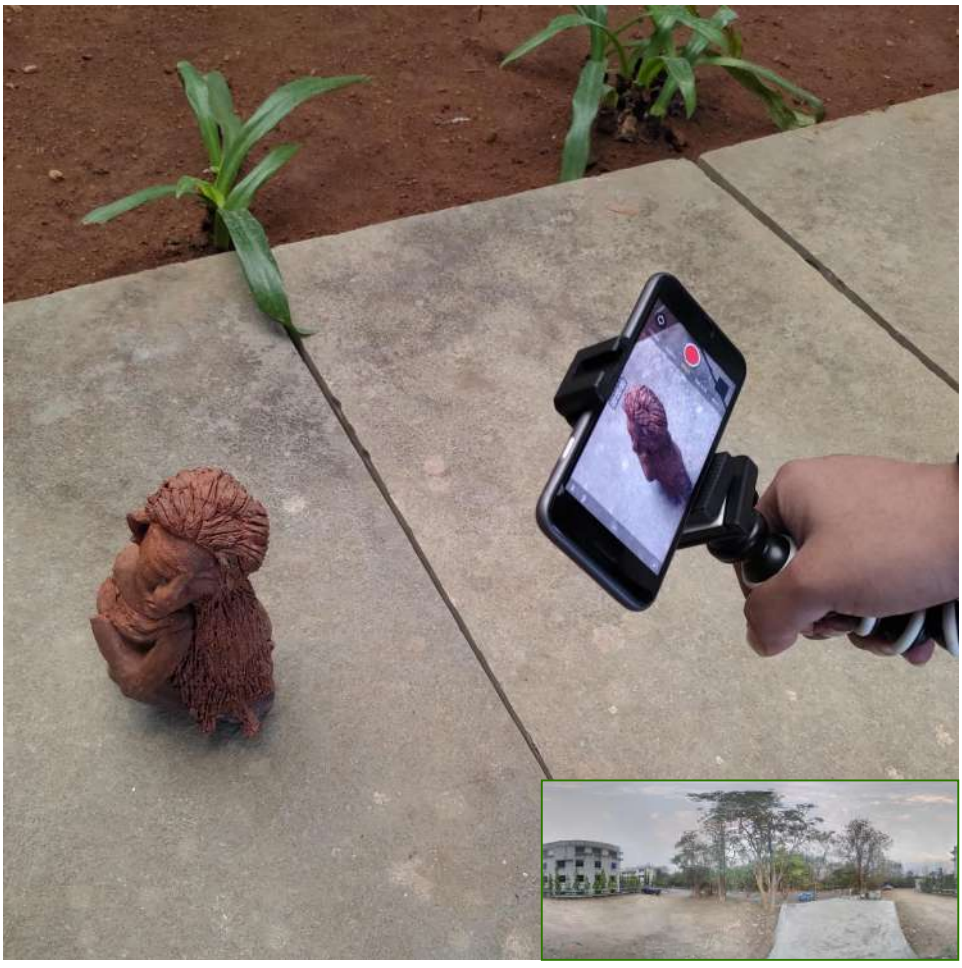


Figure 3.7 Capture Setup: We use a consumer mobile phone for 360°capture of a object. The environment map (shown in green inset) is also captured with the same mobile phone using the Google Street View app.

3.3.4 Results on Real Scenes

Fig. 3.7 shows our capture setup. We capture four scenes with a handheld mobile phone: *Plant*, *Fish*, *Cushion* and *Woman*. We capture a 60fps video of all scenes by moving the camera around the object and extract every 3rd frame for use with training, resulting in roughly 400 frames. Out of these, we use 100 as the test set. As mentioned before, we capture environment illumination with the Google Street View app.

3.3.4.1 Novel View Synthesis

Fig. 3.6 shows the qualitative results and tab. 3.2 shows the quantitative values for our own scenes and the *Hands* scene in comparison with DNR. Even though our quantitative values are slightly lesser, our method produces plausible and photo-realistic view synthesis results. Note that PhySG already fails on complex synthetic geometry. In our experiments, PhySG performed much worse for such complex geometry with estimated poses. Hence we do not directly compare to them. As stated in their paper, they focus on specular objects only.

3.3.4.2 Appearance Editing

We perform appearance edits on all three real scenes. Results are shown in fig. 3.11, with the changed diffuse texture highlighted in green. Note that for real scenes, the UV map does not necessarily make semantic sense. Our network preserves details that naive colmap geometry rendering is unable to reproduce, such as the texture detail and the underlying geometry variation.

3.4 Discussions

3.4.1 Material Estimation

We compare our method with an alternative material estimation pipeline. In this pipeline, we use Mitsuba 2 [70] to perform differentiable rendering optimization as a preprocessing step for material estimation. We freeze the learned material. The LIF is learned and combined with the learned material in a similar fashion. Fig. 3.10 shows the qualitative comparisons between the "Mitsuba" pipeline and our pipeline. Although the Mitsuba pipeline is more powerful for synthetic scenes and can learn different material models, we notice that there are distinct purple artifacts appearing with real scenes view synthesis. This artifact is not visible in our joint estimation pipeline. We infer that our pipeline performs better as compared to Mitsuba pipeline. Our joint estimation eliminates these purple artifacts.

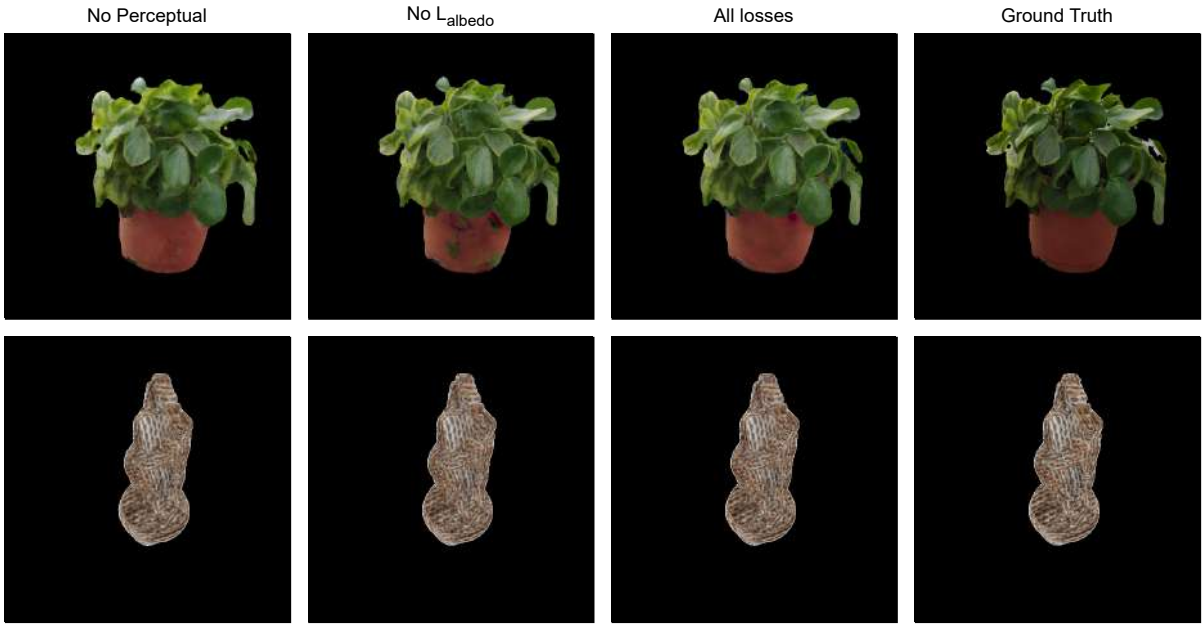


Figure 3.8 Loss function ablation: Comparison without perceptual loss, without albedo loss, with all losses and ground truth. The composite loss helps preserve the sharp details. Perceptual loss helps preserve sharp details, while the albedo loss heps better disentangle the BRDF from the GT.

3.4.2 Loss Ablation

We show the importance of our combined loss (eq. (3.9)). We train two variants of our pipeline, one with ℓ_1 loss instead of perceptual loss for $\mathcal{L}_{rend}$ and the other without the $\mathcal{L}_{albedo}$ loss. We observe that perceptual loss makes the output sharper as opposed to ℓ_1 loss. On the other hand, we observe that $\mathcal{L}_{albedo}$ loss is crucial in learning the albedo texture. Without this, we observe that the outputs are very poor as albedo is not learnt properly. We can observe visual artefacts in *Plant* scene. Also, the $\mathcal{L}_{albedo}$ loss is responsible for disentangling the diffuse albedo texture from LIF and ensures that there is no leakage of the albedo to LIF. We show this comparison in fig. 3.8, on two scenes: *Fish* and *Plant*.

3.4.3 Geometry Ablation

Next, we study the effect of geometry quality on the results of our method. To do this, we obtain two decimated versions of the *Fish* and *Plant*. We then train four separate networks on these decimation levels (two of Buddha and two of Plant). While the quality of results slightly decreases, we observe that the network is able to cover for imperfect geometry and performs well. Fig. 3.9 shows this analysis.

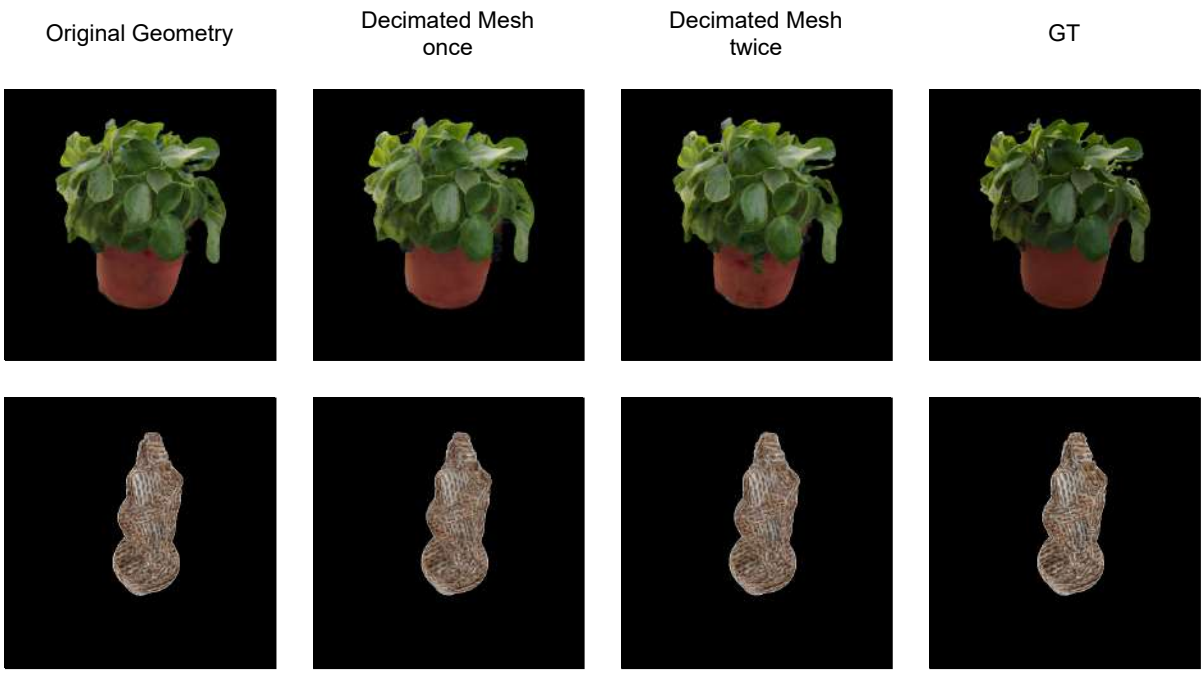


Figure 3.9 Geometry ablation: We decimate the *Plant* and the *Fish* twice and obtain two decimation levels, and train our pipeline on both levels. Results show that our method is fairly robust to degrading geometry levels.

3.4.4 Albedo Initialization

Our albedo texture is initialized with the vertex colors from COLMAP. Neural networks can heavily compensate for missing information, and thus not initializing the albedo texture may result in poor albedo and overcompensated LIF. This directly means that the appearance is no longer editable, since its now a part of the LIF. This is especially true for complex scenes.

Obtaining the albedo texture is a implementational challenge for synthetic scenes, since we assume ground truth geometry (without vertex colors). However, for simplistic scenes like *Kitty*, an albedo texture which is initialized to a random value works out reasonably well.

3.5 Limitations

One limitation of our method is the inability to handle specular objects. Specular objects are challenging to SfM methods like Colmap and can't be handled well by most methods. Other material estimation techniques also struggle when shininess increases. Our pipeline is currently tuned for single objects. With complete scenes, the challenge is to mitigate the error in geometry such that material edits do not reveal any artefacts. Another limitation is that the material change cannot be highly glossy or

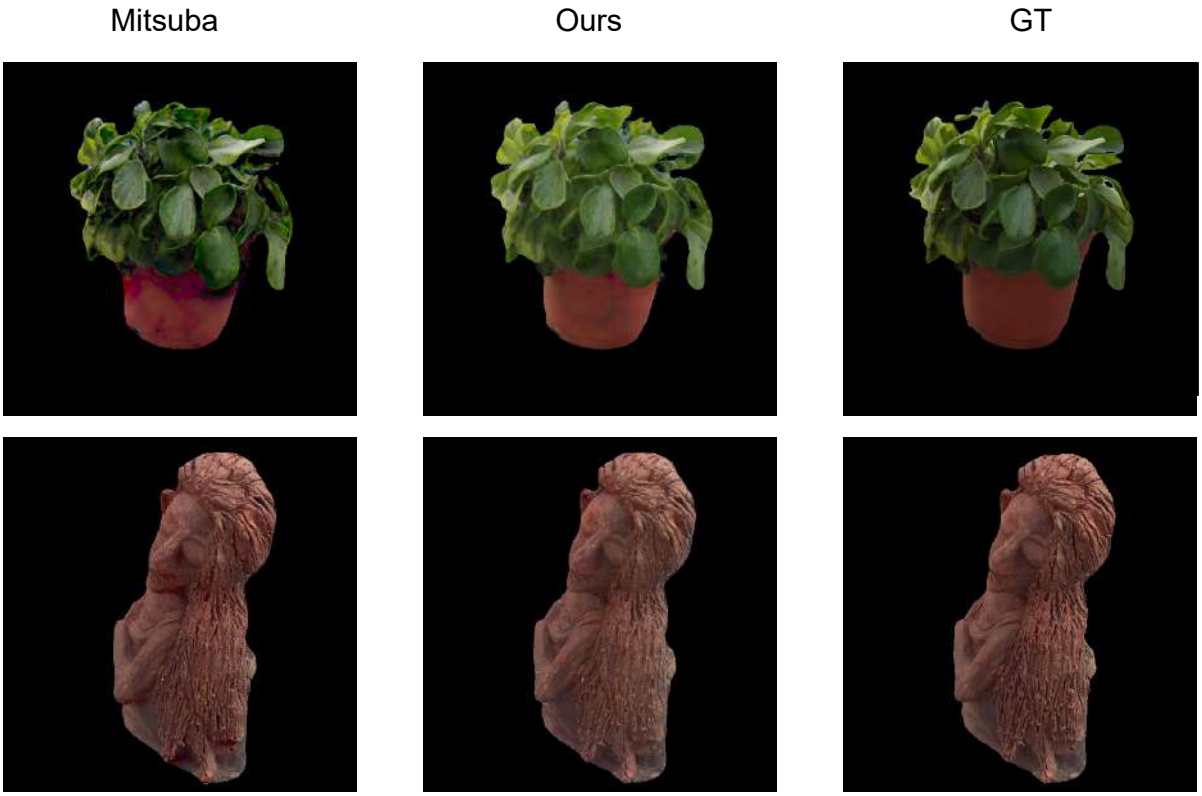


Figure 3.10 Material Estimation: Alternatively, we independently estimate albedo texture A using Mitsuba 2 as the differentiable renderer (Left). We then use it in our pipeline to train the LIF net. Note that we freeze the optimization for A. We compare this with our full pipeline (middle) and ground truth (right).

specular. This is in part due to the inability of SH to represent high-frequency details and in part due to the inadequacy of the learnt LIF representation. Handling glossy edits to the appearance requires further analysis and possibly a different basis representation.

For future work, we would like to explore other basis representations for representing the LIF and the BRDF. Such bases could help extend our method to handle glossy or even nearly-specular material edits. Alternatively, we can improve the material disentanglement capabilities by designing and using stronger priors like they are in DNL [19] where they use radiance cues to better disentangle lighting. Another interesting direction of research is to incorporate relighting into our method while retaining the ability to perform appearance editing and view synthesis.

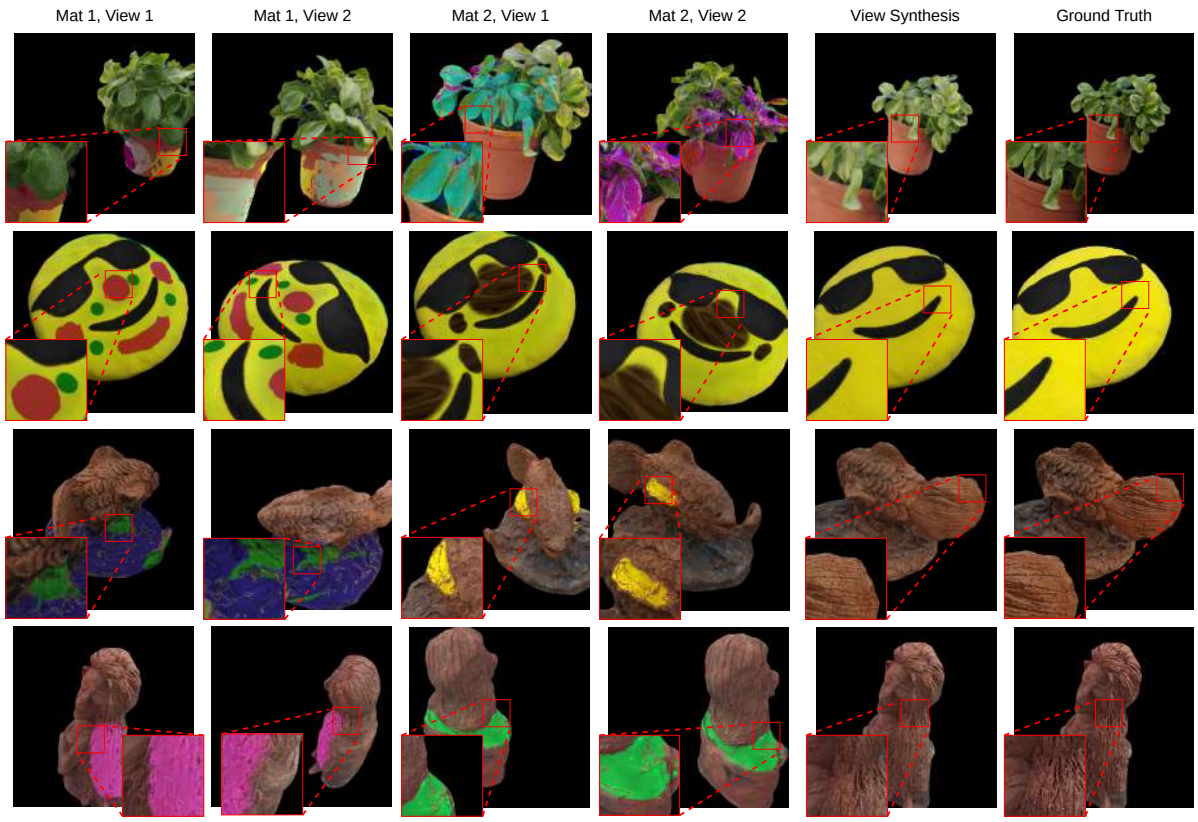


Figure 3.11 View synthesis and material editing results of our pipeline on four scenes: *Plant*, *Cushion*, *Fish* and *Woman*. Our method is able to produce plausible results for both tasks.

3.6 Conclusions

In this chapter, we propose a neural rendering framework that achieves simultaneous appearance editing and view synthesis. This problem is orthogonal to current approaches that achieve relighting along with view synthesis. Our method proposes a BRDF independent formulation of the local irradiance function (LIF) and its subsequent projection to SH basis. We further present a method to recover the BRDF and LIF from unstructured photographs, where only the latter component involves learning. This allows us to modify the underlying material of the scene to generate photo-realistic and plausible renderings, as demonstrated. Our method's performance is at par with the current state-of-the-art approaches in view synthesis while adding the capability of appearance editing.

Chapter 4

Casual Indoor HDR Radiance Capture from Omnidirectional Images

Figure 4.1 We capture the continuous HDR radiance of an indoor scene. Our PanoHDR-NeRF approach takes a) casually captured LDR images from an off-the-shelf camera (shown in inset) as input, and performs b) novel view synthesis of the indoor scene. c) As oppposed to existing techniques such as NeRF++ [114] (left), PanoHDR-NeRF (right) properly estimates the HDR radiance of the scene, visualized by relighting virtual test objects.

4.1 Introduction

In this chapter, we present PanoHDR-NeRF, a novel neural representation of the full HDR radiance field of an indoor scene that can be captured casually. HDR radiance at any point in the scene can be produced from PanoHDR-NeRF subsequently (fig. 4.1). Our method does not require any special equipment or complicated capture protocols. It accepts as input a video sequence captured by freely moving a commercial 360° camera around the scene. As output, it produces the HDR radiance at any given location in the scene. To do so, we leverage two deep neural networks: 1) an LDR2HDR model that predicts the HDR radiance from a single LDR panorama captured by the camera and 2) a modified NeRF++ model trained on the predicted HDR outputs of the first network. To evaluate our proposed

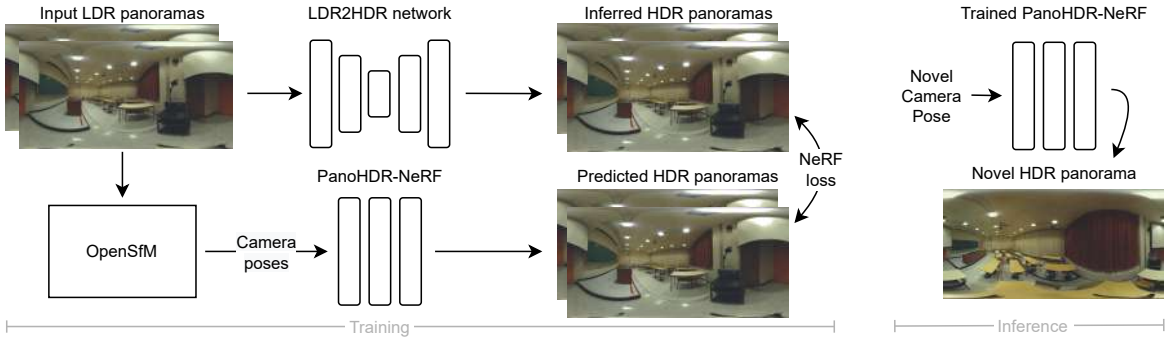


Figure 4.2 Overview of our pipeline. At training time (left), the captured panoramas are linearized (calibrated using a color checker, not shown) and processed by a pre-trained LDR2HDR network to obtain HDR estimates. The HDR panoramas, along with the camera poses obtained with OpenSfM [56], are used to train the PanoHDR-NeRF network, which learns to synthesize HDR scene radiance at any point in the scene. At inference time (right), we simply provide the novel camera pose and obtain the corresponding novel HDR panorama.

method, we capture a set of six different indoor scenes, which we augment with a set of ground truth HDR light probes at each scene. Our experiments demonstrate that, despite the simplicity of the capture procedure, PanoHDR-NeRF can accurately predict HDR radiance in a variety of challenging conditions. Our approach can render 360° HDR light probes, which can be used to provide correct lighting effects when the scene is augmented with virtual objects.

4.2 PanoHDR-NeRF Method

Given a set of LDR panoramas $\{I_k\}_{k=1}^N$ captured freely using an off-the-shelf 360° camera and associated camera poses $\{\mathbf{P}_k\}_{k=1}^N$ obtained with SfM, our objective is to predict the HDR radiance of the scene at any novel viewpoint. We achieve this by recovering HDR values from LDR frames using learning-based inverse tone mapping and then using them as the supervision for novel view synthesis. An overview of our method is given in fig. 4.2.

4.2.1 High dynamic range with LDR2HDR network

Extrapolating HDR from LDR inputs is typically framed as recovering values in the over- and under-exposed regions. Our work focuses on recovering the over-exposed regions exclusively, with the goal of predicting accurate HDR radiance values (specially the intensities of light sources) for realistic virtual object insertion.

4.2.2 Network architecture

To this end, we borrow the network architecture proposed in Luminance Attentive Networks (LANet) [110], which is designed as a multi-task network with two *luminance attention* and *HDR reconstruction* streams. The former attempts to create a spatially-weighted attention map of the over-exposed regions in the input image, while the latter uses the attention map to estimate the HDR images. Note that we do not use their proposed adaptation[1] for panoramas since it did not improve the performance in our case. So, we train the model on equirectangular images directly.

4.2.3 Loss functions

For the LDR2HDR module, we use the same loss function ℓ_{lanet} as [110], which combines scale invariant and luminance segmentation losses. In addition, we use a rendering loss ℓ_{rend} that leverages pre-computed radiance transfer [91]

$$\ell_{\text{rend}} = ||\mathbf{T}\,\mathbf{y}_{\text{HDR}} - \mathbf{T}\,\mathbf{t}_{\text{HDR}}||_2\,, \tag{4.1}$$

where $\mathbf{T}$ is a pre-computed transport matrix for a lambertian scene (without interreflections), $\mathbf{y}_{\text{HDR}}$ is the predicted HDR panorama, and $\mathbf{t}_{\text{HDR}}$ is the ground truth. To ensure high frequency lighting is properly captured, a "bumpy sphere" on a flat ground plane seen from above is used for rendering as in [112]. The final loss for training the LDR2HDR network is an equally weighted combination $\ell_{\text{hdr}} = \ell_{\text{lanet}} + \ell_{\text{rend}}$.

4.2.4 Datasets and training

We pretrain the LDR2HDR network on the Laval Indoor HDR Dataset [22], which consists of 2,400 HDR panoramas captured in a variety of indoor settings, with a train, validation and test split as 80:10:10. We augment the training set with random rotations (about the vertical axis), intensity changes (multiply the image by 2^γ, with $\gamma \sim \text{U}(-0.1, 0.1)$) and exposure changes (make the median intensity of image $0.5 + \gamma$). Here, $\text{U}(a, b)$ indicates a uniform distribution in the $[a, b]$ interval. After augmentation, the resulting HDR panorama is used as target $\mathbf{t}$ for training. The input $\mathbf{x}$ is created by clipping $\mathbf{t}$ to the [0,1] interval. We further apply hue shift and unsharp mask (amount =1, $\sigma \sim \text{U}(0, 3)$), add small amount of per-pixel gaussian noise ($\sigma = 0.01$), and augment the tonemapping $\mathbf{x}^{1+\gamma}$ to simulate the behaviour of a real camera. We train the network for 1500 epochs using the Adam [42] optimizer with a learning rate $\eta = 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$.

Since the sensor used for Laval Indoor HDR Dataset (Canon 5D Mark iii camera) and the sensor we use to capture indoor scenes (Ricoh Theta Z1) are different, a domain gap was observed. To alleviate this, we finetune the LDR2HDR network on a small dataset of 78 HDR panoramas captured using the test camera by running 130 additional epochs using the Adam [42] optimizer with the same parameters as above.

[1] This involves projecting the upper half of the hemisphere onto a plane and using this as an additional input to the network.

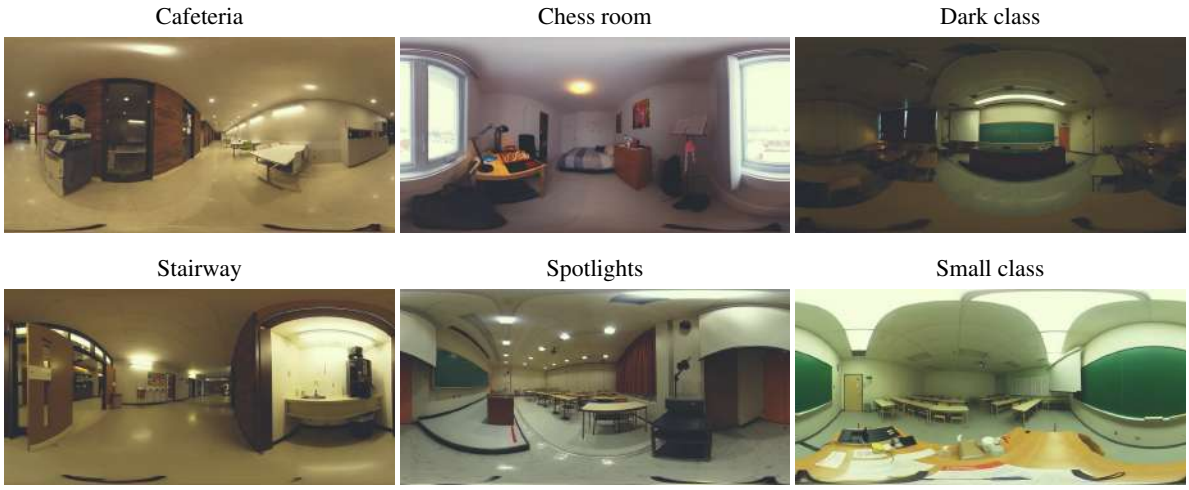


Figure 4.3 Representative images from each test scene used in the experiments and captured with the Theta Z1 camera.

4.2.5 Continuous HDR radiance with PanoHDR-NeRF

We consider a scene as a HDR radiance field and train a network to predict radiance and density at any given 3D point in the scene.

4.2.6 Network architecture

We take inspiration from and combine several recent work on NeRF. First, we employ the NeRF++ architecture (sec. 1.3.6.2) as the base. Second, similar to [5], we incorporate the anti-aliased conical frustums from Mip-NeRF [4].

Equirectangular images correspond to the projection of a spherical signal onto a 2D plane, where (normalized) pixel coordinates (i, j) are related to azimuth $\varphi \in [-\pi, \pi]$ and elevation $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ angles by

$$i = \frac{1}{2\pi}\varphi\cos\theta + \frac{1}{2}\,, \text{ and } j = \frac{\frac{\pi}{2} - \theta}{\pi}\,. \tag{4.2}$$

To train PanoHDR-NeRF, we sample rays in spherical coordinates instead of pixel coordinates, where $\theta \sim \mathrm{U}(-\pi, \pi)$, $\varphi = \cos^{-1}(2\beta - 1)$, and $\beta \sim \mathrm{U}(0, 1)$.

4.2.7 Loss functions

We train PanoHDR-NeRF using supervision from LDR2HDR network. Traditional volume rendering methods are used to predict the radiance and densities of points sampled on the ray as in [114, 4]. We define NeRF loss ℓ_{nerf} between predicted radiance $\hat{e}$ and ground truth HDR e as

$$\ell_{\text{nerf}} = \sum_{r \in \mathcal{R}(\mathbf{P})} \|\hat{E}(\mathbf{r}) - E(\mathbf{r})\|^2\,, \tag{4.3}$$

	Input LDR			LDR2HDR pre-trained			LDR2HDR finetuned		
Dataset	PU-PSNR↑	RMSE↓	HDR-VDP3↑	PU-PSNR↑	RMSE↓	HDR-VDP3↑	PU-PSNR↑	RMSE↓	HDR-VDP3↑
Chess room	31.659	0.051	8.067	33.994	0.048	8.234	36.995	0.005	8.492
Stairway	31.964	0.224	7.881	33.297	0.213	8.016	33.685	0.019	8.489
Cafeteria	25.299	5.378	6.098	26.664	5.268	6.418	28.499	4.061	7.164
Spotlights	23.939	3.489	6.001	25.118	3.438	6.097	28.966	0.877	7.667
Dark classroom	30.657	0.364	7.429	32.125	0.340	7.592	32.594	0.262	8.135
Small classroom	32.353	2.162	8.018	34.095	1.889	8.267	35.465	0.221	8.678
Overall	28.399	1.703	7.243	30.913	1.634	7.406	33.651	0.696	8.209

Table 4.1 Quantitative comparison of different strategies for recovering radiance across captured scenes. "Input LDR" are on the images captured by the camera, "LDR2HDR pre-trained" is on our network pre-trained on the Laval Indoor Dataset, and "LDR2HDR finetuned" is after the network finetuned to test camera. As expected, finetuning helps bridge the domain gap and significantly improves the results.

where $E = log(1 + e)$ and $\mathcal{R}(\mathbf{P})$ is the set of camera rays at pose $\mathbf{P}$.

The photographer, who inevitably appears in the captured images, is segmented using an off-the-shelf segmentation algorithm [28], and the corresponding pixels are ignored in the loss.

4.2.8 Datasets and training

To obtain training data for a given indoor scene, we casually capture a set of LDR panoramas using a commercial 360° camera (we use the Ricoh Theta Z1 model). We attach the camera to a portable tripod and capture a 360° video while waving the camera around the scene to cover the entire volume as much as possible for a few minutes (typically 3–5 minutes for the scenes used in the experiments). Approximately, 200 frames are then extracted from the video at even intervals. The camera parameters of the input LDR panoramas are recovered using OpenSFM[56] and given as input to PanoHDR-NeRF. An eight-layer MLP with 256 channels is used for predicting radiance and densities at the sampled points. Along each ray, we sample 64 points for training the coarse network and 128 points for training the fine network. The batch size of rays is 1024. We use integrated positional encoding to encode the inputs of the network as used in MipNeRF [4]. Similarly a single MLP is used to encode the scene. In addition, we also use spherical sampling, which weights pixels at the poles less with respect to pixels in the middle. The network is trained using the Adam optimizer [42] with learning rate $\eta = 10^{-4}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. The resolution of the training images is 960×480. The network is trained for approximately 500,000 iterations, which takes around 36 hours on a NVIDIA V100 GPU.

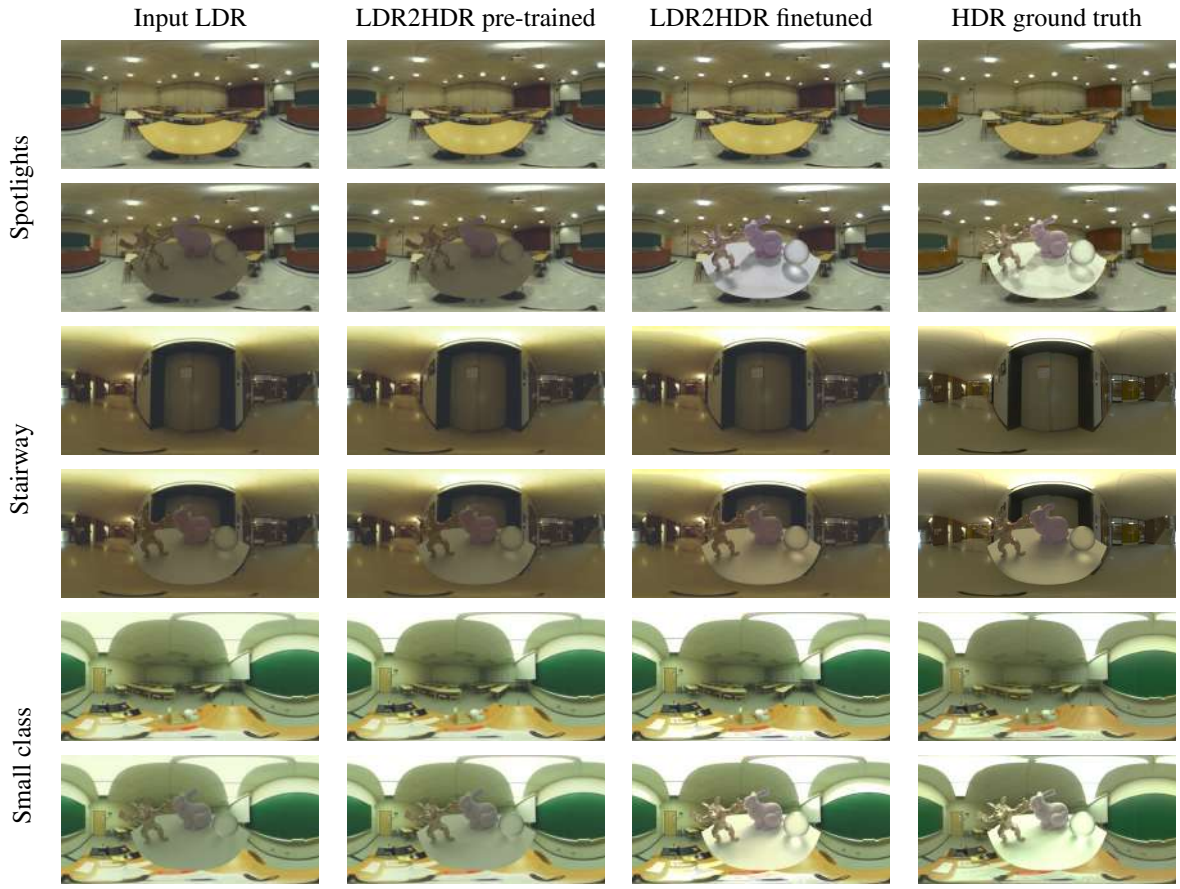


Figure 4.4 Qualitative comparison of different strategies for recovering radiance across captured scenes. For each example, the figure shows virtual test objects relit to demonstrate the dynamic range. Note that despite some color imbalance (e.g. "Spotlights"), fine-tuning helps bridge the domain gap between the training data and the captured images. Images tonemapped for display with $\gamma = 2.2$.

4.3 Evaluation

In this section, we evaluate our approach against a set of challenging real scenes, where the ground truth HDR radiance is also captured at several locations. We further establish the sensitivity of design choices and compare them to closely related techniques.

4.3.1 Test dataset

For evaluation, we capture different real indoor scenes from a variety of different environments (fig. 4.3). For each scene, we first capture a 360° video using an off-the-shelf panoramic camera as described in sec. 4.2.5. We also capture a set of HDR panoramas for evaluation to use as ground truth. To obtain these, we set the camera on a tripod at certain locations in the scene (between 3–10 locations

	w/o render loss		render loss	
	PU-PSNR↑	HDR-VDP3↑	PU-PSNR↑	HDR-VDP3↑
Hall. Net [51]	30.06	7.54	31.51	8.12
LANet [110]	32.43	8.26	38.20	9.67

Table 4.2 Quantitative comparison of two single image HDR estimation architectures on the Laval Indoor HDR test set, with and withough the rendering loss ℓ_{render} while training the network. Render Loss with LANet significantly improves the results.

per scene) and program it using exposure bracketing to capture 11 exposures spanning over 22 f-stops that are subsequently merged to HDR using the PTGui Pro commercial software. We also capture a short video with the same camera parameters as the captured video at the same location as the HDR. We then extract a single frame from that video, allowing us to have an LDR image at the exact same location as its HDR counterpart. The resulting LDR image is linearized using a pre-calibrated camera response function obtained with a Macbeth color checker. In total, we capture six different scenes containing a total number of 10 LDR/HDR ground truth panorama pairs.

As discussed in sec. 4.2.1, we rely on a small set of HDR images captured with the Theta Z1 camera for fine-tuning the LDR2HDR network. We capture 78 HDR panoramas at various locations, different from the test scenes above.

4.3.2 LDR2HDR evaluation

The LDR2HDR network presented in sec. 4.2.1 is evaluated on the test set described in sec. 4.3.1. Tab. 4.2.7 compares the performance obtained by: using the LDR images directly, the LDR2HDR network pre-trained on the Laval Indoor HDR Dataset [22], and after fine-tuning on the 78 HDR dataset captured with the same test camera. For evaluation, we use the "PU-PSNR" metric [55], which is a perceptually-uniform PSNR adjusted for HDR images. In addition, the "RMSE" corresponds to the rendering loss in eq. (4.1). Finally, we also report the HDR-VDP3 [68] metric, where a value of 10 indicates a perfect match. Here, color encoding is set as "rgb-bt.709" for HDR evaluation, assuming a 24-inch display, 1920×1080 resolution, and a viewing distance of 1 meter.

As shown in tab. 4.2.7 and illustrated qualitatively in fig. 4.4, there exists a significant domain gap between the training dataset and the test camera: simply pre-training on [22] works marginally better than the input LDR image itself, but finetuning results in a significant performance gain on all metrics. Visually, finetuning produces renderings that look very similar to the ground truth (fig. 4.4).

Figure 4.5 Comparing the dynamic range of a single RAW image (top, as used in [63]) with the output of our LDR2HDR network (bottom) at different exposures. A single RAW image is insufficient to capture the full dynamic range of typical indoor scenes.

4.3.3 PanoHDR-NeRF evaluation

To evaluate how well PanoHDR-NeRF works in terms of capturing the high dynamic range radiance field, we use the same set of ground truth HDR images as described in sec. 4.3.1. We infer environment maps at the locations of HDR panoramas and use them to render a synthetic scene.

4.3.3.1 Comparison to the NeRF++ baseline

We modified NeRF++ to work with equirectangular image representation and trained directly on the LDR frames of the input video. The rendering results of NeRF++ appear dark compared to ground truth, and the lighting is not realistic (fig. 4.7). In addition, the generated shadows are soft and faded. In contrast, PanoHDR-NeRF produces well-lit results, with sharp shadows that are similar to ground truth.

4.3.3.2 Comparison to other methods

The closest methods to our work are HDR-NeRF[35] and NeRF in the Dark [63]. First, HDR-NeRF [35] requires images captured at different exposures. Capturing a single HDR panorama spanning 22 f-stops requires the acquisition of 11 exposures, which takes approximately two minutes with our camera. In addition, we train our model on around 200 images. It would therefore take more than six hours to capture an indoor scene with this technique, a process we bring down to 2–3 minutes. Second, NeRF in the Dark [63] trains a NeRF model on RAW images. Unfortunately, as shown in fig. 4.5, RAW images do not contain a sufficiently high dynamic range to capture light sources properly. In contrast, our proposed learning-based reverse tone mapping approach creates images containing a much higher dynamic range.

	Planar		Spherical	
Datasets	PSNR↑	SSIM↑	PSNR↑	SSIM↑
Chess room	29.635	0.919	31.389	0.929
Stairway	25.381	0.892	27.381	0.891
Cafeteria	22.940	0.845	24.038	0.842
Spotlights	25.557	0.847	26.128	0.852
Dark class	29.955	0.911	31.368	0.917
Small class	29.231	0.906	30.611	0.911
Overall	27.115	0.886	28.486	0.902

Table 4.3 Quantitative comparison between planar and spherical sampling (on LDR images only) averaged over all captured scenes. Spherical sampling has better results.

4.3.4 Sensitivity analysis

We proceed to analyze the sensitivity of the method to several design choices, namely: the architecture for the LDR2HDR network, the spherical ray sampling, log-space loss, and the order of operations.

4.3.4.1 LDR2HDR network architecture

We compare between two popular single image HDR estimation architectures, namely HallucinationNet [51] and LANet [110], with and without our rendering loss. For this, we evaluate on the Laval Indoor HDR [22] test set (as opposed to our real test set from sec. 4.3.1) and report results in sec. 4.3. With its saturation-driven attention, the LANet architecture outperforms HallucinationNet. In addition, the use of the rendering loss ℓ_{rend} (eq. (4.1)) helps the network focus on the bright light sources, which is crucial for accurate radiance reproduction.

4.3.4.2 Planar vs spherical sampling

We compare the impact of planar vs spherical sampling (c.f. sec. 4.2.5) for training our network on LDR images. In both cases, we follow the hierarchical sampling strategy of NeRF [65] and train the network with 64 coarse samples and 128 fine samples. Sec. 4.3.4.2 shows that sphere sampling outperforms planar sampling for omnidirectional images as it does not oversample points at the poles but does so uniformly on the sphere.

	Linear loss		PanoHDR-NeRF		NeRF-LDR2HDR	
Dataset	PU-PSNR↑	RMSE↓	PU-PSNR↑	RMSE↓	PU-PSNR↑	RMSE↓
Chess room	35.152	0.011	35.941	0.012	35.991	0.006
Stairway	31.810	0.055	32.169	0.056	32.707	0.055
Cafeteria	24.139	4.376	27.029	4.179	26.537	5.298
Spotlights	26.719	0.909	27.657	0.431	27.324	1.619
Dark class.	28.431	1.367	29.687	1.509	29.819	0.621
Small class.	36.829	0.043	36.687	0.054	38.529	0.006
Overall	29.725	1.071	31.528	1.038	31.650	1.301

Table 4.4 Quantitative comparison between linear (left) and log (middle) losses used for training. Comparison of PanoHDR-NeRF with the NeRF done before LDR2HDR is at the right.

4.3.4.3 Loss in log space

We evaluate the importance of computing the loss in log space (c.f. sec. 4.2.5) in sec. 4.3.4.2, which suggests that the network is able to estimate the high dynamic range equally well with or without the log-space loss. However, fig. 4.6 shows that training in linear space results in additional floating artifacts and blurrier images than those obtained with the log loss.

4.3.4.4 Order of operations

PanoHDR-NeRF uses a NeRF network trained on HDR images. It is also possible to reverse the order by training the NeRF on LDR images and pass its output through the LDR2HDR network (dubbed "NeRF-LDR2HDR"). We compare these options in sec. 4.3.4.2 and fig. 4.7. Though the metrics given in the table don't differ much, the supplementary video shows PanoHDR-NeRF produces temporally more stable results. This could be due to the averaging that naturally happens within the NeRF network.

4.4 Discussion

The main contribution of this work is a novel representation of the full HDR radiance of an indoor scene. As opposed to methods that require careful scanning of a scene [15, 111], specialized hardware [12, 13] or intricately calibrated camera configurations [63, 35], our representation can be capture using a single, off-the-shelf 360° camera that is moved around the scene. PanoHDR-NeRF can render novel 360° views from any point within the scene in high dynamic range. We show their use for the realistic relighting of virtual objects in real scenes, bringing augmented reality closer to more people.

4.4.1 Limitations and future work

A limitation of our work is the blurriness of the NeRF results, despite using cone-casting and integrated positional embeddings from Mip-NeRF [4]. We hypothesize that further improvements such as [5] could help in reconstructing sharper estimates. Another limitation is that the photographer is themself unwittingly modifying the light field by moving around in it and casting shadows, creating reflections off of shiny surfaces, etc. Unfortunately, the intensity changes this creates are too soft for existing shadow detectors [102] which are trained for hard cast shadows outdoors. Here, methods modeling transient changes [58] could potentially be of help. In addition, the LDR2HDR network needs to be finetuned for the specific camera being used—the acquisition of a small finetuning dataset captured using the target camera proved critical to obtain good performance (c.f. sec. 4.3.2). Future research should investigate how to improve generalization performance, perhaps by leveraging multiple cameras for training, or through more advanced data augmentation techniques. Next, our approach learns radiance and view synthesis in two independent steps by specialized networks. Exploring how both can be done simultaneously, potentially in conjunction with geometry and material estimation [8, 115], is an exciting direction for future work.

Finally, recent efforts have demonstrated how to significantly shorten training [67, 109] and inference [79, 20] times of NeRF-based approaches, which can be incorporated into PanoHDR-NeRF. By reducing the time between capture and visualization, PanoHDR-NeRF can be used for AR/VR applications such as virtual tours and VFX generation.

4.5 Conclusion

In this chapter, we present PanoHDR-NeRF, a neural representation of the high dynamic range (HDR) radiance field of an indoor scene that can be captured casually without elaborate setups or complex capture protocols. First, a user captures a low dynamic range (LDR) omnidirectional video of the scene by freely waving an off-the-shelf camera around the scene. Then, an LDR2HDR network converts the captured LDR frames to HDR, which are subsequently used to train a modified NeRF++ model. The resulting PanoHDR-NeRF representation can synthesize full HDR panoramas from at any location in the scene. Through experiments on a novel test dataset of a variety of real scenes with the ground truth HDR radiance captured at locations not seen during training, we show that PanoHDR-NeRF accurately predicts the radiance compared to the ground truth, and in greater accuracy than what was previously possible. We also show that the HDR images produced by PanoHDR-NeRF can synthesize correct lighting effects, enabling the augmentation of indoor scenes with synthetic objects that are lit correctly.

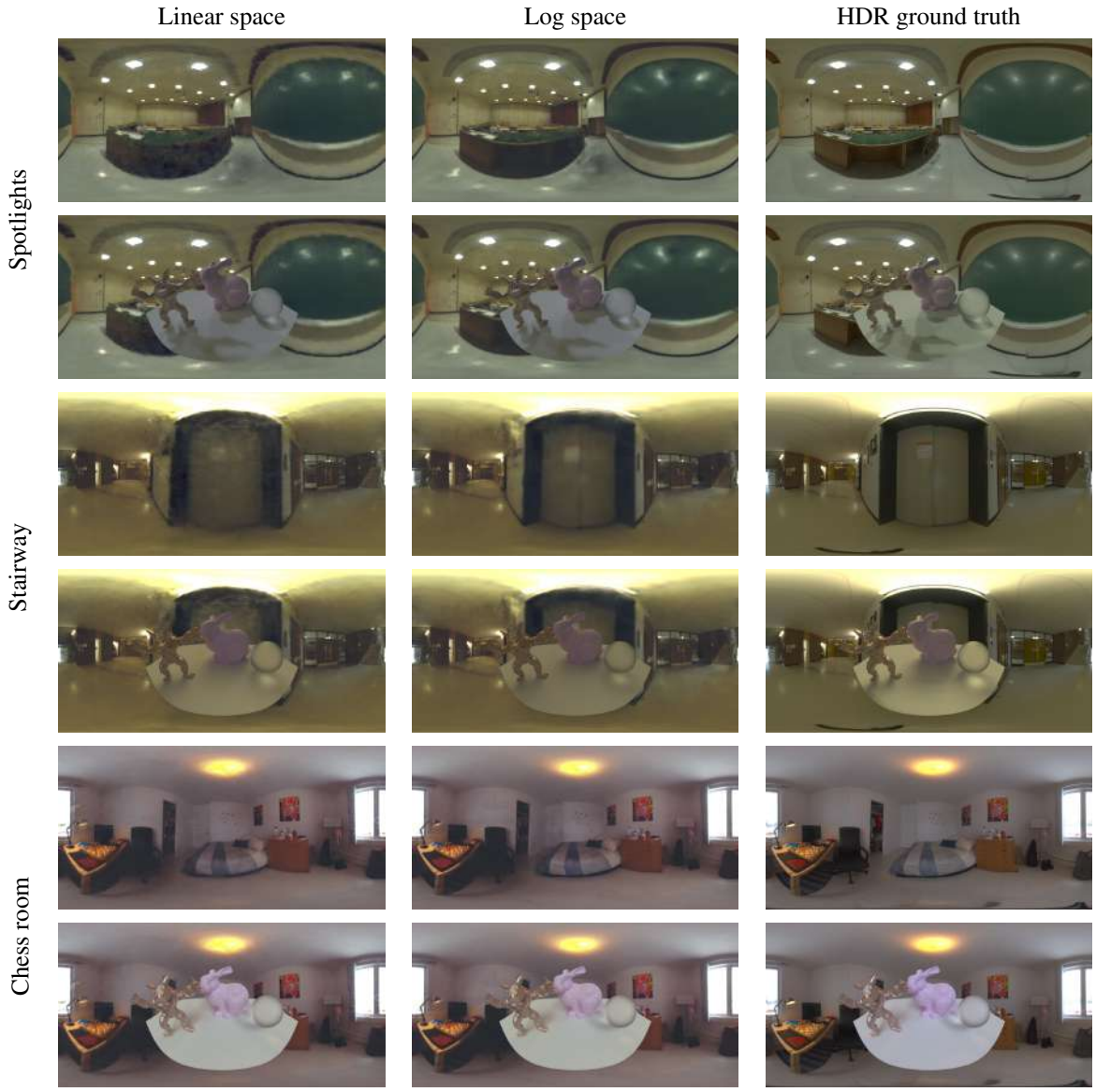
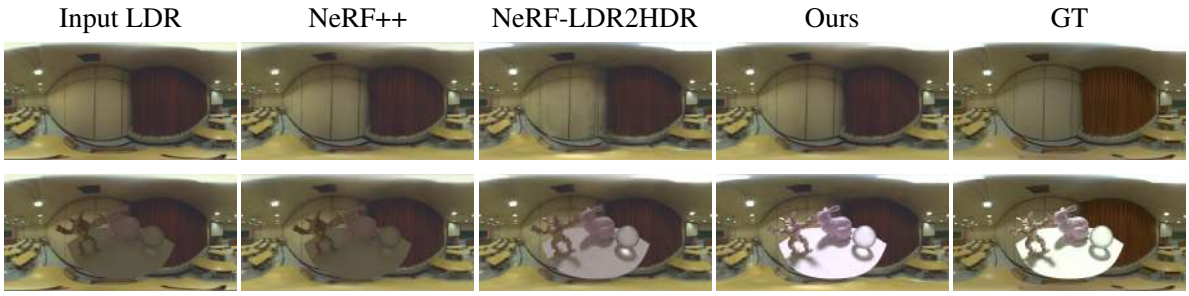
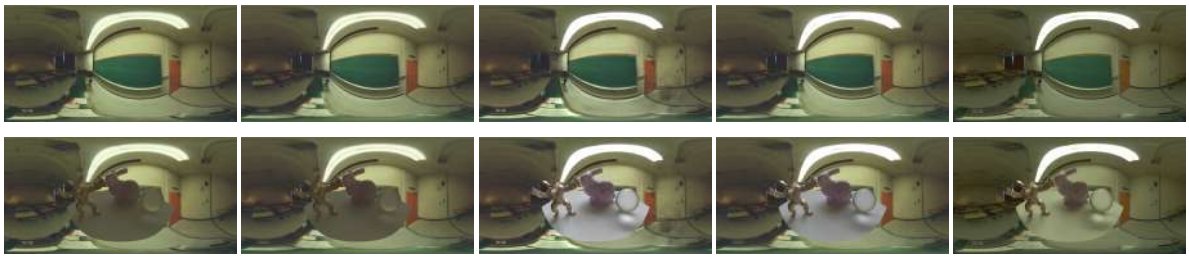


Figure 4.6 Comparing panoramas generated by PanoHDR-NeRF after loss in linear space and log space. For each example, the figure shows (top) the panorama with (bottom) virtual test objects relit to show the dynamic range. While the network learns the high dynamic range in both cases, we observe that taking loss in linear space leads to poor visual quality and floating artifacts in the output panoramas. PanoHDR-NeRF produces better results when trained in log space, consistent with [110, 82, 46].



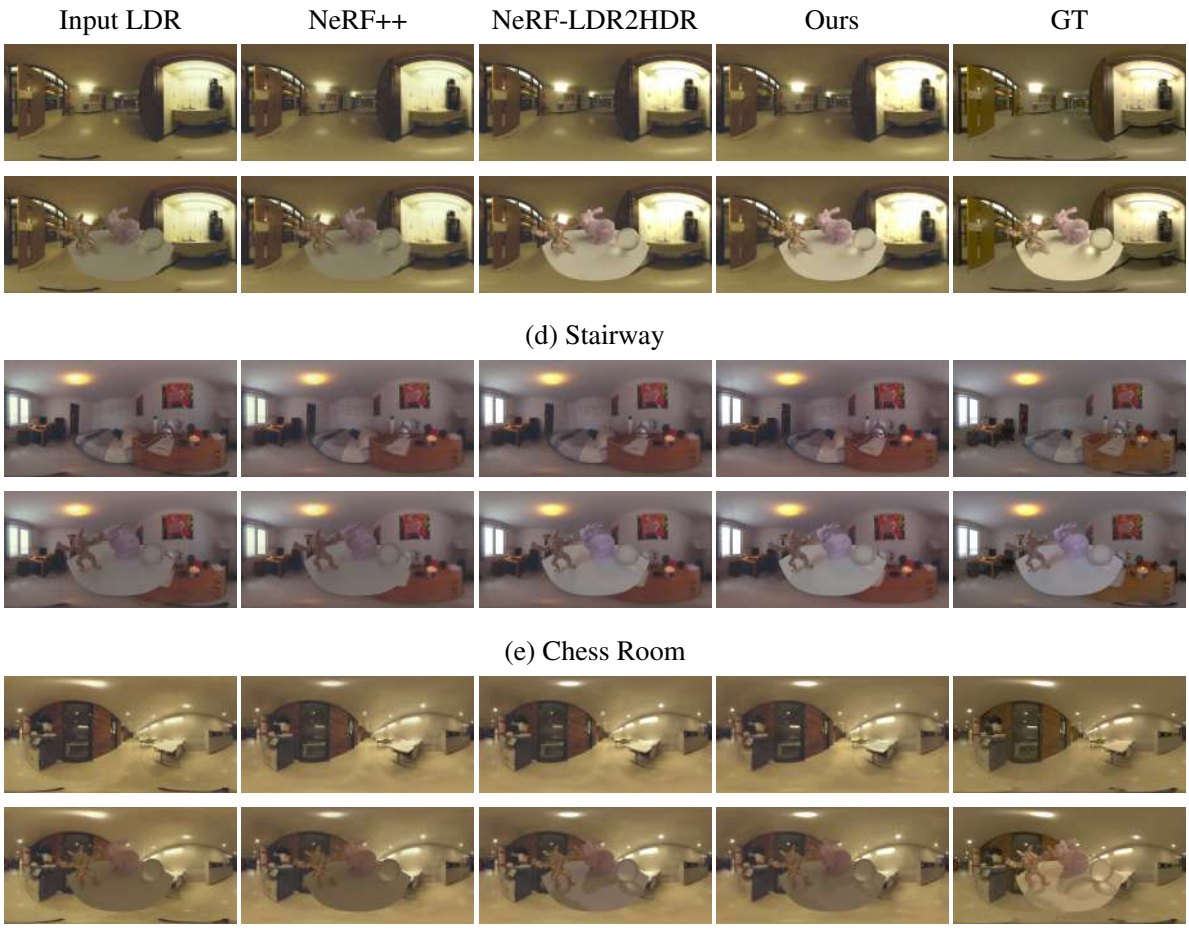
(a) Spotlights



(b) Dark Class

(b) Small Class

Figure 4.7 Comparing input LDR, NeRF++, NeRF-LDR2HDR, PanoHDR-NeRF (ours), and GT panoramas. For each example, the figure shows a virtual test object relit to show the dynamic range. While NeRF++ is able to model the scene correctly, it is unable to capture the radiance of the scene. PanoHDR-NeRF is able to faithfully capture the radiance of the scene. We compare it with NeRF-LDR2HDR which estimates HDR from NeRF++ outputs. Although it is able to closely estimate the radiance, it leads to flickering between consecutive frames. Images tonemapped for display with $\gamma = 2.2$.



(d) Stairway

(e) Chess Room

(f) Cafeteria

Figure 4.8 Comparing input LDR, NeRF++, NeRF-LDR2HDR, PanoHDR-NeRF (ours), and GT panoramas. For each example, the figure shows a virtual test object relit to show the dynamic range. Continued example from fig. 4.7. Images tonemapped for display with $\gamma = 2.2$.

Chapter 5

Conclusions and Future Work

In this thesis, we study how to produce novel views of a casually captured scene and modify them in interesting ways for augmented/virtual (AR/VR) applications.

We first present a neural rendering framework for simultaneous novel view synthesis and appearance editing of a causally captured scene from off-the-shelf smartphone cameras under known illumination. Our method explicitly disentangles appearance from lighting while estimating radiance and learns an independent lighting estimation of the scene from casually captured images. This enables us to generalize arbitrary changes in the scene's materials while performing novel view synthesis. Our appearance changes are consistent with the topology of the underlying geometry.

The neural rendering framework proposed however does not work with specular objects. We would like to explore other basis representations which would enable us to handle glossy or even nearly-specular material edits. Further, we would like to incorporate relighting while retaining the ability to perform appearance editing and view synthesis.

Next, we present PanoHDR-NeRF, a neural representation of an indoor scene's high dynamic range (HDR) radiance that can be captured casually without elaborate setups or complex capture protocols. Given a low dynamic range (LDR) omnidirectional video of the scene captured freely waving an off-the-shelf camera around the scene, our method can synthesize full HDR panoramas from any location in the scene. The generated HDR panoramas can synthesize correct lighting effects, enabling the augmentation of indoor scenes with synthetic objects that are lit correctly.

View synthesis results from PanoHDR-NeRF can be improved by using similar ideas to [95] as we encode a large unbounded indoor scene. The photographer modifies the light field by moving around, and modeling the transient changes separately should help. Finally, we want to remove the camera finetuning step for the LDR2HDR module and generalize it for any sensor.

Related Publications

- **Pulkit Gera**, Aakash K.T, Dhawal Sirikonda, P.J.Narayanan. *Neural View Synthesis with Appearance Editing from Unstructured Images*. Twelfth Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP '21). Article 40, 1–9.https://doi.org/10.1145/3490035.3490299
- **Pulkit Gera**, Aakash KT, Dhawal Sirikonda, Parikshit Sakurikar, and P. J.Narayanan. 2021. *Appearance Editing with Free-viewpoint Neural Rendering*.arXiv:2110.07674 (2021)
- **Pulkit Gera**, Mohammed Reza Karimi, Charles Renaud, P.J.Narayanan, Jean-François Lalonde. *Casual Radiance Capture and Synthesis for Indoor Scenes*. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops 2022.
- **Pulkit Gera**, Mohammed Reza Karimi, Charles Renaud, P.J.Narayanan, Jean-François Lalonde. *Casual Indoor HDR Radiance Capture from Omnidirectional Images*. ACM SIGGRAPH Asia Conference 2022 **(under review)**.

Bibliography

[1] A. F. Agarap. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018. 28

[2] B. Attal, S. Ling, A. Gokaslan, C. Richardt, and J. Tompkin. Matryodshka: Real-time 6dof video view synthesis using multi-sphere images. In *Eur. Conf. Comput. Vis.*, 2020. x, 9, 15, 16

[3] F. Banterle, A. Artusi, K. Debattista, and A. Chalmers. *Advanced High Dynamic Range Imaging, Second Edition*. 2017. 11

[4] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *IEEE/CVF Int. Conf. Comput. Vis.*, 2021. 16, 43, 44, 50

[5] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2022. 16, 43, 50

[6] A. Ben-Artzi, R. Overbeck, and R. Ramamoorthi. Real-time brdf editing in complex lighting. *ACM Trans. Graph.*, 2006. 10, 25

[7] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart. Vision based mav navigation in unknown and unstructured environments. In *2010 IEEE International Conference on Robotics and Automation*, 2010. 9

[8] M. Boss, R. Braun, V. Jampani, J. T. Barron, C. Liu, and H. P. A. Lensch. Nerd: Neural reflectance decomposition from image collections. *IEEE/CVF Int. Conf. Comput. Vis.*, 2021. 17, 50

[9] H. Chen, B. He, H. Wang, Y. Ren, S.-N. Lim, and A. Shrivastava. Nerv: Neural representations for videos. In *Adv. Neural Inform. Process. Syst.*, 2021. 17

[10] Z. Chen, A. Chen, G. Zhang, C. Wang, Y. Ji, K. N. Kutulakos, and J. Yu. A neural rendering framework for free-viewpoint relighting. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2020. 16, 25

[11] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Conf. Emp. Meth. Nat. Lang. Proc.*, 2014. 14

[12] P. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Conf. Comp. Graph. Int. Tech.*, SIGGRAPH, pages 189–198, 1998. 11, 49

[13] P. Debevec, P. Graham, J. Busch, and M. Bolas. A single-shot light probe. In *ACM SIGGRAPH Talks*, pages 10:1–10:1, New York, NY, USA, 2012. ACM. 49

[14] P. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *ACM SIGGRAPH*, pages 369–378, 1997. 11, 19

[15] P. Debevec, Y. Yu, and G. Boshokov. Efficient view-dependent ibr with projective texture-mapping. In *EG Rend. Works.*, 1998. 11, 14, 49

[16] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. *IEEE/CVF Int. Conf. Comput. Vis.*, 2015. 19

[17] Y. Endo, Y. Kanamori, and J. Mitani. Deep reverse tone mapping. *ACM Trans. Graph.*, 2017. 21

[18] J. Flynn, M. Broxton, P. E. Debevec, M. DuVall, G. Fyffe, R. S. Overbeck, N. Snavely, and R. Tucker. Deepview: View synthesis with learned gradient descent. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2019. 14

[19] D. Gao, G. Chen, Y. Dong, P. Peers, K. Xu, and X. Tong. Deferred neural lighting: free-viewpoint relighting from unstructured photographs. *ACM Trans. Graph.*, December 2020. 16, 27, 28, 37

[20] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *IEEE/CVF Int. Conf. Comput. Vis.*, 2021. 16, 50

[21] M.-A. Gardner, Y. Hold-Geoffroy, K. Sunkavalli, C. Gagné, and J.-F. Lalonde. Deep parametric indoor lighting estimation. *IEEE/CVF Int. Conf. Comput. Vis.*, 2019. 22

[22] M.-A. Gardner, K. Sunkavalli, E. Yumer, X. Shen, E. Gambaretto, C. Gagné, and J.-F. Lalonde. Learning to predict indoor illumination from a single image. *ACM Trans. Graph.*, 2017. 42, 46, 48

[23] V. gay bellile, A. Bartoli, K. Hamrouni, P. Sayd, S. Bourgeois, and A. Belhedi. Noise modelling in time-of-flight sensors with application to depth noise removal and uncertainty estimation in three-dimensional measurement. *IET Computer Vision*, 2015. 2

[24] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. ACM, 1996. 14

[25] K. Gu, T. Maugey, S. Knorr, and C. Guillemot. Omni-nerf: neural radiance field from 360° image captures. In *Int. Conf. Multi. Expo*, 2022. 17

[26] P. Hansen, P. Corke, W. Boles, and K. Daniilidis. Scale invariant feature matching with wide angle images. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007. 9

[27] T. Hara and T. Harada. Enhancement of novel view synthesis using omnidirectional image completion. *ArXiv*, abs/2203.09957, 2022. 9

[28] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2017. 44

[29] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2016. 27

[30] P. Hedman, J. Philip, T. Price, J. Frahm, G. Drettakis, and G. J. Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph.*, 2018. 14

[31] P. Hedman, J. Philip, T. Price, J.-M. Frahm, G. Drettakis, and G. Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Trans. Graph.*, 2018. x, 5, 6

[32] P. Hedman, T. Ritschel, G. Drettakis, and G. J. Brostow. Scalable inside-out image-based rendering. *ACM Trans. Graph.*, 2016. 14

[33] C.-Y. Hsu, C. Sun, and H.-T. Chen. Moving in a 360 world: Synthesizing panoramic parallaxes from a single panorama. *ArXiv*, 2021. 9, 17

[34] J. Huang, Z. Chen, D. Ceylan, and H. Jin. 6-DOF VR videos with a single 360-camera. In *IEEE Virt. Real.*, 2017. 9, 16

[35] X. Huang, Q. Zhang, F. Ying, H. Li, X. Wang, and Q. Wang. Hdr-nerf: High dynamic range neural radiance fields. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2022. x, 17, 18, 47, 49

[36] W. Jarosz. *Efficient Monte Carlo Methods for Light Transport in Scattering Media*. PhD thesis, UC San Diego, September 2008. 25, 27

[37] Y. Jeong, S. Ahn, C. B. Choy, A. Anandkumar, M. Cho, and J. Park. Self-calibrating neural radiance fields. *IEEE/CVF Int. Conf. Comput. Vis.*, 2021. 16

[38] J. T. Kajiya. The rendering equation. *SIGGRAPH Comput. Graph.*, 1986. 25

[39] J. T. Kajiya and B. P. Von Herzen. Ray tracing volume densities. In *Annual Conference on Computer Graphics and Interactive Techniques*, 1984. 9

[40] N. K. Kalantari and R. Ramamoorthi. Deep high dynamic range imaging of dynamic scenes. *ACM Trans. Graph.*, 2017. 19

[41] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. Eurographics Association, 2006. 27

[42] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015. 28, 42, 44

[43] Y. Kinoshita and H. Kiya. itm-net: Deep inverse tone mapping using novel loss function considering tone mapping operator. *IEEE Access*, 2019. 21

[44] L. Lechlek, D. Meneveaux, M. Ribardière, R. Perrot, and M. C. Babahenini. Interactive hdr image-based rendering from unstructured ldr photographs. *Computers Graphics*, 2019. 22

[45] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 2000. 2

[46] J. Li and P. Fang. Hdrnet: Single-image-based hdr reconstruction using channel attention cnn. In *Int. Conf. Mult. Sign. Proc.*, 2019. xi, xii, 20, 21, 51

[47] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE/CVF Int. Conf. Comput. Vis.*, 2021. 16

[48] K.-E. Lin, Z. Xu, B. Mildenhall, P. P. Srinivasan, Y. Hold-Geoffroy, S. DiVerdi, Q. Sun, K. Sunkavalli, and R. Ramamoorthi. Deep multi depth panoramas for view synthesis. In *Eur. Conf. Comput. Vis.*, 2020. 9, 16

[49] Y.-C. Lin, P. R. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin. inerf: Inverting neural radiance fields for pose estimation. *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021. 16

[50] S. Liu and W. Deng. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, 2015. 28

[51] Y.-L. Liu, W.-S. Lai, Y.-S. Chen, Y.-L. Kao, M.-H. Yang, Y.-Y. Chuang, and J.-B. Huang. Single-image hdr reconstruction by learning to reverse the camera pipeline. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2020. xi, 20, 21, 46, 48

[52] Z. Liu, W. Lin, X. Li, Q. Rao, T. Jiang, M. Han, H. Fan, J. Sun, and S. Liu. Adnet: Attention-guided deformable convolutional network for high dynamic range imaging. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2021. 19

[53] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 2019. x, 6, 7

[54] Y. Luo and M. L. Gavrilova. 3d building reconstruction from lidar data. In M. Gavrilova, O. Gervasi, V. Kumar, C. J. K. Tan, D. Taniar, A. Laganá, Y. Mun, and H. Choo, editors, *Computational Science and Its Applications - ICCSA 2006*. Springer Berlin Heidelberg, 2006. 1

[55] R. K. Mantiuk and M. Azimi. Pu21: A novel perceptually uniform encoding for adapting existing quality metrics for hdr. *Pict. Coding Symp.*, 2021. 46

[56] Mapillary. Opensfm. https://github.com/mapillary/OpenSfM, 2022. xii, 41, 44

[57] D. Marnerides, T. Bashford-Rogers, J. Hatchett, and K. Debattista. Expandnet: A deep convolutional neural network for high dynamic range expansion from low dynamic range content. *Comput. Graph. Forum (CGF)*, 2018. 21, 22

[58] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2021. 50

[59] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2021. x, 17, 18

[60] J. Masci, D. Migliore, M. M. Bronstein, and J. Schmidhuber. *Descriptor Learning for Omnidirectional Image Matching*. 2014. 9

[61] L. M. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2019. x, 3, 4

[62] M. Meshry, D. B. Goldman, S. Khamis, H. Hoppe, R. Pandey, N. Snavely, and R. Martin-Brualla. Neural rerendering in the wild. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2019. x, 5, 6, 14

[63] B. Mildenhall, P. Hedman, R. Martin-Brualla, P. P. Srinivasan, and J. T. Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2022. xii, 17, 47, 49

[64] B. Mildenhall, P. P. Srinivasan, R. O. Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion. *ACM Trans. Graph.*, 2019. x, 14, 15

[65] 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 *Eur. Conf. Comput. Vis.*, 2020. x, 7, 8, 16, 48

[66] K. Moriwaki, R. Yoshihashi, R. Kawakami, S. You, and T. Naemura. Hybrid loss for learning single-image-based hdr reconstruction. *ArXiv*, abs/1812.07134, 2018. 21

[67] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 2022. 17, 50

[68] M. Narwaria, R. K. Mantiuk, M. P. D. Silva, and P. L. Callet. Hdr-vdp-2.2: a calibrated method for objective quality prediction of high-dynamic range and standard images. *Journ. Elect. Imag.*, 2015. 46

[69] M. Niemeyer, L. M. Mescheder, M. Oechsle, and A. Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2020. 7

[70] M. Nimier-David, D. Vicini, T. Zeltner, and W. Jakob. Mitsuba 2: A retargetable forward and inverse renderer. *ACM Trans. Graph.*, 2019. 28, 34

[71] Y. Niu, J. Wu, W. Liu, W. Guo, and R. W. H. Lau. Hdr-gan: Hdr image reconstruction from multi-exposed ldr images with large motions. *IEEE Transactions on Image Processing*, 2021. 19

[72] J. J. Park, P. R. Florence, J. Straub, R. A. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2019. 4

[73] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Adv. Neural Inform. Process. Syst.* 2019. 27

[74] M. Pollefeys and L. V. Gool. From images to 3d models. *Commun. ACM*, 2002. 2

[75] K. Prabhakar, R. Arora, A. Swaminathan, K. P. Singh, and R. V. Babu. A fast, scalable, and reliable deghosting method for extreme exposure fusion. *2019 IEEE International Conference on Computational Photography (ICCP)*, 2019. 19

[76] K. Prabhakar, V. S. Srikar, and R. V. Babu. Deepfuse: A deep unsupervised approach for exposure fusion with extreme exposure image pairs. *IEEE/CVF Int. Conf. Comput. Vis.*, 2017. 19

[77] X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. Zaiane, and M. Jagersand. U2-net: Going deeper with nested u-structure for salient object detection. 2020. 27

[78] R. Ramamoorthi. *Precomputation-Based Rendering*. NOW Publishers Inc, 2009. 25

[79] C. Reiser, S. Peng, Y. Liao, and A. Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *IEEE/CVF Int. Conf. Comput. Vis.*, 2021. x, 16, 18, 50

[80] G. Riegler and V. Koltun. Free view synthesis. In *Eur. Conf. Comput. Vis.*, 2020. 5, 14

[81] G. Riegler and V. Koltun. Stable view synthesis. *CVPR*, 2021. x, 5, 15, 16

[82] M. S. Santos, I. R. Tsang, and N. K. Kalantari. Single image hdr reconstruction using a cnn with masked features and perceptual loss. *ACM Trans. Graph.*, 2020. xii, 21, 51

[83] Sara Fridovich-Keil and Alex Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2022. 7, 17

[84] D. Scaramuzza. Omnidirectional vision: From calibration to root motion estimation. 2007. 9

[85] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2016. x, 2, 27

[86] J. L. Schönberger, E. Zheng, J.-M. Frahm, and M. Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *Eur. Conf. Comput. Vis.*, 2016. x, 2

[87] A. Serrano, I. Kim, Z. Chen, S. DiVerdi, D. Gutierrez, A. Hertzmann, and B. Masia. Motion parallax for 360° rgbd video. *IEEE Trans. Vis. Comput. Graph.*, 2019. 16

[88] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2019. 7

[89] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2019. 14

[90] V. Sitzmann, M. Zollhöfer, and G. Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019. 7

[91] P.-P. Sloan, J. Kautz, and J. Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Trans. Graph.*, 2002. 10, 25, 27, 42

[92] M. P. B. Slomp, M. M. de Oliveira Neto, and D. I. Patrício. A gentle introduction to precomputed radiance transfer. *RITA*, 2006. 10

[93] N. Snavely, S. M. Seitz, and R. Szeliski. Modeling the world from internet photo collections. *Int. J. Comput. Vis.*, 80:189–210, 2007. 2

[94] S. Song and T. A. Funkhouser. Neural illumination: Lighting prediction for indoor environments. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2019. 21

[95] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2022. 16, 54

[96] J. Tarko, J. Tompkin, and C. Richardt. Real-time virtual object insertion for moving 360° videos. In *Int. Conf. Virt. Real. Cont. App. Indus.*, 2019. xi, 22

[97] J. Thies, M. Zollhöfer, and M. Nießner. Deferred neural rendering. *ACM Trans. Graph.*, 2019. x, 5, 14, 15, 16, 23, 25, 26, 27, 29, 30, 33

[98] J. Thies, M. Zollhöfer, C. Theobalt, M. Stamminger, and M. Nießner. Image-guided neural object rendering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 5

[99] H. Turki, D. Ramanan, and M. Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2022. 16

[100] D. R. Walton and A. Steed. Dynamic hdr environment capture for mixed reality. *ACM Symp. Virt. Real. Soft. Tech.*, 2018. 22

[101] L. Wang and K.-J. Yoon. Deep learning for hdr imaging: State-of-the-art and future trends. *IEEE transactions on pattern analysis and machine intelligence*, 2021. x, 11

[102] T. Wang, X. Hu, Q. Wang, P. Heng, and C. Fu. Instance shadow detection. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2020. 50

[103] S. Wu, J. Xu, Y.-W. Tai, and C.-K. Tang. Deep high dynamic range imaging with large foreground motions. In *Eur. Conf. Comput. Vis.*, 2018. 19

[104] H. Xu, J. Ma, Z. Le, J. Jiang, and X. Guo. Fusiondn: A unified densely connected network for image fusion. In *AAAI*, 2020. 19

[105] J. Xu, J. Zheng, Y. Xu, R. Tang, and S. Gao. Layout-guided novel view synthesis from a single indoor panorama. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2021. 9, 16

[106] Z. Xu, S. Bi, K. Sunkavalli, S. Hadap, H. Su, and R. Ramamoorthi. Deep view synthesis from sparse photometric images. *ACM Trans. Graph.*, 2019. 5

[107] Q. Yan, D. Gong, Q. Shi, A. van den Hengel, C. Shen, I. D. Reid, and Y. Zhang. Attention-guided network for ghost-free high dynamic range imaging. *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, pages 1751–1760, 2019. 19

[108] B. Yang, Y. Zhang, Y. Li, Z. Cui, S. Fanello, H. Bao, and G. Zhang. Neural rendering in a room: Amodal 3d understanding and free-viewpoint rendering for the closed scene composed of pre-captured objects. *ACM Trans. Graph.*, 2022. 22

[109] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *IEEE/CVF Int. Conf. Comput. Vis.*, 2021. 17, 50

[110] H. Yu, W. Liu, C. Long, B. Dong, Q. Zou, and C. Xiao. Luminance attentive networks for hdr image and panorama reconstruction. *Comput. Graph. Forum (CGF)*, 2021. xi, xii, 20, 21, 22, 42, 46, 48, 51

[111] E. Zhang, M. F. Cohen, and B. Curless. Emptying, refurnishing, and relighting indoor spaces. *ACM Trans. Graph.*, 2016. 22, 49

[112] J. Zhang and J.-F. Lalonde. Learning high dynamic range from outdoor panoramas. In *IEEE/CVF Int. Conf. Comput. Vis.*, 2017. 21, 42

[113] K. Zhang, F. Luan, Q. Wang, K. Bala, and N. Snavely. Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2021. x, 17, 18, 28, 29, 30

[114] K. Zhang, G. Riegler, N. Snavely, and V. Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv:2010.07492*, 2020. xii, 8, 16, 40, 43

[115] X. Zhang, P. P. Srinivasan, B. Deng, P. Debevec, W. T. Freeman, and J. T. Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Trans. Graph.*, 2021. 17, 50

[116] Z. Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 2012. 2

[117] Z. Zhang, Q. Liu, and Y. Wang. Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters*, 2018. 27

[118] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely. Stereo magnification. *ACM Trans. Graph.*, 2018. 7

[119] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Trans. Graph.*, 2018. x, 6, 14