Incorporating dense depth into neural 3D representations for view synthesis and relighting

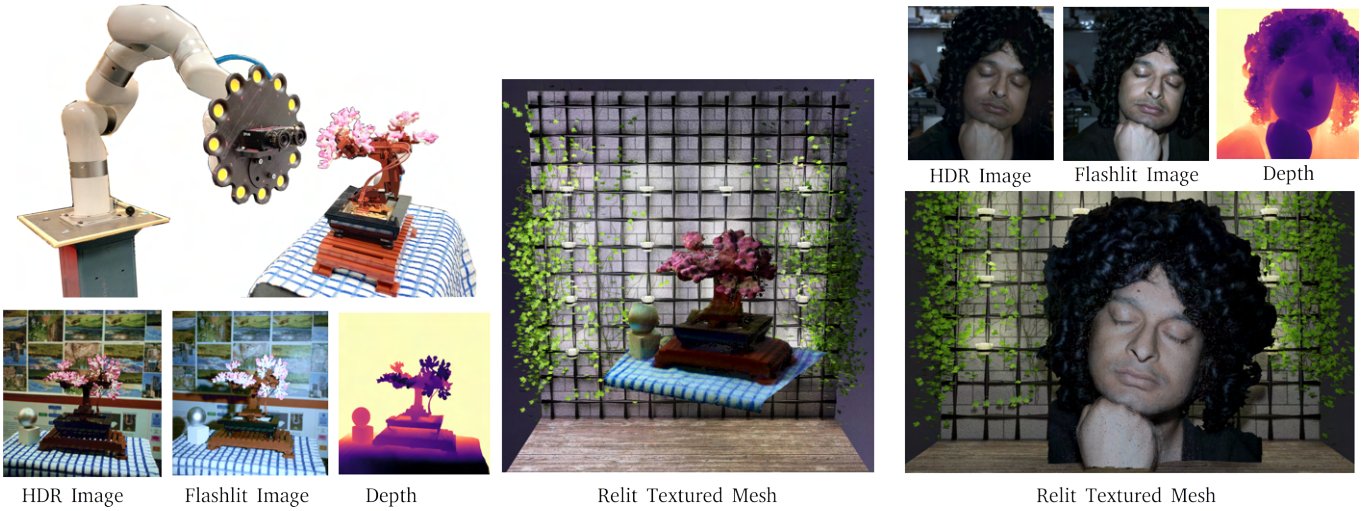


Fig. 1. We present an approach for the photo-realistic capture of small scenes by incorporating dense metric depth, multi-view, and multi-illumination images into neural 3D scene understanding pipelines. We use a robot mounted multi-flash stereo camera system, developed in-house, to capture the necessary supervision signals needed to optimize our representation with a few input views. The reconstruction of the LEGO plant and the face were generated with 11 and 2 stereo pairs respectively. We relight the textured meshes using [Blender 2024]. Background design by [Ginibird 2021].

Synthesizing accurate geometry and photo-realistic appearance of small scenes is an active area of research with compelling use cases in gaming, content creation, virtual reality, and robotics. When applying scene geometry and appearance estimation techniques to robotics, we found that the narrow cone of possible viewpoints due to the limited range of robot motion and scene clutter caused current estimation techniques to produce poor quality estimates or even fail. On the other hand, in robotic applications, dense metric depth can often be measured directly using stereo and illumination can be controlled. Depth can provide a good initial estimate of the object geometry to improve reconstruction. In this work we demonstrate a method to incorporate dense metric depth into the training of neural 3D representations, address an artifact observed while jointly refining geometry and appearance by disambiguating between texture and geometry edges, and propose a new pipeline which accelerates training of neural fields by using dense metric depth. We also discuss a multi-flash stereo camera system developed in-house to capture the necessary data for our pipeline and show results on relighting and view synthesis with a few training views.

1 INTRODUCTION

Capturing photo realistic appearance and geometry of scenes is a fundamental problem in computer vision and graphics with a set of mature tools and solutions for content creation [Boudoin 2023; LumaLabs 2023], large scale scene mapping [3DZephyr 2022], augmented reality and cinematography [AliceVision 2022; RealityCapture 2022; Tancik et al. 2023]. Enthusiast level 3D photogrammetry, especially for small or tabletop scenes, has been supercharged by more capable smartphone cameras and new toolboxes like RealityCapture and NeRFStudio. A subset of these solutions are geared towards view synthesis where the focus is on photo-realistic view interpolation rather than recovery of accurate scene geometry. These solutions take the "shape-radiance ambiguity"[Kutulakos and Seitz 1999] into stride by decoupling the scene transmissivity (related to geometry) from the scene appearance prediction. But without of diverse training views, several neural scene representations (e.g. [Fridovich-Keil et al. 2022; Mildenhall et al. 2021; Müller et al. 2022]) are prone to poor shape reconstructions while estimating accurate appearance.

By only reasoning about appearance as cumulative radiance weighted with the scene's transmissivity, one can achieve convincing view interpolation results, with the quality of estimated scene geometry improving with the diversity and number of training views. However, capturing a diverse set of views, especially for small scenes, often becomes challenging due to the scenes' arrangement.

Without of metric depth measurements, researchers have used sparse depth from structure from motion [Kang et al. 2021; Roessle et al. 2022], and dense monocular depth priors [Yu et al. 2022] to improve reconstruction, with a focus on appearance. Assimilating dense non-metric depth (e.g. [Cheng et al. 2019; Eftekhar et al. 2021]) is often challenging due to the presence of an unknown affine degree of freedom which needs to be estimated across many views.

However, without diversity of viewpoints, measuring the geometry directly is often useful. Several hardware solutions for digitizing objects exist, ranging from consumer level 3D scanners (e.g. [Shining3D 2023]), and room scale metrology devices ([Ens 2023; Pho 2023; Matterport 2023]) to high precision hand held 3D scanners

Author's address:

(e.g. [Art 2023]). Although these systems measure geometry very accurately, they interpret appearance as diffuse reflectance and often fall short in modelling view-dependent appearance.

Despite the known effectiveness of incorporating depth and widespread availability of dense metric depth sensors in smartphone cameras ([ZDNet 2023; Zhang et al. 2020]) and as standalone devices ([azu 2023; Keselman et al. 2017], incorporation of dense metric depth into neural 3D scene understanding is underexplored.

In this work:

(1) we present a method to incorporate dense metric depth into the training of neural 3D fields, enabling state-of-the-art methods to use dense metric depth with minor changes.
(2) We investigate an artifact (Fig. 3) commonly observed while jointly refining shape and appearance. We identify its cause as existing methods' inability to differentiate between depth and texture discontinuities. We address it by using depth edges as an additional supervision signal.
(3) We present a new approach to accelerate training of neural fields for photo-realistic scene capture and relighting from multi-view and multi-illumination images by incorporating metric depth.

We demonstrate our ideas using a robot mounted multi-flash stereo camera rig developed in-house. This device allows us to capture a diverse range of scenes with varying complexity in both appearance and geometry. Using the captured data, we demonstrate results in reconstruction, view interpolation, geometry capture, and relighting with a few views. We hope that our full-stack solution comprising of the camera system and algorithms will serve as a test bench for automatically capturing small scenes in the future. Additional results may be viewed at https://stereomfc.github.io

2 RELATED WORK

View synthesis and reconstruction of shapes from multiple 3D measurements is an important problem in computer vision with highly efficient and general solutions like volumetric fusion ([Curless and Levoy 1996]), screened Poisson surface reconstruction ([Kazhdan and Hoppe 2013]), patch based dense stereopsis ([Galliani et al. 2015]) and joint refinement of surface and appearance [Dai et al. 2017]. While these continue to serve as robust foundations, they fall short in capturing view-dependent appearance. Additionally, even with arbitrary levels of discretization, they often oversmooth texture and surfaces due to data association relying on weighted averages along the object surface.

Recent neural 3D scene understanding approaches (e.g. [Li et al. 2023; Wang et al. 2021; Yariv et al. 2021]) have avoided this by adopting a continuous implicit volumetric representation to serve as the geometric and appearance back-end of the view synthesizer. Together with continuous models, reasoning about appearance as radiance, and high frequency preserving embeddings[Tancik et al. 2020], these approaches serve as highly capable view interpolators by reliably preserving view dependent appearance and minute geometric details. More recent work has included additional geometric priors in the form of monocular depth supervision ([Yu et al. 2022]), sparse depth supervision from structure-from-motion toolboxes ([Sun et al. 2022]), dense depth maps ([Azinović et al. 2022; Sandström et al. 2023]), patch based multi-view consistency [Fu et al. 2022], and multi-view photometric consistency under assumed surface reflectance functions ([Guizilini et al. 2023]). Our work builds on the insights from using dense depth supervision to improve scene understanding with only a few training views available.

Novel hardware is often used for collecting supervision signals in addition to color images to aid 3D scene understanding. [Attal et al. 2021] demonstrate a method to incorporate a time-of-flight sensor. [Shandilya et al. 2023] demonstrate a method to extract geometric and radiometric cues from scenes captured with a commercial RGBD sensor and improve view synthesis with a few views. Event based sensors have also been used to understand poorly lit scenes with fast moving cameras ([Klenk et al. 2023; Low and Lee 2023]). Researchers have also combined illumination sources with cameras to capture photometric and geometric cues for dense 3D reconstruction of scenes with known reflectances ([Chaudhury et al. 2024; Gotardo et al. 2015]). Similarly, [Cheng et al. 2023; Schmitt et al. 2023, 2020] capture geometry and reflectance of objects by refining multi-view color, depth and multi-illumination images. Given the recent advances in stereo matching ([Xu et al. 2022]) we use a stereo camera to collect data for view synthesis to disambiguate between shape and appearance at capture.

Pairing illumination sources with imaging can improve reasoning about the appearance in terms of surface reflectance parameters. [Kang et al. 2019; Schmitt et al. 2023; Zhou et al. 2013] approaches the problem of material capture using a variety of neural and classical techniques. [Bi et al. 2020; Cheng et al. 2023; Kang et al. 2021; Zhang et al. 2022] leverage recent neural scene understanding techniques to jointly learn shape and appearance as reflectance of the scene. Our work also pairs illumination sources with stereo cameras to capture multi-illumination images from the scene and we build on modern neural techniques for view synthesis and relighting the scene.

3 METHODS

We describe our method of incorporating dense metric depth in Section 3.2 which enables a variety of neural 3D representations to use it. In Section 3.3 we jointly optimize shape and appearance of a scene using information about scene depth edges. In Section 3.4, we propose a new method of neural view synthesis: AdaShell^{++}that accelerates training by incorporating dense depth. In Section 3.5 we discuss metric depth augmented versions of prior works.

3.1 Primer on neural 3D fields

Our scene representation consists of two neural networks – an intrinsic network $\mathcal{N}(\theta)$ and an appearance network $\mathcal{A}(\phi)$ which are jointly optimized to capture the shape and appearance of the object. $\mathcal{N}(\theta)$, an a multi-layer perceptron (MLP) with parameters (θ) with multi-level hash grid encoding ([Li et al. 2023]), is trained to approximate the intrinsic properties of the scene – the scene geometry as a neural signed distance field $\mathcal{S}(\theta)$ and an embedding $\mathcal{E}(\theta)$. The appearance network $\mathcal{A}(\phi)$ is another MLP which takes $\mathcal{E}(\theta)$ and a frequency encoded representation of the viewing direction and returns the scene radiance along a ray.

(a) (b) (c) (d) (e)

Fig. 2. A snapshot of the important supervision signals. We capture a high dynamic range image [Mertens et al. 2007] and display it after tonemapping [Reinhard et al. 2023] in Fig. 2a. Figure 2b shows the scene depth (in mm) from stereo. Figure 2c displays the likelihood of each pixel falling on a depth edge. Figure 2d shows the object surface normals. We note that unlike conventional stereo matching ([Hirschmuller 2005]), [Xu et al. 2022] returns locally smooth surfaces and often ignores local texture variations but is less noisy. The inset shows the surface normals on the textured aluminum plate calculated as gradients of depth from conventional stereo matching. Finally, Fig. 2e identifies the pixels with the largest appearance variation due to moving lights.

The neural signed distance field $\mathcal{S}(\theta)$ is optimized to return the signed distance of a point from its nearest surface $\mathcal{S}(\theta) : \mathbb{R}^3 \rightarrow \mathbb{R}$. The surface of the object can be obtained from the zero-level set of $\mathcal{S}(\theta)$ – i.e. for all surface points $\mathbf{x}_s \in \mathbb{R}^3 \mid \mathcal{S}(\mathbf{x}_s|\theta) = 0$. We train $\mathcal{S}(\theta)$ by minimizing a geometric loss ℓ_D (Eq. (5)). We follow [Yariv et al. 2021] to transform the distance of a point $\vec{p}_i = \vec{r}|_{t_i}$ in a ray to its closest surface $s_i = \mathcal{S}(\theta, \vec{p}_i)$ to the scene density (or transmissivity).

$$\Psi_\beta(s) = \begin{cases} 0.5 \exp(\frac{s}{\beta}), & s \leq 0 \\ 1 - 0.5 \exp(\frac{-s}{\beta}), & \text{otherwise.} \end{cases} \tag{1}$$

To render the color C of a single pixel of the scene at a target view with a camera centered at $\vec{o}$ and an outgoing ray direction $\vec{d}$, we calculate the ray corresponding to the pixel $\vec{r} = \vec{o} + t\vec{d}$, and sample a set of points t_i along the ray. The networks $\mathcal{N}(\theta)$ and $\mathcal{A}(\phi)$ are then evaluated at all the $\mathbf{x}_i$ corresponding to t_i and the per point color $\mathbf{c}_i$. The transmissivity τ_i is obtained and composited together using the quadrature approximation from [Max 1995] as:

$$\mathbf{C} = \sum_i \exp(-\sum_{\mathrm{j<i}} \tau_\mathrm{j}\delta_\mathrm{j})(1 - \exp(-\tau_\mathrm{j}\delta_\mathrm{j}))\mathbf{c}_\mathrm{i}, \quad \delta_\mathrm{i} = \mathrm{t_i} - \mathrm{t_{i-1}} \tag{2}$$

The appearance can then be learned using a loss on the estimated and ground truth color $\mathbf{C}_{gt}$

$$\ell_C = \mathbb{E}\left[||\mathbf{C} - \mathbf{C}_{gt}||^2\right] \tag{3}$$

The appearance and geometry are jointly estimated by minimizing the losses in Eq. (4) using stochastic gradient descent [Kingma and Ba 2014].

$$\ell = \ell_C + \lambda_g \ell_D + \lambda_c \mathbb{E}(|\nabla^2_\mathbf{x} \mathcal{S}(\mathbf{x}_s)|) \tag{4}$$

λs are hyperparameters and the third term in Eq. (4) is the mean surface curvature minimized against the captured surface normals. As the gradients of the loss functions ℓ_C and ℓ_D propagate through $\mathcal{A}$ and $\mathcal{N}$ (and $\mathcal{S}$ as it is part of $\mathcal{N}$) the appearance and geometry are learned together.

3.2 Incorporating dense metric depth

Prior work has jointly learned $\mathcal{S}, \mathcal{N}, \mathcal{A}$ with only multi-view images, in contrast, we have access to estimates of true surface depth to any surface point $\mathbf{x}_s$. In this section we describe our method to directly optimize $\mathcal{S}$ with depth.

The depth estimates can be directly used to optimize appearance and render surfaces following [Dai et al. 2017; Oechsle et al. 2021; Zollhöfer et al. 2015]. However, to be able to approximate view-dependent appearance, we elect to learn a continuous and locally smooth function that approximates the signed distance function of the surface $\mathbf{x}_s$ which can then be transformed to scene density (Eq. (1)). To do this, we roughly follow [Gropp et al. 2020] and consider a loss function of the form

$$\ell_D(\theta) = \ell_{\mathbf{x}_s} + \lambda \mathbb{E}(||\nabla_\mathbf{x} \mathcal{S}(\mathbf{x}^\Delta, \theta)|| - 1)^2 \tag{5}$$

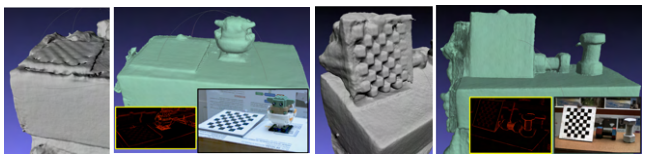
$$\text{where, } \ell_{\mathbf{x}_s} = \frac{1}{N} \Sigma_{\forall \mathbf{x}} \left[\mathcal{S}(\mathbf{x}, \theta) + 1 - \langle \nabla_\mathbf{x} \mathcal{S}(\mathbf{x}, \theta), \mathbf{n}_x \rangle\right]$$

Through the two components, the loss encourages the function $\mathcal{S}(\mathbf{x}, \theta)$ to vanish at the observed surface points and the gradients of the surface to align at the measured surface normals. The second component in Eq. (5) is the Eikonal term ([Crandall and Lions 1983]) which encourages the gradients of $\mathcal{S}$ to have a unit L_2 norm everywhere. The individual terms of Eq. (5) are averaged across all samples in a batch corresponding to N rays projected from a known camera.

The Eikonal constraint applies to the neighborhood points $\mathbf{x}^\Delta_s$ of each point in $\mathbf{x}_s$. [Gropp et al. 2020] identifies candidate $\mathbf{x}^\Delta_s$ through a nearest neighbor search, where as [Yariv et al. 2021] identifies $\mathbf{x}^\Delta$ through random perturbations of the estimated surface point along the projected ray. As we have access to depth maps, we identify the variance of the neighborhood of $\mathbf{x}_s$ through a sliding window maximum filter on the depth images. This lets us avoid expensive nearest neighbor lookups for a batch of $\mathbf{x}_s$ to generate better estimates of $\mathbf{x}^\Delta_s$ than [Yariv et al. 2021] at train time. As a result, convergence is accelerated – ($\sim$ 100$\times$ over [Gropp et al. 2020]) with no loss of accuracy. As we used metric depth, noisy depth estimates for parts of the scene are implicitly averaged by $\mathcal{S}$ optimized by minimizing Eq. (5), making us more robust to errors than [Yu et al. 2022]. We provide more details in appendix A.

3.3 Incorporating depth edges in joint optimization of appearance and geometry

Prior works show the benefits of jointly refining geometry and appearance as it yields lesser independent hyperparameters, some



(a) (b)

Fig. 3. We demonstrate a corner case of jointly refining appearance and geometry. The left insets of Figs. 3a and 3b are the scene geometries recovered in the worst cases, the right insets display the better meshes recovered using the method described in Section 3.3. An image used for training and the edge map used for sampling are in the insets. We recommend zooming into the figure for details. Corresponding quantitative results are in Table 4.

degree of geometric super-resolution, and more stable training. However, some pathological cases may arise when the scene has a large variation in appearance corresponding to a minimal variation in geometry across $\mathbf{x}_s$ and $\mathbf{x}_s^\Delta$. We investigate this effect by considering an extreme case – a checkerboard printed on matte paper with an inkjet printer, where there is no geometric variation (planar geometry) or view dependent artifacts (ink on matte paper is close to Lambertian) corresponding to a maximum variation in appearance (white on black). The qualitative results are presented in Fig. 3.

Consider two rays $\vec{r}_{\mathbf{x}_s}$ and $\vec{r}_{\mathbf{x}_s^\Delta}$ connecting the camera center and two neighboring points $\mathbf{x}_s$ and $\mathbf{x}_s^\Delta$ on two sides of an checkerboard edge included in the same batch of the gradient descent. The total losses for those rays depend on the sum of the geometry and appearance losses (Eq. (4)). By default, the current state of the art ([Li et al. 2023; Wang et al. 2023; Yariv et al. 2023] etc.) do not have a mechanism to disambiguate between texture and geometric edges (depth discontinuities).

As seen in Fig. 3, given unsuitable hyperparameters, the approaches will continue to jointly update both geometry and appearance to minimize a combined loss (Eq. (4)). This can often result in pathological reconstructions (left insets in Fig. 3) due to ℓ_C gradients dominating over ℓ_D. By gradually increasing the modelling capacity of $\mathcal{N}$ we can somewhat avoid this artifact and force the gradient updates to focus on $\mathcal{A}$ to minimize the cumulative loss. [Li et al. 2023] recognize this and provide an excellent set of hyperparameters and training curricula to gradually increase the modelling capacity of $\mathcal{N}(\phi)$. This results in remarkable geometric reconstructions for well known datasets ([Jensen et al. 2014; Knapitsch et al. 2017]). Alternatively, if we have per-pixel labels of geometric edges ($\mathbb{E}$), we can preferentially sample image patches with low variation of geometric features when the model capacity is lower ($\mathcal{S}(\theta)$ tends to represent smoother surfaces), and focus on image patches with geometric edges when the model capacity has increased. The modelling capacity of $\mathcal{A}(\phi)$ never changes.

Figure 5 describes our sampling procedure while learning a scene with a variety of geometric and texture edges. Equation (6) is used to draw pixel samples – the probability of drawing pixel p_i is calculated as a linear blend of the likelihood that it belongs to the set of edge pixels $\mathbb{E}$ and α is a scalar ($\alpha \in [0, 1]$) proportional to the progress of the training.

$$P(p_i|\alpha) = (1-\alpha)P(p_i \in \mathbb{E}) + \alpha P(p_i \notin \mathbb{E}) \tag{6}$$

To preserve the geometric nature of the edges while ruling out high frequency pixel labels, we use Euclidean distance transform ([Felzenszwalb and Huttenlocher 2012]) to dilate $\mathbb{E}$ before applying Eq. (6). We provide implementation details in appendix A for reproducibility.

3.4 AdaShell^{++}: Accelerating training with dense depth

The slowest step in training and inference for neural volumetric representations is generally the evaluation of Eq. (2). In this section we describe our method to accelerate training by incorporating metric depth.

A method to make training more efficient involves drawing the smallest number of the most important samples of t_i for any ray. The sampling of t_i is based on the current estimate of the scene density and although these samples can have a large variance, given a large number of orthogonal view pairs (viewpoint diversity), and the absence of very strong view dependent effects, the training procedure is expected to recover an unbiased estimate of the true scene depth (see e.g. [Goli et al. 2024]). We can accelerate the convergence by a) providing high quality biased estimate of the scene depth and b) decreasing the number of samples for t_i along the rays.

Given the high quality of modern deep stereo (we use[Xu et al. 2022]) and a well calibrated camera system, stereo depth can serve as a good initial estimate of the true surface depth. We use stereo depth, aligned across multiple views of the scene to pre-optimize the geometry network $\mathcal{S}(\theta)$. The other channels $\mathcal{E}(\theta)$ of $\mathcal{N}$ remain un-optimized. A pre-optimized $\mathcal{S}$ can then be used for high quality estimates of ray termination depths.

[Oechsle et al. 2021; Wang et al. 2021; Yariv et al. 2021] recommend using root finding techniques (e.g. bisection method) on scene transmissivity (Eq. (1)) to estimate the ray termination depth. The samples for Eq. (2) are then generated around the estimated surface point. Drawing high variance samples as $\mathcal{N}$ and $\mathcal{A}$ are jointly optimized reduces the effect of low quality local minima, especially in the initial stages of the optimization. As we have a pre-trained scene transmissivity field ($\mathcal{S}$ transformed with Eq. (1)), we can draw a few high-quality samples to minimize the training effort.

We found uniformly sampling around the estimated ray-termination depth (UniSurf^{++}baseline in Section 3.5 and Fig. 3) to be unsuitable. Instead, we pre-calculated a discrete sampling volume by immersing $\mathcal{S}$ in an isotropic voxel grid and culling the voxels which report a lower than threshold scene density. We then used an unbiased sampler from [Yariv et al. 2021] to generate the samples in this volume. This let us greatly reduce the number of root-finding iterations and samples, while limiting the variance by the dimensions of the volume along a ray. As the training progresses, we decrease the culling threshold to converge to a thinner sampling volume around the surface while reducing the number of samples required.

We show the sampling volume (at convergence) and our reconstruction results in Figs. 8 and 9 respectively. We retain the advantages of volumetric scene representation as demonstrated by the reconstruction of the thin structures in the scene, while reducing training effort. We dub our method AdaShell^{++}to acknowledge

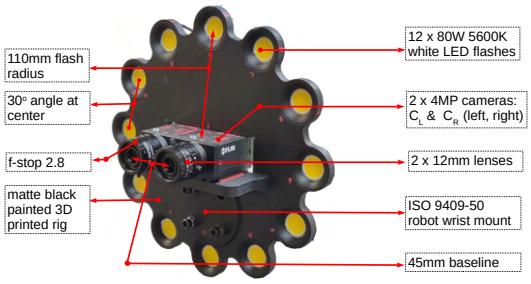


Fig. 4. Our system used to capture the data in Fig. 2.

Property	BMVS	DTU	ReNe	DGT*	PaNDoRa	OpenIllum.	Ours
depth	✓	✓	×	✓	×	×	✓
OLAT/ Flash	OLAT	×	OLAT	OLAT	×	OLAT	Flash
polarization	×	×	×	×	✓	✓	×
specularity	×	×	×	×	✓	×	✓
depth edges	×	×	×	×	×	×	✓
HDR	✓	×	×	✓	×	✓	✓
illum. model	✓	×	×	✓	✓	✓	×

Table 1. We identify some differences between our dataset and a few established datasets: BMVS[Yao et al. 2020], DTU[Jensen et al. 2014], ReNe[Toschi et al. 2023], DiLiGenT[Shi et al. 2016], DiLiGenT-MV[Shi et al. 2016] (both abbreviated as DGT*), PaNDoRa[Dave et al. 2022] and OpenIllumination[Liu et al. 2023a]. OLAT: one light at a time.

[Wang et al. 2023], which demonstrates a related approach to accelerate inference.

3.5 Baselines augmented with depth

In addition to AdaShell^{++}we augment three other state-of-the-art methods to incorporate metric depth:

VolSDF^{++}is our augmented version of [Yariv et al. 2021, 2023], where we use the metric depths along the rays to optimize the geometry (Eq. (4)). All the other parts of the original approaches, including the method for generating samples for Eq. (2), and analogs for Eq. (1) are left intact.

NeUS^{++}is our augmented version of [Li et al. 2023], where we also use metric depths to optimize the geometry. The rest of the algorithm including the background radiance field is left intact.

UniSurf^{++}is our deliberately hamstrung version of [Oechsle et al. 2021] where we force the the samples generated for Eq. (2) to have a very low variance around the current biased estimate of the surface. This makes the algorithm necessarily indifferent to the relative magnitudes of ℓ_D and ℓ_C in Eq. (4), and helps us exaggerate the pathological effects of not segregating texture and geometric edges. We choose to name the method UniSurf^{++}after we (and [Brahimi et al. 2024]) observed that original method was vulnerable to this artifact under certain hyperparameter choices. Across all the methods, we implement and train $\mathcal{N}(\theta)$ following [Li et al. 2023], and all of them use the same appearance network $\mathcal{A}(\phi)$.

AdaShell^{++}and UniSurf^{++}require a warm start – $\mathcal{S}$ pre-optimized for 5K gradient steps. All methods except UniSurf^{++}use the sampling strategy from Section 3.3. More details are in appendix A.

4 SETUP AND DATASET

4.1 A multi-flash stereo camera

In addition to multi-view images, scene depth and depth edges are valuable signals to train neural 3D representations. To capture all the supervision signals, we designed and fabricated a multi-flash stereo camera based on insights from [Feris et al. 2004; Raskar et al. 2004]. We capture data by moving our camera rig in front of objects. For each camera pose we capture a stereo pair of high dynamic range (HDR) images, two depth maps from left and right stereo, two corresponding image aligned surface normals (as gradient of depth maps). We augment the HDR images by in-painting them with the depth edges before using [Xu et al. 2022] to preserve intricate surface details in the depth maps. Additionally we capture 12 pairs of multi-illumination images for 12 flash lights around the cameras, one light at a time. From the multi-flash images we recover a per pixel likelihood of depth edges in the scene and a label of pixels with a large appearance variation under changing illumination – relating to the specularity. We detail the design of our rig and the capture processes in appendix B. Figure 2 shows a snapshot of the data captured, Fig. 4 illustrates our camera rig prototype.

4.2 Dataset

Although a data set is not the primary contribution of our research, we capture some salient aspects of the scene that are not present in several established datasets. We identify these aspects in Table 1.

In the rows labeled "specularity" and "depth edges" we note if the dataset has explicit labels for the specular nature of the pixel or a presence of a depth edge at that pixel respectively. Under "illum. model" we note if an explicit illumination model is present per scene – we do not capture an environment illumination model, and instead provide light poses. PaNDoRa does not have explicit specularity labels but polarization measurements at pixels may be used to derive high quality specularity labels, which are better than what our system natively captures. We differentiate between "OLAT" (one light at a time) and "flash" by the location of the source of illumination. Similar to WildLight [Cheng et al. 2023], our flashes are parallel to the imaging plane, located close (~0.1f) to the camera, as opposed to ReNE and OpenIllumination.

5 EXPERIMENTS AND RESULTS

5.1 Accuracy of incorporating metric depth

We reconstruct synthetic scenes with ground truth depth from [Azinović et al. 2022; Sandström et al. 2023] to measure the accuracy of our technique. We use 12-15 RGBD images to reconstruct the scenes and train for an average of 30k gradient steps (~1500 epochs) in about 75 minutes. In contrast, [Azinović et al. 2022; Sandström et al. 2023] use 300+ RGBD tuples and 9+ hours of training on comparable hardware. Notably, [Azinović et al. 2022] also optimizes for noise in camera poses and reports metrics with ground truth and optimized poses. We report the best metric among these two. [Dai et al. 2017] registers the images themselves. We register the RGBD images with a combination of rigid and photometric registration ([Park et al. 2017; Sarlin et al. 2019; Zhou et al. 2016]).

Scene	NeuralRGBD	BundleFusion	AdaShell^{++}	NeUS^{++}
greenroom	0.013	0.024	**0.015**	0.016
staircase	0.045	0.091	0.024	**0.009**
kitchen I	0.252	0.234	0.044	**0.036**
kitchen II	**0.032**	0.089	0.045	**0.032**

Table 2. Accuracy of reconstruction from un-posed RGBD images. For ***un-posed*** RGBD images, we compare the accuracy of reconstructing the scene using AdaShell^{++}and NeUS^{++}with [Azinović et al. 2022] and [Dai et al. 2017]. We report normalized chamfer distances (lower is better) across four synthetic scenes from [Azinović et al. 2022]. We report the metrics for AdaShell^{++}and NeUS^{++}– performance for AdaShell^{++}and VolSDF^{++}were very similar for this dataset.

Scene	Rm 0	Rm 1	Rm 2	Off 0	Off 1	Off 2	Off 3	Off 4
PointSLAM	0.61	0.41	0.37	0.38	0.48	0.54	0.69	0.72
AdaShell^{++}	**0.11**	**0.10**	**0.09**	**0.12**	**0.06**	**0.08**	**0.14**	**0.11**

Table 3. Accuracy of reconstruction from posed RGBD images. For RGBD images with ***ground-truth poses***, we compare the accuracy of reconstructing the scene between AdaShell^{++}and PointSLAM[Sandström et al. 2023]. We report the mean L_1 distances in cm (lower is better) across eight synthetic scenes from the Replica Dataset [Straub et al. 2019]. For posed images, we report an average improvement of 81% over PointSLAM.

Scene	stereo	VolSDF^{++}	NeUS^{++}	AdaShell^{++}	UniSurf^{++}
horizontal (Fig. 3a)	6.22	4.74	**2.77**	5.42	13.21
vertical (Fig. 3b)	6.82	3.87	**3.68**	6.34	16.02

Table 4. Depth edges help prioritize learning of texture discontinuities over geometric ones. We report the RMS deviation from a plane (lower is better) for the reconstructed checkerboard surfaces in mm. We note that AdaShell^{++}performs slightly worse than volumetric methods VolSDF^{++}and NeUS^{++}. Except for UniSurf^{++}, all improve the quality of the surface measured with only stereo. Qualitative results are shown in Fig. 3.

We present the quantitative results in Tables 2 and 3. We replicate or out-perform the baselines by using a fraction of the training data and gradient steps. Among all methods discussed in Section 3.5, AdaShell^{++}and VolSDF^{++}demonstrate similar performance, NeUS^{++}recovers a smoother surface at the expense of $\sim 1.25\times$ more gradient steps. Our errors on these synthetic datasets closely reflect the performance of [Gropp et al. 2020] on approximating surfaces from low noise point clouds. These datasets do not have large view dependent appearance variations to affect the gradient updates.

5.2 The effect of depth edges in training

We tested fused RGBD maps from stereo and four baselines from Section 3.5 to investigate the effect of depth and texture edges. We use edge guided sampling (Section 3.3 and Eq. (6)) for all except stereo and UniSurf^{++}to prioritize learning geometric discontinuities over appearance. We present the results in Fig. 3 and Table 4. All of the baselines except UniSurf^{++}improve the reconstruction accuracy due to segregation of texture and depth edges. The smoothness enforced by the curvature loss in Eq. (4) also improves the surface reconstruction over stereo.

PSNR	VolSDF^{++}			NeUS^{++}			AdaShell^{++}			UniSurf^{++}		
20	6.42	6.60	9.43	16.1	12.5	23.4	3.6	1.52	4.85	2.5	1.81	– –
25	15.0	20.4	24.7	41.4	74.8	72.2	16.7	7.24	54.8	– –	– –	– –
27.5+	21.3	33.1	40.0	70.5	100+	100+	22.6	23.2	94.6	– –	– –	– –

Table 5. Training performance for view synthesis. We compare the number of gradient steps (in thousands, lower is faster) to reach a target test-time accuracy of reconstructing the scene (PSNR in dB), for scenes in Fig. 6(a,b,c). We warm-start AdaShell^{++}and UniSurf^{++}by pre-optimizing $\mathcal{S}$ for 5k gradient steps (not included in the table). Cases of divergences of UniSurf^{++}marked with $--$ in the table.

5.3 View synthesis with dense depth

Incorporating dense metric depth and our sampling strategy from Sections 3.2 and 3.3 enables AdaShell^{++}, VolSDF^{++}, and NeUS^{++}to perform competitively across challenging scenes. Scene A (Fig. 6(a)) looks at a couple of reflective objects with large variation in view dependent appearance. Additionally, there are large local errors in the captured depth maps due to specularities in the scene. We capture six stereo pairs, train on 11 images and test on one image. Scene B (Fig. 6(b)) features a rough metallic object of relatively simple geometry captured by a 16mm lens (450 mm focal length, shallow depth of field). We capture four stereo pairs, train on seven images and test on one image. Scene C (Fig. 6(c)) features a fairly complicated geometry and is captured with 12 stereo pairs. We train on 22 images and test on two. Quantitative results of our experiments are in Table 5. We observe that AdaShell^{++}, which is roughly 15% faster per gradient step than VolSDF^{++}, generally converges the fastest (wall clock time) to a target PSNR. When the geometry is very complicated (scene C), an equally complicated sampling volume negates the efficiency gains of our sampler. We could not find good parameters for UniSurf^{++}for any of these sequences.

View synthesis was unsuccessful without the inclusion of dense depth. We trained VolSDF^{++}with no depth supervision (equivalent to [Yariv et al. 2021]) until saturation (less than 0.1 PSNR increase for 1000 consecutive epochs). The reconstructions, none of which had a PSNR of 18 or higher, are shown in the last column of Fig. 6.

condition	Fig. 7(a,b)[5]	Fig. 7(c,d)[7]	Fig. 7(e,f)[5]
edge sampling	**491**	**403**	**225**
no edge sampling	593	419	251
noisy stereo	600	523	369
27.5+ PSNR w/ noise(AdaShell^{++})	7.93	20.2	5.12
27.5+ PSNR w/o noise (AdaShell^{++})	7.85	23.2	2.71
25.0+ PSNR w/ noise (NeUS^{++})	49.4	36.0	32.9

Table 6. Effect of noisy depth and depth edges. Top: The surface reconstruction quality (Hausdorff distance, lower is better) with conventional (noisy) stereo compared with surface recovered by NeUS^{++}on learned stereo. Qualitative comparison in Fig. 7(b,d,f). Bottom: gradient steps (in 1000s lower is faster) required to surpass a test time accuracy of 27.5dB with AdaShell^{++}. We specify the count of training views in [ ] braces.

Scene	face_1	face_2	face_3	face_4	shiny_1	shiny_2
mask	**28.95**	**31.19**	**29.56**	**27.28**	**25.18**	**24.51**
no mask	27.17	30.05	27.82	25.46	23.65	21.88

Table 7. Relighting scenes with a volumetric renderer. We report PSNR (higher is better) under two heads – masked and unmasked relit images, to offset the effects of incorrect shadows cast on the background. The unmasked reconstructions generally have a poorer PSNR because our implicit scene understanding approach does not approximate a ray tracer and cannot cast correct shadows on the background. The lower PSNR for reconstructing the shiny objects is mainly due to the inability of the network to model saturation caused by reflection. Results in Fig. 11.

5.4 Using noisy depth

To investigate the effects of noise in the depth maps, we obtain the depths of scenes using conventional stereo. We used semi-global matching stereo ([Hirschmuller 2005]) with a dense census cost ([Zabih and Woodfill 1994]) and sub-pixel refinement on tone mapped HDR images to calculate the surface depth. Surface normals were calculated using the spatial gradients of the depth maps. To focus on the performance of our approaches, we did not filter or smooth the depth obtained from conventional stereo. From the top of Table 6, we observe that NeUS^{++}strictly improves the quality of the surface reconstructed from just noisy stereo (row 1 and 2 versus row 3), especially when edge sampling is enabled. If the end goal is just view synthesis, AdaShell^{++}, which blends the advantages of volumetric and surface based rendering, performs equally well with large noise in depth, whereas NeUS^{++}takes many more iterations to converge. This indicates that photorealistic view synthesis with a volumetric renderer is possible with noisy depth data. However conventional stereo often introduces large local errors (see e.g. surface patterns on Fig. 7(b,f)) which our approaches were unable to improve significantly.

In the presence of noisy depth, the quality of the reconstructed surface was enhanced through edge-based sampling (Section 3.3 and Eq. (6)). Our sampling strategy allocated samples away from depth edges, where the noise was more prevalent, leading to fewer gradient steps spent modelling areas with higher noise. Table 6 presents the quantitative details of the experiment.

5.5 Relighting

We capture multi-illumination images with known light poses and recover geometry independently of appearance. This allows us to infer the illumination dependent appearance using a combination of physically based appearance parameters – e.g. the Disney Principled BRDF[Burley 2012]. As a benchmark, we upgraded the closest related work, [Cheng et al. 2023], which uses the full gamut of the Disney BRDF parameters, with NeUS^{++}, to incorporate dense depth. For the data we collected, the optimization process as implemented by [Cheng et al. 2023], was quite brittle and some parameters (e.g. 'clearcoat-gloss') would often take precedence over other appearance parameters (e.g. 'specular-tint') and drive the optimization to a poor local minima. Figure 10 demonstrates this problem with textured meshes. We found the optimization of a subset of appearance parameters ('base-color', 'specular-tint', and 'roughness') to be the most stable. [Brahimi et al. 2024; Zhang et al. 2022] conclude the same.

For relighting, we explore two avenues – the inference step of our approach as a volumetric renderer and a mesh created with the appearance parameters as texture. Quantitative and qualitative results of the volumetric renderer are shown in Table 7 and Fig. 11. We used [Blender 2024] to unwrap the geometry and generate texture coordinates whose quality exceeded [Young 2024] and our implementation of [Srinivasan et al. 2023]. None of our approaches worked on the ReNe dataset (Table 1) due to low view diversity, and the absence of metric depth. We used the labels in Fig. 2e to allocate more gradient steps in learning the regions with higher appearance variation. We provide more details in appendix A.

6 LIMITATIONS

Although we achieve state of the art results in view-synthesis and relighting with a few views, our approach struggles to represent transparent objects and accurately capture the geometry of reflective surfaces. [Liu et al. 2023b] address the problem of reflective objects by modelling background reflections and is based on the architecture proposed by [Wang et al. 2021]. As NeUS^{++}enables [Wang et al. 2021] to use possibly noisy metric depth, it can potentially be extended to model reflective objects.

Our approaches require metric depth and depth edges for the best performance. Our approach relies on capture devices with reasonable quality depth measurements. Future work will address incorporation of monocular and sparse depth priors with depth edges.

Incorporation of metric depth introduces a strong bias, often limiting super resolution of geometry sometimes achieved in neural 3D scene representation (see e.g. [Li et al. 2023]). Decreasing the effect of Eq. (5) during training may potentially encourage geometric superresolution and is future work.

Finally, modern grid based representations (see e.g. [Duckworth et al. 2023; Reiser et al. 2023]) produce very compelling view interpolation results at a fraction of the computational cost of a state of the art volumetric renderer (e.g. [Müller et al. 2022; Wang et al. 2023]). However, they need to be "distilled" from a pre-trained volumetric view interpolator. Future work can investigate the use of depth priors to train a grid based representation directly from color and depth images.

7 CONCLUSIONS

We present a solution to incorporate dense metric depth into neural 3D reconstruction which enables state of the art geometry reconstruction. With dense metric depth available, we demonstrate an a to accelerate training of neural representations for view synthesis and relighting of small scenes. We examine a corner case of jointly learning appearance and geometry and address it by incorporating additional supervision signals. Additionally, we describe a variant of the multi-flash camera to capture the salient supervision signals needed to improve photorealistic 3D reconstruction.

REFERENCES

2023. ArtecLeo. https://www.artec3d.com/portable-3d-scanners/artec-leo
2023. Azure Kinect 3D sensors. https://www.microsoft.com/en-us/d/azure-kinect-dk

2023. Ensenso XR series scanners. https://www.ids-imaging.us/ensenso-3d-camera-xr-series.html
2023. Photoneo PhoXi3D scanners. https://www.photoneo.com/phoxi-3d-scanner/
3DZephyr. 2022. 3DF Zephyr - photogrammetry software - 3D models from photos. https://www.3dflow.net/3df-zephyr-photogrammetry-software/
AliceVision. 2022. AliceVision Meshroom. https://alicevision.org/
Benjamin Attal, Eliot Laidlaw, Aaron Gokaslan, Changil Kim, Christian Richardt, James Tompkin, and Matthew O'Toole. 2021. Törf: Time-of-flight radiance fields for dynamic scene view synthesis. *Advances in neural information processing systems* 34 (2021), 26289–26301. https://proceedings.neurips.cc/paper_files/paper/2021/file/dd03de08bfdff4d8ab01117276564cc7-Paper.pdf
Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. 2022. Neural RGB-D Surface Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 6290–6301. https://dazinovic.github.io/neural-rgbd-surface-reconstruction/
Sai Bi, Zexiang Xu, Pratul Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. 2020. Neural reflectance fields for appearance acquisition. *arXiv preprint arXiv:2008.03824* (2020). https://arxiv.org/pdf/2008.03824.pdf
Blender. 2024. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam. http://www.blender.org
Pierre Boudoin. 2023. Hyper capture: 3D object scan. https://apps.apple.com/us/app/hyper-capture-3d-object-scan/
Mohammed Brahimi, Bjoern Haefner, Tarun Yenamandra, Bastian Goldluecke, and Daniel Cremers. 2024. SupeRVol: Super-Resolution Shape and Reflectance Estimation in Inverse Volume Rendering. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 3139–3149. https://openaccess.thecvf.com/content/WACV2024/html/Brahimi_SupeRVol_Super-Resolution_Shape_and_Reflectance_Estimation_in_Inverse_Volume_Rendering_WACV_2024_paper.html
Brent Burley. 2012. Physically-based shading at disney. In *Acm Siggraph*, Vol. 2012. vol. 2012, 1–7. https://media.disneyanimation.com/uploads/production/publication_asset/48/asset/s2012_pbs_disney_brdf_notes_v3.pdf
Manmohan Chandraker, Jiamin Bai, and Ravi Ramamoorthi. 2012. On differential photometric reconstruction for unknown, isotropic BRDFs. *IEEE transactions on pattern analysis and machine intelligence* 35, 12 (2012), 2941–2955. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6327191
Arkadeep Narayan Chaudhury, Leonid Keselman, and Christopher G Atkeson. 2024. Shape From Shading for Robotic Manipulation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 8389–8398. https://openaccess.thecvf.com/content/WACV2024/html/Chaudhury_Shape_From_Shading_for_Robotic_Manipulation_WACV_2024_paper.html
Xinjing Cheng, Peng Wang, and Ruigang Yang. 2019. Learning depth with convolutional spatial propagation network. *IEEE transactions on pattern analysis and machine intelligence* 42, 10 (2019), 2361–2379. https://arxiv.org/pdf/1810.02695.pdf
Ziang Cheng, Junxuan Li, and Hongdong Li. 2023. WildLight: In-the-wild Inverse Rendering with a Flashlight. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 4305–4314. https://junxuan-li.github.io/wildlight-website/
Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. 2015. Robust reconstruction of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5556–5565. http://www.open3d.org/docs/release/tutorial/pipelines/multiway_registration.html
Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. 2008. MeshLab: an Open-Source Mesh Processing Tool. In *Eurographics Italian Chapter Conference*, Vittorio Scarano, Rosario De Chiara, and Ugo Erra (Eds.). The Eurographics Association. https://doi.org/10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136
Michael G Crandall and Pierre-Louis Lions. 1983. Viscosity solutions of Hamilton-Jacobi equations. *Transactions of the American mathematical society* 277, 1 (1983), 1–42. https://www.ams.org/journals/tran/1983-277-01/S0002-9947-1983-0690039-8/S0002-9947-1983-0690039-8.pdf
CREE. 2024. CREE XLamp CXA2540. https://www.digikey.com/en/products/detail/creeled-inc/CXA2540-0000-000N00W257F/4437055
Brian Curless and Marc Levoy. 1996. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 303–312. https://dl.acm.org/doi/pdf/10.1145/237170.237269
Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. 2017. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)* 36, 4 (2017), 1. https://dl.acm.org/doi/pdf/10.1145/3072959.3054739
Akshat Dave, Yongyi Zhao, and Ashok Veeraraghavan. 2022. Pandora: Polarization-aided neural decomposition of radiance. In *European Conference on Computer Vision*. Springer, 538–556. https://akshatdave.github.io/pandora/index.html
Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. 2022. Depth-supervised NeRF: Fewer Views and Faster Training for Free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://www.cs.cmu.edu/~dsnerf/
Daniel Duckworth, Peter Hedman, Christian Reiser, Peter Zhizhin, Jean-François Thibert, Mario Lučić, Richard Szeliski, and Jonathan T. Barron. 2023. SMERF: Streamable Memory Efficient Radiance Fields for Real-Time Large-Scene Exploration. arXiv:2312.07541 [cs.CV] https://smerf-3d.github.io/
EdmundOptics. 2024. C series fixed focal length lenses: Edmund Optics. https://www.edmundoptics.com/f/c-series-fixed-focal-length-lenses/13679/
Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. 2021. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 10786–10796. https://openaccess.thecvf.com/content/ICCV2021/html/Eftekhar_Omnidata_A_Scalable_Pipeline_for_Making_Multi-Task_Mid-Level_Vision_Datasets_ICCV_2021_paper.html
Pedro F Felzenszwalb and Daniel P Huttenlocher. 2012. Distance transforms of sampled functions. *Theory of computing* 8, 1 (2012), 415–428. https://theoryofcomputing.org/articles/v008a019/v008a019.pdf
Rogerio Feris, Ramesh Raskar, Longbin Chen, Kar-Han Tan, and Matthew Turk. 2005. Discontinuity preserving stereo with small baseline multi-flash illumination. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, Vol. 1. IEEE, 412–419. https://rogerioferis.com/multi-flash-stereo/
Rogerio Feris, Ramesh Raskar, Kar-Han Tan, and Matthew Turk. 2004. Specular reflection reduction with multi-flash imaging. In *Proceedings. 17th Brazilian symposium on computer graphics and image processing*. IEEE, 316–321. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1352976
Martin A Fischler and Robert C Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (1981), 381–395.
FLIR. 2024. Grasshopper3 USB3 Model: GS3-U3-41C6C. https://www.flir.com/products/grasshopper3-usb3/?model=GS3-U3-41C6C-C
Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5501–5510. https://openaccess.thecvf.com/content/CVPR2022/html/Fridovich-Keil_Plenoxels_Radiance_Fields_Without_Neural_Networks_CVPR_2022_paper.html
Qiancheng Fu, Qingshan Xu, Yew Soon Ong, and Wenbing Tao. 2022. Geo-NeUS: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. *Advances in Neural Information Processing Systems* 35 (2022), 3403–3416. https://proceedings.neurips.cc/paper_files/paper/2022/file/16415eed5a0a121bfce79924db05d3fe-Paper-Conference.pdf
Silvano Galliani, Katrin Lasinger, and Konrad Schindler. 2015. Massively Parallel Multiview Stereopsis by Surface Normal Diffusion. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. https://openaccess.thecvf.com/content_iccv_2015/html/Galliani_Massively_Parallel_Multiview_ICCV_2015_paper.html
Ginibird. 2021. Garden Tea Lights [License: CC-0]. https://www.blendswap.com/blend/27683
Lily Goli, Cody Reading, Silvia Sellán, Alec Jacobson, and Andrea Tagliasacchi. 2024. Bayes' Rays: Uncertainty Quantification in Neural Radiance Fields. *CVPR* (2024). https://bayesrays.github.io/
Paulo F. U. Gotardo, Tomas Simon, Yaser Sheikh, and Iain Matthews. 2015. Photogeometric Scene Flow for High-Detail Dynamic 3D Reconstruction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. https://openaccess.thecvf.com/content_iccv_2015/html/Gotardo_Photogeometric_Scene_Flow_ICCV_2015_paper.html
Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. 2020. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099* (2020). https://arxiv.org/abs/2002.10099
Vitor Guizilini, Igor Vasiljevic, Jiading Fang, Rares Ambrus, Sergey Zakharov, Vincent Sitzmann, and Adrien Gaidon. 2023. DeLiRa: Self-Supervised Depth, Light, and Radiance Fields. *arXiv preprint arXiv:2304.02797* (2023). https://sites.google.com/view/tri-delira
Heiko Hirschmuller. 2005. Accurate and efficient stereo processing by semi-global matching and mutual information. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, 807–814. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1467526
Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. 2014. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 406–413. https://www.cv-foundation.org/openaccess/content_cvpr_2014/papers/Jensen_Large_Scale_Multi-view_2014_CVPR_paper.pdf
Kaizhang Kang, Minyi Gu, Cihui Xie, Xuanda Yang, Hongzhi Wu, and Kun Zhou. 2021. Neural Reflectance Capture in the View-Illumination Domain. *IEEE Transactions on Visualization and Computer Graphics* 29, 2 (2021), 1450–1462. https://ieeexplore.ieee.org/abstract/document/9557800
Kaizhang Kang, Cihui Xie, Chengan He, Mingqi Yi, Minyi Gu, Zimin Chen, Kun Zhou, and Hongzhi Wu. 2019. Learning efficient illumination multiplexing for joint

capture of reflectance and shape. *ACM Trans. Graph.* 38, 6 (2019), 165–1. https://svbrdf.github.io/publications/jointcap/jointcap.pdf

Michael Kazhdan and Hugues Hoppe. 2013. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)* 32, 3 (2013), 1–13. https://www.cs.jhu.edu/~misha/MyPapers/ToG13.pdf

Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. 2017. Intel RealSense stereoscopic depth cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 1–10. https://openaccess.thecvf.com/content_cvpr_2017_workshops/w15/papers/Keselman_Intel_RealSense_Stereoscopic_CVPR_2017_paper.pdf

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014). https://arxiv.org/pdf/1412.6980.pdf

Simon Klenk, Lukas Koestler, Davide Scaramuzza, and Daniel Cremers. 2023. E-nerf: Neural radiance fields from a moving event camera. *IEEE Robotics and Automation Letters* 8, 3 (2023), 1587–1594. https://ieeexplore.ieee.org/abstract/document/10028738

Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)* 36, 4 (2017), 1–13. https://dl.acm.org/doi/pdf/10.1145/3072959.3073599

Kiriakos N Kutulakos and Steven M Seitz. 1999. A theory of shape by space carving. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Vol. 1. IEEE, 307–314. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=791235

Ruilong Li, Matthew Tancik, and Angjoo Kanazawa. 2022. Nerfacc: A general nerf acceleration toolbox. *arXiv preprint arXiv:2210.04847* (2022). https://www.nerfacc.com/

Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. 2023. Neuralangelo: High-Fidelity Neural Surface Reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://research.nvidia.com/labs/dir/neuralangelo/

Chao Liu, Srinivasa G Narasimhan, and Artur W Dubrawski. 2018. Near-light photometric stereo using circularly placed point light sources. In *2018 IEEE International Conference on Computational Photography (ICCP)*. IEEE, 1–10. https://www.cs.cmu.edu/~ILIM/projects/IM/nearPS/

Isabella Liu, Linghao Chen, Ziyang Fu, Liwen Wu, Haian Jin, Zhong Li, Chin Ming Ryan Wong, Yi Xu, Ravi Ramamoorthi, Zexiang Xu, et al. 2023a. OpenIllumination: A Multi-Illumination Dataset for Inverse Rendering Evaluation on Real Objects. *arXiv preprint arXiv:2309.07921* (2023). https://oppo-us-research.github.io/OpenIllumination/

Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. 2023b. NeRO: Neural Geometry and BRDF Reconstruction of Reflective Objects from Multiview Images. *arXiv preprint arXiv:2305.17398* (2023). https://arxiv.org/abs/2305.17398

Weng Fei Low and Gim Hee Lee. 2023. Robust e-NeRF: NeRF from Sparse & Noisy Events under Non-Uniform Motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 18335–18346. https://openaccess.thecvf.com/content/ICCV2023/html/Low_Robust_e-NeRF_NeRF_from_Sparse__Noisy_Events_under_Non-Uniform_ICCV_2023_paper.html

LumaLabs. 2023. Luma AI: AI for gorgeous 3D capture. https://lumalabs.ai/

Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. 2021. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *CVPR*. https://nerf-w.github.io/

Matterport. 2023. Matterport: Drive results with Digital Twins. https://matterport.com/

Nelson Max. 1995. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics* 1, 2 (1995), 99–108. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=468400

Tom Mertens, Jan Kautz, and Frank Van Reeth. 2007. Exposure fusion. In *15th Pacific Conference on Computer Graphics and Applications (PG'07)*. IEEE, 382–390. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4392748

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. 2019. Occupancy Networks: Learning 3D Reconstruction in Function Space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://openaccess.thecvf.com/content_CVPR_2019/html/Mescheder_Occupancy_Networks_Learning_3D_Reconstruction_in_Function_Space_CVPR_2019_paper.html

Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P. Srinivasan, and Jonathan T. Barron. 2022. NeRF in the Dark: High Dynamic Range View Synthesis from Noisy Raw Images. *CVPR* (2022). https://bmild.github.io/rawnerf/

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106. https://dl.acm.org/doi/abs/10.1145/3503250

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)* 41, 4 (2022), 1–15. https://dl.acm.org/doi/pdf/10.1145/3528223.3530127

Michael Oechsle, Songyou Peng, and Andreas Geiger. 2021. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 5589–5599. https://moechsle.github.io/unisurf/

Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. 2017. Colored point cloud registration revisited. In *Proceedings of the IEEE international conference on computer vision*. 143–152. http://www.open3d.org/docs/release/tutorial/pipelines/colored_pointcloud_registration.html

Ramesh Raskar, Kar-Han Tan, Rogerio Feris, Jingyi Yu, and Matthew Turk. 2004. Non-photorealistic camera: depth edge detection and stylized rendering using multi-flash imaging. *ACM transactions on graphics (TOG)* 23, 3 (2004), 679–688. https://dl.acm.org/doi/pdf/10.1145/1015706.1015779

RealityCapture. 2022. RealityCapture: 3D Models from Photos and/or Laser Scans. https://www.capturingreality.com/

Erik Reinhard, Michael Stark, Peter Shirley, and James Ferwerda. 2023. Photographic tone reproduction for digital images. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*. 661–670. https://dl.acm.org/doi/10.1145/566654.566575

Christian Reiser, Rick Szeliski, Dor Verbin, Pratul Srinivasan, Ben Mildenhall, Andreas Geiger, Jon Barron, and Peter Hedman. 2023. Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–12. https://creiser.github.io/merf/

Barbara Roessle, Jonathan T. Barron, Ben Mildenhall, Pratul P. Srinivasan, and Matthias Nießner. 2022. Dense Depth Priors for Neural Radiance Fields from Sparse Input Views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://barbararoessle.github.io/dense_depth_priors_nerf/

Erik Sandström, Yue Li, Luc Van Gool, and Martin R. Oswald. 2023. Point-SLAM: Dense Neural Point Cloud-based SLAM. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. https://github.com/eriksandstroem/Point-SLAM

Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. 2019. From Coarse to Fine: Robust Hierarchical Localization at Large Scale. In *CVPR*. https://github.com/cvg/Hierarchical-Localization

Carolin Schmitt, Božidar Antić, Andrei Neculai, Joo Ho Lee, and Andreas Geiger. 2023. Towards Scalable Multi-View Reconstruction of Geometry and Materials. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023). https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10251698

Carolin Schmitt, Simon Donne, Gernot Riegler, Vladlen Koltun, and Andreas Geiger. 2020. On Joint Estimation of Pose, Geometry and svBRDF From a Handheld Scanner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. https://openaccess.thecvf.com/content_CVPR_2020/html/Schmitt_On_Joint_Estimation_of_Pose_Geometry_and_svBRDF_From_a_CVPR_2020_paper.html

Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. 2016. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*. https://colmap.github.io/

Aarrushi Shandilya, Benjamin Attal, Christian Richardt, James Tompkin, and Matthew O'toole. 2023. Neural Fields for Structured Lighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 3512–3522. https://openaccess.thecvf.com/content/ICCV2023/html/Shandilya_Neural_Fields_for_Structured_Lighting_ICCV_2023_paper.html

Boxin Shi, Zhe Wu, Zhipeng Mo, Dinglong Duan, Sai-Kit Yeung, and Ping Tan. 2016. A benchmark dataset and evaluation for non-lambertian and uncalibrated photometric stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3707–3716. https://openaccess.thecvf.com/content_cvpr_2016/papers/Shi_A_Benchmark_Dataset_CVPR_2016_paper.pdf

Shining3D. 2023. EinScanSP. https://www.einscan.com/desktop-3d-scanners/

Sketchfab. 2024. Sketchfab: Online 3D geometry viewer. https://sketchfab.com/

Pratul P. Srinivasan, Stephan J. Garbin, Dor Verbin, Jonathan T. Barron, and Ben Mildenhall. 2023. Nuvo: Neural UV Mapping for Unruly 3D Representations. *arXiv* (2023). https://pratulsrinivasan.github.io/nuvo/

Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. 2019. The Replica Dataset: A Digital Replica of Indoor Spaces. *arXiv preprint arXiv:1906.05797* (2019). https://github.com/facebookresearch/Replica-Dataset

Jiaming Sun, Xi Chen, Qianqian Wang, Zhengqi Li, Hadar Averbuch-Elor, Xiaowei Zhou, and Noah Snavely. 2022. Neural 3d reconstruction in the wild. In *ACM SIGGRAPH 2022 Conference Proceedings*. 1–9. https://dl.acm.org/doi/abs/10.1145/3528233.3530718

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. 2020. Fourier features let networks learn high frequency functions in

low dimensional domains. *Advances in Neural Information Processing Systems* 33 (2020), 7537–7547. https://proceedings.neurips.cc/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf

Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. 2023. Nerfstudio: A Modular Framework for Neural Radiance Field Development. In *ACM SIGGRAPH 2023 Conference Proceedings (SIGGRAPH '23)*. https://docs.nerf.studio/

Marco Toschi, Riccardo De Matteo, Riccardo Spezialetti, Daniele De Gregorio, Luigi Di Stefano, and Samuele Salti. 2023. ReLight My NeRF: A Dataset for Novel View Synthesis and Relighting of Real World Objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 20762–20772. https://eyecan-ai.github.io/rene/

Shinji Umeyama. 1991. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 13, 04 (1991), 376–380. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=88573

Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. 2021. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689* (2021). https://proceedings.neurips.cc/paper/2021/file/e41e164f7485ec4a28741a2d0ea41c74-Paper.pdf

Zian Wang, Tianchang Shen, Merlin Nimier-David, Nicholas Sharp, Jun Gao, Alexander Keller, Sanja Fidler, Thomas Müller, and Zan Gojcic. 2023. Adaptive Shells for Efficient Neural Radiance Field Rendering. *ACM Trans. Graph.* 42, 6, Article 259 (2023), 15 pages. https://doi.org/10.1145/3618390

Sven Woop, Louis Feng, Ingo Wald, and Carsten Benthin. 2013. Embree ray tracing kernels for cpus and the xeon phi architecture. In *ACM SIGGRAPH 2013 Talks*. 1–1. https://dl.acm.org/doi/pdf/10.1145/2504459.2504515

Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. 2022. GMFlow: Learning Optical Flow via Global Matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8121–8130. https://github.com/autonomousvision/unimatch

Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. 2020. BlendedMVS: A Large-scale Dataset for Generalized Multi-view Stereo Networks. *Computer Vision and Pattern Recognition (CVPR)* (2020). https://github.com/YoYo000/BlendedMVS

Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. 2021. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems* 34 (2021), 4805–4815. https://lioryariv.github.io/volsdf/

Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P Srinivasan, Richard Szeliski, Jonathan T Barron, and Ben Mildenhall. 2023. BakedSDF: Meshing Neural SDFs for Real-Time View Synthesis. *arXiv preprint arXiv:2302.14859* (2023). https://bakedsdf.github.io/

Jonathan Young. 2024. Sketchfab: Online 3D geometry viewer. https://github.com/jpcy/xatlas

Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. 2022. MonoSDF: Exploring Monocular Geometric Cues for Neural Implicit Surface Reconstruction. *Advances in Neural Information Processing Systems (NeurIPS)* (2022). https://niujinshuchong.github.io/monosdf/

Ramin Zabih and John Woodfill. 1994. Non-parametric local transforms for computing visual correspondence. In *Computer Vision—ECCV'94: Third European Conference on Computer Vision Stockholm, Sweden, May 2–6 1994 Proceedings, Volume II 3*. Springer, 151–158. https://woodfill.com/Papers/Census94.pdf

ZDNet. 2023. https://www.zdnet.com/article/how-to-use-lidar-on-the-iphone-and-ipad/

Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. 2022. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5565–5574. https://openaccess.thecvf.com/content/CVPR2022/html/Zhang_IRON_Inverse_Rendering_by_Optimizing_Neural_SDFs_and_Materials_From_CVPR_2022_paper.html

Yinda Zhang, Neal Wadhwa, Sergio Orts-Escolano, Christian Häne, Sean Fanello, and Rahul Garg. 2020. Du 2 net: Learning depth estimation from dual-cameras and dual-pixels. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer, 582–598. https://augmentedperception.github.io/du2net/

Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. 2016. Fast global registration. In *European Conference on Computer Vision*. Springer, 766–782. https://link.springer.com/chapter/10.1007/978-3-319-46475-6_47

Zhenglong Zhou, Zhe Wu, and Ping Tan. 2013. Multi-view photometric stereo with spatially varying isotropic materials. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1482–1489. https://openaccess.thecvf.com/content_cvpr_2013/papers/Zhou_Multi-view_Photometric_Stereo_2013_CVPR_paper.pdf

Michael Zollhöfer, Angela Dai, Matthias Innmann, Chenglei Wu, Marc Stamminger, Christian Theobalt, and Matthias Nießner. 2015. Shading-based refinement on volumetric signed distance functions. *ACM Transactions on Graphics (ToG)* 34, 4 (2015), 1–14. https://dl.acm.org/doi/pdf/10.1145/2766887

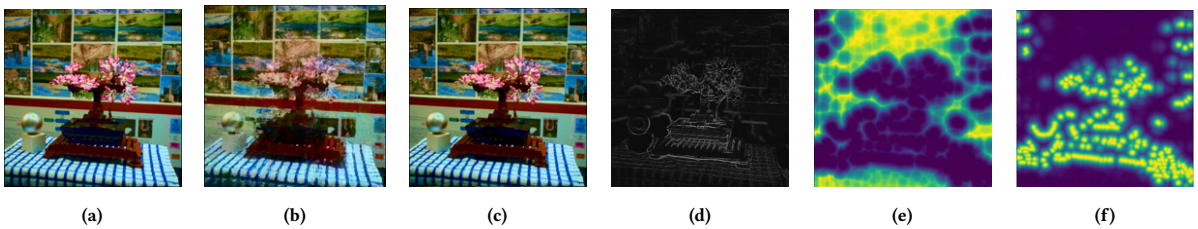


Fig. 5. Overview of our sampling process during training. Figure 5a is the ground truth test image. Figure 5b is the reconstruction of the test image after training has progressed 15% (15k gradient steps), Fig. 5c is the reconstruction of the test image at the end of training (100k gradient steps). Figure 5d denotes the per-pixel likelihoods of depth edges in the scene at the same view captured with our device. We note in Fig. 5b, the parts of scene with complicated geometry (foliage with many depth edges) have lower fidelity of appearance in the reconstruction at an earlier stage of training, which gradually improves in Fig. 5c. Figure 5e indicates the per-pixel sampling likelihood *if the test view were to be used for training* at a training progress of 10%, Fig. 5f indicates the same at a progress of 90%. Equation (6) is used to draw the samples: $\alpha = 0.1$ and 0.9 respectively for Figs. 5e and 5f. Brighter color indicates higher sampling likelihood.

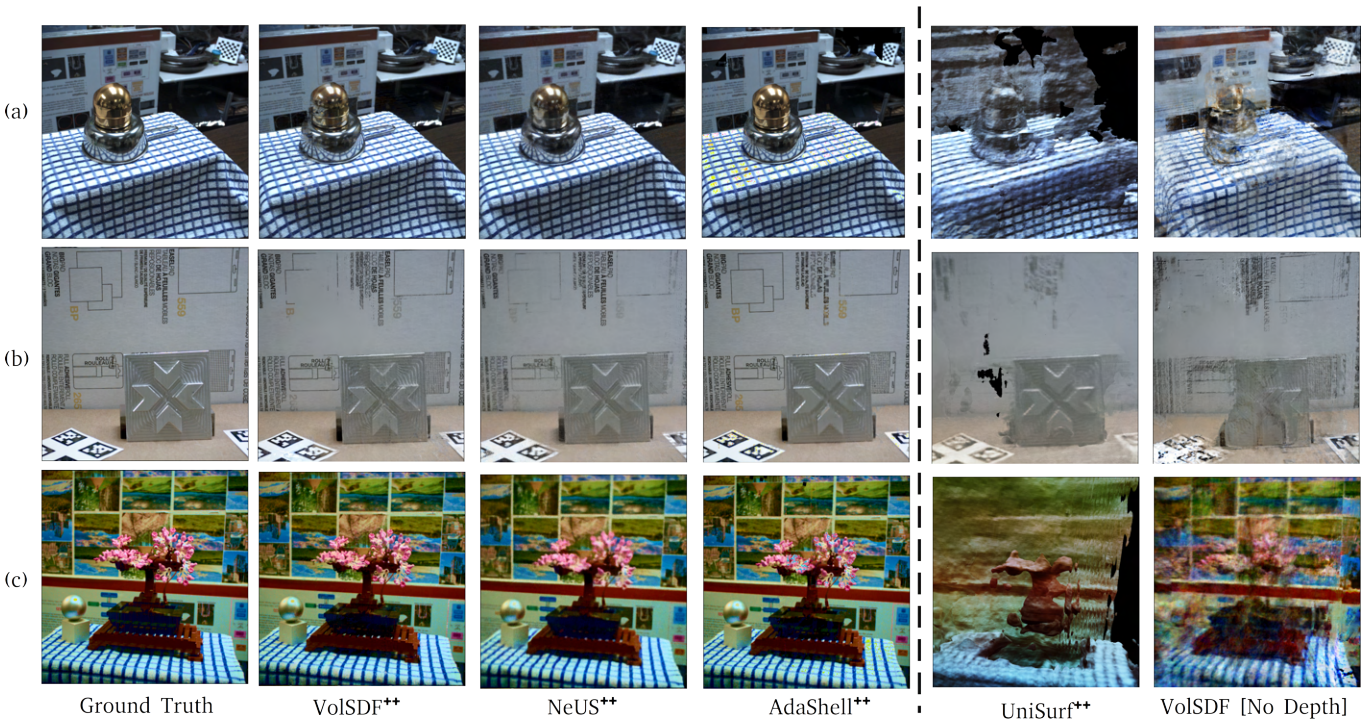


Fig. 6. Relative performance of all the methods. Details in Section 5.3, quantitative results in Table 5. Each method was allocated a budget of 100,000 gradient steps or fewer, and the enhancement in reconstruction exhibited a monotonically increasing trend until (and beyond) the cutoff.

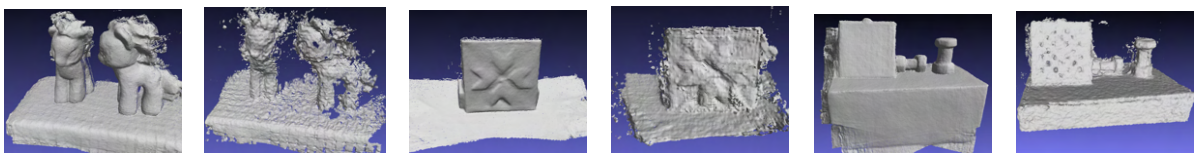


Fig. 7. AdaShell^{++}can work with noisy depth with little loss in view interpolation performance and training time. However, the surface recovered is also noisy. Figures 7a, 7c and 7e are the geometries recovered with no noise in depth and Figs. 7b, 7d and 7f are with noisy depths. Details in Section 5.4.

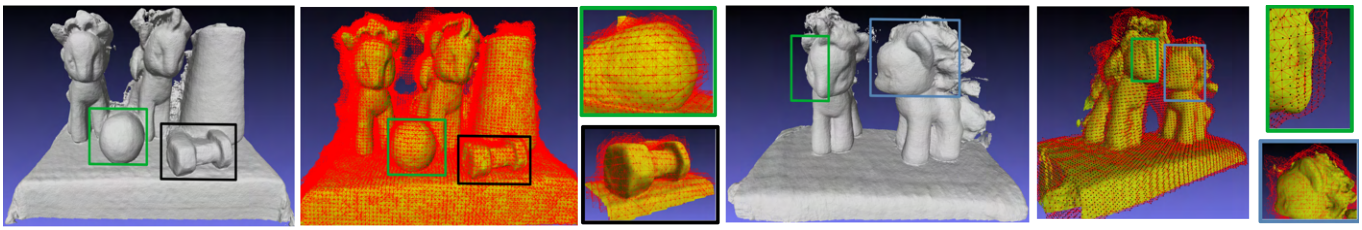


Fig. 8. AdaShell^{++}recovers sampling volumes similar to [Wang et al. 2023]. Figures a,d are the geometries recovered after optimization of Eq. (5) and is the starting point of AdaShell^{++}. Figures b, e display the sampling volumes around the starting geometry after AdaShell^{++}has converged. Details in Section 3.4.

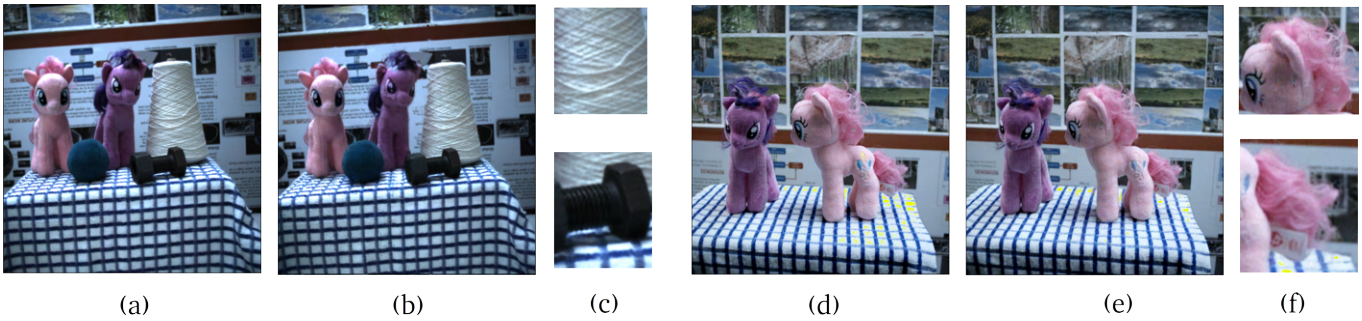


Fig. 9. Sampling reconstructions with AdaShell^{++}. Figures (a,d) are ground truth test images, (b,e) are reconstructed views, and (c,f) are zoomed in crops. Optimized geometries in Fig. 8(a,d). Both the scenes were trained on 9 and tested on 1 view for 30K gradient steps. For the diffuse scenes above, AdaShell^{++}recovered thin 'shells' around the estimated geometry, accelerating convergence and rendering.

(a) Limited parametrization **(b)** Full Disney BDRF parametrization

Fig. 10. Optimizing for the full Disney BRDF is difficult. Figure 10a shows our results with only specularity, roughness and metallic BRDF parameters. Figure 10b depicts identical results utilizing the complete range of Disney BRDF parameters as outlined in [Cheng et al. 2023]. Note the excessive glossy appearance of Fig. 10b due to the dominance of the clearcoat and clearcoat-gloss parameters. Details in appendix A, meshes rendered with [Sketchfab 2024].

(a) face_1 **(b)** face_2 **(c)** face_4 **(d)** shiny_1 **(e)** shiny_2

Fig. 11. Relighting scenes can be achieved with AdaShell^{++}trained with multi-illumination images. Discussion and results in Section 5.5 and Table 7. We capture 12 flash lit images for all the camera views and we use alternate flashes for all the training views (6 per view). Figures above show one (of six) flash configurations for the test view. Results of face_3 will be added after reviews. More results on the project website.

Supplementary Material: Incorporating dense depth into neural 3D representations for view synthesis and relighting

Submission ID: 704

This document inherits the figure, table, equation numbers, and references from the main document. Additional results may be viewed at **https://stereomfc.github.io**

A REPRESENTATIONS AND IMPLEMENTATION DETAILS

A.1 Model details

Our scene representation consists of two networks – an intrinsic network $\mathcal{N}(\theta)$ and an appearance network $\mathcal{A}(\phi)$. We follow [Li et al. 2023] to build and train $\mathcal{N}(\theta)$. We use 18 levels of hashgrid encodings [Müller et al. 2022] to encode the input and a two layer (128 neurons/layer) MLP to generate the intrinsic embedding. The first channel of the embedding, $\mathcal{S}(\theta)$ is trained with Eq. (5) to recover a signed distance field of the scene as described in Section 3.2. The rest of the 127 channels of the embedding $\mathcal{E}(\theta)$ are passed on to the appearance network $\mathcal{A}(\phi)$ as an input.

The appearance network takes $\mathcal{E}(\theta)$, the viewing direction (encoded with 6 levels of sinusoidal encodings following [Tancik et al. 2020]), and optionally the illumination direction (if recovering BRDF) to generate colors. The neural network is built with 2 layers of fully connected MLPs (128 neurons/layer) with skip connections.

VolSDF^{++} is our method similar to VolSDF [Yariv et al. 2021] and MonoSDF[Yu et al. 2022]. We represent the scene with $\mathcal{N}$ and $\mathcal{A}$ and train it with metric depth and color by minimizing Eq. (4). The samples for Eq. (3) are drawn using the "error-bounded sampler" introduced by [Yariv et al. 2021].

NeUS^{++} represents a modified version of NeUS [Wang et al. 2021], where we use the training schedule and structure of $\mathcal{N}$ from [Li et al. 2023], the appearance network $\mathcal{A}$ is adopted from NeUS and we optimize Eq. (5) along with Eq. (3). In addition to $\mathcal{A}$, NeUS^{++}also has a small 4 layer MLP (32 neurons per layer) to learn the radiance of the background as recommended in the original work by [Wang et al. 2021].

UniSurf^{++} is our method inspired by UniSurf[Oechsle et al. 2021]. We represent the scene's geometry using a pre-optimized implicit network $\mathcal{N}$ as outlined in Section 3.2. We follow the recommendations of [Oechsle et al. 2021] to optimize $\mathcal{A}$. UniSurf exposes a hyperparameter to bias sampling of Eq. (2) towards the current estimate of the surface. As we pre-optimize the surface, we can find the surface point $\mathbf{x}_s = \mathbf{o} + t_s\mathbf{d}$ through sphere tracing $\mathcal{S}$ along a ray. The intersection point t_s can then be used to generate N samples along the ray to optimize Eq. (3).

$$t_i = \mathcal{U}\left[t_s + \left(\frac{2i-2}{N} - 1\right)\Delta, t_s + \left(\frac{2i}{N} - 1\right)\Delta\right] \tag{7}$$

Equation (7) is the distribution used to draw samples and Δ is the hyperparameter that biases the samples to be close to the current surface estimate. We optimize $\mathcal{S}$ independent of Eq. (2) by just minimizing Eq. (5) with registered depth maps (see Section 3.2). We use this method to study the effects of volumetric rendering versus surface rendering. We found this strategy to be very sensitive to the hyperparameter Δ and its decay schedule as the training progressed. While best parameters for some sequences resulted in very quick convergence, they were very hard to come across and generally, poorer choices led to undesirable artifacts (see e.g. Fig. 3).

Finally, **AdaShell^{++}** is our method inspired by AdaptiveShells[Wang et al. 2023] and [Müller et al. 2022; Sun et al. 2022]. We warm start with a pre-optimized $\mathcal{S}$ by minimizing Eq. (5) in Section 3.2. We then enclose the surface represented by $\mathcal{S}$ in an isotropic voxel grid and progressively cull the voxels at a pre-set distance from the zero-level set of $\mathcal{S}$. This leaves us with a "shell" of voxels around (both inside and outside) the surface (zero-level set of $\mathcal{S}$) of the object which serves a similar purpose to the "shell" around the learned surface in [Wang et al. 2023]. We roughly follow [Sun et al. 2022] to generate samples along a segment of the ray guided by the voxels it intersects. The spatial density of samples is inversely proportional to their distance from the estimated surface. We implement this using the tools from NerfStudio[Li et al. 2022; Tancik et al. 2023]. Figure 8(a,d). Figure 8(b,e) denote the sampling volume as a wire frame around the estimated geometry (Fig. 8(a,d)) and Fig. 8(c,f) are zoomed in views of sections of the scene.

The "shells" shown in Fig. 8 and the shells recovered by [Wang et al. 2023] for small scenes are physically similar quantities. [Wang et al. 2023] dilate and erode the original level-set of the scene (approximated by $\mathcal{S}$) using a hyperparameter. Our "shells" are jointly estimated with the geometry as the training progresses. [Wang et al. 2023] estimate the fall-off of the volume density values along a ray to determine the hyperparameters, which in turn determines the thickness of the "shell". They subsequently use uniform sampling (similar to Eq. (7), where the Δ now denotes the local thickness of the shell) to generate samples for rendering. Our work takes a discrete approach by immersing the zero-level set (in form of pre-optimized $\mathcal{S}$) in a dense isotropic voxel grid and culling the voxels which have a lower volume density, according to a preset hyperparameter that determines the thickness of the shell. Once the shell has been estimated, we use a unbiased density weighted sampler (instead of a uniform sampler) to generate samples along the ray inside the shell. We expect our sampling strategy to be more robust to errors in estimated geometry (as shown in Section 5.4) than [Wang et al. 2023]. However, at the time of writing, an implementation of AdaptiveShells was not available to validate this claim.

A.2 Training Details

We ran our experiments on a Linux workstation with an Intel Core i9 processor, 64GB RAM, and an Nvidia RTX3090Ti graphics card with 25GB of vRAM. Across all the experiments for learning scene radiance, we implemented a hard cut-off of 100K gradient steps amounting to less than 4.5 hours of training time across all the experiments. The implementations of our baselines, design of the multi-flash camera system, dataset and the hyperparameters will be released in future. Figure 8 denotes the training steps graphically.

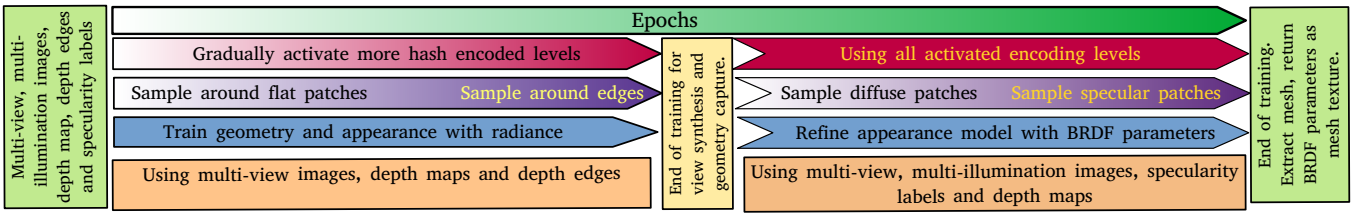


Fig. 12. Our approach to recovering 3D assets from captured data. In the first part, for NeUS^{++}and VolSDF^{++}we jointly optimize geometry and appearance by minimizing Eqs. (3) and (5). For AdaShell^{++}we first optimize Eq. (5) for a fixed number of gradient steps before the joint optimization. At this stage the geometry is optimized and appearance is recovered as radiance. Following this, we use multi-illumination images with a truncated BRDF parametrization to refine the appearance model, given the geometry, to learn the reflectance parameters.

A.3 Difference between our and prior work on neural scene understanding with depth

IGR[Gropp et al. 2020] were among the first to fit a neural surface to point samples of the surface. Our pipeline is largely inspired by that work. However, we have two main differences – we use a smaller network, and periodically activate multi-resolution hash encodings as recommended by [Li et al. 2023] instead of using a fully connected set of layers with skip connections. Additionally, as we have access to depth maps, we identify the variance of the neighborhood of a point on the surface through a sliding window filter. We use this local estimate of variance in a normal distribution to draw samples for $\mathbf{x}_s^{\Delta}$ along each ray. Our strategy assumes that image-space pixel neighbors are also world space neighbors, which is incorrect along the depth edges. However, as the Eikonal equation should be generally valid in $\mathbb{R}^3$ for $\mathcal{S}$, the incorrect samples do not cause substantial errors and only contribute as minor inefficiencies in the pipeline. A more physically based alternative, following [Gropp et al. 2020], would be executing nearest neighbor queries at each surface point along the rays to estimate the variance for sampling. With about 80k rays per batch, ~ 200K points in $(\mathbf{x}_s)$, and about 40k gradient steps executed till convergence, and a smaller network, our approach was more than two orders of magnitude faster than [Gropp et al. 2020], with no measurable decrease in accuracy of approximating the zero-level set of the surface.

NeuralRGBD[Azinović et al. 2022] is the closest prior work based on data needed for the pipeline and its output. The scene is reconstructed using color and aligned dense metric depth maps. The authors aggregate the depth maps as signed distance fields and use the signed distance field to calculate weights for cumulative radiance along samples on a ray (Eq. (2) in text). The weights are calculated with

$$w_i = \sigma\left(\frac{D_i}{tr}\right) \times \sigma\left(-\frac{D_i}{tr}\right) \tag{8}$$

where the D_i is the distance to the surface point along a ray, and the truncation tr denotes how fast the weights fall off away from the surface. Equation (8) yields surface biased weights with a variance controlled by the parameter tr. Notably, the depth map aggregation does not yield a learned sign distance field (no Eikonal regularizer in the loss). The authors also include a 'free-space' preserving loss to remove "floaters". As implemented, the pipeline needs the truncation factor to be selected per-scene. As the depth maps are implicitly averaged by a neural network, it is implicitly smoothed and therefore the pipeline is robust to local noise in the depth map.

MonoSDF[Yu et al. 2022] is mathematically the closest prior method to our work and it uses dense scene depths and normals obtained by a monocular depth and normal prediction network (OmniData[Eftekhar et al. 2021]). MonoSDF defines the ray length weighted with the scene density as the scene depth $\mathbf{d}_{pred}$ and minimizes

$$\ell_D = \sum_r ||\mathbf{w}\mathbf{d}_{mono} + \mathbf{q} - \mathbf{d}_{pred}||_2^2 \tag{9}$$

where $\{\mathbf{w}, \mathbf{q}\}$ are scale and shift parameters. Estimating an affine transformation on the monocular depth $\mathbf{d}_{mono}$ is important because in addition to gauge freedom $(\mathbf{w})$, monocular depths also have an affine degree of freedom $(\mathbf{q})$. The scale and shift can be solved using least squares to align $\mathbf{d}_{mono}$ and $\mathbf{d}_{pred}$. The scene normals are calculated as gradients of $\mathcal{S}$ weighted with scene density along a ray. Through a scale and shift invariant loss, MonoSDF calculates one set of $(\mathbf{w}, \mathbf{q})$ for all the rays in the batch corresponding to a single training RGBD tuple. In the earlier stages of the training, this loss helps the scene geometry converge. The underlying assumption is that there is an unique tuple $\{\mathbf{w}, \mathbf{q}\}$ per training image that aligns $\mathbf{d}_{mono}$ to the actual scene depth captured by the intrinsic network $\mathcal{N}$.

Our experiments with MonoSDF indicate that the network probably memorizes the set of $(\mathbf{w}, \mathbf{q})$ tuples per training image. Explicitly passing an unique scalar tied to the training image (e.g. image index as proposed in [Martin-Brualla et al. 2021]) speeds up convergence significantly. Success of MonoSDF in recovering both shape and appearance strongly depends on the quality of the monocular depth and normal predictions. Our experiments on using MonoSDF on the WildLight dataset([Cheng et al. 2023]) or the ReNe dataset ([Toschi et al. 2023]) failed because the pre-trained Omnidata models performed poorly on these datasets. Unfortunately, as implemented, MonoSDF also failed to reconstruct scene geometry when the angles between the training views were small – ReNe dataset views are maximally 45° apart. However, it demonstrates superior performance on the DTU and the BlendedMVS sequences while training with as low as three pre-selected views. Finally, our scenes were captured with a small depth of field and most of the background was out of focus, so the scene background depth was significantly

(a) AdaShell^{++} (b) VolSDF^{++} (c) NeUS^{++}

Fig. 13. All the pipelines can be used to extract the "base-color" of the scene. We calculate texture of the meshes from the radiance at convergence (PSNR 27.5+) for one of the scenes in Fig. 1. The textured meshes are rendered with MeshLab[Cignoni et al. 2008].

more noisy than the foreground depth. We sidestepped this problem by assigning a fixed 1m depth to all the pixels that were in the background. Although this depth mask simplifies our camera pose estimation problem (by segregating the foreground from the background), it assigns multiple infeasible depths to a single background point. As we aggregate the depth maps into the intrinsic network ($\mathcal{N}$) by minimizing Eq. (5), the network learns the mean (with some local smoothing) of the multiple depths assigned to the single background point. However, the scale and shift invariant loss is not robust to this and with masked depth maps, we could not reliably optimize MonoSDF on our sequences. We suspect that this is because the scale and shift estimates for each instance of Eq. (9) on the background points yielded very different results, de-stabilizing the optimization.

[Roessle et al. 2022] and **[Deng et al. 2022]** use sparse scene depth in the form of SfM triangulated points. [Roessle et al. 2022] use learnt spatial propagation [Cheng et al. 2019] to generate dense depth maps from the sparse depth obtained by projecting the world points triangulated by SfM. [Deng et al. 2022] assign the closest surface depth at a pixel obtained by projecting the triangulated points to the image plane. Neither of these pipelines recover a 3D representation of the scene and focus on view synthesis using few views.

[Sandström et al. 2023] introduce a novel 3D representation – "Neural point clouds" which includes geometric and appearance feature descriptors (small MLPs) grounded to a point in 3D. The geometry is recovered as the anchors of the "neural points". The appearance is calculated using a volumetric renderer which composits the outputs of the appearance descriptors of the neural points with the transmissivity of the neural points along the ray. The transmissivity of a neural point is calculated as a function of distances of a pre-set number of neighboring neural points.

A.4 Capturing approximate BRDF and generating textured meshes

Multi-illumination images captured by our camera system can be used to estimate surface reflectance properties. We recover a truncated Disney BRDF model([Burley 2012]. Our model consists of a per pixel specular albedo, a diffuse RGB albedo, and a roughness value to interpret the observed appearance under varying illumination. To estimate the spatially varying reflectance, we first train a model (AdaShell^{++}, VolSDF^{++}or NeUS^{++}) to convergence to learn the appearance as radiance. At convergence, the first channel $\mathcal{S}$ of the intrinsic network $\mathcal{N}$ encodes the geometry and the appearance network $\mathcal{A}$ encodes the radiance. We use two of the embedding channels of $\mathcal{E}$ to predict the roughness and specular albedo at every point on the scene. The diffuse albedo is obtained as the output of the converged appearance network $\mathcal{A}$. To calculate the appearance, we apply the shading model ([Burley 2012]) to calculate the color at every sample along a ray and volumetrically composite them using Eq. (2) to infer appearance as reflectance. Figure 12 describes our steps graphically. Optimizing for the full set of the Disney BRDF parameters, following [Cheng et al. 2023] did not work with our aproach as the optimization the optimization often got stuck at local minima. Figure 10b shows one instance of optimizing the pipeline of [Cheng et al. 2023], where the strengths of the recovered 'clearcoat' and 'clearcoat-gloss' parameters dominated over the optimization of the other parameters, resulting in a waxy appearance. Choosing a more conservative set of parameters (only 'base-color', 'specular' and 'roughness') in Fig. 10a led to a more realistic appearance.

Our process of generating texture and material properties roughly follows the methods described by [Cheng et al. 2023] and [Tancik et al. 2023]. We proceed through the following steps:

(1) At convergence (see Fig. 12), we extracted the scene geometry using the method described in [Mescheder et al. 2019].
(2) We calculate a depth mask by thresholding the depth images at every training view with an estimate of the scene depth to segregate the foreground from the background.
(3) Next, we cull the resulting triangular mesh (step 1) by projecting rays from every unmasked (foreground) pixel corresponding to all the camera views. This lets us extract the main subject of our scene as a mesh. We use Embree[Woop et al. 2013] to implement this.
(4) We generate texture coordinates on the culled mesh using "Smart UV Unwrap" function from [Blender 2024]. These results were qualitatively better than [Young 2024] and our re-implementation of [Srinivasan et al. 2023]. We then rasterize the culled mesh from step 3 to get points on the surface corresponding to the texture coordinates.
(5) We project each of these surface points back on to each of the training views to get the image coordinates. Rays originating from a rasterized surface point and intersecting the surface before reaching the camera are removed to preserve self occlusion.

(6) For all the valid projected points, we cast a ray onto the scene and use either AdaShell^{++}, VolSDF^{++}, or NeUS^{++}to generate the color at the pixel along the ray using Eq. (2). This is repeated for all the training views.
(7) At the end of the previous step we have several measurements of colors at every texture coordinate of the scene. We apply a median filter (per color channel) to choose the color – taking averages or maxima of the samples introduces artifacts. If using the radiance as texture is sufficient (often the case for diffuse scenes) this textured mesh can be exported. Figure 13 demonstrates using each of AdaShell^{++}, VolSDF^{++}and, NeUS^{++}to calculate the diffuse color of the scene in Fig. 1.
(8) To generate material textures, we follow the same procedures with the corresponding material channels after AdaShell^{++}, VolSDF^{++}or NeUS^{++}has been trained on multi illumination images using the schedule outlined in Fig. 12.
(9) The material properties are also volumetrically composited using Eq. (2) and median filtered like the base colors. This is different from just querying the value of the network at the estimated surface point in [Cheng et al. 2023].

We use [Sketchfab 2024], a web browser based tool that supports physically based rendering with the Disney BRDF parameters, and [Blender 2024] to generate the images in Figs. 1 and 10 respectively.

B A MULTI-FLASH STEREO CAMERA

We capture the scene using a binocular stereo camera pair with a ring of lights that can be flashed at high intensity. For our prototype, we use a pair of machine vision cameras ([FLIR 2024]) with a 1", 4MP CMOS imaging sensor of resolution of 2048 × 2048 pixels. As we focus mainly on small scenes, we use two sets of lenses that yield a narrow field of view – 12mm and 16mm fixed focal length lens ([EdmundOptics 2024]). We use 80W 5600K white LEDs ([CREE 2024]) flashed by a high-current DC power supply switched though MOSFETs controlled with an Arduino microcontroller. At each pose of our rig, we captured 12 images with each of the flash lights on (one light at a time) and one HDR image per camera. The cameras are configured to return a 12 bit Bayer image which is then de-Bayered to yield a 16 bit RGB image.

For the HDR images, we performed a sweep of exposures from the sensor's maximum (22580 microseconds) in 8 stops and used [Mertens et al. 2007] to fuse the exposures captured with ambient illumination (fluorescent light panels in a room). Following the recommendations of [Jensen et al. 2014] we used an f-stop of 2.8 to ensure the whole scene is in the depth of field of the sensors. We found the recommendations from [Mildenhall et al. 2022] to be incompatible with our pipeline, so we used Reinhard tone-mapping ([Reinhard et al. 2023]) to re-interpret the HDR images. Our image localization pipeline, and stereo matching also worked better with tonemapped images.

We set the left and right cameras to be triggered simultaneously by an external synchronization signal. We configured the camera frame acquisition and the illumination control programs to run in the same thread and synchronized the frame acquisition with the flashes through blocking function calls. Figure 4 presents a schematic of our prototype device.

Through experiments we observed that the vignetting at the edges of the frames were detrimental to the quality of reconstruction, so we only binned the central 1536 × 1536 pixels. A 16bit 1536×1536 frame saved as a PNG image was often larget than 10MB. To achieve a faster capture and training time without sacrificing the field of view, we down sampled the images to a resolution of 768 × 768 pixels for our experiments. Centered crops of our initial larger frames lead to failures of our pose-estimation pipelines due to the field of view being too narrow(Section 4.2), so we chose to down sample the images instead. For the images lit by a single LED, we used the camera's auto exposure function to calculate an admissible exposure for the scene and used 80% of the calculated exposure time for imaging – the built-in auto-exposure algorithm tended to over-expose the images a bit. Estimating the exposure takes about 2 seconds. Once the exposure value is calculated, it is used for all of the 12 flashes for each camera.

Several instances of these RGBD tuples are collected and the colored depth maps are registered in the 3D space in two stages – first coarsely using FGR [Zhou et al. 2016] and then refined by optimizing a pose graph[Choi et al. 2015]. At the end of this global registration and odometry step, we retain a reprojection error of about 5 - 10 pixels. If the reprojection errors are not addressed, they will cause the final assets to have smudged color textures. To address it, we independently align the color images using image-feature based alignment techniques common in multi-view stereo ([Sarlin et al. 2019; Schönberger et al. 2016]), so that a sub 1 pixel mean squared reprojection error is attained. The cameras aligned in the image-space are then robustly transformed to the world space poses using RANSAC[Fischler and Bolles 1981] with Umeyama-Kabsch's algorithm[Umeyama 1991]. Finally, we mask out the specular parts of the aligned images and use ColorICP [Park et al. 2017] to refine the poses. The final refinement step helps remove any small offset in the camera poses introduced by the robust alignment step. A subset of the data collected can be viewed on the project website.

B.1 Identifying pixels along depth edges

To identify pixels along depth edges, we follow [Raskar et al. 2004] and derive per-pixel likelihoods of depth edges. Assuming that the flashes are point light sources and the scene is Lambertian, we can model the observed image intensity for the k^{th} light illuminating a point $\mathbf{x}$ with reflectance $\rho(\mathbf{x})$ on the object as

$$\mathbf{I}_k(\mathbf{x}) = \mu_k \rho(\mathbf{x}) \langle \mathbf{l}_k(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle \tag{10}$$

where μ_k is the intensity of the k^{th} source and $\mathbf{l}_k(\mathbf{x})$ is the normalized light vector at the surface point. $\mathbf{I}_k(\mathbf{x})$ is the image with the ambient component removed. With this, we can calculate a ratio image across all the illumination sources

$$\mathbf{R}(\mathbf{x}) = \frac{\mathbf{I}_k(\mathbf{x})}{\mathbf{I}_{max}(\mathbf{x})} = \frac{\mu_k \langle \mathbf{l}_k(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle}{\max_i(\mu_i \langle \mathbf{l}_i(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle)} \tag{11}$$

It is clear that the ratio image $\mathbf{R}(\mathbf{x})$ of a surface point is exclusively a function of the local geometry. As the light source to camera baselines are much smaller than the camera to scene distance, except for a few detached shadows and inter-reflections, the ratio images (Eq. (11)) are more sensitive to the variations in geometry than

any other parameters. We exploit this effect to look for pixels with largest change in intensity along the direction of the epipolar line between the camera and the light source on the image. This yields a per-light confidence value of whether $\mathbf{x}$ is located on a depth edge or not. Across all 12 illumination sources, we extract the maximum values of the confidences as the depth edge maps. Unlike [Raskar et al. 2004], we use 12 illumination sources 30° apart, and we do not threshold the confidence values to extract a binary edge map. This lets us extract more edges especially for our narrow depth of field imaging system and gets rid of hyper parameters used for thresholding and connecting the edges.

Often parts of our scene violate the assumption of Lambertian reflectances resulting in spurious depth edges. When we use depth edges for sampling, these errors do not affect the accuracy of our pipeline. When using depth edges for enhancing stereo matching (Section 4.1) we ensure that the stereo pairs do not contain too many of these spurious edge labels to introduce noise in our depth maps.

B.2 Identifying patches with non-Lambertian reflectances

We modified the definition of differential images in the context of near-field photometric stereo introduced by [Chandraker et al. 2012; Liu et al. 2018] to identify non-Lambertian patches. Assuming uniform Lambertian reflectances, Eq. (10) can be expanded as

$$\mathbf{I}_k(\mathbf{x}) = \mu_k^* \rho(\mathbf{x}) \mathbf{n}(\mathbf{x})^T \frac{\mathbf{s}_k - \mathbf{x}}{|\mathbf{s}_k - \mathbf{x}|^3} \tag{12}$$

where $\mathbf{s}_k$ is the location and μ_k^* is the power of the k^{th} light source. We define the differential images as $\mathbf{I}_t = \frac{\partial \mathbf{I}}{\partial \mathbf{s}} \mathbf{s}_t$ where, $\mathbf{s}_t = \frac{\partial \mathbf{s}}{\partial t}$, which when applied to Eq. (12) can be expanded as

$$\mathbf{I}_t(\mathbf{x}) = \mathbf{I}(\mathbf{x}) \frac{\mathbf{n}^T \mathbf{s}_t}{\mathbf{n}^T (\mathbf{s} - \mathbf{x})} - 3\mathbf{I}(\mathbf{x}) \frac{(\mathbf{s} - \mathbf{x})^T \mathbf{s}_t}{|\mathbf{s} - \mathbf{x}|^2} \tag{13}$$

Observing that the light sources move in a circle around the center of projection on the imaging plane, $\mathbf{s}^T \mathbf{s}_t = 0$. Also, the second term of Eq. (13) is exceedingly small given that the plane spanned by $\mathbf{s}_t$ is parallel to the imaging plane and our choice of lenses limit the field of view of the cameras. The second term is further attenuated by the denominator $|\mathbf{s} - \mathbf{x}|^2$ because the camera-to-light baselines ($\mathbf{s}$) are at least an order of magnitude smaller than the camera to object distance ($\mathbf{x}$). As a result, under isotropic reflectances (Lambertian assumed for this analysis) the differential images $\mathbf{I}_t(\mathbf{x})$ are invariant to circular light motions. Any observed variance therefore can be attributed to the violations of our isotropic BRDF assumptions. We identify specular patches by measuring the variance of this quantity across the 12 instances of the flashlit images.

Although our pipelines for identifying depth edges and patches of varying appearances demonstrate satisfactory qualitative performance, sometimes they yield wrong labels because Eqs. (11) and (13) do not include additional terms for spatially varying BRDFs and interreflections respectively. These errors do not have any significant effect in our reconstruction pipeline as we use this information to generate samples during different phases of training to minimize photometric losses and we do not directly infer shape or reflectances from these steps.

B.3 Difference between [Feris et al. 2005; Raskar et al. 2004] and our hardware

[Raskar et al. 2004] was the first to propose pairing flashes with cameras and laid the groundwork for identifying depth edges from multi-flash images from a single viewpoint. However, [Raskar et al. 2004] considered a monocular camera and only four flashes along the horizontal and vertical directions of the camera in the demonstrated device. Researchers (see e.g. [Chaudhury et al. 2024]) have since extended it by placing multiple light sources far apart from a monocular camera and have demonstrated locating depth edges on objects with strictly Lambertian reflectances. In this work, we retain the original light and camera configuration from [Raskar et al. 2004] and increase the number of lights from four to 12.

[Feris et al. 2005] also investigated a stereo camera in a multi-flash configuration aimed at edge preserving stereo depth maps. They do not extend the application to synthesizing geometry or appearance by capturing and assimilating multiple views of the scene. For obtaining stereo depth maps, we use [Xu et al. 2022], which performs much better than conventional stereo matching ([Hirschmuller 2005; Zabih and Woodfill 1994]) largely deployed in off-the shelf systems ([Keselman et al. 2017]).

Both [Feris et al. 2004; Raskar et al. 2004] discuss methods to detect specularities (termed "material edges") through different transforms of the multi-light images. However, we achieve a more continuous circular motion of the lights around the cameras, so we choose to use the photometric invariants described by [Chandraker et al. 2012] instead.