# A Multi-flash Stereo Camera for Photo-realistic Capture of Small Scenes

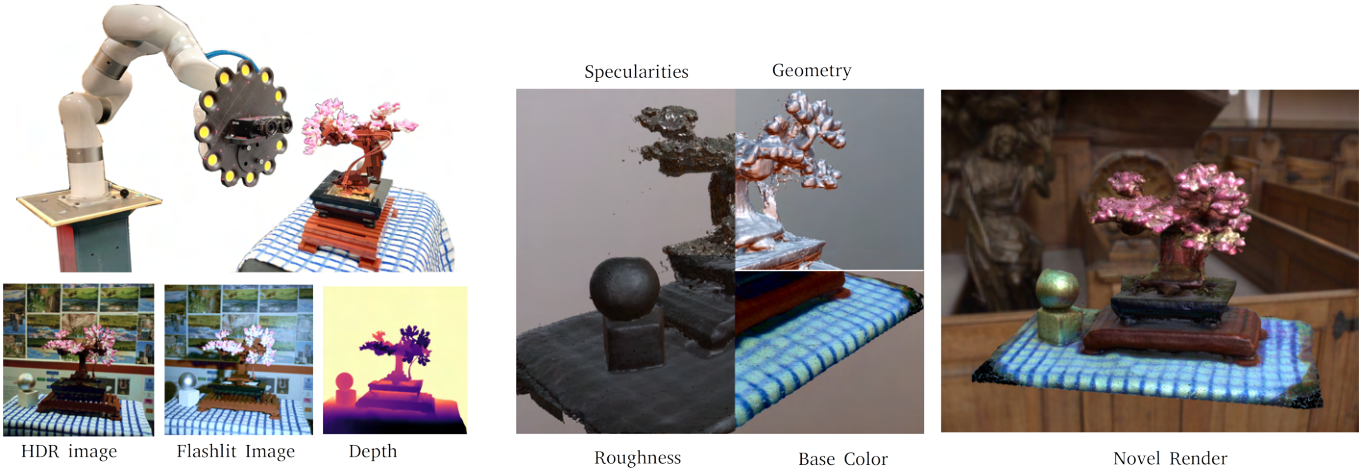



**Fig. 1. We present an applied approach to the photo-realistic capture of small scenes** using a multi-flash stereo camera and recent neural 3D scene understanding pipelines. Users often encounter restrictions on the number of views they can capture for a small scene. This quantity may be insufficient for conventional or modern view synthesis methods to function effectively. However, having scene depth proves beneficial in such cases. To that end, we introduce a technique that can be easily incorporated to use current state-of-the-art 3D view interpolation with metric depth. Additionally we demonstrate a device to capture multi-view color, depth and multi-illumination images and generate portable, photo-realistic 3D assets from a few instances of the captured data. The reconstruction of this scene was generated with only 11 stereo pairs captured with a robot mounted multi-flash stereo camera rig.

Automating the synthesis of photo-realistic digital twins of small scenes such as objects on a table is an active area of research with compelling use cases in gaming, content creation, virtual reality and robotics. Due to the configuration of the scene and capture system, it is often difficult to capture diverse views for the synthesis – especially for robotics applications. Out of the box, several of the modern neural 3D scene understanding pipelines are incompatible with, or perform poorly with limited viewpoint diversity. Images acquired at the same camera pose with varying illumination further deteriorates the quality of the results. We demonstrate a multi-flash stereo camera system for capturing geometry and approximate spatially varying reflectance of small scenes. Using a binocular stereo camera, we acquire an estimate of the metric shape of the object, while multiple flash lights around the cameras accentuate the depth edges and capture the illumination dependent appearance of the scene. A small number of these instances are fused and refined using modified versions of recent neural 3D shape representations to obtain a portable photo-realistic representation of the scene in the form or a volumetric representation or a textured mesh. Through this work, we provide an analysis of exporting assets from small scenes, and propose a 'drop-in' modification to three recent neural 3D scene understanding pipelines to work with the data collected by our system. Additionally, we open-source the design of our system, the capture pipeline, and a data set of diverse small scenes captured with our device.

## 1 INTRODUCTION

Multi-view 3D reconstruction and view synthesis is a fundamental problem in computer vision 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]. 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 (e.g. [Art 2023]). However, with renewed excitement around virtual reality, enthusiast level 3D photogrammetry, especially for small or tabletop scenes, has been supercharged by more capable cellphone cameras ([ZDNet 2023; Zhang et al. 2020]), and toolboxes like NerfStudio[Tancik et al. 2023] and RealityCapture. A subset of the new toolboxes and cutting edge solutions in the literature are geared towards view synthesis where the focus is rendering quality rather than accurate scene geometry, especially when a very diverse set of training views are absent. This is due to the "shape-radiance ambiguity"[Kutulakos and Seitz 1999]–by effectively by decoupling the scene transmissivity estimation from the radiance prediction, neural scene representations[Fridovich-Keil et al. 2022; Mildenhall et al. 2021; Müller et al. 2022] are prone to estimating accurate color but poor shape reconstructions. By only reasoning about appearance as cumulative radiance weighted with the scene's transmissivity along the viewing direction, these methods can achieve very convincing view interpolation results, with the quality of estimated scene geometry (a derivative of scene transmissivity) improving with the diversity and number of training views. Researchers ([Wang et al. 2021; Yariv et al. 2021]) have also described methods to effectively decouple scene geometry and appearance by using a geometric back-end, paving the way towards physically based neural scene understanding [Brahimi et al. 2024; Cheng et al. 2023; Zhang et al. 2022]. The geometric back-end, often a neural network approximating the signed distance field of the scene, is jointly trained with a neural appearance model to represent the scene. Geometric back-ends are a natural choice for research on

Author's address:

including scene depth in the process of generating better scene representations from fewer views – [Azinović et al. 2022; Yu et al. 2022b] notably use dense depth priors whereas [Deng et al. 2022; Roessle et al. 2022] use true scene depth to supervise view synthesis.

Recognizing the effectiveness of scene depth in view-synthesis especially with fewer views, our work takes an applied approach to generate high fidelity portable 3D assets from bounded scenes from captured instances with a custom built multi-flash camera. We combine insights from classical computer vision systems – multi-flash cameras ([Feris et al. 2005; Raskar et al. 2004]), appearance clustering ([Feris et al. 2004; Koppal and Narasimhan 2006; Liu et al. 2018]) with recent deep learning based stereo matchers ([Xu et al. 2022]) and neural scene understanding pipelines and propose a system to capture multi-illumination images of a bounded scene and generate textured 3D meshes and a volumetric view interpolator from the captured data. In this work,

(1) we present a "drop-in" modification to three current state of the art view synthesis pipelines with geometric back-ends : VolSDF[Yariv et al. 2021], NeuralAngelo[Li et al. 2023], and AdaptiveShells[Wang et al. 2023]), enabling them to use metric depth, accelerating their convergence and view synthesis, using only a handful of training views.
(2) Inspired by [Feris et al. 2005; Raskar et al. 2004], we design a multi-flash stereo camera system to capture multi-view, multi-illumination images of small scenes, and show how information captured by the system may be used to jointly refine appearance and geometry from the captures.

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 effects. Additionally, even with arbitrary levels of discretization, they often result in the smoothing of texture and surfaces due to data association relying on weighted averages along the object surface.

Recent neural 3D scene understanding pipelines (e.g. [Li et al. 2023; Mildenhall et al. 2021; 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. Together with continuous models, reasoning about appearance along the direction of the rays and high frequency preserving embeddings[Tancik et al. 2020], these pipelines serve as highly capable view interpolators by reliably preserving view dependent appearance and minute geometric details. Prior work has improved upon this by including additional geometric priors in the form of monocular depth supervision ([Yu et al. 2022b]), sparse depth supervision from structure-from-motion toolboxes ([Sun et al. 2022]), dense depth maps ([Azinović et al. 2022]), 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, especially when operating with a small number of available views. [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 ([Keselman et al. 2017]) and improve view synthesis with as few as ten training 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 assumed reflectances ([Chaudhury et al. 2024; Gotardo et al. 2015]), and capturing geometry and reflectance of objects by refining multi-view color, depth and multi-illumination images ([Cheng et al. 2023; Schmitt et al. 2023, 2020]). Our work also pairs illumination sources with stereo cameras to capture multiple supervision signals from the scene.

**Portability** of the 3D representations is also an important aspect and research on this area has taken two distinct directions in the pursuit of a common goal – high fidelity rendering of the scene delivered at interactive rates on consumer devices. Volumetric representations([Hedman et al. 2021; Reiser et al. 2021; Yu et al. 2021]) need a custom back-end hosted through a web server and recent iterations ([Duckworth et al. 2023; Reiser et al. 2023; Wang et al. 2023]) achieve upwards of 120FPS at full HD resolution on consumer hardware. On the other hand, representations based on geometric primitives e.g. Gaussians[Kerbl et al. 2023] and triangular meshes ([Chen et al. 2023; Cheng et al. 2023; Yariv et al. 2023]) exploit the simplicity of the primitives for rendering content. We chose triangular meshes for rendering and exporting our 3D scene models given the mature set of accelerators available ([Parker et al. 2010; Woop et al. 2013]). For view interpolation, we use a volumetric rendering pipeline inspired by [Wang et al. 2023].

## 3 METHODS

We synthesize novel views of a small scene with a small set of views. To do that, our approach decouples appearance and geometry during capture, by using metric depth from stereo as an estimate of the geometry and the color image as the source of appearance models. We describe our method of incorporating dense metric depth in Section 3.1. We discuss the effects of jointly optimizing shape and appearance of a scene in Section 3.2 and identify a few ambiguous cases that can be resolved with additional supervision signals. We briefly describe our system to capture the necessary measurements of the scene in Section 3.3 and finally in Section 3.4, we describe a "drop-in" modification to current state of the art to incorporate the data collected by our system.


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

**Fig. 2. A snapshot of the data collected by our system.** Figure 2a shows the color image captured by the right camera. We capture a high dynamic range image [Mertens et al. 2007] and display it after tonemapping [Reinhard et al. 2023]. Figure 2b shows the scene depth captured by matching the left and the right stereo pairs – we use GMFlow[Xu et al. 2022] to calculate the disparities, the inset scale is in mm. Figure 2c displays the likelihood of each pixel being on a depth edge on the object. Figure 2d shows the object surface normals calculated as a spatial gradient scene depth. We note that unlike conventional stereo matching ([Hirschmuller 2005; Zabih and Woodfill 1994]), [Xu et al. 2022] is returns locally smooth surfaces and ignores local texture variations but is also much 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 asymmetries of the textured aluminum plate in Fig. 2e is due to multiple bounce light paths.

### 3.1 Incorporating dense metric depth

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. Like prior work, we choose to represent the geometry of the scene with a signed distance field approximated with a neural network. Consequently, the first embedding channel of $\mathcal{N}$ is optimized to return a continuous value corresponding to the signed distance of a point from its nearest surface. We denote the function approximated by the first embedding channel of $\mathcal{N}(\theta)$ as $\mathcal{S}(\theta) : \mathbb{R}^3 \rightarrow \mathbb{R}$. The surface of the object, therefore, is learned as the zero-level set of $\mathcal{S}$ – i.e. for all surface points $\mathbf{x}_s \in \mathbb{R}^3 \mid \mathcal{S}(\mathbf{x}_s|\theta) = 0$. Prior work has jointly learnt $\mathcal{S}, \mathcal{N}, \mathcal{A}$ with only multi-view images, in contrast, we have access to estimates of true surface depth along any ray connecting $\mathbf{x}_s$ to camera poses $\mathbf{T}_i|i = 1, ..., N$ through intrinsics $\mathbf{C}_L$ and $\mathbf{C}_R$.

While the depth estimates can be directly used to optimize appearance and render surfaces, as recommended by [Dai et al. 2017; Oechsle et al. 2021; Zollhöfer et al. 2015], to avoid some pathological local minima discussed later, we elect to learn a continuous and locally smooth function that approximates the signed distance function of the surface $\mathbf{x}_s$. To do this, we follow [Gropp et al. 2020] and consider a loss function of the form

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

where, $\ell_{\mathbf{x}_s} = \frac{1}{N} \Sigma_{\forall \mathbf{x}} [\mathcal{S}(\mathbf{x}, \theta) + 1 - \langle \nabla_{\mathbf{x}} \mathcal{S}(\mathbf{x}, \theta), \mathbf{n}_x \rangle]$, 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. (1) 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. (1) 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 $\mathbf{x}_{nei}$ of each point in $\mathbf{x}_s$. [Gropp et al. 2020] identifies candidate $\mathbf{x}_{nei}$ through a nearest neighbor search, where as [Yariv et al. 2021] identifies $\mathbf{x}_{nei}$ 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 a point on the surface through a sliding window maximum filter. Finally, we adopt the scene density network and the training methodology from [Li et al. 2023] to optimize $\mathcal{S}(\theta)$.

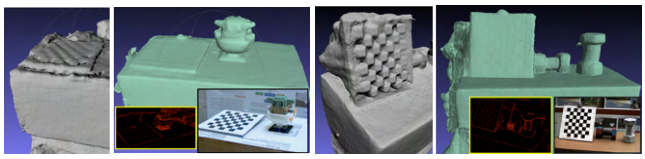
Our method is fundamentally different from prior work. It enables us to recover a geometric representation of the scene (as opposed to [Deng et al. 2022; Roessle et al. 2022]), is faster and more sample efficient than [Azinović et al. 2022], and more robust to local and global measurement errors than [Yu et al. 2022b]. We explain the details in appendix A.

**Table 1. We investigate a pathological case of learning shape and appearance jointly.** The numbers (lower is better) denote the RMS deviation of the reconstructed surface from a plane and have been normalized to a single metric scale. We note that surface based methods (AdaShell$^\dagger$, UniSurf$^\dagger$) perform worse than volumetric methods (VolSDF$^\dagger$, NeUS$^\dagger$), and except UniSurf$^\dagger$, all improve the quality of the surface measured with stereo. More details in Section 3.2, qualitative results in Fig. 3. Description of the modified methods ( $\cdot^\dagger$) in Section 3.4.

| Position | stereo | VolSDF$^\dagger$ | NeUS$^\dagger$ | AdaShell$^\dagger$ | UniSurf$^\dagger$ |
|---|---|---|---|---|---|
| horizontal (Fig. 3a) | 622 | 474 | 277 | 542 | 1321 |
| vertical (Fig. 3b) | 682 | 387 | 368 | 634 | 1602 |

### 3.2 Joint refinement of appearance and shape

Following recent work, we jointly optimize an implicit representation consisting of two neural networks – $\mathcal{N}(\theta)$ representing the implicit properties of the scene and $\mathcal{A}(\phi)$ representing the appearance of the scene using differentiable volumetric rendering. To render the color $\mathbf{C}$ of a single pixel of the scene at a target view with a camera centered at $\mathbf{o}$ and an outgoing ray direction $\mathbf{d}$, we calculate the ray corresponding to the pixel $\mathbf{r} = \mathbf{o} + t\mathbf{d}$, and sample a set of points $t_i$ along the ray. The networks $\mathcal{N}(\theta)$ and $\mathcal{A}(\phi)$ are then



(a) (b)

**Fig. 3. We demonstrate a pathological 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 by restricting the capacity of the model during training. Additionally, along the right insets, we provide an image used for training and the edge map used for sampling – we recommend zooming in or referring the project website for high resolution versions of this figure. Corresponding quantitative results in Table 1.

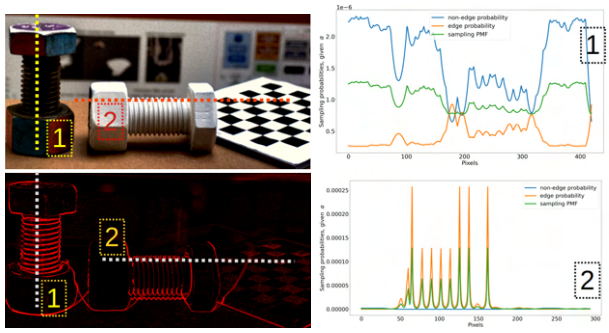
evaluated at all the $\mathbf{x}_i$ corresponding to $t_i$ and the per point color $\mathbf{c}_i$. The transmissivity $\tau_i$ is obtained and composited together using the quadrature approximation from [Max 1995] as:

$$\mathbf{C} = \sum_i \exp\left(-\sum_{j<i} \tau_j \delta_j\right)(1 - \exp(-\tau_j \delta_j))\mathbf{c}_i, \quad \delta_i = t_i - t_{i-1} \tag{2}$$

The neural implicit representations can then be trained jointly using a loss on the estimated and ground truth color $\mathbf{C}_{gt}$

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

We follow recent work and obtain the optical density (related to bulk transmissivity $\tau_i$) of the scene by transforming the value of the signed distance function $\mathcal{S}(\mathbf{x}_i, \theta)$ and jointly minimize the losses $\ell_D$ and $\ell_C$ in Eqs. (1) and (3) using stochastic gradient descent [Kingma and Ba 2014]. As the gradients of the loss functions $\ell_C$ and $\ell_D$ propagate through $\mathcal{A}$ and $\mathcal{N}$ (and $\mathcal{S}$ as it is part of $\mathcal{N}$) the appearance and geometry are learned together. Our 3D reconstruction pipeline can be considered adjacent to prior works ([Li et al. 2023; Sun et al. 2022; Wang et al. 2021; Yariv et al. 2021]). However, unlike those, we bias the optimization of $\mathcal{A}$ and $\mathcal{N}$ with metric depth captured using a stereo rig by incorporating $\ell_D$ (Eq. (1) and Section 3.1) in the optimization process. Although prior work shows the benefits of jointly refining shape and appearance, some pathological cases may arise when the scene has a large variation in appearance corresponding to a minimal variation in geometry. We investigate this effect by considering an extreme case – a checkerboard printed on matte paper where there is no geometric variation (all the texture is on a plane) corresponding to a maximum variation in appearance (white on black). We tested five methods to capture the appearance and geometry of the surface: color aligned depth maps through stereo and four modified versions of prior work. The qualitative results are presented in Fig. 3 and the quantitative results are presented in Table 1. We describe the modifications in Section 3.4. Through this experiment we learnt that the models do not have the inherent capacity to disambiguate between texture and geometric edges (depth discontinuities). Given enough iterations and capacity, the models will continue to jointly update geometry $\mathcal{N}$ and appearance $\mathcal{A}$ to minimize a perceptual loss (e.g. $L_1$ or MSE loss in RGB space), often




(a) (b)

**Fig. 4. We use geometric edge guided sampling** to train our model to disambiguate between depth and texture edges. Figure 4a shows two objects with geometric edges and texture edges, the top inset is the color image captures and the bottom inset shows each pixel identified with its likelihood of being a depth edge. To demonstrate the effect of our sampling strategy (Eq. (4)) we look at two rows of pixels **1** and **2** in Fig. 4a. Figure 4b shows the likelihood $P(p_i \in \mathbb{E})$ in **orange**, $P(p_i \notin \mathbb{E})$ in **blue** and $P(p_i|\alpha = 0.5)$ in **green**. Details in Section 3.2.

resulting in pathological cases (left insets in Fig. 3). However, by restricting the modelling capacity of $\mathcal{N}$ we can bypass this artifact and force the gradient updates to focus on $\mathcal{A}$ to minimize the perceptual loss. [Li et al. 2023] recognize this and provide an excellent set of hyperparameters and training curricula for well known datasets ([Jensen et al. 2014; Knapitsch et al. 2017]) based on sophisticated heuristics. As our hardware identifies areas with geometric edges, we opt to preferentially sample image patches with low variation on geometric features when the model capacity is lower (low number of hash encodings active), and focus on image patches with geometric edges when the model capacity has increased. Figure 6 pictorially represents our training curriculum, Fig. 4 describes our sampling procedure and Eq. (4) 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 $\alpha$ is a scalar ($\alpha \in [0, 1]$) proportional to the progress of the training.

$$P(p_i|\alpha) = (1 - \alpha)P(p_i \in \mathbb{E}) + \alpha P(p_i \notin \mathbb{E}) \tag{4}$$

To preserve the geometric nature of the edges, we use Euclidean distance transform ([Felzenszwalb and Huttenlocher 2012]) to calculate a smooth neighborhood around $\mathbb{E}$ identified by our multi-flash camera before applying Eq. (4).

### 3.3 A multi-flash stereo camera

So far, we have identified scene depth and depth edges as valuable signals, in addition to images. To capture all the supervision signals, we designed and fabricated a multi-flash stereo camera. We image the scene using two machine vision cameras with fixed focal length lenses. The cameras ($\mathbf{C}_L$ and $\mathbf{C}_R$) are synchronized with each other and also to each of the twelve flash lights ($L_{1:12}$) in a ring around them.

It is known from the literature (see e.g. [Liu et al. 2012; Raskar et al. 2004]) that under directional illumination located in the imaging

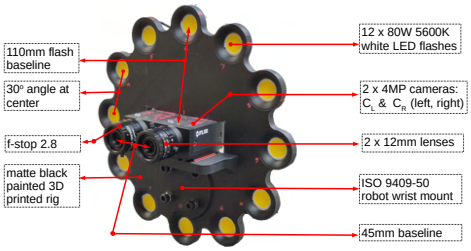



**Fig. 5. Schematic of our prototype system** used to capture the data in Fig. 2. Details of system components and data capture in Appendix B

plane, intensity variations due to depth edges are more prominent than intensity variations due to texture edges – we use this insight to identify the pixel locations of depth edges in the images. We use insights from [Chandraker et al. 2012; Koppal and Narasimhan 2006; Liu et al. 2018]) to identify areas with specular reflectances. We defer a detailed discussion of the choice of components and practical considerations in designing the multi-flash camera system to appendix B, Fig. 5 presents a schematic of our capture device and Fig. 2 presents a snapshot of the data collected by our device.

### 3.4 Representations

We consider four modified versions of the current state of the art methods for joint 3D shape and appearance learning. As noted by [Yu et al. 2022a], these methods mainly differ in how the ray samples are generated to calculate Eq. (2). The different methods of drawing samples along the rays can be distinguished by their degree of bias towards the current estimate of location of the surface. Surprisingly, all 'methods with a geometric back-end can be adopted to use metric depth by substituting the part of the pipelines estimating the geometry with the optimization of Eq. (1). We study the following four variations:

**Volumetric representations** ([Wang et al. 2021; Yariv et al. 2021]) sample from heavier tailed distributions to ensure enough variance in samples so that the minimization of Eq. (2) can escape low quality local minima at the cost of training and inference time. We present two modified versions of the current state of the art: NeUS$^\dagger$([Wang et al. 2021]) and VolSDF$^\dagger$([Yariv et al. 2021]) to explore the incorporation of dense metric depth.

**Surface based representations** on the other hand draw biased samples with lesser variance and demonstrate quicker convergence and rendering times. We study a modification of [Oechsle et al. 2021]: UniSurf$^\dagger$, where the bias can be controlled using a hyper-parameter. Alternatively, [Wang et al. 2023] recover an adaptive parameter that dictates the sampling bias to accelerate inference. As we measure object surface independently from appearance (through stereo depth), we can apply the same insights to accelerate training and inference. We call this modification AdaShell$^\dagger$ after the original work "Adaptive Shells".

Figure 6 graphically describes the general procedure we follow to capture scene representations, and we present the details of the four methods in appendix C. Additionally, with varying illumination and pre-optimized scene geometry, we can estimate material properties of the objects. We present our method for approximating spatially varying BRDF in appendix D.

## 4 RESULTS

### 4.1 Dataset

Although a data set is not the primary contribution of our research, our device captures some salient aspects of the scene that are not present in several established datasets. We identify these aspects in Table 2. 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 the location respectively. Under "illum. model" we note if an explicit illumination model is present per scene – we do not capture an environment illumination model, 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.

For each camera pose, we capture an HDR stereo pair, two depth maps, and two surface normals (as gradient of depth maps) aligned to the $C_L$ and $C_R$, two depth masks aligned to $C_L$ and $C_R$, two depth edge labels for the left and right frames, and two sets of 12 flash images using $L_{1:12}$ for $C_1$ and $C_2$. Several instances of these image sets are collected and the colored depth maps are registered in the 3D space in two staged – 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, which, if not addressed, will cause baked assets with smudged color textures. To address it, we independently align the 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 to 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.

### 4.2 Experiments

In this section we describe our experiments on the data collected by our system (Section 4.1) with the scene understanding pipelines we proposed in Section 3.4.

*4.2.1 Accuracy.* To measure the accuracy of our technique of incorporating metric depth, we reconstruct synthetic scenes with ground truth depth – in particular, we use the scenes curated by [Azinović et al. 2022]. We use 12-15 RGBD tuples to reconstruct the scenes and train for an average of 30k gradient steps (~1500 epochs) in about 75 minutes, in contrast to 300+ RGBD tuples and 9+ hours of training for [Azinović et al. 2022] on comparable hardware. Notably,

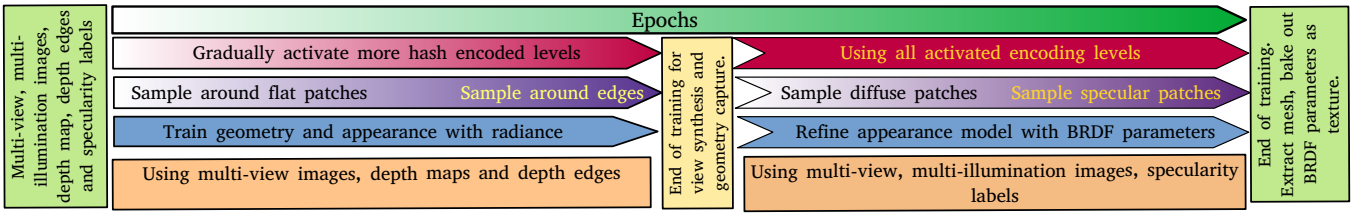



**Fig. 6. Our approach to recovering 3D assets from captured data**. In the first part, for NeUS$^\dagger$([Wang et al. 2021]) and VolSDF$^\dagger$([Yariv et al. 2021]) we jointly optimize geometry and appearance by minimizing Eqs. (1) and (3). For AdaShell$^\dagger$, we first optimize Eq. (1) for a 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. More details in appendix D.

**Table 2. We identify some differences between our dataset and a few established datasets**: BMVS[Yao et al. 2020], DTU[Jensen et al. 2014], DiLiGenT[Shi et al. 2016], DiLiGenT-MV[Shi et al. 2016] (both abbreviated as DGT*), PaNDoRa[Dave et al. 2022] and Open-Illumination[Liu et al. 2023a].

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

**Table 3. Accuracy of reconstruction.** We compare the accuracy of reconstructing the scene using VolSDF$^\dagger$ and NeUS$^\dagger$ with [Azinović et al. 2022] and [Dai et al. 2017]. We compare normalized chamfer distances (lower is better) across four synthetic scenes from [Azinović et al. 2022]. Scenes from BlendSwap.

| Scene | NeuralRGBD | BundleFusion | VolSDF$^\dagger$ | NeUS$^\dagger$ |
|---|---|---|---|---|
| greenroom | 0.013 | 0.024 | **0.012** | 0.016 |
| staircase | 0.045 | 0.091 | 0.020 | **0.009** |
| kitchen I | 0.252 | 0.234 | 0.047 | **0.036** |
| kitchen II | **0.032** | 0.089 | 0.060 | **0.032** |

[Azinović et al. 2022] also optimizes for noise in poses and reports metrics with ground truth poses in addition to optimized poses. We echo the best metric among these two. [Dai et al. 2017] registers the images themselves and for our experiments, we estimate the camera poses using the method described in Section 4.1. We present the quantitative results in Table 3. We replicate or out-perform the baselines by using a fraction of the training data and gradient steps, and among the methods discussed in Section 3.4, NeUS$^\dagger$ and VolSDF$^\dagger$ demonstrate similar performance, but, VolSDF$^\dagger$ converges about 1.25x faster than NeUS$^\dagger$.

*4.2.2 Training performance.* To identify the comparative performance of the methods proposed in Section 3.4, we test them under three scenes. Scene a (Fig. 7(a)) looks at a couple of specular objects with large variation in view dependent appearance. Additionally, there are large local errors in the captured depth maps due the specularities in the scene. We capture six stereo pairs and train on 11 images and test on one image. Scene b (Fig. 7(b)) features a rough metallic object in a very shallow depth of field captured by a 16mm lens (450 mm focal length). We capture four stereo pairs, train on seven images and test on on the remaining image. Scene c (Fig. 7) 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 4. We observe that VolSDF$^\dagger$ converges the fastest, NeUS$^\dagger$ recovers the highest quality geometry. If the primary consideration is not geometry (e.g., in the context of view interpolation alone), we observed that AdaShell performs best for scenes characterized by simpler geometries. We did not find ideal parameters for Unisurf$^\dagger$ to succeed on any of these sequences.

**Table 4. Speed of convergence.** 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). We report the numbers for scenes in Fig. 7(a,b,c) for each of the methods tested. As a pre-processing step for AdaShell$^\dagger$ and Unisurf$^\dagger$ we optimize Eq. (1) for 10k steps. We use exponentially moving average (EMA) with a smoothing factor of 0.99 to report the numbers. All of the trends were monotonically increasing till the cutoff at 100k steps. Cases of divergences of Unisurf$^\dagger$ marked with – – ––.

| PSNR | VolSDF$^\dagger$ | | | NeUS$^\dagger$ | | | AdaShell$^\dagger$ | | | Unisurf$^\dagger$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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+ | 25.4 | 23.2 | 94.6 | – – | – – | – – |

*4.2.3 View interpolation.* AdaShell$^\dagger$ uses the pre-computed scene geometry to learn a scene representation and combines the best of both surface rendering and volumetric rendering towards our goal of interpolating views. Starting with a pre-trained geometric backend obtained by minimizing Eq. (1), we draw heavily biased samples using the pre-computed surface to learn appearance as radiance. Due to the surface bias, the sampling process is efficient and needs few samples (as low as 20 samples per ray) to capture the appearance. Fig. 10 shows AdaShell$^\dagger$ capturing the finer details of the scene. However, our experiments (Table 1) indicate that the optimization deteriorates the quality of surface by a small but noticeable amount, by producing some high frequency artifacts.

*4.2.4 Using noisy depth.* To investigate the effects noise in the depth maps, we re-calculated the depths of some of the 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, and surface normals were calculated using the spatial gradients of the depth maps. These noisy depths and normals substituted the depths and normals calculated using learnt stereo in our original pipeline and the scenes were reconstructed with the noisy data. To isolate the performance of our pipelines, we did not filter the depth obtained from stereo. From the top of Table 5, we observe that NeUS$^{\dagger}$ strictly improves the quality of the surface reconstructed from just noisy stereo (row 1 and 2 versus row 3). Nevertheless, from the bottom of Table 5 we note that, if the end goal is just view synthesis, AdaShell$^{\dagger}$ performs equally well with smooth or noisy depth, indicating that photorealistic view synthesis is still possible with noisy depth data. However, unprocessed conventional stereo often introduces large local errors (see e.g. surface patterns on Fig. 8(b,f)) which our pipelines could not correct using a handful of views.

*4.2.5 Effect of supervision Signals.* We found the role of high quality depth from stereo as a supervision signal to be disproportionately influential in the success of our pipeline. It frequently took precedence over the subtle advantages of edge sampling (Section 3.2), particularly in the case of simpler geometries. Nevertheless, in the presence of noisy depth, the quality of the reconstructed surface was enhanced through edge-based sampling Section 3.2 and Eq. (4)). Our hierarchical sampling strategy allocated samples away from depth edges, where the noise was more prevalent, thus less gradient steps were spent modelling areas with higher noise. Table 5 presents the quantitative details of the experiment.

We discuss our experiment on recovering reflectance parameters and exporting assets in appendix D.

# 5 CONCLUSIONS

## 5.1 Discussions

Although we achieve state of the art results in view-synthesis, our approach struggles to represent transparent objects and accurately capture the geometry of reflective surfaces. [Liu et al. 2023b] address the problem with reflective objects by modelling background reflections and is based on the architecture proposed by [Wang et al. 2021]. As NeUS$^{\dagger}$ adapts the same back-end to use metric depth and can incorporate noisy depth estimates. Starting at Section 4.2.4, extending our work to accommodate shiny objects should be straightforward.

Our pipelines require metric depth to be functional, which ties it with our capture device. Future work will address enrollment of monocular depth priors, comparison between our method and [Yu et al. 2022b] in appendix A may be a good starting point in that direction. Incorporation of metric depth also introduces a strong bias, often limiting super resolution of geometry typically achieved in neural 3D scene representation (see e.g. [Li et al. 2023]). Limiting the extent of use of metric depth in our pipelines is expected to address this issue and is future work.

At the moment, calculating good exposure values during capture and color grading across the tonemapped HDR images and flash-lit images (e.g. bottom left insets in Fig. 1) is best done manually, per scene. If done improperly, this can significantly deteriorate the reconstruction quality. Modern smartphone cameras ([ZDNet 2023; Zhang et al. 2020]) address this issue through sophisticated software, while also capturing metric depth and can be used to collect data for our pipelines.

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.

**Table 5. 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$^{\dagger}$ on learnt stereo. Qualitative comparison in Fig. 8(b,d,f). Bottom: gradient steps (in 1000s lower is faster) required to surpass a test time accuracy of 27.5dB with AdaShell$^{\dagger}$. Like Table 4, we use EMA with a smoothing factor of 0.99 and the trends were monotonically increasing till (and beyond) the reported steps. We specify the count of views utilized in [ ] braces.

| condition | Fig. 8(a,b)[5] | Fig. 8(c,d)[7] | Fig. 8(e,f)[5] |
|---|---|---|---|
| edge sampling | 491 | 403 | 225 |
| no edge sampling | 593 | 419 | 251 |
| noisy stereo | 600 | 523 | 369 |
| 27.5+ PSNR with noise | 7.93 | 20.2 | 5.12 |
| 27.5+ PSNR without noise | 7.85 | 23.2 | 2.71 |

## 5.2 Conclusions

We presented a solution for automated capture of portable tabletop assets by proposing (1) a novel multi-flash camera system and (2) an implicit representation architecture that takes advantage of the unique data available to our multi-flash camera system. Compared to state-of-the-art implicit 3D representations, our system generates assets of higher quality both in terms of visual fidelity and shape reconstruction, while requiring lesser compute and data. We use the data collected with our system and generate photo-realistic portable representations of the scene in the form of volumetric renderings or a mesh with color and material properties as texture.

# REFERENCES


2023. ArtecLeo. https://www.artec3d.com/portable-3d-scanners/artec-leo

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/

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

Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. 2023. MobileNeRF: Exploiting the Polygon Rasterization Pipeline for Efficient Neural Field Rendering on Mobile Architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 16569–16578. https://openaccess.thecvf.com/content/CVPR2023/html/Chen_MobileNeRF_Exploiting_the_Polygon_Rasterization_Pipeline_for_Efficient_Neural_Field_CVPR_2023_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

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

Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. 2021. Baking Neural Radiance Fields for Real-Time View Synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 5875–5884. https://openaccess.thecvf.com/content/ICCV2021/html/Hedman_Baking_Neural_Radiance_Fields_for_Real-Time_View_Synthesis_ICCV_2021_paper.html

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

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

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics* 42, 4 (2023). https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

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

Sanjeev J Koppal and Srinivasa G Narasimhan. 2006. Clustering appearance for scene analysis. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Vol. 2. IEEE, 1323–1330. https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1640911

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/

Ming-Yu Liu, Oncel Tuzel, Ashok Veeraraghavan, Yuichi Taguchi, Tim K Marks, and Rama Chellappa. 2012. Fast object localization and pose estimation in heavy clutter for robotic bin picking. *The International Journal of Robotics Research* 31, 8 (2012), 951–973. https://journals.sagepub.com/doi/abs/10.1177/0278364911436018

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

Steven G Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, et al. 2010. Optix: a general purpose ray tracing engine. *Acm transactions on graphics (tog)* 29, 4 (2010), 1–13. https://dl.acm.org/doi/abs/10.1145/1778765.1778803

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, Songyou Peng, Yiyi Liao, and Andreas Geiger. 2021. KiloNeRF: Speeding Up Neural Radiance Fields With Thousands of Tiny MLPs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 14335–14345.

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/

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/

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

Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. 2021. PlenOctrees for Real-Time Rendering of Neural Radiance Fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 5752–5761. https://openaccess.thecvf.com/content/ICCV2021/html/Yu_PlenOctrees_for_Real-Time_Rendering_of_Neural_Radiance_Fields_ICCV_2021_paper.html

Zehao Yu, Anpei Chen, Bozidar Antic, Songyou Peng, Apratim Bhattacharyya, Michael Niemeyer, Siyu Tang, Torsten Sattler, and Andreas Geiger. 2022a. SDFStudio: A Unified Framework for Surface Reconstruction. https://github.com/autonomousvision/sdfstudio

Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. 2022b. 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

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

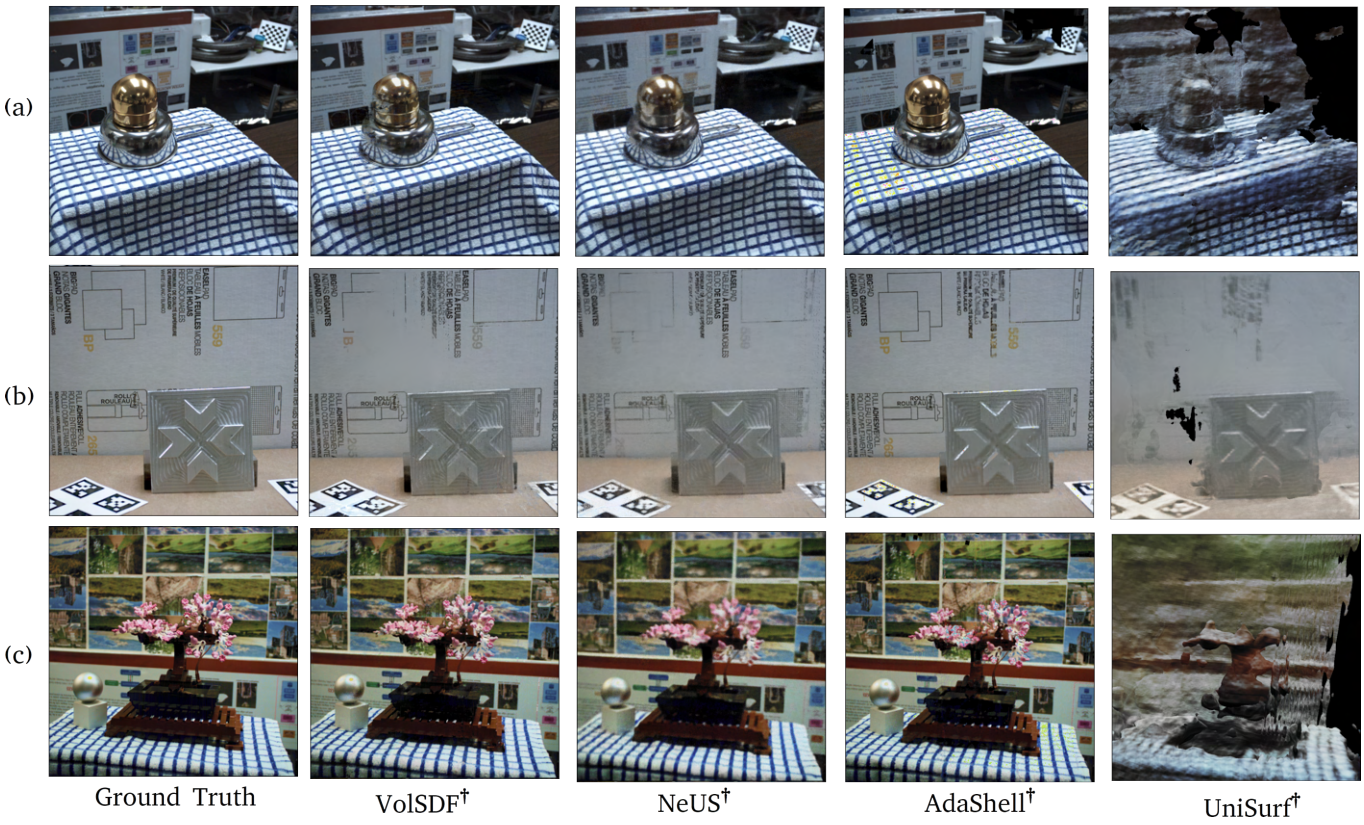



**Fig. 7. Relative performance of all the methods**. Details in Section 4.2.2, quantitative results in Table 4. Each method was allocated a budget of 100,000 gradient steps or fewer, and the enhancement in reconstruction exhibited a monotonically increasing trend until (and beyond) the cutoff.

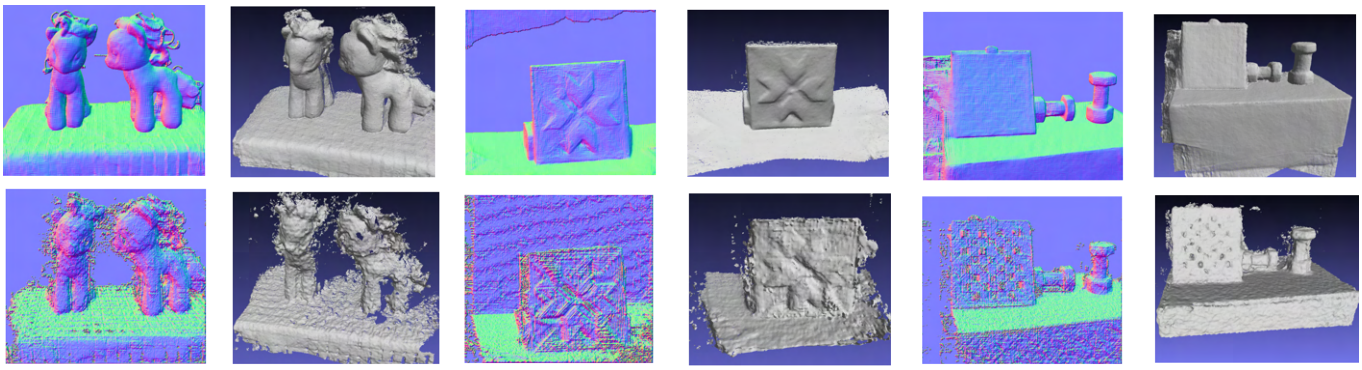



**Fig. 8. Using noisy data for reconstruction**. Descriptions in Sections 3.4 and 4.2.3. Top row denotes the surface normals and surfaces reconstructed using learnt stereo ([Xu et al. 2022], and bottom row denotes the same using conventional stereo matching. We present normals in figures a,c and, e instead of the surface depths to highlight the noise in the data.

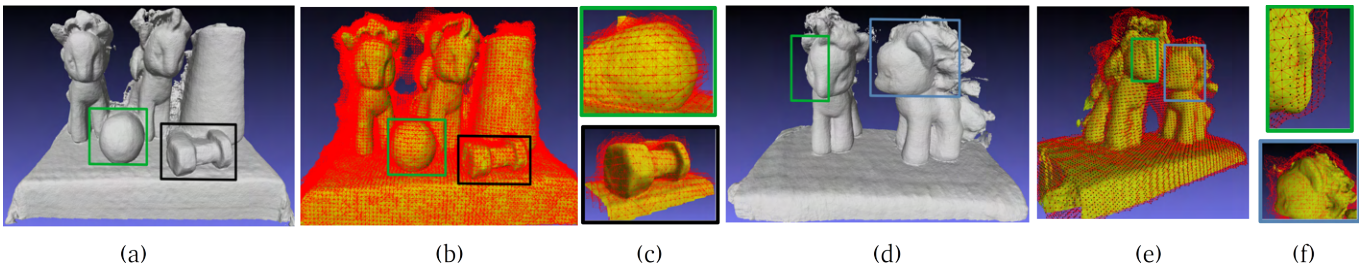



**Fig. 9. AdaShell† and Adaptive Shells [Wang et al. 2023] recover similar sampling volumes.** Figures a,d are the geometries recovered after optimization of Eq. (1) and is the starting point of AdaShell†. Figures b, e display the sampling volumes around the starting geometry after AdaShell† has converged. Details in appendix C.

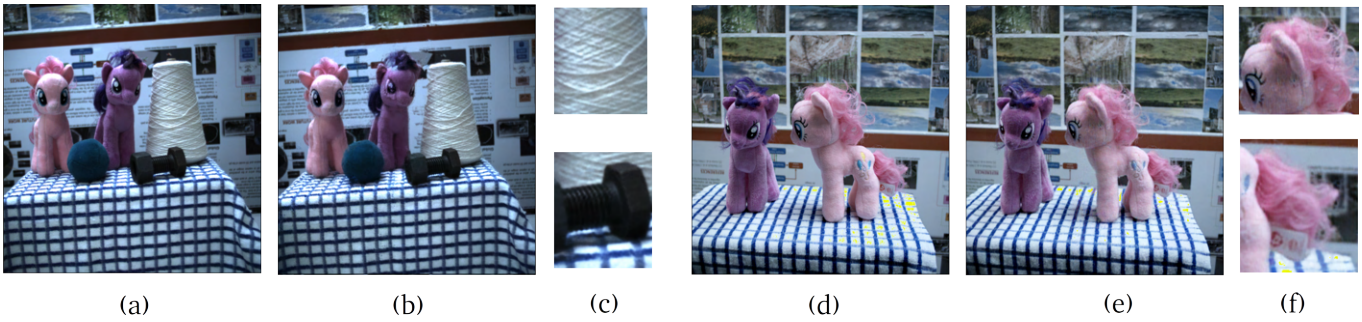



**Fig. 10. Sampling reconstructions with AdaShell†**. Details in Section 4.2.3. Figures (a,d) are ground truth test images, (b,e) are reconstructed views, and (c,f) are cropped and zoomed in sections. Starting geometries in Fig. 9(a,d). Both the scenes were trained on 9 and tested on 1 view for 30K gradient steps. For the diffuse scenes above, AdaShell† recovered thin 'shells' around the estimated geometry, accelerating convergence and rendering.

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

**Fig. 11. Optimizing for the full Disney BRDF is difficult**. Figure 11a shows our results with only specularity, roughness and metallic BRDF parameters. Figure 11b 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. 11b due to the dominance of the clearcoat and clearcoat-gloss parameters. Details in appendix D, meshes rendered with [Sketchfab 2024].

**(a)** VolSDF† **(b)** AdaShell† **(c)** NeUS†

**Fig. 12. All the pipelines** can be used to extract the "base-color" of the scene. We bake out the radiance as texture of the meshes at the end of the experiment in Table 4 for the scene in Fig. 1. Details in appendix D. The textured meshes are rendered with MeshLab[Cignoni et al. 2008].

# Supplementary Material: A Multi-flash Stereo Camera for Photo-realistic Capture of Small Scenes.

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 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 them. However, we have two man differences – we use a smaller network, and periodically activated multi-resolution hash encodings as recommended by [Li et al. 2023] instead of a fully connected cascade 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 maximum filter. We use this variance in a normal distribution to draw samples for $\mathbf{x}_{nei}$ at 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 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, $\sim$ 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 to us based on data needed for the pipeline and its outcome – 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

$$w_i = \sigma\left(\frac{D_i}{tr}\right) \times \sigma\left(-\frac{D_i}{tr}\right) \tag{5}$$

where the $D_i$ are 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 (5) yields surface biased weights and this is jointly trained with the color. 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. 2022b]** 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

$$\ell_D = \sum_r ||\mathbf{w}\mathbf{d}_{mono} + \mathbf{q} - \mathbf{d}_{pred}||_2^2 \tag{6}$$

where $\{\mathbf{w}, \mathbf{q}\}$ are scale and shift parameters, because the monocular depth $\mathbf{d}_{mono}$, in addition to gauge freedom ($\mathbf{w}$) also has 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 also calculated as scene density weighted spatial gradients of $\mathcal{S}$. Through a scale and shift invariant loss, MonoSDF calculates one set of $(\mathbf{w}, \mathbf{d})$ for all the rays in the batch and in the earlier stages of the training, this loss helps the scene geometry converge. The underlying assumption being, 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 per training image shift and scale – 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 almost exclusively 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 views were small (ReNe dataset views are maximally 45° apart) for the scenes we captured. 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 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. (1), 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 in this sense 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. (6) 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.

## B TECHNICAL DETAILS OF OUR HARDWARE

To capture data from the scene we integrate a binocular stereo camera pair with a ring of flash lights around them. 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]) as flashes driven by a DC power supply and switched though MOSFETs controlled with a Arduino over USB. 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. We set the left and right cameras to be triggered simultaneously by an external synchronization signal. We configured the camera frame acquiring and the flash triggering programs to run on the same thread and synchronized the frame acquisition with the flashes through blocking function calls. Figure 5 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. Additionally, a 16bit 1536 × 1536 frame saved as a PNG image were often upward of 10MB, so, to achieve a faster capture time and a faster training time using the captured images, 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 (Section 4.1), so we chose to down sample the images instead. For the flashlit images, we use the camera's auto exposure function to calculate an admissible exposure for the scene and use 80% of the calculated exposure time for imaging – the built-in auto-exposure algorithm tended to over-expose the images a bit. 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. 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 work better with tonemapped images.

**To identify pixels along depth edges**, we followed [Raskar et al. 2004] to 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 $k^{\text{th}}$ light illuminating a point $\mathbf{x}$ with reflectance $\rho(\mathbf{x})$ on the object as

$$\mathbf{I}_k(\mathbf{x}) = \mu_k \rho(\mathbf{x}) \langle \mathbf{l}_k(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle \tag{7}$$

where $\mu_k$ is the intensity of the $k^{\text{th}}$ 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

$$\mathbf{R}(\mathbf{x}) = \frac{\mathbf{I}_k(\mathbf{x})}{\mathbf{I}_{max}(\mathbf{x})} = \frac{\mu_k \langle \mathbf{l}_k(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle}{\max_i (\mu_i \langle \mathbf{l}_i(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle)} \tag{8}$$

It is clear that the ratio image $\mathbf{R}(\mathbf{x})$ of a surface point is exclusively a function of the local geometry and 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. (8)) is 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.

**To identify pixels 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]. Equation (7), assuming uniform Lambertian reflectances, can be expanded as

$$\mathbf{I}_k(\mathbf{x}) = \mu_k^* \rho(\mathbf{x}) \mathbf{n}(\mathbf{x})^T \frac{\mathbf{s}_k - \mathbf{x}}{|\mathbf{s}_k - \mathbf{x}|^3} \tag{9}$$

where $\mathbf{s}_k$ is the location and $\mu_k^*$ is the power of the $k^{\text{th}}$ 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. (9) can be expanded as

$$\mathbf{I}_t(\mathbf{x}) = \mathbf{I}(\mathbf{x}) \frac{\mathbf{n}^T \mathbf{s}_t}{\mathbf{n}^T (\mathbf{s} - \mathbf{x})} - 3\mathbf{I}(\mathbf{x}) \frac{(\mathbf{s} - \mathbf{x})^T \mathbf{s}_t}{|\mathbf{s} - \mathbf{x}|^2} \tag{10}$$

Observing that the light sources move in a circle around the center of projection on the imaging plane, $\mathbf{s}^T \mathbf{s}_t = 0$, and the second term of Eq. (10) 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 ($\mathbf{x}$). Therefore, under isotropic reflectances (Lambertian assumed for this analysis) the differential images $\mathbf{I}_t(\mathbf{x})$ are invariant to circular light motions – any variance 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. (8) and (10) 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.1 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 investigate a stereo camera in a multi-flash configuration aiming at edge preserving stereo depth maps, and do not extend the application to synthesizing views 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.

## C REPRESENTATIONS AND IMPLEMENTATION DETAILS

To ensure interoperability and modularity of our pipelines between the proposed architectures – VolSDF$^\dagger$, NeUS$^\dagger$, AdaShell$^\dagger$, and UniSurf$^\dagger$, we used a common intrinsic network $\mathcal{N}$ and appearance network $\mathcal{A}$. The intrinsic network $\mathcal{N}$ models the scene geometry and surface properties through its embedding channels. Across all the experiments, we used $\mathcal{N}$ with two fully connected layers of 128 neurons. Following NeuralAngelo [Li et al. 2023], we used 16-18 levels of hash encodings activated periodically. The gradients of the parameters of $\mathcal{N}$ were numerically calculated numerically (not with automatic differentiation) as recommended by [Li et al. 2023]. The appearance network $\mathcal{A}$ for learning scene radiance is inspired by MonoSDF([Yu et al. 2022b]) and comprises of two fully connected layers of 128 neurons and 4-6 orders of frequency encoding for the viewing directions. In addition to $\mathcal{A}$, NeUS$^\dagger$ 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].

We ran our experiments on a Linux workstation with an Intel Corei9 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.

**Unisurf**$^\dagger$ 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 3.1. 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 samples along the ray to optimize Eq. (3).

$$t_i = \mathcal{U}\left[t_s + \left(\frac{2i-2}{N} - 1\right)\Delta, t_s + \left(\frac{2i}{N} - 1\right)\Delta\right] \tag{11}$$

Equation (11) is the distribution used to draw samples and $\Delta$ is the hyperparameter that biases the samples to be cose to the current surface estimate. As we can optimize $\mathcal{S}$ independent of Eq. (2) by just minimizing Eq. (1) with registered depth maps (see Section 3.1), 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 $\Delta$ and its decay schedule as the training progressed. While best parameters for some sequences resulted in very quick convergence, poorer choices led to undesirable artifacts (see e.g. Fig. 3).

**AdaShell**$^\dagger$ is our method inspired by AdaptiveShells[Wang et al. 2023] and [Müller et al. 2022; Sun et al. 2022]. We start with a pre-optimized $\mathcal{S}$ by minimizing Eq. (1) in Section 3.1. We then immerse the surface represented by $\mathcal{S}$ in an isotropic voxel grid and pregressively cull the voxels at a adaptive 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]. Following [Sun et al. 2022] we then generate samples along a pixel 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 Nerf-Studio[Li et al. 2022; Tancik et al. 2023], $\mathcal{A}$ is derived from $\mathcal{A}$ is directly adopted from [Yu et al. 2022b]. Figure 9(a,d) denote the result of minimizing Eq. (1) and is the starting point of our AdaShell$^\dagger$ pipeline. Figure 9(b,e) denote the sampling volume as a wire frame around the estimated geometry and Fig. 9(c,f) are zoomed in views of sections of the scene.

The "shells" recovered in Fig. 9 and by [Wang et al. 2023] are physically similar quantities – [Wang et al. 2023] dilate and erode the original level-set of the scene (approximated by $\mathcal{S}$ in both our work and Adaptive Shells) through a hyperparameter. [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 width of the "shell", and subsequently use uniform sampling (similar to Eq. (11), where the $\Delta$ 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 density weighted sampling (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 4.2.4) than [Wang et al. 2023], however, at the time of writing, an implementation of AdaptiveShells was not

available to validate this claim.

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

Finally, **NeUS**$^\dagger$ 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. (1) along with Eq. (3).

## D CAPTURING APPROXIMATE BRDF AND BAKING TEXTURE

Multi-illumination images captured by our camera can be used to estimate surface reflectance properties. We largely follow the appearance parametrization described by [Zhang et al. 2022] and recover a truncated Disney BRDF model([Burley 2012]. Our model consists of a per pixel specular albedo, a diffuse RGB albedo, and roughness value to interpret the observed appearance under varying illumination. To estimate the spatially varying reflectance, we first train a volumetric model (VolSDF$^\dagger$ or NeUS$^\dagger$) to convergence to learn the appearance as radiance. At convergence, the first channel $\mathcal{S}$ of the intrinsic network $\mathcal{N}$ encodes the geometry, the appearance network $\mathcal{A}$ encodes the radiance. We use two of the remaining channels of the intrinsic network to predict the roughness and specular albedo at every point on the scene, the diffuse albedo is obtained as the output of the converged network. 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 radiance as reflectance. Figure 6 describes our steps graphically. Optimizing for the full set of the Disney BRDF parameters, following [Cheng et al. 2023] did not work with our pipeline as the optimization landscape was filled with with several local minima. Figure 11b 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, whereas choosing a more conservative set of parameters (only 'base-color', 'specular' and 'roughness') in Fig. 11a led to a more realistic appearance.

Our process of baking out 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:

- At convergence (see Fig. 6), we extracted the scene geometry using the method described in [Mescheder et al. 2019].
- 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.
- Next, we cull the resulting triangular mesh 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.
- We then generate texture coordinates on the culled mesh using [Young 2024] and rasterize the mesh to get points on the surface corresponding to the texture coordinates.
- We then 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, intersecting the surface before reaching the camera are removed.
- For all the valid projected points, we cast a ray onto the scene and use either of VolSDF$^\dagger$ or NeUS$^\dagger$ to generate the color at the pixel along the ray using Eq. (2). This is repeated for all the training views.
- At the end of this step we are left with several measurements of colors at every texture coordinate of the scene. We apply a median filter to choose the color – taking averages or maxima of the samples introduces artifacts. If baking the radiance as texture is sufficient (often the case for diffuse scenes) this textured mesh can be exported. Figure 12 demonstrates using each of VolSDF$^\dagger$, NeUS$^\dagger$ and AdaShell$^\dagger$ to calculate the diffuse color of the scene in Fig. 1.
- To bake out material textures, we follow the same procedures with the corresponding material channels after VolSDF$^\dagger$ or NeUS$^\dagger$ has been trained on multi illumination images using the schedule outlined in Fig. 6.
- 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 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 to generate the representations in Figs. 1 and 11.