SuperNormal: Neural Surface Reconstruction via Multi-View Normal Integration

Xu Cao Takafumi Taketomi
CyberAgent
{xu_cao, taketomi_takafumi}@cyberagent.co.jp

Abstract

We present SuperNormal, a fast, high-fidelity approach to multi-view 3D reconstruction using surface normal maps. With a few minutes, SuperNormal produces detailed surfaces on par with 3D scanners. We harness volume rendering to optimize a neural signed distance function (SDF) powered by multi-resolution hash encoding. To accelerate training, we propose directional finite difference and patch-based ray marching to approximate the SDF gradients numerically. While not compromising reconstruction quality, this strategy is nearly twice as efficient as analytical gradients and about three times faster than axis-aligned finite difference. Experiments on the benchmark dataset demonstrate the superiority of SuperNormal in efficiency and accuracy compared to existing multi-view photometric stereo methods. On our captured objects, SuperNormal produces more fine-grained geometry than recent neural 3D reconstruction methods. Our code is available at https://github.com/CyberAgentAILab/SuperNormal.

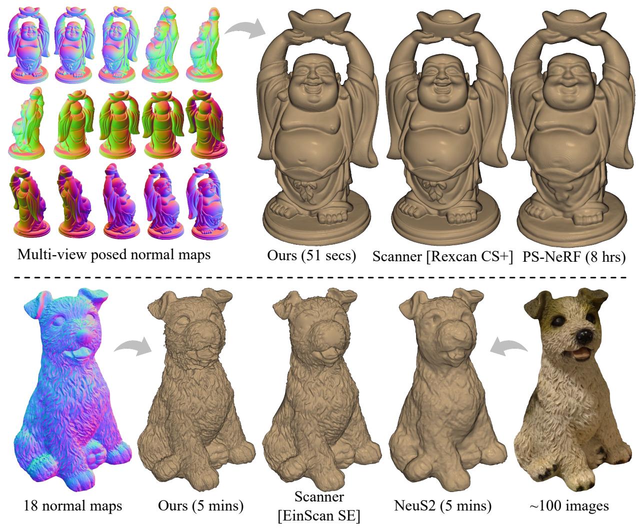


Figure 1. **(Top)** From multi-view surface normal maps, SuperNormal recovers fine-grained surface details comparable to 3D scanners while being orders of magnitude faster than the existing MVPS method PS-NeRF [31]. **(Bottom)** Using normal maps produces more faithful high-frequency details than the MVS-based method NeuS2 [30], although both use multi-resolution hash encoding [20].

1. Introduction

Recovering high-quality 3D geometry of real-world objects from their multi-view images is a long-standing challenge in computer vision. Recently, neural implicit surface-based methods have shown remarkable reconstruction results. Compared to traditional multi-view stereo (MVS) methods [25], neural methods are more robust to view-dependent observations and textureless surfaces [29]. Furthermore, the reconstruction procedure has become highly efficient [30] by introducing multi-resolution hash coding [20]. However, even though this spatial encoding allows fine-grained geometry to be represented, shape reconstruction from multi-view images tends to smooth out high-frequency surface details, as shown in Fig. 1 bottom right.

Multi-view photometric stereo (MVPS), on the other hand, aims to recover pixel-level high-frequency surface detail by introducing additional lighting conditions during the image acquisition [9]. In a typical workflow, a surface normal map is first recovered for each view, recording per-pixel surface orientation information. The normal maps estimated at different viewpoints are then fused into a 3D model, also known as multi-view normal integration [5]. However, existing normal fusion methods struggle to reflect the details of the normal maps in the shape of the recovered 3D model. Moreover, the reconstruction process takes hours even when only a few low-resolution normal maps are used [4, 31]. Due to the lack of a fast and accurate multi-view normal fusion method, current multi-view photometric stereo results remain unsatisfactory.

This paper presents SuperNormal to unite the best of both worlds. Normal maps with pixel-level surface details are utilized to exploit the expressive power and efficiency of multi-resolution hash encoding-powered neural implicit surface. Using normal maps alone avoids per-scene reflectance modeling and optimization, thus improving training efficiency compared to using color images. In a few minutes, our method can produce fine-grained geometry on par with commercial 3D scanners.

Specifically, we use volume rendering to optimize a neural signed distance function (SDF) such that its rendered normal maps are consistent with input normal maps. To further improve the training efficiency, we propose a directional finite difference and patch-based ray marching strategy to approximate the SDF gradients numerically. This strategy kills two birds with one stone: SDF values evaluated at sampled points are *all* used both for SDF-based volume rendering and SDF gradient approximation. This way, we avoid second-order derivative computation during training. Unlike traditional axis-aligned finite difference, directional finite difference avoids redundant SDF evaluations (Fig. 2). Without compromising reconstruction quality, our method is nearly twice as fast as using analytical gradients and about three times faster than finite difference.

Compared to existing neural MVPS methods, our method is two orders of magnitude faster and recovers better surface details (Fig. 1 above). We also capture objects with complex geometry and show that our method recovers more faithful surface details than the recent neural 3D reconstruction method [30] (Fig. 1 bottom). Taking advantage of recent advances in photometric stereo, we can estimate high-quality normal maps from images captured under common lighting conditions with inexpensive equipment [10]. Combined with SuperNormal, high-fidelity 3D objects can be reconstructed using inexpensive equipment in widely accessible environments.

In summary, this paper's key contributions are:

- An effective neural 3D reconstruction approach using multi-view normal maps, enhancing the reconstruction quality to near 3D scanner levels;
- An efficient method for computing numerical gradients in SDF-based neural rendering, speeding up the reconstruction process; and
- A comprehensive evaluation using benchmark objects with MVPS approaches and an assessment using our captured objects with MVS approaches.

2. Related work

2.1. Multi-view photometric stereo (MVPS)

MVPS aims at high-fidelity 3D reconstruction using multi-view observations under varying light conditions [9] (usually one-light-at-time (OLAT) images [16, 26]).

Existing methods differ in what geometry-related information is inferred in each view and how information from different views is fused into a complete 3D model. Chang *et al*. [5] proposes a level set method [22] to fuse multi-view normal maps into a volumetric representation of the shape. R-MVPS [23] uses the multi-view normal maps to refine a coarse mesh estimated by multi-view stereo. B-MVPS [16] initialize the shape as a sparse point cloud estimated by structure from motion (SfM), then propagates the known point spatially using the normal maps.

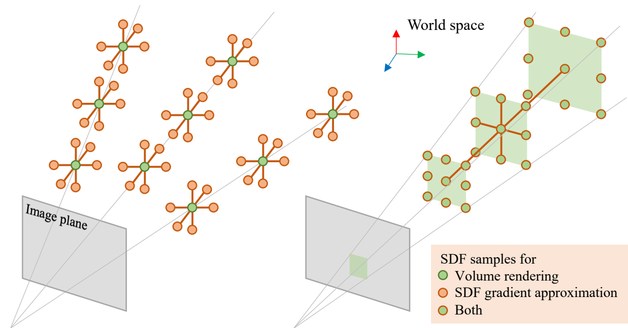


Figure 2. **(Left)** In SDF-based volume rendering, approximating SDF gradients using axis-aligned finite difference [18, 36] requires additional SDF samples at neighboring positions of volume rendering points, increasing computational complexity and training time. **(Right)** Our strategy of patch-based ray marching and directional finite difference effectively uses all SDF samples for both SDF gradient approximation and volume rendering.

Recent MVPS methods have undergone a paradigm shift since the advent of NeRF [19] by using a neural implicit scene representation, usually parameterized as a multi-layer perception (MLP). PS-NeRF [31] recovers a neural field based on UNISURF [21] by regularizing the gradients of the neural field using the normal maps. UA-MVPS [12] fuses the multi-view depth and normal maps into a neural SDF using neural inverse rendering while considering per-pixel confidence. MVAS [4] shows utilizing the azimuth components of multi-view normal maps is sufficient to recover the neural implicit surface. While most methods only use images in the same view for normal map estimation, MVPSNet [35] utilizes neighboring views to infer per-view depth and normal maps jointly and applies screened Poisson surface reconstruction (sPSR) [13] to obtain the shape.

However, the reconstructed surfaces of these methods tend to be smoothed and do not reflect the high-frequency details of the input normal maps. Our approach introduces multi-resolution hash encoding and directional finite difference to improve the implicit neural field's expressive power and training efficiency.

2.2. Neural 3D reconstruction

The seminal work NeRF [19] and follow-up works [2, 3, 27] have demonstrated remarkable novel view synthesis quality by volume rendering a neural implicit field. For better geometry reconstruction quality, IDR [32] imposes eikonal regularization [8] such that the neural field approximates a valid SDF. Later, the concurrent works VolSDF [33] and NeuS [29] develop SDF-based volume rendering, which bridges SDF values to volume rendering weights and improves the geometry fidelity. However, these methods require several hours to recover one object due to optimizing a deep coordinate-based MLP.

Multi-resolution hash encoding has been proposed to improve expressive power and accelerate the learning of neural implicit field [20]. Several efforts have been made to incorporate this expressive spatial encoding into neural SDF learning. Since the official CUDA implementation does not support the double propagation deduced by the eikonal regularization, PermutoSDF [24] and NeuS2 [30] re-implement the CUDA programming. InstantNSR [36] and Neuralangelo [18] use finite difference instead of automatic differentiation to avoid double propagation.

We propose directional finite difference, a numerical way to compute SDF gradients that avoids double propagation and is even faster than automatic differentiation when combined with a tailored patch-based sampling strategy.

2.3. Neural 3D reconstruction with normals

Normal maps have been used as auxiliary information to facilitate neural scene reconstruction in MonoSDF [34] and NeuRIS [28]. These works target scene-level reconstruction, and normal maps help infer the geometry of textureless smooth regions (*e.g.*, walls) where using color images is likely to break down.

Instead, our method is particularly useful for single objects with complex surface details. We use photometric stereo [10] to estimate normal maps from images under varying light conditions. This produces high-quality normal maps that can be used alone for shape reconstruction.

3. Approach

We aim to recover the surface given a set of normal maps, object masks, and camera intrinsic and extrinsic parameters. Since photometric stereo methods estimate the normal maps in camera space, we first apply the camera-to-world rotation to normal maps to obtain world-space normal maps. Figure 3 depicts our pipeline. We represent the geometry as a parametric SDF and optimize SDF parameters such that its rendered gradient and opacity values are consistent with input normal vectors and mask values, respectively.

3.1. Pipeline

SDF-based volume rendering (Forward pass) A signed distance function (SDF) implicitly captures the geometry by assigning each spatial point with the signed distance to its closest surface point. The surface $\mathcal{M}$ can then be represented as the zero-level-set of the SDF:

$$\mathcal{M} = \{\mathbf{x} \mid f(\mathbf{x}) = 0\}. \tag{1}$$

We parameterize the SDF using multi-resolution hash encoding followed by a shallow MLP. In multi-resolution hash encoding, the 3D space is discretized into a set of multi-resolution 3D grids. Each grid cell is associated with a unique hash value mapping the 3D coordinates to a learnable feature vector. Specifically, $h(\mathbf{x}; \phi)$ is the feature vector obtained by concatenating feature vectors of different levels $h(\mathbf{x}) = [\mathbf{x}, h_1(\mathbf{x}), ..., h_L(\mathbf{x})]$. Then, the SDF can be parameterized as

$$f(\mathbf{x}) = f(h(\mathbf{x}; \phi); \theta), \tag{2}$$

where ϕ and θ are learnable hash-encoding feature vectors and MLP parameters. Multi-resolution hash encoding has proven to speed up training and improve the expressive power of the neural SDF [20, 30, 36].

During training, we evaluate the SDF values at points sampled on the rays cast from the camera centers toward the scene. NeuS [29] proposes an unbiased and occlusion-aware way to transfer the SDF samples to volume rendering opacities. Given N ordered 3D points $\{\mathbf{x}_i\}_{i=0}^N$ on a ray and their SDF values $\{f(\mathbf{x}_i)\}_{i=0}^N$, the opacity at each point is designed as

$$\alpha_i = \max\left(\frac{\phi_s(f(\mathbf{x}_i)) - \phi_s(f(\mathbf{x}_{i+1}))}{\phi_s(f(\mathbf{x}_i))}, 0\right), \tag{3}$$

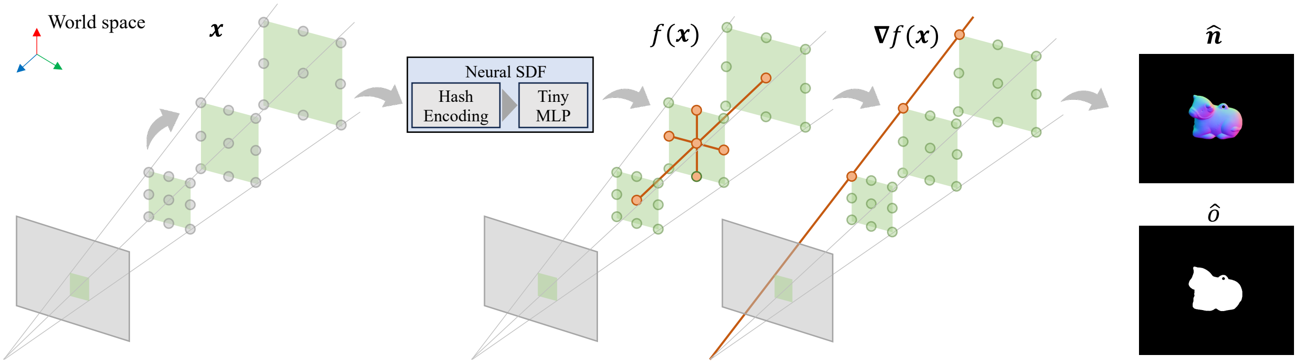
where $\phi_s(x) = \frac{1}{1+\exp(-sx)}$ is the sigmoid function with a trainable sharpness s.

With this SDF-based opacity, we can render the normal and opacity for the pixel $\mathbf{p}$ as:

$$\hat{\mathbf{n}}(\mathbf{p}) = \sum_{i=0}^{N} T_i \alpha_i \nabla f(\mathbf{x}_i), \quad \hat{o}(\mathbf{p}) = \sum_{i=0}^{N} T_i \alpha_i \quad \text{with} \quad T_i = \prod_{j=0}^{i-1}(1-\alpha_j). \tag{4}$$

From Eqs. (3) and (4), the neural SDF must be evaluated at least once at sampled points for volume rendering. In Secs. 3.2 and 3.3, we show that these SDF samples can be reused for SDF gradient approximation, thus reducing the computational amount and speeding up the training.

Neural SDF Training (Backward pass) To train the neural SDF, we supervise the rendered normals and opacities using input ones $\mathbf{n}(\mathbf{p})$ and $\hat{o}(\mathbf{p})$. The loss function consists



(a) Patch-based ray marching **(b)** Directional finite difference **(c)** SDF-based volume rendering Normal and opacity maps

Figure 3. **Approach overview.** **(a)** Given input views, we randomly sample patches of pixels, cast rays from those pixels, and march a plane along the patch of rays (Sec. 3.3). We treat the ray-plane intersections as sampled points and evaluate their SDF values via the neural SDF. **(b)** SDF samples neighbor to a point are used to compute the SDF gradients using directional finite difference (Sec. 3.2). **(c)** SDF samples on the same rays are used for SDF-based volume rendering (Sec. 3.1).

of three terms: The normal, the mask, and the eikonal term:

$$\mathcal{L} = \overbrace{\sum_{\mathbf{p}} (\hat{\mathbf{n}}(\mathbf{p}) - \mathbf{n}(\mathbf{p}))^2}^{normal} + \lambda_1 \overbrace{\sum_{\mathbf{p}} \mathrm{BCE}\left(\hat{o}(\mathbf{p}), o(\mathbf{p})\right)}^{mask} + \lambda_2 \underbrace{\sum_{\mathbf{x}} (\|\nabla f(\mathbf{x})\|_2 - 1)^2}_{eikonal}, \tag{5}$$

where $\mathbf{p}$ is the sampled pixels in a batch, $\mathbf{x}$ is the sampled points on rays, and BCE is the binary cross entropy function. The normal term encourages the rendered SDF gradients to be consistent with the input normal vectors so that accurate geometry details are recovered. The mask term aligns the shape's silhouette with input masks and can be considered the boundary condition. The eikonal term enforces the SDF gradient norm to be close to 1 almost everywhere so that the neural SDF is approximately valid [22].

3.2. Directional Finite Difference

As seen, both the normal and eikonal terms backpropagate the loss gradient via the SDF gradient to neural SDF parameters (*i.e.*, double backpropagation $\frac{\partial \mathcal{L}}{\partial \nabla f} \frac{\partial \nabla f}{\partial \theta}$). The most common way is to use Pytorch's automatic differentiation (AD) due to implementation simplicity. However, the multi-resolution hash encoding package does not officially support double propagation. Common workarounds include CUDA programming [24, 30] or use finite difference (FD) [18, 36] to bypass double propagation. However, CUDA programming can be inflexible in that it shrinks the design space (*e.g.*, NeuS2 [30] requires ReLU activation), and FD will slow down the training. Consider the ordinary axis-aligned FD:

$$\nabla f(\mathbf{x}) \approx \begin{bmatrix} \frac{f(\mathbf{x}+\boldsymbol{\epsilon}_x) - f(\mathbf{x}-\boldsymbol{\epsilon}_x)}{2\epsilon} \\ \frac{f(\mathbf{x}+\boldsymbol{\epsilon}_y) - f(\mathbf{x}-\boldsymbol{\epsilon}_y)}{2\epsilon} \\ \frac{f(\mathbf{x}+\boldsymbol{\epsilon}_z) - f(\mathbf{x}-\boldsymbol{\epsilon}_z)}{2\epsilon} \end{bmatrix}, \tag{6}$$

where $\boldsymbol{\epsilon}_{* \in \{x,y,z\}}$ is an offset vector along x, y, or z axis with a tiny step size ϵ. Computing the gradient at one point requires evaluating the SDF at its nearby points. These additional SDF samples are also included in the computation graph in the forward pass. Consequently, using FD significantly slows the forward rendering and backward optimization even if the efficient spatial encoding is used [18].

To avoid redundant SDF evaluations and double backpropagation, we propose directional finite difference (DFD) to reuse the volume rendering SDF samples for SDF gradients approximation. Consider the SDF value at a 3D point $\mathbf{x}$ parameterized by the ray origin $\mathbf{o}$ and direction $\mathbf{v}$:

$$f(\mathbf{x}) = f(\mathbf{o} + t\mathbf{v}), \tag{7}$$

where t is the distance to the ray origin. From Eq. (7), we have

$$\frac{df}{dt} = \nabla f(\mathbf{x})^\top \mathbf{v} := \nabla_{\mathbf{v}} f. \tag{8}$$

Equation (8) is exactly the directional derivative of the SDF, which projects the SDF gradient onto the ray direction. Suppose we parameterize the same point along three different directions as

$$\mathbf{x} = \mathbf{o}_i + t_i \mathbf{v}_i, \quad i \in \{1, 2, 3\}. \tag{9}$$

Applying Eq. (8) then yields three linear equations:

$$\underbrace{\begin{bmatrix} -\mathbf{v}_1^\top - \\ -\mathbf{v}_2^\top - \\ -\mathbf{v}_3^\top - \end{bmatrix}}_{\mathbf{V}} \nabla f = \underbrace{\begin{bmatrix} \frac{df}{dt_1} \\ \frac{df}{dt_2} \\ \frac{df}{dt_3} \end{bmatrix}}_{\nabla_{\mathbf{V}} f}. \tag{10}$$

From Eq. (10), we can compute the SDF gradient by

$$\nabla f = \mathbf{V}^{-1} \nabla_{\mathbf{V}} f. \tag{11}$$

Numerically, we can approximate the directional derivative by

$$\begin{aligned} \frac{df}{dt_i} &\approx \frac{f(\mathbf{o}_i + (t_i + \Delta t)\mathbf{v}_i) - f(\mathbf{o}_i + (t_i - \Delta t)\mathbf{v}_i)}{2\Delta t} \\ &= \frac{f(\mathbf{x}_i + \Delta t \mathbf{v}_i) - f(\mathbf{x}_i - \Delta t \mathbf{v}_i)}{2\Delta t}, \quad i \in \{1, 2, 3\} \end{aligned} \tag{12}$$

Equations (11) and (12) imply that, to numerically approximate the SDF gradient, we can first compute finite differences along three directions $\mathbf{V}$, then linearly transform the concatenated finite difference by $\mathbf{V}^{-1}$. We call this SDF gradient approximation directional finite difference. Equation (11) has a unique solution if the three directions are non-coplanar.

The ordinary finite difference can be viewed as the special case of Eq. (11) by choosing the three coordinate axes as the ray directions. In this case, $\mathbf{V}$ becomes the identity matrix. Both DFD and FD require nearby SDF samples to approximate the SDF gradient. However, by tailoring the sampling strategy, we can reuse the SDF samples in volume rendering for DFD without additional SDF samples, as described in the following.

3.3. Patch-based Ray Marching

DFD requires computing the finite difference in three directions. In volume rendering, we naturally have SDF samples along the viewing ray direction that can be used for finite difference. For the other two directions, we propose to sample patches of pixels instead of single pixels and march a plane on each patch of rays to locate the sampled points on parallel planes.

Specifically, we perform ordinary ray marching for the center pixel/ray of the patch and treat the sampled points as the intersections between a marching plane and the center ray. The points to be sampled on the remaining rays within the patch can then be found as the intersection of the planes and the rays:

$$t_j = \frac{t_i \mathbf{v}_i^\top \mathbf{m}}{\mathbf{v}_j^\top \mathbf{m}}, \tag{13}$$

where $\mathbf{m}$ is the unit normal direction of the marching plane, t_i is the ray marching distance on the center ray with direction $\mathbf{v}_i$, and t_j is the distance from the intersection point to the ray origin (*i.e.*, camera center) along the direction $\mathbf{v}_j$; for derivation see Sec. B.2 in SupMat.

We choose the marching plane to parallel the image plane where the patch is located (Fig. 3). With this sampling strategy, the three directions for DFD computation are the viewing ray direction and the world-space directions of x- and y-axis of the camera coordinates. The marching plane normal $\mathbf{m}$ becomes the world-space direction of the z-axis of the camera coordinates and is the same for all pixels/rays from the same view. This way, the SDF samples on the same ray can be used for volume rendering as Eq. (4), and the SDF samples neighbor to a point can be used for SDF gradient computation as Eq. (11). We use the central difference (*i.e.*, Eq. (12)) for points with neighbors in both directions and forward/backward difference for boundary points.

This sampling strategy is beneficial in three aspects. First, it allows us to reuse the SDF samples from neighboring rays to compute DFD. Second, it stabilizes training by ensuring the existence of $\mathbf{V}^{-1}$, since the viewing ray direction will never co-planer to the image plane. Third, it reduces the computation overhead during training because the three ray directions are identical for samples from the same pixel/ray. The inverse of ray directions $\mathbf{V}^{-1}$ can be pre-computed for each pixel before training. In our experiments, the computation takes about $0.6\,\mathrm{sec}$ for two million pixels. During training, only the 3×3 matrix-vector multiplication Eq. (11) is required, which can almost be ignored compared to AD or FD.

4. Experiments

We evaluate our method quantitatively on the MVPS benchmark in Sec. 4.1, investigate different components of our method in Sec. 4.2 and evaluate our method qualitatively on our captured real-world data in Sec. 4.3.

Implemetation details We apply SDM-UniPS [10] to images in each view to obtain normal maps. In each batch, we randomly sample 2048 patches of 3×3 pixels from all input normal maps. Like INGP [20], we skip the empty and occluded space during ray marching using NerfAcc [17]. The scene is bounded within a unit sphere, and we use a coarse-to-fine ray marching step decreasing log-linearly from $1e^{-2}$ to $1e^{-3}$. We optimize the shape of DiLiGenT-MV [16] benchmark objects for 5000 batches (roughly equal to 20 epochs), and 30000 batches for our captured data. The weights of loss terms are set as 1. We use the Adam optimizer with an initial learning rate $5e^{-3}$. More details can be found in Sec. B in SupMat.

4.1. Quantitative evaluation

Dataset We evaluate our method using the only MVPS benchmark dataset DiLiGenT-MV [16]. It consists of five objects captured from 20 views, and 96 OLAT images are captured in each view, yielding a total of 1920 images per object. Each image is 512×612, and the total number of foreground object pixels is about 2.2 million. The camera is fixed during capture, and the object is placed on a turntable in a darkroom. The "ground truth" meshes are created using a 3D scanner.

Baselines We benchmark SuperNormal against several MVPS methods grouped into three categories. (1) Those that refine an initial point cloud or mesh and require manual effort, including R-MVPS [23] and B-MVPS [16]. (2) Those that use Poisson surface reconstruction (PSR) [13] to fuse multi-view depth and normal maps, including MVPSNet [35]. (3) Those that optimize an implicit neural surface representation, including UA-MVPS [12], PS-NeRF [31], MVAS [4], and our method.

Evaluation metrics We use L2 Chamfer distance (CD) and F-score with threshold $\tau_F = 0.5\,\mathrm{mm}$ to evaluate geometry accuracy [12, 15]. For CD and F-score, we only consider visible points from the input views by casting rays for all pixels and finding the first ray-mesh intersections (More details in Sec. B.4 in SupMat.)

Results As reported in Table 1, our method achieves the best geometry accuracy among all compared MVPS methods (See Sec. A in SupMat. for normal accuracy). We attribute this superior performance to two aspects: 1) the powerful expressive capability of multi-resolution hash encoding [20] and 2) the fine-grained normal estimation method SDM-UniPS [10], as discussed in Sec. 4.2. Figure 4 visualizes the recovered shapes of different methods. Our method recovers fine-grained surface details on par with the "GT" meshes produced by a 3D scanner.

Table 1 also reports the average runtime measured on our machine with one RTX 4090Ti graphics card. The runtime of R-MVPS [23] and B-MVPS [16] is difficult to measure since both methods require intensive manual effort. The code of UA-MVPS [12] is not publicly available, and several hours per object are required, as reported in MVPSNet [35]. Both PS-NeRF [31] and MVAS [4] require hours per object since they optimize a dense MLP. Our method is the fastest among neural implicit representation-based methods, with less than one minute per object.

On the other hand, MVPSNet [35] is highly efficient because its pipeline mainly consists of feedforward inferencing a trained neural network and an sPSR [13] step, the latter of which has been optimized over one decade. Although our method is slower than MVPSNet [35], we achieve better reconstruction quality within a reasonable time.

4.2. Ablation study

Effect of normal maps Since multi-view normal integration is not bound to specific normal estimation methods, we investigate the effect of different normal estimation methods on normal integration methods. We test three sets of normal maps, estimated by SDPS [6] or SDM-UniPS [10], and the one rendered from the scanned mesh (*i.e.*, "GT" normal maps provided by DiLiGenT-MV [16]). Table 2 reports the L2-Chamfer distance and F-score averaged on five DiLiGenT-MV [16] objects. Our method consistently achieves better results than PS-NeRF [31] and MVAS [4] when using the same set of normal maps. This verifies that our method can better fuse the information from 2.5D normal maps to 3D shapes. Using the SoTA method SDM-UniPS [10] for normal estimation, our method realizes the best reconstruction quality.

Effect of spatial encoding Table 2 also compares the geometry accuracy between multi-resolution hash encoding (hash enc.) and positional encoding (pos. enc.). Our results verify again that multi-resolution hash encoding surpasses positional encoding in expressiveness, enhancing geometry accuracy. Therefore, the keys to superior MVPS performance are high-quality normal estimation, an expressive surface representation, and an effective normal fusion approach.

Effect of SDF gradient computation Table 3 compares the geometry accuracy and runtime using finite difference (FD), Pytorch's automatic differentiation (AD), and directional finite difference (DFD) to compute the SDF gradient. All other parameter settings are kept the same. We do not observe a significant difference regarding reconstruction quality. However, DFD performs consistently faster than FD and AD by accelerating both the forward rendering and backward optimization.

Generalizability of DFD In principle, DFD and patch-based sampling strategy apply to any SDF-based volume rendering like NeuS [29] or Neuralangelo [18]. To verify this, we incorporate DFD and patch-based sampling into Neuralangelo [18][1]. We follow the coarse-to-fine sampling strategy of Neuralangelo [18] but use DFD instead of FD for SDF gradient approximation. We do not use mask supervision for Neuralangelo [18]. As shown in Figure 5, training time is reduced by 25% on a scene from the DTU benchmark [11], even $2.25\times$ more pixels are looped in training. The result suggests that DFD could be a general strategy to accelerate NeuS-like multi-view reconstruction [29].

4.3. Qualitative results on our captured data

To further demonstrate the performance of SuperNormal in fine-grained surface detail recovery, we also captured objects with more complex details, as shown in Fig. 6. Thanks to the generalizability of SDM-UniPS [10], we can capture the objects under casual light conditions (*e.g.*, within an apartment) without ensuring a darkroom setup. We manually move a video light during the capture without any photometric or geometric light calibration. In total, we capture 18 viewpoints $\times$ 13 light conditions for each object (1 under ambient light for SfM). We use Metashape [1] to calibrate camera parameters, use SAM [14] to create foreground masks, and apply SDM-UniPS [10] to every 12 image to infer the per-view normal map.

[1]Our modified code of Neuralangelo [18] is available at https://github.com/xucao-42/Neuralangelo_DFD.git.

Table 1. Quantitative evaluation on DiLiGenT-MV [16] benchmark. Red and orange cells indicate **the best** and the second best results respectively. On average, our approach achieves the best reconstruction quality at the second fastest speed.

Methods	L2 Chamfer distance [mm] (↓) Bear	Buddha	Cow	Pot2	Reading	Average	F-score (τ_F = 0.5 mm) (↑) Bear	Buddha	Cow	Pot2	Reading	Average	Runtime (↓) Average
R-MVPS [23]	1.070	0.397	0.440	1.504	0.561	0.794	0.262	0.698	0.760	0.198	0.519	0.487	NA
B-MVPS [16]	0.212	0.254	**0.091**	0.201	0.259	0.203	0.958	0.902	**0.986**	0.946	0.914	0.941	NA
MVPSNet [35]	0.317	0.279	0.255	0.310	0.248	0.282	0.813	0.866	0.889	0.838	**0.918**	0.865	**37 secs**
UA-MVPS [12]	0.414	0.452	0.326	0.414	0.382	0.398	0.707	0.669	0.798	0.731	0.762	0.733	several hrs
PS-NeRF [31]	0.260	0.314	0.287	0.254	0.352	0.293	0.898	0.806	0.856	0.919	0.785	0.853	8 hrs
MVAS [4]	0.243	0.357	0.216	0.197	0.522	0.307	0.909	0.754	0.907	0.962	0.546	0.816	70 mins
Ours	**0.184**	**0.218**	0.193	**0.150**	**0.223**	**0.194**	**0.983**	**0.950**	0.977	**0.994**	0.916	**0.962**	48 secs

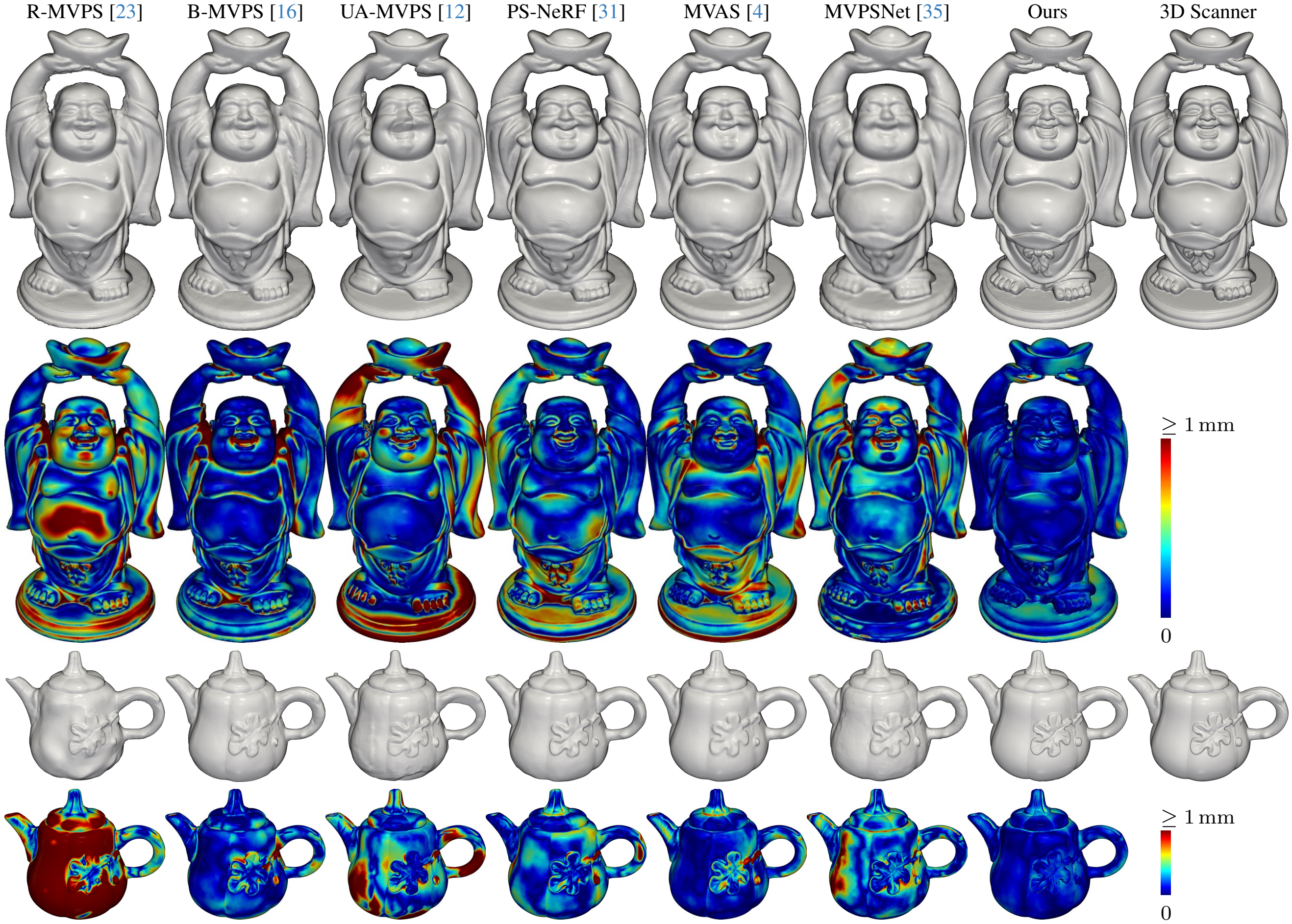


Figure 4. Qualitative comparison of recovered shapes of *Buddha* and *Pot2* and their error maps. Best viewed on screen.

Table 2. Quantitative evaluation using normal maps estimated by different methods as inputs. The metrics are averaged over five DiLiGenT-MV [16] objects. Darker colors indicate better results.

	L2 Chamfer distance [mm] (↓) SDPS [6]	SDM-UniPS [10]	GT	F-score (τ_F = 0.5 mm) (↑) SDPS [6]	SDM-UniPS [10]	GT
MVAS [4]	0.307	0.332	0.292	0.816	0.812	0.859
PS-NeRF [31]	0.293	0.296	0.224	0.853	0.844	0.938
Ours (pos. enc.)	0.301	0.228	0.195	0.849	0.942	0.977
Ours (hash enc.)	0.281	0.194	0.114	0.860	0.962	0.998

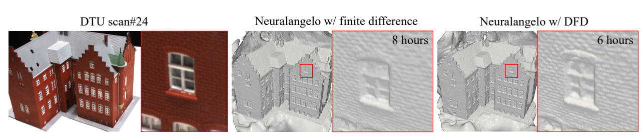


Figure 5. DFD and patch-based sampling accelerate Neuralangelo's training without quality degradation. We train both alternatives for 20k batches. For FD, 512 random rays are sampled per batch; for DFD, 128 random patches of 3 × 3 rays per batch.

Table 3. Ablation study using different strategies for SDF gradients computation in volume rendering. We report the metrics averaged over five DiLiGenT-MV [16] objects. **FD**: Finite difference, **AD**: Automatic differentiation, **DFD**: Directional finite difference.

	Accuracy		Runtime (Sec)		
	CD	F-Score	Forward	Backward	Total
FD	0.192	0.963	47.7	70.3	124.6
AD	0.195	0.961	29.4	42.4	78.1
DFD	0.194	0.961	19.6	22.5	48.3

Figure 6. Our real-world data capture setup. The target object is placed on a turntable before a dark sheet under general indoor illumination. There is no need to prepare a darkroom. We use an iPhone to take images and move a video light to illuminate the object from the camera side. Objects are 6 cm to 15 cm high.

Figures 1 and 7 display the results from NeuS2 [30], SuperNormal, and a structured-light based commercial 3D scanner EinScan SE. Although NeuS2 [30] and our method use the same multi-resolution hash encoding for fine-grained scene representation, the surface details recovered from multi-view color images are smoothed out. Our method, benefitting from the normal map inputs, recovers more fine-grained surface details.

We find that our scanner performs poorly in highly concave regions (Fig. 7 above). This is because the scanner uses a wide baseline stereo camera (about 15 cm apart), and small concave regions create self-occlusions, thus reducing the reconstruction quality. In principle, normal estimation does not suffer from self-occlusion because it is based on single-view observations. Fusing multi-view normal maps can then produce better results in concave regions.

5. Discussion

We have demonstrated SuperNormal, a fast, high-fidelity 3D reconstruction approach using multi-view normal maps estimated by photometric stereo. SuperNormal outperforms other neural MVPS methods in speed and reconstruction quality on the benchmark dataset. Compared to multi-view neural reconstruction methods, using normal maps can exploit the expressive power of multi-resolution hash encoding, and produce scanner-level surface details. Further, we have shown the effectiveness of directional finite differences in expediting training without sacrificing quality.

SuperNormal provides a promising direction to make high-quality 3D scanning more accessible. Unlike costly commercial 3D scanners, our approach is viable with everyday devices like smartphones and basic equipment such as a tripod, video light, and a turntable.

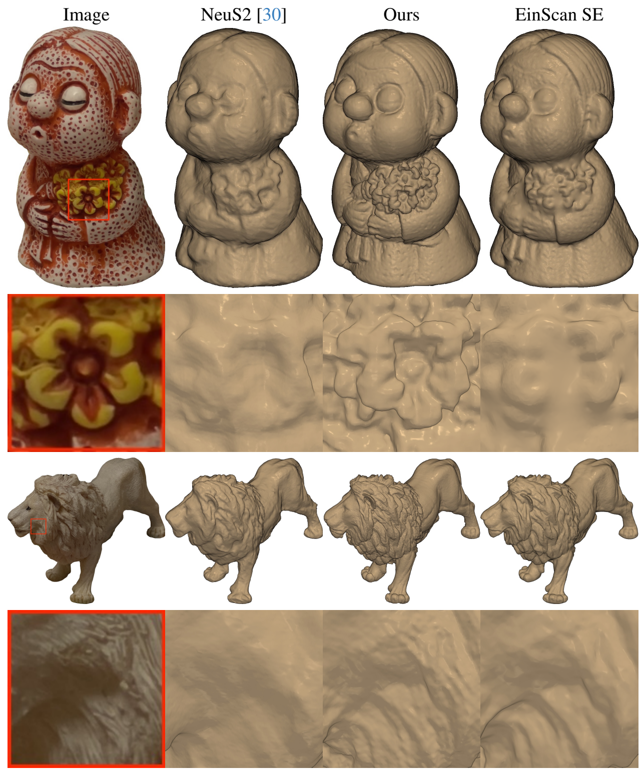


Figure 7. Qualitative comparison using our objects. Our approach recovers better surface detail than multi-view stereo approaches and achieves competitive results with commercial 3D scanners.

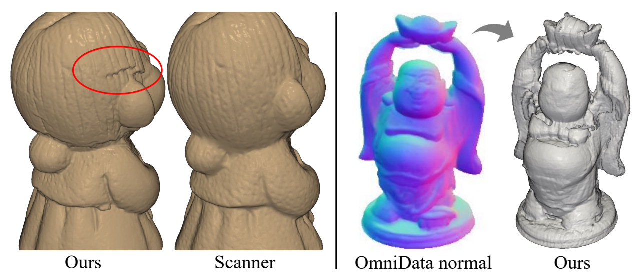


Figure 8. **Limitations.** (Left) Fault artifacts are highlighted in red circles. (Right) Our method is sensitive to the inaccuracy in normal maps estimated by monocular methods (*e.g.*, OmniData [7]).

Limitations Our method may produce fault-like artifacts, as shown in Fig. 8. The artifacts exist no matter our SDF gradient computation approach (*i.e.*, AD, FD, or DFD) and require further investigation. Moreover, the performance depends on mask supervision and high-quality normal maps (*e.g.*, those obtained from photometric stereo). Our method does not work well with the monocular normal estimation methods like OmniData [7] (Fig. 8).

References

[1] Agisoft metashape. https://www.agisoft.com. 6

[2] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *ICCV*, 2021. 2

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-NeRF: Anti-aliased grid-based neural radiance fields. *ICCV*, 2023. 2

[4] Xu Cao, Hiroaki Santo, Fumio Okura, and Yasuyuki Matsushita. Multi-view azimuth stereo via tangent space consistency. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 825–834, 2023. 1, 2, 6, 7, 11, 12

[5] Ju Yong Chang, Kyoung Mu Lee, and Sang Uk Lee. Multiview normal field integration using level set methods. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007. 1, 2

[6] Guanying Chen, Kai Han, Boxin Shi, Yasuyuki Matsushita, and Kwan-Yee K. Wong. SDPS-Net: Self-calibrating deep photometric stereo networks. In *CVPR*, 2019. 6, 7

[7] Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *ICCV*, pages 10786–10796, 2021. 8

[8] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proceedings of Machine Learning and Systems 2020*, pages 3569–3579. 2020. 3, 11

[9] Carlos Hernandez, George Vogiatzis, and Roberto Cipolla. Multiview photometric stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 30(3):548–554, 2008. 1, 2

[10] Satoshi Ikehata. Scalable, detailed and mask-free universal photometric stereo. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, 2023. 2, 3, 5, 6, 7

[11] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *CVPR*, pages 406–413. IEEE, 2014. 6

[12] Berk Kaya, Suryansh Kumar, Carlos Oliveira, Vittorio Ferrari, and Luc Van Gool. Uncertainty-aware deep multi-view photometric stereo. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 12601–12611, 2022. 2, 6, 7

[13] Michael Kazhdan and Hugues Hoppe. Screened Poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. 2, 6

[14] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment anything. In *Proc. of International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023. 6

[15] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 6

[16] Min Li, Zhenglong Zhou, Zhe Wu, Boxin Shi, Changyu Diao, and Ping Tan. Multi-view photometric stereo: A robust solution and benchmark dataset for spatially varying isotropic materials. *IEEE Transactions on Image Processing*, 29:4159–4173, 2020. 2, 5, 6, 7, 8, 11, 12

[17] Ruilong Li, Hang Gao, Matthew Tancik, and Angjoo Kanazawa. NerfAcc: Efficient sampling accelerates nerfs. In *Proc. of International Conference on Computer Vision (ICCV)*, pages 18537–18546, 2023. 5, 13

[18] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H. Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 8456–8465, 2023. 2, 3, 4, 6

[19] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proc. of European Conference on Computer Vision (ECCV)*, 2020. 2

[20] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 1, 3, 5, 6

[21] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proc. of International Conference on Computer Vision (ICCV)*, 2021. 2

[22] Stanley Osher, Ronald Fedkiw, and K Piechor. Level set methods and dynamic implicit surfaces. *Appl. Mech. Rev.*, 57(3):B15–B15, 2004. 2, 4

[23] Jaesik Park, Sudipta N Sinha, Yasuyuki Matsushita, Yu-Wing Tai, and In So Kweon. Robust multiview photometric stereo using planar mesh parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 39(8):1591–1604, 2016. 2, 6, 7, 11, 12

[24] Radu Alexandru Rosu and Sven Behnke. Permutosdf: Fast multi-view reconstruction with implicit surfaces using permutohedral lattices. In *CVPR*, 2023. 3, 4

[25] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proc. of European Conference on Computer Vision (ECCV)*, 2016. 1

[26] Marco Toschi, Riccardo De Matteo, Riccardo Spezialetti, Daniele De Gregorio, Luigi Di Stefano, and Samuele Salti. Relight my NeRF: A dataset for novel view synthesis and relighting of real world objects. In *Proc. of Computer Vision and Pattern Recognition (CVPR)*, pages 20762–20772, 2023. 2

[27] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022. 2

[28] Jiepeng Wang, Peng Wang, Xiaoxiao Long, Christian Theobalt, Taku Komura, Lingjie Liu, and Wenping Wang. Neuris: Neural reconstruction of indoor scenes using normal priors. In *ECCV*, 2022. 3

[29] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view recon-

struction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 1, 3, 6

[30] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. NeuS2: Fast learning of neural implicit surfaces for multi-view reconstruction. In *Proc. of International Conference on Computer Vision (ICCV)*, 2023. 1, 2, 3, 4, 8, 14, 15

[31] Wenqi Yang, Guanying Chen, Chaofeng Chen, Zhenfang Chen, and Kwan-Yee K. Wong. PS-NeRF: Neural inverse rendering for multi-view photometric stereo. In *Proc. of European Conference on Computer Vision (ECCV)*, 2022. 1, 2, 6, 7, 11, 12, 14

[32] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *Advances in Neural Information Processing Systems (NeurIPS)*, 33, 2020. 2

[33] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems (NeurIPS)*, 34:4805–4815, 2021. 3

[34] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. MonoSDF: Exploring monocular geometric cues for neural implicit surface reconstruction. *NIPS*, 2022. 3

[35] Dongxu Zhao, Daniel Lichy, Pierre-Nicolas Perrin, Jan-Michael Frahm, and Soumyadip Sengupta. MVPSNet: Fast generalizable multi-view photometric stereo. In *Proc. of International Conference on Computer Vision (ICCV)*, pages 12525–12536, 2023. 2, 6, 7

[36] Fuqiang Zhao, Yuheng Jiang, Kaixin Yao, Jiakai Zhang, Liao Wang, Haizhao Dai, Yuhui Zhong, Yingliang Zhang, Minye Wu, Lan Xu, and Jingyi Yu. Human performance modeling and rendering via neural animated mesh. *ACM Trans. Graph.*, 41(6), 2022. 2, 3, 4