Francois Darmon, Benedicte Bascle, Jean-Clement Devaux, Pascal Monasse, Mathieu Aubry

Improving neural implicit surfaces geometry with patch warping

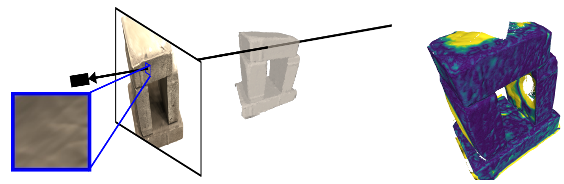
François Darmon[12] Bénédicte Bascle[1] Jean-Clément Devaux[1] Pascal Monasse[2] Mathieu Aubry[2]
[1]Thales LAS France [2]LIGM (UMR 8049), École des Ponts, Univ. Gustave Eiffel, CNRS, Marne-la-Vallée, France
http://imagine.enpc.fr/~darmonf/NeuralWarp/

Abstract

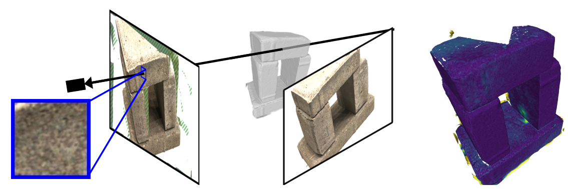
Neural implicit surfaces have become an important technique for multi-view 3D reconstruction but their accuracy remains limited. In this paper, we argue that this comes from the difficulty to learn and render high frequency textures with neural networks. We thus propose to add to the standard neural rendering optimization a direct photo-consistency term across the different views. Intuitively, we optimize the implicit geometry so that it warps views on each other in a consistent way. We demonstrate that two elements are key to the success of such an approach: (i) warping entire patches, using the predicted occupancy and normals of the 3D points along each ray, and measuring their similarity with a robust structural similarity (SSIM); (ii) handling visibility and occlusion in such a way that incorrect warps are not given too much importance while encouraging a reconstruction as complete as possible. We evaluate our approach, dubbed NeuralWarp, on the standard DTU and EPFL benchmarks and show it outperforms state of the art unsupervised implicit surfaces reconstructions by over 20% on both datasets. Our code is available at https://github.com/fdarmon/NeuralWarp

1. Introduction

Multi-view 3D reconstruction is the task of recovering the geometry of objects by looking at their projected views. Multi-View Stereo (MVS) methods rely on the photo-consistency of multiple views and typically provide the best results [29, 44]. However, they require a cumbersome multi-step procedure, first estimating then merging depth maps. Recent 3D optimization methods [22, 24, 25, 34, 41, 42] avoid this issue by representing the surface implicitly and jointly optimizing neural networks encoding occupancy and color for all images, but their accuracy remains limited. In this work, we bridge these two types of approaches by optimizing multi-view photo-consistency for a geometry represented by implicit functions. We show that this enables our method to leverage high-frequency textures present in the input images that existing implicit methods struggle to represent, resulting in significant accuracy gains.



(a) Volumetric rendering (left) and geometric error map (right).



(b) Our method's image warping (left) and geometric error map (right)

Figure 1. Standard neural implicit surface approaches jointly optimize a geometry and color network, but struggle to represent high frequency textures and therefore lack accuracy (top). We propose to additionally warp image patches with the implicit geometry, which allows to directly optimize photo-consistency between images and significantly improves the reconstruction accuracy (bottom).

The idea behind our approach is visualized on Figure 1. The top row shows a rendering and the geometric error map (1a) for a state of the art implicit method [41]. The rendering fails at producing high frequency textures, resulting in low 3D accuracy. To overcome this limitation we use the original images, reprojecting them using the geometry described by the implicit occupancy function. This is shown on the bottom row (1b) where our warped patch includes high frequency texture. Consequently, we can optimize the geometry much more accurately, resulting in smaller geometric errors in the reconstruction.

Optimizing the implicit geometry for photo-consistency poses two main challenges. First, since we do not have perfectly Lambertian materials, directly minimizing the difference between colors is not meaningful and would lead to artefacts. We thus compare entire patches using a robust similarity (SSIM [36]) which requires performing patch warping using the implicit geometry. Building on the volumetric neural implicit surface framework, we start by sampling 3D points on the ray associated to each pixel in a reference image. We then propose to warp for each sampled point a source image patch to the reference image using a planar scene approximation, and finally combine all warped patches. Second, opposite to standard neural rendering methods that can associate a color to each 3D point, a warping-based approach must deal with the fact that many 3D points do not project correctly in the source view, e.g. are not visible or are occluded; this will typically happen for points sampled on any ray. We thus define for each reference image pixel and each source image a soft visibility mask. We then completely remove from the loss the contribution of pixels in the reference image that have no valid reprojection in any of the source views and, for the other pixels, weight the loss associated to each source view depending on how reliable the associated projection is. This downweights invalid reprojections, while encouraging a reconstruction as complete as possible.

We evaluate our method on the DTU [14] and EPFL [32] benchmarks. Our method outperforms current state-of-the-art unsupervised neural implicit surfaces methods by a large margin: the 3D reconstruction metrics are on average improved by 20%. We also show qualitatively that our image warps are able to capture high frequency details.

To summarize, we present:

- a method to warp patches using implicit geometry;
- a loss function able to handle incorrect reprojections;
- an experimental evaluation demonstrating the very significant accuracy gain on two standard benchmarks and validating each element of our approach.

2. Related Work

Multi-view 3D reconstruction, the task of recovering the 3D geometry of a scene from 2D images, is a long standing problem in computer vision. We focus on the calibrated scenario where both camera calibrations are known. In this section, we first review Multi-View Stereo (MVS) methods, then neural implicit methods that optimize neural networks for image rendering and finally methods that use projections in multiple views with implicit methods.

Multi-view stereo (MVS): Classical MVS approaches use 3D representations such as 3D point clouds like PMVS [10], voxel grids [19, 30] or depth maps [11, 29, 46]. A more detailed overview can be found in [9]. Depth map based methods are arguably the most common, with the widespread usage of COLMAP [29]. This approach relies on a graphical model and optimizes depth and normal maps in a multi-step optimization. It ends with a depth map fusion step that outputs a point cloud, which can be further processed with a meshing algorithm [17, 20]. Deep learning has also been successfully applied to MVS estimation. Most methods output depth map estimates [5, 13, 38, 39, 44], but some also produce voxels [15, 23] or point clouds [31]. These methods achieve impressive results on multiple benchmarks [14, 18] but they are supervised and trained on specific datasets [14, 40]. Unsupervised deep MVS methods have also been introduced [7, 8, 37] but their performances are still limited compared to supervised versions. Our method is fully unsupervised and requires neither training data nor pretrained networks, but has high performance.

Neural implicit surfaces: Recently, new implicit representations of 3D surfaces with neural network were introduced. The surface is represented by a neural network which will output either an occupancy field [21, 27] or a Signed Distance Function (SDF) [26]. These representations are used to perform multi-view reconstruction following two different paradigms [25]: surfacic [24, 42, 45] and volumetric [22, 25, 34, 41]. Surfacic approaches compute the surface then backpropagate through this step with implicit differentiation [1]. They are hard to optimize and typically require additional supervision: silhouette masks in [24, 42] or the output of a pretrained depth map estimator [44] in MVSDF [45]. Volumetric approaches were introduced in NeRF [22]. The latter combines classical volumetric rendering [16] with implicit functions to produce high quality renderings of images. The main focus of NeRF [22] was the quality of rendering, therefore the geometry was not evaluated. Further work adapted the geometric output [25, 34, 41] to make it better suited for surface extraction. UNISURF [25] uses an occupancy network [21] whereas VolSDF [41] and NeuS [34] use an SDF [26]. We build our method on VolSDF [41] but we believe it could be adapted to fit any volumetric neural implicit framework.

Image warping and neural implicit surfaces: In this work we combine the color matching idea of traditional MVS with neural implicit surfaces. Closest to this idea is MVSDF [45] that also uses a loss based on correspondences and works on accurate geometry optimization. However, it optimizes consistency between CNN features and the optimization requires a network pretrained on multi-view datasets [40]. Our approach does not require any pretrained network and we show that it outperforms MVSDF.

The idea of projecting information from source views to 3D then using the neural radiance field framework to render a target view has also been used in learning-based approaches [3, 4, 6, 33, 35, 43]. These approaches train on multiple scenes networks that take as input features from the source views aggregated at a given 3D point and output radiance and occupancy for this point. They focus however on the generalization to new scenes and the quality of rendered views, but the quality of their predicted geometry has not been evaluated, to the best of our knowledge. On the

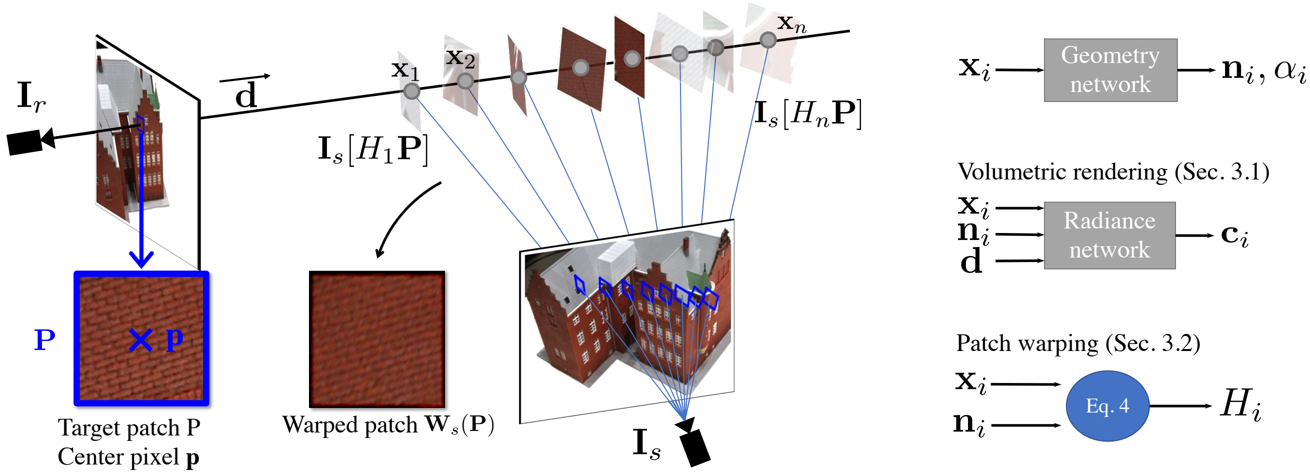


Figure 2. Approach overview. We combine volumetric rendering with a new patch warping technique. Both approaches aggregate color from points sampled along the camera ray: radiance predicted by the radiance network for volumetric rendering and patch extracted from source views for our patch warping.

contrary, we focus on the optimization framework, i.e., we do not train our network on several scenes, and we optimize the quality of geometry.

3. Method

In this section, we present our technical contributions. Section 3.1 introduces the volumetric rendering framework on which we build. Section 3.2 explains how we warp a patch from a source image to a target image given a 3D scene represented by a geometry network predicting occupancy for each 3D point. Section 3.3 discusses questions related to visibility and how we mask invalid points during the optimization. Finally, Section 3.4 presents our full optimization. An overview of our approach and notations can be seen in Figure 2.

3.1. Volumetric rendering of radiance field

Neural volumetric rendering was introduced in [22] for novel view rendering. The idea is to represent the characteristics of a 3D scene with two implicit functions that are approximated with neural networks. The geometry network encodes the geometry of the scene, we use Signed Distance Field (SDF) encoding [34, 41]. The radiance network encodes the color emitted by any region in space in all directions. The idea is to optimize the two neural networks together so that rendering the associated scene reconstructs a set of given views of the scene. Let us consider a reference image $\mathbf{I}_r$. The two networks are optimized using $\ell 1$ loss between the colors in reference image $\mathbf{I}_r[\mathbf{p}]$ and the volumetric rendering $R[\mathbf{p}]$ for a pixel $\mathbf{p}$:

$$\mathcal{L}_{vol} = \sum_{\mathbf{p}} |\mathbf{I}_r[\mathbf{p}] - \mathbf{R}[\mathbf{p}]|. \tag{1}$$

The rendered color $\mathbf{R}[\mathbf{p}]$ is computed from both networks in a differentiable way with respect to their parameters using volumetric rendering [16]. Let $\mathbf{x}_i, i = 1 \dots n$ be an ordered set of points sampled along the ray going through the reference camera center and the pixel $\mathbf{p}$.[1] The rendering of the scene at pixel $\mathbf{p}$ is approximated as a weighted sum of the radiance $\mathbf{c}_i$ at each point using weights computed from the geometry network. Intuitively, the color $\mathbf{c}_i$ will contribute to the rendering if $\mathbf{x}_i$ has a high density and if no point on the ray between $\mathbf{x}_i$ and the reference camera has a high density value. Formally, $\mathbf{c}_i = \mathbf{c}(\mathbf{x}_i, \mathbf{n}_i, \mathbf{d})$ is the radiance computed with the radiance network $\mathbf{c}$ in ray direction $\mathbf{d}$, at the points $\mathbf{x}_i$ of surface normals $\mathbf{n}_i$, computed by differentiation of the geometry network at the different positions $\mathbf{x}_i$. The rendered color is approximated with an alpha blending of the $\mathbf{c}_i$.

$$\mathbf{R}[\mathbf{p}] = \sum_{i=1}^{N} \alpha_i \prod_{j<i} (1 - \alpha_j) \mathbf{c}_i, \tag{2}$$

where we consider for simplicity that the geometry network α outputs occupancy values $\alpha_i = \alpha(\mathbf{x}_i)$ between 0 and 1. In practice, our geometry network outputs an SDF, and we refer to [41] for a detailed explanation of the mapping of SDF to occupancy values. Eq. (2) is the discrete approximation of an integral along the camera ray. Therefore, the choice of sampling points $\mathbf{x}_i$ is a key element, discussed in [22, 25, 34, 41]. Those methods improve the geometry estimation by focusing on the sampling, but the reconstructions are still worse than the traditional MVS techniques. Our hypothesis is that it comes from the difficulty of the radiance network to represent high frequency textures (see Figure 1).

[1] For simplicity, we drop from the point notations the dependency on the pixel to render $\mathbf{p}$ and the reference image index r.

3.2. Warping images with implicit geometry

Instead of memorizing all the color information present in the scene with the radiance network, we propose to directly warp images onto each other relying only on the geometry network. We consider a reference image $\mathbf{I}_r$ and a source image $\mathbf{I}_s$. Similar to the above section, we want to obtain the color of a pixel $\mathbf{p}$ or a patch centered around $\mathbf{p}$ using the occupancies $\alpha_i, i = 1 \dots N$ from the geometry network but this time using colors from projections from the source image and not the one predicted by the radiance network. In this section, we assume these projections and their colors are well defined, and deal with the general case in Section 3.3. We start by explaining how a source image can be warped to a target image pixel-by-pixel, which we refer to as pixel warping. We then extend this idea to warping full patches, a classical idea in MVS [10, 11, 29, 46].

Pixel warping: Instead of using a radiance network to compute the color of each 3D point $\mathbf{x}_i$ on the ray associated to pixel $\mathbf{p}$, we use the color of their projection in source image. Formally, we define the warped value of $\mathbf{p}$ from source image s as:

$$\mathbf{W}_s[\mathbf{p}] = \sum_{i=1}^{N} \alpha_i \prod_{j<i} (1 - \alpha_j) \mathbf{I}_s[\pi_s(\mathbf{x}_i)], \tag{3}$$

where $\mathbf{I}_s[\pi_s(\mathbf{x})]$ denotes the bilinear interpolation of colors from $\mathbf{I}_s$ at the point $\pi_s(\mathbf{x})$ where the 3D point $\mathbf{x}$ projects in $\mathbf{I}_s$. In this section, we assume that every 3D points has a valid projection in source image so that $\mathbf{I}_s[\pi_s(\mathbf{x})]$ is always defined. Eq. (3) is similar to Eq. (2) but the color comes from pixel values in source images instead of network predictions. Intuitively, the warped value is a weighted average of the source image colors along the epipolar line. Similar to Eq. 1, one could optimize the geometry using a $\ell 1$ loss function $\ell = \sum_{\mathbf{p}} |\mathbf{W}_s[\mathbf{p}] - I_r[\mathbf{p}]|$. However, this does not model changes in intensity related to the camera viewpoint, and in particular specularities, and can create artifacts in the reconstruction. One solution to this issue is to use a robust patch-based photometric loss function.

Patch warping: We now explain how to warp entire patches instead of single pixels, by locally approximating the scene at each point $\mathbf{x}_i$ as a plane in a way similar to standard techniques used in classical MVS [10]. Let $\mathbf{n}_i$ be the surface normal at $\mathbf{x}_i$, which can be computed with automatic differentiation of the geometry network at $\mathbf{x}_i$. Let H_i be the homography between $\mathbf{I}_r$ and $\mathbf{I}_s$ induced by the plane through $\mathbf{x}_i$ of normal $\mathbf{n}_i$. It can be computed as a 3×3 matrix acting on 2D homogeneous coordinates:

$$H_i = K_s \left(R_{rs} + \frac{\mathbf{t}_{rs} \mathbf{n}_i^T R_r^T}{\mathbf{n}_i^T (\mathbf{x}_i + R_r^T \mathbf{t}_r)} \right) K_r^{-1}, \tag{4}$$

where K_r and K_s are the internal calibration matrices of reference and source cameras, $(R_{rs}, \mathbf{t}_{rs})$ is the relative motion from $\mathbf{I}_r$ to $\mathbf{I}_s$ represented by a 3×3 rotation matrix R_{rs} and a 3D translation vector $\mathbf{t}_{rs}$ and $(R_r, \mathbf{t}_r)$ is the pose of reference frame in world coordinates with the same representation. This homography associates any pixel $\mathbf{q}$ in $\mathbf{I}_r$ to a pixel $H\_i * q$ in $\mathbf{I}_s$.

With a slight abuse we extend this notation to patches: for a patch $\mathbf{P}$ centered around pixel $\mathbf{p}$, we write $H_i\mathbf{P}$ the application of the homography to all pixels of the patch and $\mathbf{I}_s[H_i\mathbf{P}]$ the color interpolated at those locations in $\mathbf{I}_s$. Intuitively, $H_i\mathbf{P}$ is the location of a patch in source image that would correspond to $\mathbf{P}$ in reference frame if the true geometry was a plane of normal $\mathbf{n}_i$ passing through $\mathbf{x}_i$. We can now average the patches corresponding to each $\mathbf{x}_i$ in a manner similar to Eq. (2) to produce a warped patch $\mathbf{W}_s[\mathbf{P}]$:

$$\mathbf{W}_s[\mathbf{P}] = \sum_{i=1}^{N} \alpha_i \prod_{j<i} (1 - \alpha_j) \mathbf{I}_s[H_i\mathbf{P}]. \tag{5}$$

In all our experiments we follow COLMAP [29] and use a patch size of 11×11. Note that using a patch size of 1 would provide the same equation as Eq. (3).

3.3. Optimizing geometry from warped patches

We now want to define a loss based on the warped patches to optimize the geometry. This cannot be done by using directly (5) to warp patches and maximizing SSIM. Indeed, we assumed that all 3D points $\mathbf{x}_i$ on the camera ray have a valid projection in the source image, but in practice this is not true for many points, which may project outside of the source image for example. In that case $\mathbf{I}_s[H_i\mathbf{P}]$ is not defined, and we instead use a constant (gray) padding color in (5). This will of course affect the quality of the warped patch $\mathbf{W}_s[\mathbf{P}]$. Intuitively, if all non-valid 3D points on a ray are far from the implicit surface seen in the reference image, the padding value will contribute very little to the final warped patch which can be used in the loss. On the contrary, if there are invalid points near the implicit surface, the warped patch becomes invalid and should not be used in the loss. We formalize this intuition by assigning to a patch $\mathbf{P}$ centered around pixel $\mathbf{p}$ in the reference image a mask value $M_s[\mathbf{P}] \in [0, 1]$ for each source image s. In the rest of the section, we first explain how we define our loss for a given reference image based on the validity masks M_s associated with each source image; We then explain how we define the validity masks.

Warping-based loss: We start by selecting for a given reference image the set $\mathcal{V}$ of patches we consider in the loss. Since we want to discard patches $\mathbf{P}$ for which no source image gives ant reasonable warp, as quantified by $M_s(\mathbf{P})$, we define it as $\mathcal{V} = \{\mathbf{P} : \sum_s M_s(\mathbf{P}) > \epsilon\}$. In practice, we use $\epsilon = 0.001$ in all of our experiments. We then define

Scan	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	**Mean**
IDR [42]	1.63	1.87	0.63	0.48	1.04	0.79	0.77	1.33	1.16	0.76	0.67	0.90	0.42	0.51	0.53	0.90
MVSDF [45]*	0.83	1.76	0.88	0.44	1.11	0.90	0.75	1.26	1.02	1.35	0.87	0.84	0.34	0.47	0.46	0.88
COLMAP [29]	0.45	0.91	0.37	0.37	0.90	1.00	0.54	1.22	1.08	0.64	0.48	0.59	0.32	0.45	0.43	0.65
NeRF [22]	1.90	1.60	1.85	0.58	2.28	1.27	1.47	1.67	2.05	1.07	0.88	2.53	1.06	1.15	0.96	1.49
UNISURF [25]	1.32	1.36	1.72	0.44	1.35	$\underline{0.79}$	$\underline{0.80}$	1.49	1.37	0.89	**0.59**	1.47	0.46	$\underline{0.59}$	0.62	1.02
NeuS [34]	1.37	$\underline{1.21}$	$\underline{0.73}$	$\underline{0.40}$	$\underline{1.20}$	**0.70**	**0.72**	**1.01**	$\underline{1.16}$	0.82	$\underline{0.66}$	1.69	**0.39**	**0.49**	**0.51**	0.87
VolSDF [41]	$\underline{1.14}$	1.26	0.81	0.49	1.25	**0.70**	**0.72**	1.29	1.18	$\underline{0.70}$	$\underline{0.66}$	$\underline{1.08}$	0.42	0.61	$\underline{0.55}$	$\underline{0.86}$
NeuralWarp (ours)	**0.49**	**0.71**	**0.38**	**0.38**	**0.79**	0.81	0.82	$\underline{1.20}$	**1.06**	**0.68**	$\underline{0.66}$	**0.74**	$\underline{0.41}$	0.63	**0.51**	**0.68**

Table 1. Quantitative comparison on DTU. All the results are the ones reported in the original papers, except NeRF, IDR and COLMAP results that come from [25]. Note that MVSDF results (*) are not directly comparable since they use a custom filtering whereas results with all other methods have been cleaned using the visual hull. The bottom part of the table compares our result with neural implicit surfaces approaches that do not use additional inputs (masks, supervised depth estimation). **Bold** results have the best score and underlined the second best. Our method outperforms existing work by a large margin, more than 20% on the mean metrics.

our warping-based loss such that every valid patch in the reference image is given the same weight, but also such that invalid warping are given less weight:

$$\mathcal{L}_{\text{warp}} = \sum_{\mathbf{P}\in\mathcal{V}} \frac{\sum_{s\in\mathcal{S}} M_s[\mathbf{P}]\, d(\mathbf{I}_r[\mathbf{P}], \mathbf{W}_s[\mathbf{P}])}{\sum_{s\in\mathcal{S}} M_s[\mathbf{P}]} \tag{6}$$

where $\mathbf{I}_r[\mathbf{P}]$ is the color patch $\mathbf{P}$ in $\mathbf{I}_r$ and d is a photometric distance between image patches. We use for d the SSIM [36], except for our ablation where we use $\ell 1$.

Validity masks: We now explain how we define the validity mask M_s. We consider two reasons for warps not being valid, hence two masks: (i) a projection mask M_s^{proj} for cases in which the projection is not valid for geometric reasons, and (ii) an occlusion mask M_s^{occ} for cases where the patch is occluded by the reconstructed scene in the source image. The final mask is the product of both: $M_s[\mathbf{P}] = M_s^{\text{proj}}[\mathbf{P}]M_s^{\text{occ}}[\mathbf{P}]$.

To define the projection mask, we introduce a binary indicator V_i^s which is 0 when the projection associated with $\mathbf{x}_i$ is not valid and 1 otherwise. The projection can be invalid for three reasons: first, when the projection of the point in the source view is outside the source image; second, when the reference and source views are on two different sides of the plane defined by $\mathbf{x}_i$ and the normal $\mathbf{n}_i$; third, when a camera center is too close to the plane defined by $\mathbf{x}_i$ and the normal $\mathbf{n}_i$, for which we use a threshold of 0.001 in practice. We obtain $M_s^{\text{proj}}[\mathbf{P}]$ by averaging the validity indicators for all 3D points sampled on the ray associated to $\mathbf{P}$ weighted by the α values:

$$M_s^{\text{proj}}[\mathbf{P}] = \sum_{i=1}^{N} \alpha_i \prod_{j<i}(1-\alpha_j)V_i^s \tag{7}$$

Note that (7) produces a soft mask value between 0 and 1, which is necessary to make it differentiable with respect to the α factors. We found such a property to be important in practice.

To define the occlusion mask M_s^{occ} we check whether there are occupied regions on the ray between the points $\mathbf{x}_i$ and the source camera center. We compute how occluded a 3D point $\mathbf{x}$ is with its transmittance in source view: $T_s(\mathbf{x}) = 1 - \prod_{k=1}^{N}(1-\alpha_k^s)$, where the α_k^s are the occupancy values predicted by the geometry network on 3D points sampled on the ray from $\mathbf{x}$ to the center of view s. Intuitively, $T_s(\mathbf{x})$ is close to 1 if there is no point with a large density between the source image and $\mathbf{x}$, otherwise it is close to 0.We could average the $T_s(\mathbf{x}_i), i = 1 \dots N$ in the same manner as Eq. (7) but this would require computing the transmittance T_s for every point on the ray. For computational efficiency we instead choose to compute an intersection point on the ray corresponding to the patch $\mathbf{P}$ in the reference view and evaluate transmittance in the source views on this point only:

$$M_s^{\text{occ}}[\mathbf{P}] = T_s\left(\sum_{i=1}^{N} \alpha_i \prod_{j<i}(1-\alpha_j)\mathbf{x}_i\right) \tag{8}$$

This mask is again soft because T_s outputs a continuous value in range $[0, 1]$, which helps handling thin surfaces occlusion: if a ray comes close to a surface without being strictly occluded, it will still have a lesser influence on the warping loss compared to a ray that is far from any surface.

3.4. Optimization details

We now explain the details of our method. We first present out full loss and optimization. We then detail the network architecture. We finally discuss how we selected the source images for each reference image.

Full optimization: We optimize the geometry and radiance networks to minimize the sum of the volumetric rendering loss (Eq. 1) and the patch warping loss (Eq. 6). In order to encourage the geometry network to output a function similar to a signed distance field, we also add the eikonal loss $\mathcal{L}_{\text{eik}}$ [12] which is minimum when the gradient of the output function at each point in space is of norm 1. This results in

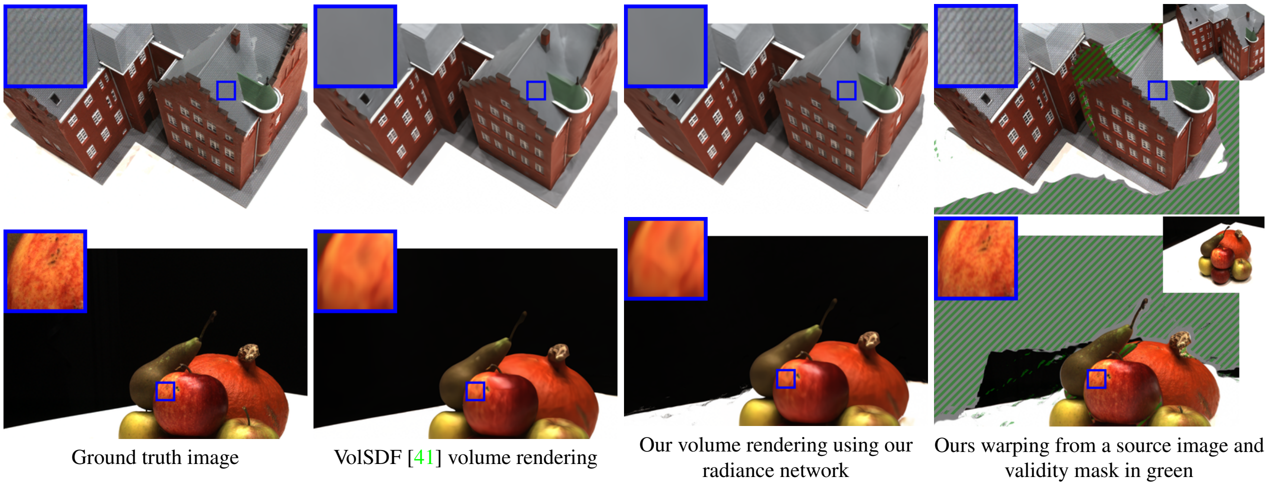


Figure 3. Examples of DTU renderings using VolSDF [41] and our method, both using our implementation. For both methods, volumetric rendering cannot produce high frequency texture whereas image warping can precisely recover them. (better viewed in electronic version)

the following complete loss:

$$\mathcal{L} = \lambda_{\text{vol}}\mathcal{L}_{\text{vol}} + \lambda_{\text{warp}}\mathcal{L}_{\text{warp}} + \lambda_{\text{eik}}\mathcal{L}_{\text{eik}} \tag{9}$$

where λ_{vol}, λ_{warp} and λ_{eik} are scalar hyperparameters. We use $\lambda_{\text{vol}} = 1$ and $\lambda_{\text{eik}} = 0.1$ for all experiments. We first optimize the networks using $\lambda_{\text{warp}} = 0$ in the same setting as VolSDF. After $100k$ iterations with a learning rate exponentially decayed from $5\text{e}{-4}$ to $5\text{e}{-5}$, we finetune for another $50k$ iterations using $\lambda_{\text{warp}} = 1$ and a fixed learning rate of $1\text{e}{-5}$. The networks are initialized with the sphere initialization of [2]. We start optimizing with volumetric rendering only because the normals are initially too noisy to compute meaningful homographies. During the first phase of training, without patch warping, we train with batches of 1024 pixels, but we finetune patch warping on batches of 512 patches due to GPU memory constraints. Also, we do not backpropagate the loss through the homography parameters in equation (4) since we noticed it leads to unstable optimization but the geometry network is still optimized through the α factors of Eq. (5).

Architecture: We use the same architecture as concurrent works [25, 41]. Both radiance and geometry networks are Multi-Layer Perceptrons (MLP). The geometry network has 8 layers with 256 hidden units. The radiance network has 4 layers with 256 hidden units. Similar to [25, 41, 42] we encode 3D position using positional encoding with 6 frequencies and viewing direction with 4 frequencies.

Rays sampling: We follow VolSDF [41] in choosing the points $\mathbf{x}_1 \dots \mathbf{x}_N$ on the camera rays with a small modification. We first estimate the opacity function with the algorithm introduced by VolSDF, we then sample $N = 64$ points on the camera ray with 90% sampled from the opacity distribution as in VolSDF but the remaining 10% sampled uniformly along the whole camera ray.

Choice of source images: Our method uses a set of 19 source images $\mathcal{S}$ following COLMAP [29] for each reference image. Those source images must be carefully chosen since very similar viewpoints will carry little geometric information and very different viewpoints will have few common points. We first build a sparse point cloud with a Structure from Motion software [28], we then compute for each image pair the total number of sparse points observed by both (co-visible points) and remove the pairs for which more than 75% of the co-visible points are observed with a triangulation angle below 5° . We finally select the 19 top views in number of co-visible points.

4. Experiments

In this section, we first show that our method outperforms state-of-the-art unsupervised neural implicit surface approaches on the DTU dataset [14], then on the EPFL benchmark [32], we present an ablation study to evaluate each of our technical contributions and finally we discuss the limitations of our method.

DTU benchmark: The DTU benchmark [14] includes scenes with 49 to 64 images associated to reference point clouds acquired with laser sensor. Each scene covers a different object: some have challenging specular materials while others have large textureless regions. The evaluation of IDR [42] selected 15 scenes and manually annotated object masks. We compare our method with existing work on the same scenes, using DTU evaluation code. The metric is the average of accuracy and completeness: the chamfer distance of prediction to reference point cloud and inversely. Similar to existing work [25, 34, 41, 42], we clean the output meshes with the visibility masks dilated by 12 pixels.

We compare with multiple baselines in Table 1. The re-

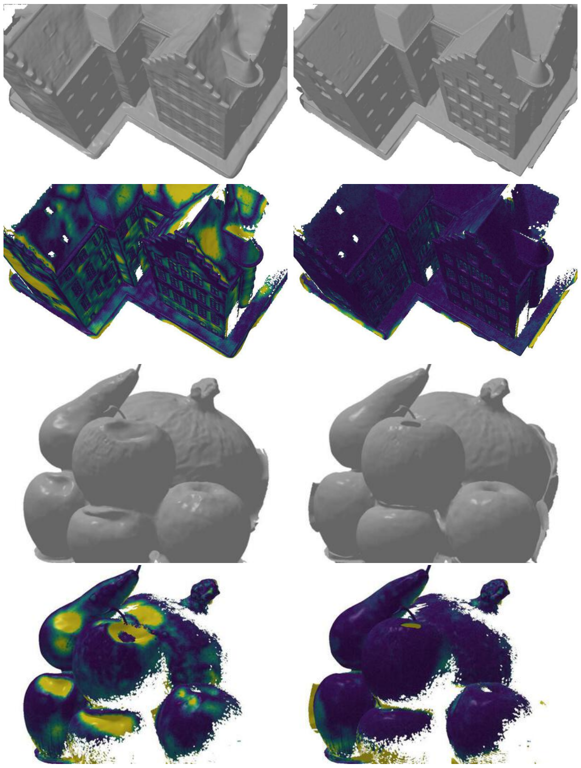


Figure 4. Rendering and geometric error maps of two DTU scenes [14] (blue=low, yellow=high). Compared to VolSDF [41] on which we build, our method significantly improves accuracy.

sults for each method are taken from their original paper, except COLMAP, NeRF and IDR that we took from [25]. Similar to [25, 34, 41], we only compare in the bottom part of the table deep implicit surface approaches that do not use masks or other data during training. In particular, MVSDF [45] uses a supervised depth estimation network. Our method outperforms existing methods by a large margin. As could be expected, the improvement is more important on highly textured scenes but our method performs on par with other methods on weakly textured scenes. Figure 3 compares original images, volume rendering obtained by VolSDF, volume rendering obtained with our radiance network and our image warping (using the pixel warping approach described in Section 3.2). Volumetric rendering only renders smoothed texture, whereas our warping is able to render high-frequency texture information. As can be qualitatively seen in Figures 1 and 4, this leads to important improvements in accuracy. Reconstructions and geometric error maps for all scenes are shown in the supplementary material.

EPFL benchmark: The EPFL benchmark [32] consists of two outdoor scenes of 7 and 11 high resolution images with a high resolution ground truth mesh. Since the extent of the ground truth does not exactly overlap the cameras

Method	Fountain-P11		Herzjesu-P7		**Mean**	
	Full	Center	Full	Center	**Full**	**Center**
COLMAP [29]	**6.47**	<u>2.45</u>	**7.95**	<u>2.31</u>	**7.21**	<u>2.38</u>
UNISURF [25]	26.16	17.72	27.22	13.72	26.69	15.72
MVSDF [45]	<u>6.87</u>	2.26	11.32	2.72	9.10	2.49
VolSDF [41]	12.89	2.99	13.61	4.58	13.25	3.78
NeuralWarp (ours)	7.77	**1.92**	<u>8.88</u>	**2.03**	<u>8.32</u>	**1.97**

Table 2. Quantitative evaluation on EPFL dataset. We use the chamfer distance on the Full scene (Full) and on a manually defined bounding box at the center of the scene (Center). Results of VolSDF [41] come from our implementation, MVSDF reconstructions were sent by the authors and and we ran MVSDF and COLMAP public implementations. Although COLMAP is the best method for the full reconstruction, our method has the best results for the center metrics. It outperforms existing neural implicit surfaces by 20% on center metrics.

viewing angle and inversely, it is necessary to remove points from both predicted and ground truth meshes for evaluation. MVSDF [45] uses manual masks to remove vertices from the ground truth mesh, which we argue might be biased. We instead automatically remove vertices from the ground truth mesh when they do not project in any input image. Similar to DTU evaluation, we use silhouette masks to clean the predicted mesh with the scene visual hull. We generate silhouette masks by rendering the ground truth mesh on each input viewpoint and marking pixels which are not covered as outside of the silhouette. Finally, we also remove from the predicted mesh any triangle that is not rendered in any image, which removes in particular faces closing the volume behind the object. To compute the distance between the filtered ground truth and predicted mesh, we sample 1 million point from each and compute their chamfer distance. We call this metrics the *full* chamfer distance. It is mainly influenced by the completeness of the reconstruction, e.g. it compares how well methods reconstruct the ground plane or rarely seen points. We therefore introduce another metric referred to as the *center* chamfer distance which only evaluates the chamfer distance in a box at the center of the scene which we manually defined so that it only includes the central part of the scene, which is reconstructed by all methods. Thus, this metric focuses more on the accuracy of the reconstruction.

We compare our method with several baselines with these two metrics in Table 2. We ran COLMAP followed by sPSR [17] with trim 5, used the official UNISURF implementation,[2] evaluated the MVSDF meshes communicated by the authors, and ran our own reimplementation of VolSDF. Qualitative comparisons between the reconstructions and error maps for each method can be seen in Figure 5. Similar to the DTU results, our method outperforms other neural implicit surfaces by more than 20%. COLMAP [29] is the best method for the full metric. This is in large part because it is

[2] https://github.com/autonomousvision/unisurf

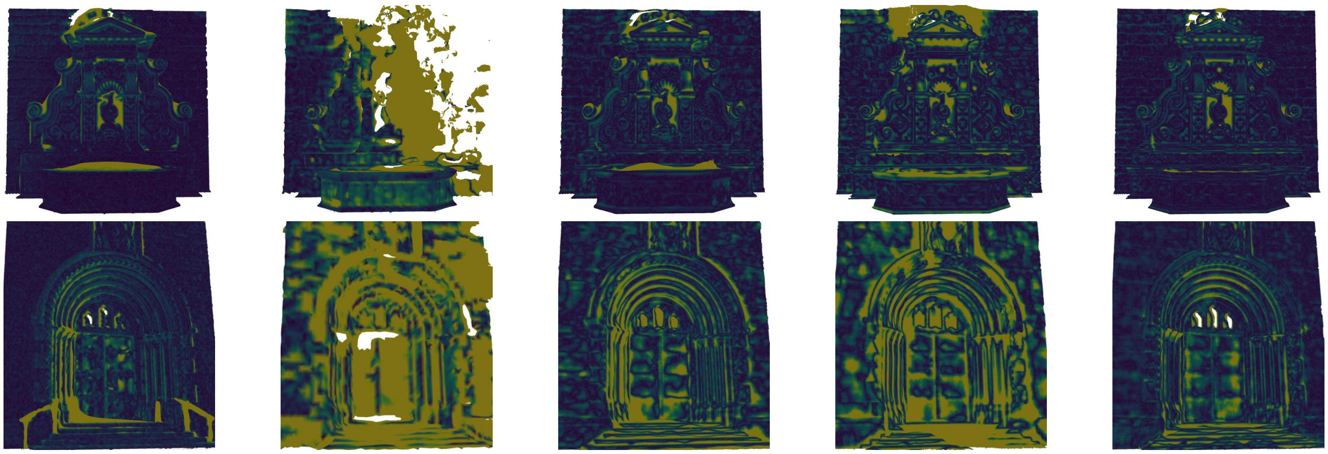


Figure 5. Geometric error maps of reconstructed meshes on EPFL benchmark [32] (blue=low, yellow=high). We ran COLMAP and UNISURF official implementations, MVSDF results were sent by the authors and we ran VolSDF with our implementation. Our method produces better results than other neural implicit surface approaches, almost on par with COLMAP [29].

Method	$\mathcal{L}_{\text{vol}}$	$\mathcal{L}_{\text{warp}}$	M_s^{occ}	Chamfer dist.
VolSDF [41]	✓	None		0.85
Pixel	✓	Pixel	✓	0.83
Patch no occ.	✓	Patch		0.74
Patch no vol.		Patch	✓	0.74
NeuralWarp (full)	✓	Patch	✓	**0.68**

Table 3. Ablation study on all scenes of DTU: $\mathcal{L}_{\text{vol}}$ denotes volumetric rendering, $\mathcal{L}_{\text{warp}}$ warping consistency for which we try none, pixel and patch warpings. M_s^{occ} denotes whether we detect self occlusions or not.

the only method able to reconstruct accurately the ground plane on both scenes. For the center metrics however, our method outperforms even COLMAP, though this might be due to some details reconstructed by COLMAP (e.g. railing) not being included in the ground truth: qualitatively, COLMAP still seems to recover finer details.

Ablation study: To evaluate the effect of our technical contributions, we perform an ablation study on the DTU dataset. Starting from the same models trained without photometric consistency, we finetune different versions of our model for 50000 iterations and compare the results. The average chamfer distance over all 15 scenes is shown in Table 3 and we report the results on each scene in the supplementary material. We first compare the results without our warping loss ('VolSDF [41]' line), with pixel warping ('Pixel' line) and with patch warping ('NeuralWarp (full)' line). Both pixel and patch warping improve the results, with a clear advantage for patches. We then evaluate the importance of masking. Removing the projection mask (not reported in the table) does not lead to meaningful reconstructions. Without the occlusion mask ('Patch no occ.' line) our method still improves over the baseline, but by a smaller margin. Finally, we tried to completely remove volumetric rendering loss, that is, use $\lambda_{\text{vol}} = 0$ ('Patch no vol.' line). Again, this improves over the baseline but is worse than combining the volumetric and warp losses.

Limitations: Our method has several limitations. First, compared to COLMAP, it struggles to reconstruct high-frequency geometry. We believe this is due to the difficulty of optimizing a geometry network at high resolution. Second, computing our loss increases the computational cost of the optimization, in particular the occlusion mask adds processing time and processing patches increases memory footprint. Finally, simply comparing patches does not model reflections, which can lead to artifacts even using a robust patch similarity. We show such an example in supplementary material.

5. Conclusion

We have presented a new method to perform multiview reconstruction with implicit functions, using image warpings in combination with volumetric rendering. Unlike existing neural implicit surface methods, our approach can easily take advantage of high-frequency texture. We show this leads to strong performance improvements on the classical DTU and EPFL datasets.

Acknowledgments

This work was supported in part by ANR project EnHerit ANR-17-CE23-0008 and was performed using HPC resources from GENCI–IDRIS 2021-AD011011756R1. We thank Tom Monnier and Bruno Lecouat for valuable feedback and Jingyang Zhang for sending MVSDF results.

References

[1] Matan Atzmon, Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron, and Yaron Lipman. Controlling neural level sets. *Adv. Neural Inform. Process. Syst.*, 2019. 2

[2] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 6

[3] Alexander W. Bergman, Petr Kellnhofer, and Gordon Wetzstein. Fast training of neural lumigraph representations using meta learning. In *Adv. Neural Inform. Process. Syst.*, 2021. 2

[4] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Int. Conf. Comput. Vis.*, 2021. 2

[5] Rui Chen, Songfang Han, Jing Xu, and Hao Su. Point-based multi-view stereo network. In *Int. Conf. Comput. Vis.*, 2019. 2

[6] Julian Chibane, Aayush Bansal, Verica Lazova, and Gerard Pons-Moll. Stereo radiance fields (srf): Learning view synthesis from sparse views of novel scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[7] Yuchao Dai, Zhidong Zhu, Zhibo Rao, and Bo Li. Mvs2: Deep unsupervised multi-view stereo with multi-view symmetry. In *Int. Conf. on 3D Vision*, 2019. 2

[8] François Darmon, Bénédicte Bascle, Jean-Clément Devaux, Pascal Monasse, and Mathieu Aubry. Deep multi-view stereo gone wild. In *Int. Conf. on 3D Vision*, 2021. 2

[9] Yasutaka Furukawa and Carlos Hernández. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2015. 2

[10] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. In *IEEE Trans. Pattern Anal. Mach. Intell.*, volume 32, 2009. 2, 4

[11] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Int. Conf. Comput. Vis.*, 2015. 2, 4

[12] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. 2020. 5

[13] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[14] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014. 2, 6, 7

[15] Mengqi Ji, Juergen Gall, Haitian Zheng, Yebin Liu, and Lu Fang. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In *Int. Conf. Comput. Vis.*, 2017. 2

[16] James T Kajiya and Brian P Von Herzen. Ray tracing volume densities. *ACM SIGGRAPH*, 1984. 2, 3

[17] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 2013. 2, 7

[18] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 2017. 2

[19] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *Int. J. Comput. Vis.*, pages 199–218, 2000. 2

[20] Patrick Labatut, Jean-Philippe Pons, and Renaud Keriven. Efficient multi-view reconstruction of large-scale scenes using interest points, delaunay triangulation and graph cuts. In *Int. Conf. Comput. Vis.*, 2007. 2

[21] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 4460–4470, 2019. 2

[22] 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 *Eur. Conf. Comput. Vis.*, 2020. 1, 2, 3, 5

[23] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. In *Eur. Conf. Comput. Vis.*, 2020. 2

[24] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 1, 2

[25] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Int. Conf. Comput. Vis.*, 2021. 1, 2, 3, 5, 6, 7, 8

[26] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[27] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Eur. Conf. Comput. Vis.*, 2020. 2

[28] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2016. 6

[29] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *Eur. Conf. Comput. Vis.*, pages 501–518, 2016. 1, 2, 4, 5, 6, 7, 8

[30] Steven M Seitz and Charles R Dyer. Photorealistic scene reconstruction by voxel coloring. *Int. J. Comput. Vis.*, pages 151–173, 1999. 2

[31] Ayan Sinha, Zak Murez, James Bartolozzi, Vijay Badrinarayanan, and Andrew Rabinovich. Deltas: Depth estimation by learning triangulation and densification of sparse points. In *Eur. Conf. Comput. Vis.*, 2020. 2

[32] Christoph Strecha, Wolfgang Von Hansen, Luc Van Gool, Pascal Fua, and Ulrich Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2008. 2, 6, 7, 8

[33] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[34] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit

surfaces by volume rendering for multi-view reconstruction. In *Adv. Neural Inform. Process. Syst.*, 2021. 1, 2, 3, 5, 6, 7

[35] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[36] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. In *IEEE Trans. Image Process.*, 2004. 1, 5

[37] Hongbin Xu, Zhipeng Zhou, Yu Qiao, Wenxiong Kang, and Qiuxia Wu. Self-supervised multi-view stereo via effective co-segmentation and data-augmentation. In *AAAI*, 2021. 2

[38] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. MVSNet: Depth inference for unstructured multi-view stereo. *Eur. Conf. Comput. Vis.*, 2018. 2

[39] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent MVSNet for high-resolution multi-view stereo depth inference. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 2

[40] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. BlendedMVS: A large-scale dataset for generalized multi-view stereo networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2

[41] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Adv. Neural Inform. Process. Syst.*, 2021. 1, 2, 3, 5, 6, 7, 8

[42] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. In *Adv. Neural Inform. Process. Syst.*, 2020. 1, 2, 5, 6

[43] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 2

[44] Jingyang Zhang, Yao Yao, Shiwei Li, Zixin Luo, and Tian Fang. Visibility-aware multi-view stereo network. In *Brit. Mach. Vis. Conf.*, 2020. 1, 2

[45] Jingyang Zhang, Yao Yao, and Long Quan. Learning signed distance field for multi-view surface reconstruction. In *Int. Conf. Comput. Vis.*, 2021. 2, 5, 7, 8

[46] Enliang Zheng, Enrique Dunn, Vladimir Jojic, and Jan-Michael Frahm. PatchMatch based joint view selection and depthmap estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2014. 2, 4