Surface-Centric Modeling for High-Fidelity Generalizable Neural Surface Reconstruction

Rui Peng[1,2], Shihe Shen[1], Kaiqiang Xiong[1], Huachen Gao[1], Jianbo Jiao[3], Xiaodong Gu[4], and Ronggang Wang[✉1,2]

[1]School of Electronic and Computer Engineering, Peking University
[2]Peng Cheng Laboratory [3]University of Birmingham [4]Alibaba
ruipeng@stu.pku.edu.cn rgwang@pkusz.edu.cn

Abstract. Reconstructing the high-fidelity surface from multi-view images, especially sparse images, is a critical and practical task that has attracted widespread attention in recent years. However, existing methods are impeded by the memory constraint or the requirement of ground-truth depths and cannot recover satisfactory geometric details. To this end, we propose *SuRF*, a new Surface-centric framework that incorporates a new Region sparsification based on a matching Field, achieving good trade-offs between performance, efficiency and scalability. To our knowledge, this is the first unsupervised method achieving end-to-end sparsification powered by the introduced matching field, which leverages the weight distribution to efficiently locate the boundary regions containing surface. Instead of predicting an SDF value for each voxel, we present a new region sparsification approach to sparse the volume by judging whether the voxel is inside the surface region. In this way, our model can exploit higher frequency features around the surface with less memory and computational consumption. Extensive experiments on multiple benchmarks containing complex large-scale scenes show that our reconstructions exhibit high-quality details and achieve new state-of-the-art performance, *i.e.*, 46% improvements with 80% less memory consumption. Code is available at `https://github.com/prstrive/SuRF`.

Keywords: Surface reconstruction · sparsification · sparse views

1 Introduction

Reconstructing surface from multi-view images is a fundamental and challenging task in computer vision with wide-ranging applications, including autonomous driving, robotics, virtual reality, and more. While many typical methods [12, 16, 38, 40, 51] have achieved satisfactory results through tedious multi-stage processes (*i.e.*, depth estimation, filtering and meshing), recent neural implicit methods [30, 33, 44, 53, 54, 58] attract increasing attention due to their concise procedures and impressive reconstructions. They can directly extract the geometry through Marching Cube [26], and avoid the accumulated errors. Despite their

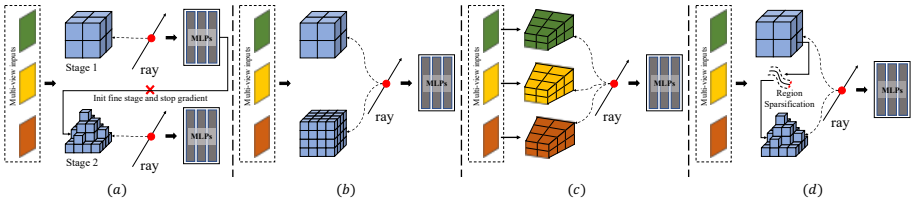


Fig. 1: Pipeline comparison with existing methods. We omit the supervised depth-fusion methods [23, 39] and only visualize two scales or stages here for convenience. (a) The multi-stage pipeline like SparseNeuS [25] is not end-to-end and tends to accumulate errors, whose coarse stages cannot be optimized together with fine stages and can no longer be corrected. (b) To achieve end-to-end training, methods like GenS [36] applied the multi-scale structure to concatenate the coarse and fine volumes together, but the memory constraints limit the volume resolution. (c) View-frustum based methods like C2F2NeuS [50] construct a separate cost volume for each view, which consumes much memory and computation, especially when there are many input views. On the contrary, we design an end-to-end and sparse pipeline (d), which can leverage higher-resolution volumes with less memory and computational consumption, and the coarse model can be optimized together with the fine model.

effectiveness, these methods are hampered by the cumbersome per-scene optimization and the requirement of a large number of input views, which makes them unsuitable for many applications. Even recent fast methods like [46,49] and 3D Gaussian Splatting methods [6, 13, 17] struggle to extract meshes in seconds and perform poorly under sparse input.

Recently, some generalizable neural surface methods [23, 25, 36, 39, 50] have been proposed to mitigate these problems by combining neural implicit representations with prior image information. However, as the pipeline comparisons shown in Fig. 1, they either rely on the non-end-to-end pipeline that leads to accumulated errors, or require constructing the dense volumes (or even separate volumes) for each view and consumes excessive memory and computation. We are interested in the question: *why unsupervised end-to-end sparsification has not been achieved yet?* To sparse the volume for the next fine model initialization, previous methods like SparseNeuS [25] require predicting SDF values for a large number of voxels and determining whether the SDF values are within a threshold. This is a time-consuming operation (about 10s), making it impossible to train the coarse and fine stages together. We note that some concurrent methods [14,22] directly use a large reconstruction model to achieve sparse reconstruction, but these methods are computational expensive and can only generate the low-resolution 3D representation, thus limiting their reconstruction fidelity.

In this paper, we present SuRF, the first attempt, to our knowledge, towards simultaneously unsupervised, sparsified and end-to-end approach, which provides good trade-offs between performance, efficiency, and scalability. The main idea behind this is the surface-centric modeling we adopt, which focuses more attention on regions near the surface, called "surface regions", a practice that improves both performance and efficiency. On the one hand, the projection feature

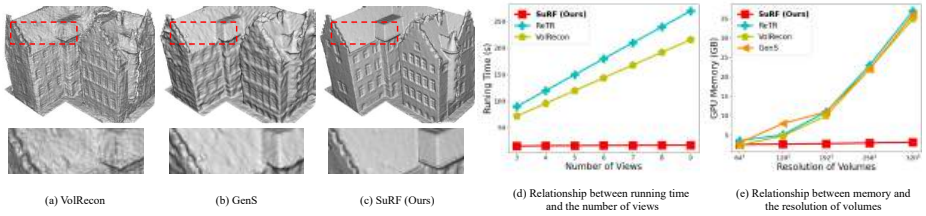
in surface regions is more multi-view consistent and more useful for geometric reasoning. On the other hand, since the surface region only occupies a small proportion of the scene bounding box, this focusing strategy can obviously save memory and computational overhead, and enable the usage of high-resolution volumes. To achieve this, we design a module called Matching Field to locate surface regions, which poses two advantages: 1) it is the first to use the weight distribution along rays to represent the geometry, and enable the use of image warping loss to achieve unsupervised training; 2) it is highly efficient that only needs an additional single-channel volume and the very-fast trilinear interpolation. Concretely, at each scale, in addition to the n-channel feature volume used for final geometric inference, we construct another single-channel matching volume for predicting the matching field.

Based on the matching field, we propose a new strategy called Region Sparsification to generate sparse volumes for later high-resolution scales. Instead of predicting the SDF values for each voxel using MLPs like existing methods, we retain only voxels in surface regions visible from at least two views, which can circumvent the influence of occlusion. Thus, we can generate multi-scale and surface-centric feature volumes to remarkably improve the geometric details of the reconstruction with less memory and computational consumption, as shown in Fig. 2. Extensive experiments on DTU [1] BlendedMVS [52], Tanks and Temples [19] and ETH3D [42] datasets validate the efficiency of the proposed model, surpassing the baseline model [25] by more than 46% and saving more than 80% memory consumption compared with previous state-of-the-art methods [36, 39]. In summary, our main contributions are highlighted below:

- We make the first attempt to achieve unsupervised end-to-end sparsification in neural surface model for high-fidelity sparse reconstruction.
- We present a novel matching field to locate surface regions, which apply the weight distribution to represent the geometry and use image warping loss to achieve unsupervised training.
- We introduce a new region sparsification strategy based on the extracted surface region that is robust to occlusions.
- Extensive experiments on standard benchmarks validate the effectiveness of our approach from the perspectives of accuracy, efficiency and scalability.

2 Related Works

Multi-view stereo. Multi-view stereo (MVS) is a type of methods that take the stereo correspondence as the main cue to reconstruct geometry from multi-view images. Taking the scene representation as an axis of taxonomy, it can be broadly categorized into three types: voxel grids-based [20, 43], point clouds-based [8, 21], and depth map-based [3, 9, 41]. Among them, depth map-based methods decompose complex 3D reconstructions into explicit 2D depth map estimates, becoming the most common one due to convenience. In particular, many learning-based methods [12, 38, 51] have been proposed to improve the



(a) VolRecon (b) GenS (c) SuRF (Ours) (d) Relationship between running time and the number of views (e) Relationship between memory and the resolution of volumes

Fig. 2: Comparisons against recent state-of-the-art methods. All experiments were conducted under the same configuration, *e.g.*, 600×800 resolution and 512 rays. The reconstruction of our method is more accurate and detailed. While the memory consumption of other methods [23, 25, 39, 50] increases exponentially with volume resolution, we can utilize higher-resolution feature volumes with smaller memory overhead to reconstruct higher-frequency details. Meanwhile, our method can directly extract meshes using Marching Cubes on the SDF like [25, 36], whose consumption is more stable with vary input numbers, and is more efficient than depth-fusion methods [23, 39].

matching accuracy through a more robust cost volume. However, the surface reconstruction of these methods is based on a multi-stage pipeline, which is cumbersome and inevitably introduces accumulated errors.

Neural surface reconstruction. Although previous volumetric methods [31] have achieved high-quality reconstructions, neural implicit functions have recently revealed significant potential in 3D reconstruction [27, 33, 34, 44, 53, 54, 58] and appearance modeling [2, 28, 29, 32, 45, 59]. Some work [30, 54, 57, 58] apply surface rendering to reconstruct plausible geometry without 3D supervision, but they often require extra priors like object masks [30, 54] or sparse points [57]. Inspired by the success of NeRF [28] in novel view synthesis, more and more methods integrate volume rendering into shape modeling. They treat the density of volume rendering as the function of different implicit representations, *e.g.*, [33] adopts the occupancy network to represent the geometry and [44, 47, 49, 53] apply the signed distance function to replace the local transparency function. Nevertheless, such methods suffer from lengthy per-scene optimization, cannot generalize to new scenes and perform poorly with sparse inputs.

Generalizable neural surface reconstruction. Similar to the generalizable novel view synthesis methods [5, 15, 45, 55], several methods [23, 25, 39, 50] are proposed to solve the generalization of neural surface reconstruction. By replacing the input from spatial coordinates with image features, these methods can achieve impressive cross-scene generalization. Method [25] is the first attempt to achieve this through a multi-stage pipeline, but still struggles to recover geometric details. Even recent methods have tried to improve this through the view-dependent representation [39], transformer architecture [23] and even build a separate cost volume for each view [50], still failing to balance performance, efficiency, and scalability. To be specific, these methods are restricted by the requirement of the ground-truth depth [38, 39], cannot use high-resolution feature volumes [25, 38, 39, 50] and cannot scale to cases with more input views [50] due to memory constraints. In this paper, we propose SuRF, which can reconstruct more geometric details with limited memory consumption.

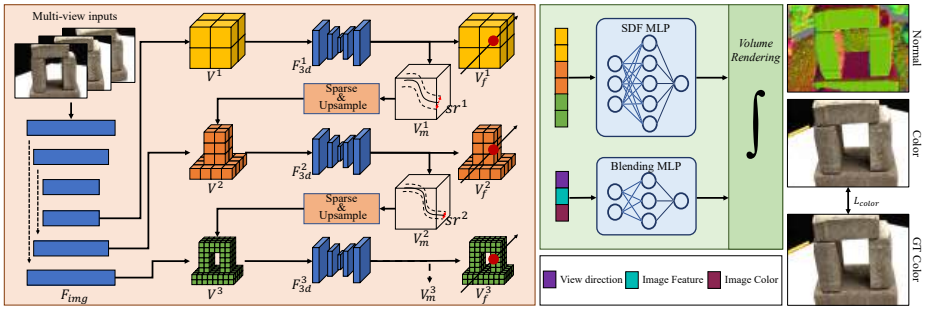


Fig. 3: Framework of SuRF. The multi-scale features are extracted through an FPN network to generate the global volume through our cross-scale fusion strategy. We then build our multi-scale surface-centric feature volumes through the region sparsification, which is based on the surface region extracted from the matching field. We employ color blending to estimate the appearance of points sampled by the surface sampling, and adopt volume rendering to recover the color of a pixel. Here, we omit some modules, *e.g.*, surface sampling and cross-scale fusion, for convenience.

3 Methodology

In this paper, our goal is to reconstruct the finely detailed and globally smooth surface $\mathcal{S}$ from an arbitrary number of inputs with limited memory and computational consumption, which is achieved through our surface-centric modeling. The overall framework of our model is illustrated in Fig. 3. We first introduce our overall pipeline in Sec. 3.1, including how to aggregate multi-view features and reason about geometry and appearance. Then we depict our matching field in Sec. 3.2, including the unsupervised training and surface regions localization, and detail how to construct the multi-scale surface-centric feature volumes based on our new sparsification strategy in Sec. 3.3. The combination of the final loss function is described in Sec. 3.4.

3.1 Overall Pipeline

Given a set of calibrated images $\{I_i \in \mathbb{R}^{3\times H\times W}\}_{i=1}^{N}$ captured from N different viewpoints, we first extract the multi-scale features $\{F_i^j \in \mathbb{R}^{C\times H\times W}\}_{i,j=1,1}^{N,L}$ through a weight-shared FPN [24] network $\mathcal{F}_{img}$. To aggregate these multi-view features, we adopt an adaptive cross-scale fusion strategy, which can grasp both global and local features and is more robust to occlusion.

Cross-scale fusion. For a volume V with U number of voxels, we project each voxel $\mathbf{v} = (x, y, z)$ to the pixel position of corresponding viewpoint with camera intrinsics $\{K_i\}_{i=1}^{N}$ and extrinsic $\{[R, \mathbf{t}]_i\}_{i=1}^{N}$:

$$\mathbf{q}_i = \pi(K_i R_i^T(\mathbf{v} - \mathbf{t}_i)), \tag{1}$$

where $\pi([x, y, z]^T) = [x/z, y/z]^T$. The corresponding multi-scale features $\{\mathbf{f}_i^j \in \mathbb{R}^C\}_{i,j=1,1}^{N,L}$ are then sampled from all image planes via bilinear interpolation.

We treat the high-scale features as detail residuals of low-scale features, and sum them together as multi-view features $\{\mathbf{f}_i \in \mathbb{R}^C\}_{i=1}^N$, which are then input to a fusion network $\mathcal{F}_{fus}$ to generate view's fusion weights $\{w_i\}_{i=1}^N$. The final fused feature for each voxel is the concatenation of weighted mean and variance features $[Mean(\mathbf{v}), Var(\mathbf{v})]$:

$$Mean(\mathbf{v}) = \sum_{i=1}^{N} w_i \mathbf{f}_i,\ Var(\mathbf{v}) = \sum_{i=1}^{N} w_i (\mathbf{f}_i - Mean(\mathbf{v}))^2. \tag{2}$$

Further regularizing above fused features through a 3D network $\mathcal{F}_{3d}$, we can get the final single-channel matching volume $V_m \in \mathbb{R}^{1\times U}$ and n-channel feature volume $V_f \in \mathbb{R}^{C'\times U}$. Note that in our surface-centric modeling, we will generate the multi-scale feature volumes $\{V_f^j\}_{j=1}^L$ and only voxels in the surface regions will be retained at high-resolution scales. The detailed procedure of surface region localization using our matching field will be stated in Sec. 3.2, and how to construct the multi-scale surface-centric feature volumes using our new region sparsification strategy will be elaborated in Sec. 3.3.

In this way, the surface can be reconstructed by the zero-level set of SDF values, which is estimated through a surface prediction network $\mathcal{F}_{sdf}$, which concatenate interpolations of multi-scale feature volumes as input:

$$\mathcal{S} = \{\mathbf{p} \in \mathbb{R}^3 | \mathcal{F}_{sdf}(\mathbf{p}, < \{V_f^j(\mathbf{p})\}_{j=1}^L >) = 0\}, \tag{3}$$

where $< \cdot >$ is a concatenation operator. Meanwhile, since the traditional sampling operator cannot interpolate from the sparse volume, we implement a sparse trilinear sampling algorithm to achieve interpolation efficiently. Inherited from [45], most methods employ a similar blending strategy to predict the color of each point on a ray:

$$\mathbf{c} = \sum_{i=1}^{N} \eta_i \hat{\mathbf{c}}_i, \tag{4}$$

where $\{\hat{\mathbf{c}}_i\}_{i=1}^N$ is the projected colors from source views, and $\{\eta\}_{i=1}^N$ is the softmax-activated blending weights estimated through a color prediction network $\mathcal{F}_{color}$, which takes projected image features and viewing direction differences as input.

Finally, alpha-composition of samples $\{\mathbf{p}(t_k) = \mathbf{o} + t_k\mathbf{d} | k = 1, ..., M\}$ is performed to produce the color of a ray emitting from camera center $\mathbf{o}$ in view direction $\mathbf{d}$:

$$\hat{C} = \sum_{k=1}^{M} T_k \alpha_k \mathbf{c}_k,\ T_k = \prod_{l=1}^{k-1} (1 - \alpha_l), \tag{5}$$

where α is formulated in an unbiased and occlusion-aware conversion of SDF values:

$$\alpha_k = \max(\frac{\Phi_s(\mathcal{F}_{sdf}(\mathbf{p}(t_k))) - \Phi_s(\mathcal{F}_{sdf}(\mathbf{p}(t_{k+1})))}{\Phi_s(\mathcal{F}_{sdf}(\mathbf{p}(t_k)))}, 0), \tag{6}$$

where Φ is the sigmoid function and s is an anneal factor, please refer to [44] for more details.

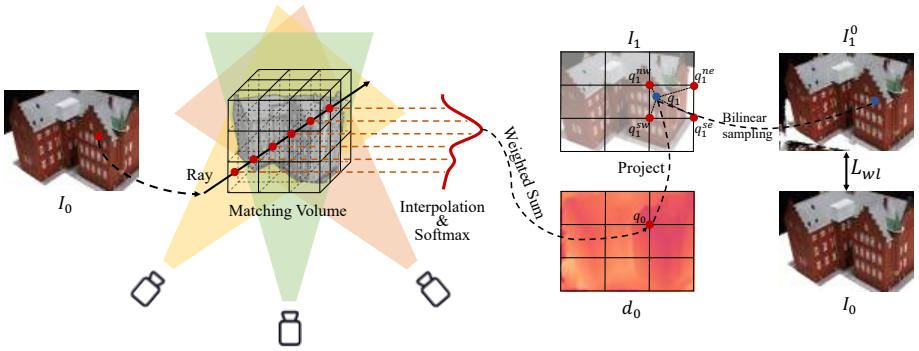


Fig. 4: Illustration of our matching field. We encode the rough scene geometry into a matching volume, and the surface position of a ray can be efficiently retrieved through interpolation. For convenience, we illustrate the surface map E_0 corresponding to all rays of image I_0 in the form of a depth map d_0. Then we leverage the warping loss L_{wl} to constrain the matching field unsupervised.

3.2 Matching Field

Fig. 4 illustrates the overall pipeline of our matching field, which is the cornerstone of surface-centric modeling. In this section, we will elaborate on it in twofold: how it achieves the surface region localization, and how it achieves unsupervised training. Note that these procedures are the same for all scales, and we omit the subscript of scales for convenience.

Surface region localization. For efficiency, we need this procedure to hold three important properties:

- It requires encoding the entire scene geometry with limited memory consumption . Existing methods like [15] or [50] borrow the main idea of MVS methods [12] to construct a separate cost volume for each view, which is sometimes impractical for surface reconstruction, especially when there are many input views.
- It needs to rapidly locate the surface region with a small computational cost, which makes multi-stage training or the use of extra networks unworkable.
- It needs to be occlusion-aware and view-dependent, *i.e.*, those surfaces that are behind or not visible from the input views are unuseful and unsolvable.

Motivated by these properties, we implement the matching field as a weight distribution along the ray obtained from matching volume interpolation.

As shown in Fig. 4, instead of representing the geometry as the occupancy, density or SDF value, we employ the view-dependent weight distribution, where larger values represent closer proximity to the surface. Concretely, to extract the surface of a ray $\mathbf{r} = (\mathbf{o}, \mathbf{d})$, we first uniformly sample M_s points $\{\mathbf{p}(t_k) = \mathbf{o} + t_k\mathbf{d}\}_{k=1}^{M_s}$ within the current surface region (Note that M_s decreases as the scale increases). Next, we directly interpolate the corresponding value for each point from the matching volume V_m, and then go through a softmax operator

to generate the weight distribution $\{\gamma_k\}_{k=1}^{M_s}$ along this ray. In this way, we can infer the rough position of the surface point $\mathbf{p}_s = \mathbf{o} + t_s\mathbf{d}$, where:

$$t_s = \sum_{k=1}^{M_s} \gamma_k t_k. \tag{7}$$

Finally, the surface region that we need is defined as: $sr = [t_s - \epsilon, t_s + \epsilon]$, and ϵ is a hyperparameter that gradually decreases as the scale increases. And the surface region is set to the length of scene bounds for the first scale.

Unsupervised training. With the surface points, we can conveniently leverage the image warping loss [10, 37] to constrain the matching field. Supposing the reference image I_0 has a resolution of $H \times W$, through the matching field, we can efficiently retrieve the surface point of all rays emitting through the pixels of I_0 to form the "surface map" $E_0 \in \mathbb{R}^{3\times H\times W}$. Then we project these points to the pixel positions of source images $\{I_i\}_{i=1}^{N}$ through Eq. (1), and interpolate the colors to generate the warped images $\{I_i^0\}_{i=1}^{N}$. Theoretically, the projected colors of these surface points should remain consistent across multiple viewpoints. Therefore, we can generate the constraints through the difference between the ground-truth reference image and the warped images from source images. Furthermore, we combine the pixel-wise color loss with the patch-based SSIM [48]:

$$WL_i = 0.8 \times \frac{1 - SSIM(I_0, I_i^0)}{2} + 0.2 \times |I_0 - I_i^0|. \tag{8}$$

To avoid the influence of occlusions, we take the average of the K smallest warping losses as the final constraint to optimize our matching field:

$$L_{wl} = \frac{1}{K} \sum_{i=1}^{K} WL_i. \tag{9}$$

In this way, we can optimize our matching field unsupervised to locate the surface region efficiently.

3.3 Feature Volume Construction based on Region Sparsification

As aforementioned and the pipeline comparisons shown in Fig. 1, previous methods [36, 50] rely on the dense volume or multi-stage training [25], and they either are limited by the memory constraints or introduce cumulative errors. To this end, based on the surface region located through our matching field, we propose a region sparsification strategy to construct the multi-scale and surface-centric feature volumes to mitigate these drawbacks.

Region sparsification. Taking a certain scale j as an example, assume that we have generated the matching volume $V_m^j \in \mathbb{R}^{1\times U^j}$ and feature volume $V_f^j \in \mathbb{R}^{C'\times U^j}$ according to the process in Sec. 3.1, and obtained the surface maps $\{E_i^j \in \mathbb{R}^{3\times H\times W}\}_{i=1}^{N}$ of all views following the pipeline in Sec. 3.2. To prune those voxels away from the surface, we project all voxels $Vox^j = \{\mathbf{v}_h\}_{h=1}^{U^j}$ to the pixel position

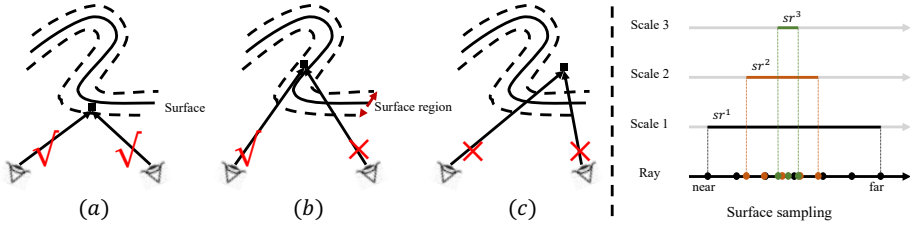


Fig. 5: Illustration of region sparsification and surface sampling. For sparsification, we illustrate three situations: voxels that fall into surface regions visible from multiple viewpoints (a) will be preserved; voxels that fall into surface regions only visible by one viewpoint (b) or are outside surface regions (c) will be pruned. The black rectangle represents the position of a voxel. For surface sampling, we sample the points for each ray within the surface region at each scale.

of all surface maps through Eq. (1) and interpolate the corresponding surface points $\{\hat{E}_i^j \in \mathbb{R}^{3\times U^j}\}_{i=1}^N$ visible from each view through bilinear sampling. We then can determine whether the voxel is inside the surface region based on the distance between the voxel and the interpolated surface point:

$$H_i^j(\mathbf{v}) = \texttt{float}(\parallel \hat{E}_i^j(\mathbf{v}) - \mathbf{v} \parallel_2 < \epsilon^j), \tag{10}$$

where `float` is the operator that converts bool values to float values. Furthermore, to maintain the view consistency, we only retain voxels that simultaneously fall into the surface region of at least two views:

$$Vox^{j+1} = \{\mathbf{v} | \texttt{sum}(H^j(\mathbf{v})) \geq 2\}, \tag{11}$$

where `sum` is the summation operator. This is an important step to mitigate the impact of occlusion, as regions visible only from a small number of views are meaningless, and we depict some examples in Fig. 5. Then we can halve these surviving voxels to aggregate higher-frequency information for the next scale.

Repeating the region sparsification for each scale, we can generate the final multi-scale feature volumes $\{V_f^j\}_{j=1}^L$. While this multi-scale strategy is beneficial for the model to reconstruct surfaces with high-frequency detail and global smoothness like [36, 56], our volumes are surface-centric and can achieve higher resolution with less memory consumption. Meanwhile, since the surface region in the coarse stage is wide, using multi-scale features to predict the geometry makes the model more robust when the surface region location in the fine stage is wrong. Before employing the volume rendering to produce the color of a ray, we propose surface sampling to efficiently sample more points for surface regions.

Surface sampling. With the off-the-shelf surface regions $\{sr^j\}_{j=1}^L$ provided by the matching field, it's natural to sample more points within these regions because voxels inside these regions contain the most valuable information about the surface. We uniformly sample a decreasing number of points within the surface region from low-resolution to high-resolution scales, which results in more sampling points near the surface as shown in Fig. 5. When interpolating from

multi-scale feature volumes, we fill the feature of those sampling points that are outside the surface region of certain scales with zero. Therefore, we do not need other networks to resample more fine points like [25, 28, 44].

3.4 Loss Function

Our overall loss function consists of three components:

$$L = L_{surf} + L_{mf}, \tag{12}$$

where L_{surf} is used to optimize surface network and L_{mf} is used to optimize multi-stage matching fields.

Following existing methods [25, 50], our surface loss L_{surf} is defined as:

$$L_{surf} = L_{color} + L_{mfc} + \alpha L_{ek} + \beta L_{pe}, \tag{13}$$

where L_{color} is computed as the average color loss of all sampled pixels Q:

$$L_{color} = \frac{1}{|Q|} \sum_{q \in Q} |C(q) - \hat{C}(q)|, \tag{14}$$

L_{mfc} is the feature consistency following [36], and eikonal loss [11] is:

$$L_{ek} = \frac{1}{|P|} \sum_{p \in P} (||\nabla \mathcal{F}_{sdf}(p)||_2 - 1)^2, \tag{15}$$

where P is a set of sampled 3D points. Similar to [50], we leverage the pseudo label generated from the unsupervised multi-view stereo method [4] to enhance and accelerate model convergence. We apply a very strict filtering strategy to obtain relatively accurate pseudo point clouds $\hat{P}$. The pseudo loss is:

$$L_{pe} = \frac{1}{|\hat{P}|} \sum_{p \in \hat{P}} |\mathcal{F}_{sdf}(p)|. \tag{16}$$

The loss of the matching field is defined as the weighted sum of all scales' warping loss:

$$L_{mf} = \sum_{j=1}^{L} \mu^j L_{wl}^j, \tag{17}$$

where μ_j is the weight of stage j, and it increases from coarse to fine scale.

4 Experiments

In this section, we first introduce our implementation details and compared datasets, then we enumerate extensive experiments and ablation studies. Note that the reported results here are based on our model without any finetuning. Please refer to Supp. Mat. for the finetuning results.

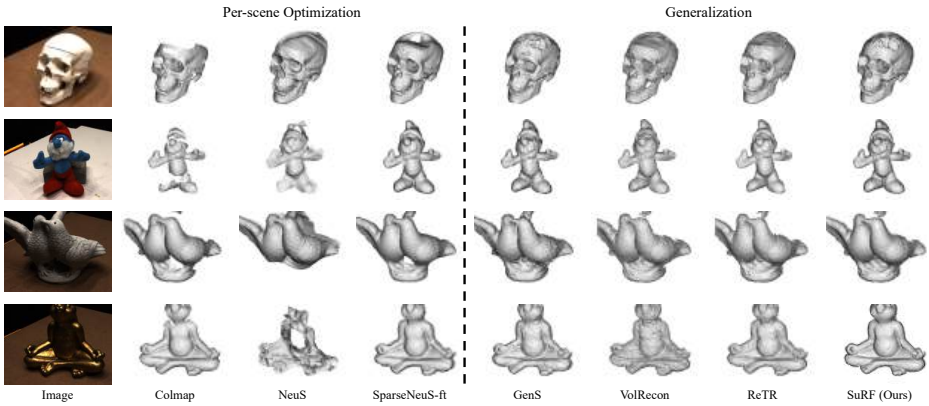


Fig. 6: Qualitative comparisons on DTU dataset.

Implementation details. We implement our model in PyTorch [35], and build our surface-centric feature volume in $L = 4$ scales. The range of surface regions for each scale is defined as ϵ^1:ϵ^2:ϵ^3:ϵ^4=1:0.3:0.1:0.01. In our matching field, we set the number of sampling points for each scale as 128, 64, 32, 16 to extract surface regions. To apply the final volume rendering, the total number of sampling points of each ray is set to $M = 120$, which consists of 64, 32, 16 and 8 for our surface sampling. During training, we adopt Adam optimizer [18] to train our model for 10 epochs. The base learning rate is set to 1e-3 for feature networks and 5e-4 for MLPs. We set the number of source images as $N = 4$ and resize the resolution to 640×480. The volume resolution of the first low-resolution scale is set to $R^1 = 64 \times 64 \times 64$. The weight of each loss term is set to $\alpha = 0.1$, $\beta = 1.0$, and the warping loss weights of each scale are set to 0.25, 0.5, 0.75 and 1.0. During testing, we take $N = 2$ source images with a resolution of 800×576 as input, and set $R^1 = 80 \times 80 \times 80$. Our meshes are extracted using Marching Cubes [26].
Datasets. Following existing practices [25, 39], we train our model on the DTU dataset [1], and we employ the same splitting strategies as [25], *i.e.*, 75 scenes for training and two sets of images from 15 non-overlapping scenes for testing. To validate our generalization ability, we further conduct some qualitative comparisons on BlendedMVS [52], Tanks and Temples [19] and ETH3D [42] datasets.

4.1 Results on DTU

For a fair comparison, we adopt the same evaluation strategy as previous methods [25,39], *i.e.*, reconstruct the surface using only three input views and report the average chamfer distance of two image sets. The quantitative results on the DTU dataset are summarized in Tab. 1, which indicate that our SuRF can bring a satisfactory improvement to the baseline, *i.e.*, more than 46% improvement compared with SparseNeuS [25]. Meanwhile, we can surpass those methods [23,39] that employ ground-truth depth for supervision. Even compared with the method that construct separate cost volumes for each input view [50], our model still offers plausible advantages in efficiency as well as scalability, that is,

Table 1: Quantitative results on DTU dataset. Best results in each category are in **bold** and the second best are in underline. '*' denotes the depth-fusion methods that need the ground-truth depth for supervision, and we report the reproduced results using their officially released model. We test [4] at the same input resolution as ours.

Method	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean
COLMAP [41]	0.90	2.89	1.63	1.08	2.18	1.94	1.61	1.30	2.34	1.28	1.10	1.42	0.76	1.17	1.14	1.52
RC-MVSNet [4]	0.93	2.64	1.92	1.00	1.55	1.62	0.88	1.29	**1.16**	1.00	0.82	0.67	0.60	1.04	1.22	1.22
NeuS [44]	4.57	4.49	3.97	4.32	4.63	1.95	4.68	3.83	4.15	2.50	1.52	6.47	1.26	5.57	6.11	4.00
VolSDF [53]	4.03	4.21	6.12	0.91	8.24	1.73	2.74	1.82	5.14	3.09	2.08	4.81	0.60	3.51	2.18	3.41
Voxurf [49]	2.51	4.32	2.88	2.17	5.43	2.01	2.88	2.11	1.94	1.60	3.02	3.65	1.20	2.10	2.08	2.65
SparseNeuS-ft [25]	1.29	**2.27**	1.57	0.88	1.61	1.86	1.06	1.27	1.42	1.07	0.99	0.87	0.54	1.15	1.18	1.27
IBRNet [45]	2.29	3.70	2.66	1.83	3.02	2.83	1.77	2.28	2.73	1.96	1.87	2.13	1.58	2.05	2.09	2.32
MVSNerf [5]	1.96	3.27	2.54	1.93	2.57	2.71	1.82	1.72	2.29	1.75	1.72	1.47	1.29	2.09	2.26	2.09
SparseNeuS [25]	2.17	3.29	2.74	1.67	2.69	2.42	1.58	1.86	1.94	1.35	1.50	1.45	0.98	1.86	1.87	1.96
VolRecon* [39]	1.54	3.05	1.84	1.08	1.67	1.84	1.13	1.63	1.49	1.19	1.06	1.42	0.76	1.22	1.29	1.48
GenS [36]	1.45	2.77	1.69	0.97	1.54	1.90	1.03	1.49	1.36	0.97	1.07	0.97	0.62	1.14	1.16	1.34
ReTR* [23]	1.23	2.63	1.62	0.98	1.38	1.56	0.90	1.39	1.39	1.02	0.93	0.83	0.59	1.10	1.16	1.25
C2F2NeuS [50]	1.12	2.42	**1.40**	**0.75**	1.41	1.77	0.85	1.16	1.26	**0.76**	0.91	0.60	**0.46**	0.88	**0.92**	1.11
SuRF (Ours)	**0.85**	2.35	1.48	0.88	**1.17**	**1.39**	**0.74**	**1.14**	1.24	0.84	**0.77**	**0.51**	0.51	**0.86**	1.03	**1.05**

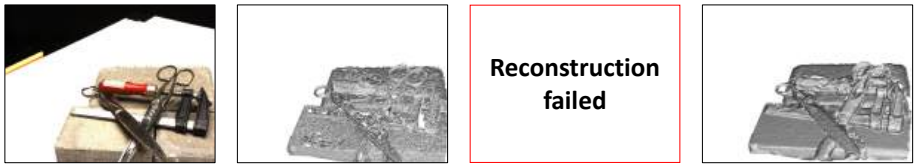


Image | SuGaR w/ 49 images | SuGaR w/ 3 images | Ours w/ 3 images

Fig. 7: Qualitative comparison on DTU dataset with SuGaR.

we only need to construct a global volume and is computation and memory insensitive to the number of input views. For those classical MVS methods [4,40], our method can win out in most metrics. Compared with the recent fast method [49] that converges in minutes, our model can still reconstruct finer details in seconds. Qualitative results in Fig. 6 further show that our SuRF can reconstruct finer surfaces only through fast network inference. The results in Fig. 7 indicate that the 3DGS-based method SuGaR [13] failed with sparse inputs and our approach shows much better performance even with sparse inputs.

Number of input views. Fig. 8 indicates that the reconstruction quality of our model gradually improves as the number of views increases, but it tends to stabilize when the input views are enough. Meanwhile, a larger number of inputs does not lead to a significant increase in consumption, which is an obvious advantage compared to C2F2NeuS [50], as the comparison shown in Tab. 2.

4.2 Generalization

To verify the generalization capabilities of our method, we further test on BlendedMVS [52], Tanks and Temples [19] and ETH3D [42] datasets using the model pre-trained on the DTU dataset [1]. Some qualitative comparisons with existing methods are shown in Fig. 9 and Fig. 10. The results prove that our method ex-

Fig. 8: The performance of mean chamfer distance w.r.t. different number of views.

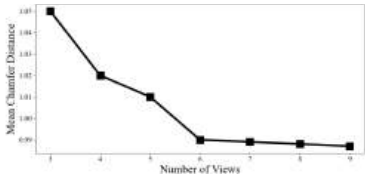


Table 2: Memory consumption with different number of views. Image resolution is 800×600, highest volume resolution of SuRF is 256^3.

Method \ Number of views	3	5	7	9
C2F2NeuS [50]	10G	16G	22G	28G
SuRF (Ours)	2.6G	2.7G	2.8G	3.1G

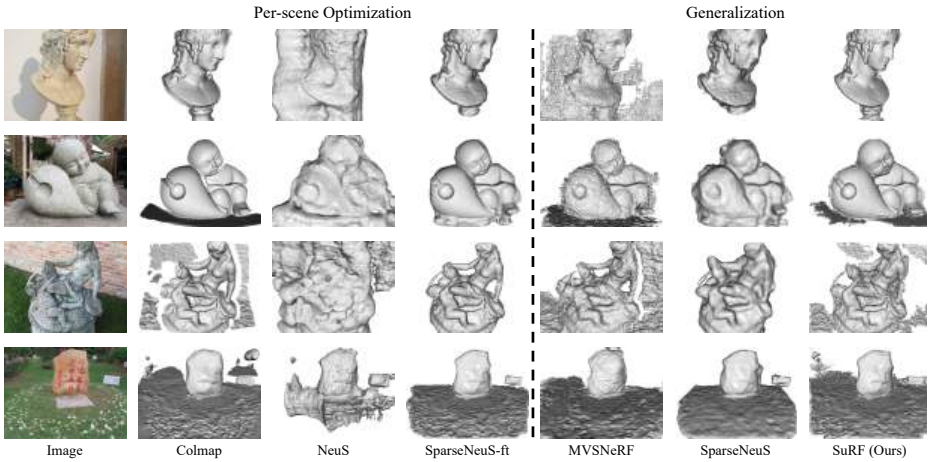


Fig. 9: Qualitative comparisons on BlendedMVS dataset.

hibits great generalization ability even under these difficult scenes. The volume-based method struggles on large scenes since most voxels are empty, but ours can still reconstruct meshes with fine details even without any finetuning.

4.3 Ablation Studies

Our ablation experiments are performed on the first image set of the DTU dataset, which is the same as [25]. To investigate the effectiveness of our end-to-end sparsification, we compare it with existing solutions, *i.e.*, the multi-stage training strategy in SparseNeuS [25] and the multi-scale dense structure in GenS [36]. The results in (a) of Tab. 3 show that all these solutions bring improvements to the baseline model, and our solution performs the best mainly because our features are surface-centric and can be continuously optimized. We perform another study to understand how the resolution of volumes and images affects the model. The results in (b) of Tab. 3 show that the higher resolution favors the model but stabilizes when a certain resolution is reached.

Efficiency and scalability. Compared with existing methods, our SuRF exhibits advantages in terms of efficiency and scalability as shown in Fig. 2. Our model can leverage volumes of higher resolution with less memory and computational consumption, *e.g.*, only 3G memory for the model with the highest 256^3

Table 3: Ablation results on DTU dataset.

(a) The effectiveness of our end-to-end sparsification using our main contributions.

Method	Mean Cham. Dist. ↓
baseline	1.70
w/ multi-stage training	1.52
w/ multi-scale dense volume	1.31
w/ end-to-end sparse.	**1.02**

(b) The performance of the model with different resolution of volumes and input images.

Res. of Vol.	Res. of Image	Mean Cham. Dist. ↓
$48 \times 48 \times 48$	640×480	1.12
$64 \times 64 \times 64$	640×480	1.07
$64 \times 64 \times 64$	800×576	1.04
$80 \times 80 \times 80$	800×576	**1.02**

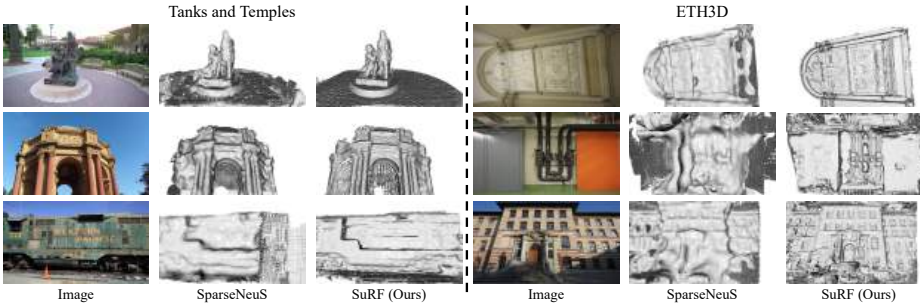


Fig. 10: Qualitative comparisons on Tanks and Temples and ETH3D datasets.

resolution volumes while at least 20G for previous methods [23, 25, 39, 50]. Compared with methods that require predicting depth maps [23, 39, 51] or constructing cost volumes [50] for each view, the running time and memory consumption of our model is relatively insensitive to the number of input views as shown in Tab. 2, making it scalable to handle different input numbers. Meanwhile, the capability of using high-resolution volumes gives our method the potential to reconstruct very large-scale scenes as the results shown in Fig. 10.

5 Conclusion

In this paper, we proposed a new generalizable neural surface model, *SuRF*, to accomplish high-fidelity reconstruction even from sparse inputs with satisfactory trade-offs between performance, efficiency and scalability. To the best of our knowledge, it is the first unsupervised method to achieve end-to-end sparsification based on our surface-centric modeling, which consists of a novel matching field module and a new region sparsification strategy. The proposed matching field adopts the weight distribution to represent geometry and introduces the image warping loss to achieve unsupervised training, which can efficiently locate the surface region. Then we adopted the region sparsification strategy to prune voxels outside the surface regions and generated the multi-scale surface-centric feature volumes. Extensive experiments on multiple public benchmarks demonstrate that our model exhibits great generalization ability in diverse scenes and can reconstruct higher-frequency details with less memory and computational consumption.

Acknowledgements

This work is financially supported by Outstanding Talents Training Fund in Shenzhen, Shenzhen Science and Technology Program-Shenzhen Cultivation of Excellent Scientific and Technological Innovation Talents project(Grant No. RCJC20200714114435057), Shenzhen Science and Technology Program-Shenzhen Hong Kong joint funding project (Grant No. SGDX20211123144400001), National Natural Science Foundation of China U21B2012, R24115SG MIGU-PKU META VISION TECHNOLOGY INNOVATION LAB, Guangdong Provincial Key Laboratory of Ultra High Definition Immersive Media Technology. Jianbo Jiao is supported by the Royal Society Short Industry Fellowship (SIF\R1\231009). In addition, we sincerely thank all assigned anonymous reviewers in ECCV 2024, whose comments were constructive and very helpful to our writing and experiments.

References

1. Aanæs, H., Jensen, R.R., Vogiatzis, G., Tola, E., Dahl, A.B.: Large-scale data for multiple-view stereopsis. IJCV **120**, 153–168 (2016)
2. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: ICCV. pp. 5855–5864 (2021)
3. Campbell, N.D., Vogiatzis, G., Hernández, C., Cipolla, R.: Using multiple hypotheses to improve depth-maps for multi-view stereo. In: ECCV. pp. 766–779 (2008)
4. Chang, D., Božič, A., Zhang, T., Yan, Q., Chen, Y., Süsstrunk, S., Nießner, M.: Rc-mvsnet: unsupervised multi-view stereo with neural rendering. In: ECCV (2022)
5. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: ICCV. pp. 14124–14133 (2021)
6. Chen, H., Li, C., Lee, G.H.: Neusg: Neural implicit surface reconstruction with 3d gaussian splatting guidance. arXiv preprint arXiv:2312.00846 (2023)
7. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., VanderBilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3d objects. In: CVPR. pp. 13142–13153 (2023)
8. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multiview stereopsis. IEEE TPAMI **32**(8), 1362–1376 (2009)
9. Galliani, S., Lasinger, K., Schindler, K.: Massively parallel multiview stereopsis by surface normal diffusion. In: ICCV. pp. 873–881 (2015)
10. Godard, C., Mac Aodha, O., Firman, M., Brostow, G.J.: Digging into self-supervised monocular depth estimation. In: ICCV. pp. 3828–3838 (2019)
11. Gropp, A., Yariv, L., Haim, N., Atzmon, M., Lipman, Y.: Implicit geometric regularization for learning shapes. In: MLSys. pp. 3569–3579 (2020)
12. Gu, X., Fan, Z., Zhu, S., Dai, Z., Tan, F., Tan, P.: Cascade cost volume for high-resolution multi-view stereo and stereo matching. In: CVPR. pp. 2495–2504 (2020)
13. Guédon, A., Lepetit, V.: Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In: CVPR (2024)
14. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: Lrm: Large reconstruction model for single image to 3d. In: ICLR (2024)

15. Johari, M.M., Lepoittevin, Y., Fleuret, F.: Geonerf: Generalizing nerf with geometry priors. In: CVPR. pp. 18365–18375 (2022)
16. Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. ACM ToG **32**(3), 1–13 (2013)
17. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM TOG **42**(4) (2023)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2014)
19. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics (ToG) **36**(4), 1–13 (2017)
20. Kutulakos, K.N., Seitz, S.M.: A theory of shape by space carving. IJCV **38**, 199–218 (2000)
21. Lhuillier, M., Quan, L.: A quasi-dense approach to surface reconstruction from uncalibrated images. IEEE TPAMI **27**(3), 418–433 (2005)
22. Li, J., Tan, H., Zhang, K., Xu, Z., Luan, F., Xu, Y., Hong, Y., Sunkavalli, K., Shakhnarovich, G., Bi, S.: Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. In: ICLR (2024)
23. Liang, Y., He, H., Chen, Y.c.: Rethinking rendering in generalizable neural surface reconstruction: A learning-based solution. In: NeurIPS (2023)
24. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR. pp. 2117–2125 (2017)
25. Long, X., Lin, C., Wang, P., Komura, T., Wang, W.: Sparseneus: Fast generalizable neural surface reconstruction from sparse views. In: ECCV. pp. 210–227 (2022)
26. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: ACM siggraph computer graphics (1987)
27. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: CVPR. pp. 4460–4470 (2019)
28. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
29. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. In: SIGGRAPH (2022)
30. Niemeyer, M., Mescheder, L., Oechsle, M., Geiger, A.: Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In: CVPR. pp. 3504–3515 (2020)
31. Nießner, M., Zollhöfer, M., Izadi, S., Stamminger, M.: Real-time 3d reconstruction at scale using voxel hashing. ACM ToG **32**(6), 1–11 (2013)
32. Oechsle, M., Mescheder, L., Niemeyer, M., Strauss, T., Geiger, A.: Texture fields: Learning texture representations in function space. In: ICCV. pp. 4531–4540 (2019)
33. Oechsle, M., Peng, S., Geiger, A.: Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: ICCV. pp. 5589–5599 (2021)
34. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: CVPR. pp. 165–174 (2019)
35. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019)
36. Peng, R., Gu, X., Tang, L., Shen, S., Yu, F., Wang, R.: Gens: Generalizable neural surface reconstruction from multi-view images. In: NeurIPS (2023)

37. Peng, R., Wang, R., Lai, Y., Tang, L., Cai, Y.: Excavating the potential capacity of self-supervised monocular depth estimation. In: ICCV. pp. 15560–15569 (2021)
38. Peng, R., Wang, R., Wang, Z., Lai, Y., Wang, R.: Rethinking depth estimation for multi-view stereo: A unified representation. In: CVPR. pp. 8645–8654 (2022)
39. Ren, Y., Zhang, T., Pollefeys, M., Süsstrunk, S., Wang, F.: Volrecon: Volume rendering of signed ray distance functions for generalizable multi-view reconstruction. In: CVPR. pp. 16685–16695 (2023)
40. Schonberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: CVPR. pp. 4104–4113 (2016)
41. Schönberger, J.L., Zheng, E., Frahm, J.M., Pollefeys, M.: Pixelwise view selection for unstructured multi-view stereo. In: ECCV. pp. 501–518 (2016)
42. Schops, T., Schonberger, J.L., Galliani, S., Sattler, T., Schindler, K., Pollefeys, M., Geiger, A.: A multi-view stereo benchmark with high-resolution images and multi-camera videos. In: CVPR. pp. 3260–3269 (2017)
43. Seitz, S.M., Dyer, C.R.: Photorealistic scene reconstruction by voxel coloring. IJCV **35**, 151–173 (1999)
44. Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W.: Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In: NeurIPS (2021)
45. Wang, Q., Wang, Z., Genova, K., Srinivasan, P.P., Zhou, H., Barron, J.T., Martin-Brualla, R., Snavely, N., Funkhouser, T.: Ibrnet: Learning multi-view image-based rendering. In: CVPR. pp. 4690–4699 (2021)
46. Wang, Y., Han, Q., Habermann, M., Daniilidis, K., Theobalt, C., Liu, L.: Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In: ICCV. pp. 3295–3306 (2023)
47. Wang, Y., Skorokhodov, I., Wonka, P.: Hf-neus: Improved surface reconstruction using high-frequency details. In: NeurIPS (2022)
48. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE TIP **13**(4), 600–612 (2004)
49. Wu, T., Wang, J., Pan, X., Xu, X., Theobalt, C., Liu, Z., Lin, D.: Voxurf: Voxel-based efficient and accurate neural surface reconstruction. In: ICLR (2022)
50. Xu, L., Guan, T., Wang, Y., Liu, W., Zeng, Z., Wang, J., Yang, W.: C2f2neus: Cascade cost frustum fusion for high fidelity and generalizable neural surface reconstruction. In: ICCV (2023)
51. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: Mvsnet: Depth inference for unstructured multi-view stereo. In: ECCV. pp. 767–783 (2018)
52. Yao, Y., Luo, Z., Li, S., Zhang, J., Ren, Y., Zhou, L., Fang, T., Quan, L.: Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In: CVPR. pp. 1790–1799 (2020)
53. Yariv, L., Gu, J., Kasten, Y., Lipman, Y.: Volume rendering of neural implicit surfaces. NeurIPS **34**, 4805–4815 (2021)
54. Yariv, L., Kasten, Y., Moran, D., Galun, M., Atzmon, M., Ronen, B., Lipman, Y.: Multiview neural surface reconstruction by disentangling geometry and appearance. NeurIPS **33**, 2492–2502 (2020)
55. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelnerf: Neural radiance fields from one or few images. In: CVPR. pp. 4578–4587 (2021)
56. Yu, Z., Peng, S., Niemeyer, M., Sattler, T., Geiger, A.: Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. In: NeurIPS (2022)
57. Zhang, J., Yao, Y., Li, S., Fang, T., McKinnon, D., Tsin, Y., Quan, L.: Critical regularizations for neural surface reconstruction in the wild. In: CVPR. pp. 6270–6279 (2022)

58. Zhang, J., Yao, Y., Quan, L.: Learning signed distance field for multi-view surface reconstruction. In: ICCV. pp. 6525–6534 (2021)
59. Zhang, X., Srinivasan, P.P., Deng, B., Debevec, P., Freeman, W.T., Barron, J.T.: Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. ACM ToG **40**(6), 1–18 (2021)

Supplementary Materials

A Results of Fine-tuning

As the qualitative and quantitative comparisons shown in our main paper, the reconstructions of our model without fine-tuning exhibit finest geometric details even compared with some fine-tuned models like SparseNeuS-ft [25]. Here, we illustrate some results of our model after fast fine-tuning. Different with methods [15,50] that require reconstructing separate cost volume for each view, our model only builds the global volume, which makes our model easily fine-tuned (only 2.5k iterations, about 10 minutes). The quantitative results in Tab. A show that our model still ranks the first in most scenes and has the best mean chamfer distance. Meanwhile, it is worth noting that our volume is sparse and more memory and computationally efficient. And the qualitative results of some scenes are visulized in Fig. A. Note that there are only three input views during fine-tuning.

Table A: Quantitative results of the fine-tuned model on DTU dataset. Best results in each category are in **bold** and the second best are in underline.

Method	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean
IBRNet-ft [45]	1.67	2.97	2.26	1.56	2.52	2.30	1.50	2.05	2.02	1.73	1.66	1.63	1.17	1.84	1.61	1.90
SparseNeuS-ft [25]	1.29	2.27	1.57	0.88	1.61	1.86	1.06	1.27	1.42	1.07	0.99	0.87	0.54	1.15	1.18	1.27
GenS-ft [36]	0.91	2.33	1.46	**0.75**	**1.02**	1.58	0.74	1.16	1.05	**0.77**	0.88	0.56	**0.49**	**0.78**	**0.93**	1.03
SuRF-ft (Ours)	**0.73**	**2.11**	**1.39**	0.83	1.05	**1.53**	**0.68**	**1.03**	**1.02**	0.84	**0.85**	**0.46**	**0.49**	0.84	1.00	**0.99**

Table B: Reproduced results of VolRecon [39] and ReTR [23] in two image sets.

Method	Image Set	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean
VolRecon [39]	Set0	1.27	2.66	1.54	1.04	1.41	1.94	1.10	1.53	1.36	1.08	1.18	1.37	0.74	1.22	1.26	1.38
	Set1	1.80	3.46	2.14	1.12	1.92	1.74	1.17	1.72	1.63	1.31	0.94	1.46	0.78	1.23	1.30	1.58
	Average	1.54	3.05	1.84	1.08	1.67	1.84	1.13	1.63	1.49	1.19	1.06	1.42	0.76	1.22	1.29	1.48
ReTR [23]	Set0	1.05	2.32	1.47	0.97	1.22	1.52	0.88	1.30	1.29	0.87	1.07	0.76	0.58	1.11	1.12	1.17
	Set1	1.42	2.95	1.76	0.99	1.55	1.59	0.92	1.49	1.50	1.19	0.79	0.89	0.60	1.09	1.21	1.33
	Average	1.23	2.63	1.62	0.98	1.38	1.56	0.90	1.39	1.39	1.02	0.93	0.83	0.59	1.10	1.16	1.25

B More Comparisons with RC-MVSNet

We show some visual and metric comparisons with the TSDF fusion result of RC-MVSNet [4] in Fig. B. The results show that the reconstruction of our model is smoother and more complete, especially in low-texture regions, leading to better results in the chamfer distance metric. To further verify the effectiveness of our surface-centric modeling, we compare with two baselines which directly use the

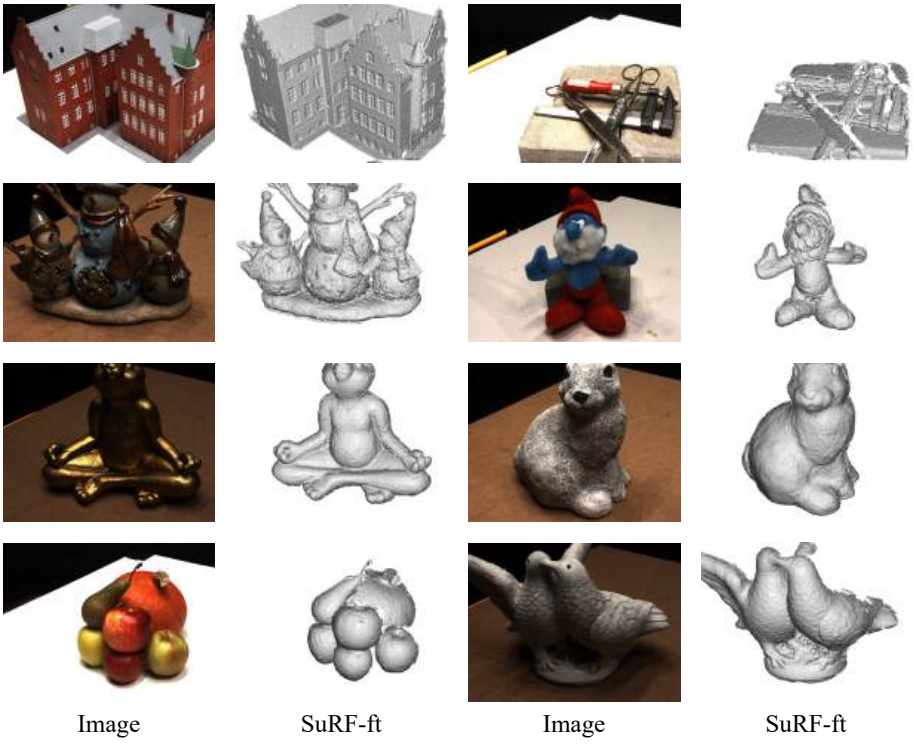


Fig. A: Visualization of fine-tuning results.

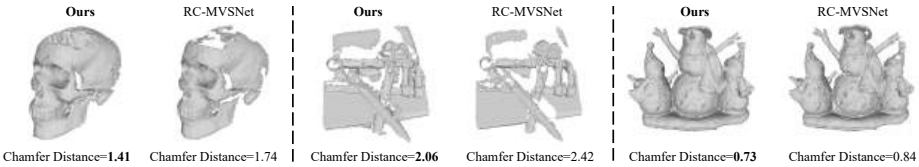


Fig. B: Visual and metric comparisons with RC-MVSNet.

surface point of RC-MVSNet to prune voxels: Baseline1 directly replaces the surface region of our trained model with that of RC-MVSNet; Baseline2 uses the surface region of RC-MVSNet to train a new model. Results in Tab. C show that even simply using the surface region of RC-MVSNet can achieve superior results. And our full model, trained together with the surface location module (Our matching field), achieves the best performance. This is reasonable because the surface region of these two baselines was not optimized or corrected with the model when directly using the results of RC-MVSNet.

C Detailed Results of VolRecon and ReTR

As mentioned in our main paper, we report the reproduced results of VolRecon [39] and ReTR [23] on two image sets using their official repositories and

Table C: More ablation studies about directly using the surface region of RC-MVSNet.

	RC-MVSNet	Baseline1	Baseline2	Ours
Chamfer Distance	1.22	1.13	1.16	**1.05**

released model checkpoints. The detailed reproduction results of all scenes at two image sets are illustrated in Tab. B, which are slightly different from the results reported in their papers. We speculate that there are something inconsistent in the experimental configurations, but this inconsistency doesn't affect the valuable of their contributions.

D More Ablation Results

Here, we report more ablation results of our model, and we set the training time to a quarter of the overall process (different from our main paper to save time) and only test on the first image set for convenience.

Number of scales. We conduct some ablations to evaluate the effect of the number of scales. We set the resolution of the finest stage of each model to be similar. The results in Tab. D show that the overall quality first remarkably increases and then slightly decreses, reaching the optimum in 4 scales. We illustrate some visual results of the model with differnet scales in Fig. C. The single-scale model performs the worst, with reconstructions that are noisy and lack geometric detail, while the four-scales model can reconstruct smooth geometry and restore more geometric details.

Table D: Ablation results on DTU dataset.

Number of scales	Surface sampling	Cross-scale fusion	Mean
1 scales	✓	✓	1.38
2 scales	✓	✓	1.22
3 scales	✓	✓	1.15
4 scales	✓	✓	**1.11**
5 scales	✓	✓	1.13
4 scales	✓	✗	1.15
4 scales	✗	✓	1.13

Ablations on loss weight. We further conduct some experiments to verify the effect of the weight of each loss term on model performance. Concretely, we change the weight of the pseudo loss β and the weight combination of different stages of matching field loss μ^j. The ablation model is based on the 4-scales model and the results are shown in Tab. E. Through the comparison between model A and model B, we can see that the pseudo point clouds generated from the unsupervised multi-view stereo method [4] can guide the model towards better convergence. To avoid the influence of erroneous pseudo points, we apply a very strict filtering strategy, *i.e.*, Only point clouds whose projection distance from at least 3 viewpoints does not exceed 0.2 pixels and whose relative depth error does not

Table E: Ablation results of loss weight on DTU dataset.

Method	β	$\mu^1 : \mu^2 : \mu^3 : \mu^4$	Mean
A	0.0	0.25 : 0.50 : 0.75 : 1.00	1.17
B	1.0	0.25 : 0.50 : 0.75 : 1.00	**1.11**
C	1.0	1.00 : 1.00 : 1.00 : 1.00	1.16
D	1.0	1.00 : 0.75 : 0.50 : 0.25	1.25

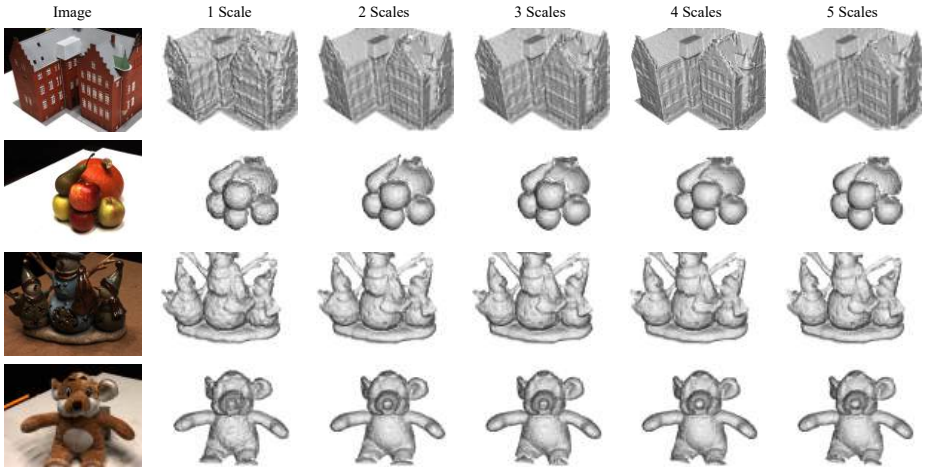


Fig. C: Visual comparison with different number of scales on DTU dataset.

exceed 0.001 can be left. From the results of the model (B, C, D) adopt different weight combinations of the matching field loss, we can see that model B which has $\mu^1 : \mu^2 : \mu^3 : \mu^4 = 0.25 : 0.50 : 0.75 : 1.00$ performs the best and model D performs the worst. This indicates that applying greater weight to the high-resolution scale is beneficial to model convergence. Because there is no need to obtain very accurate predictions in the low-resolution scale, and the gradient of the high-resolution scale will be transmitted back to the low-resolution scale, it is reasonable to have a lower weight in the low-resolution scale. And we show some visual comparisons of these models in Fig. D.

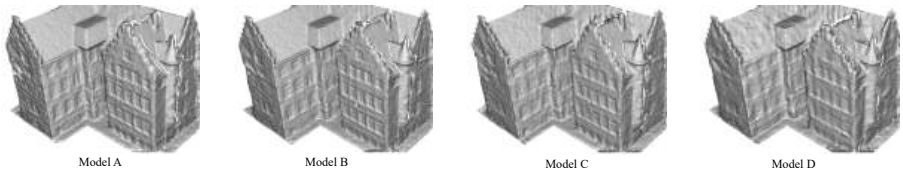


Fig. D: Visual comparison of reconstructions with different loss weights.

Ablations on the range of surface region. Here, we employ an additional ablation experiment to study the sensitivity of the range of surface regions. ϵ^0 is the range of the first scale, and its value is fixed at 1, which represents covering the entire near and far area. And the value of later scales means the percentage of coverage. As the results shown in Tab. F, the differences of these three groups of experiments are not large, as long as the surface region is gradually tightened, and model F which has a range combination of $\epsilon^1 : \epsilon^2 : \epsilon^3 : \epsilon^4 = 1.00 : 0.30 : 0.10 : 0.01$ performs the best.

Table F: Ablation results of the range of surface regions.

Method	$\epsilon^1 : \epsilon^2 : \epsilon^3 : \epsilon^4$	Mean
B	1.00 : 0.40 : 0.10 : 0.01	1.11
E	1.00 : 0.30 : 0.10 : 0.01	**1.10**
F	1.00 : 0.30 : 0.05 : 0.01	1.12

E More Results

Because C2F2NeuS [50] doesn't release the code, the memory of C2F2NeuS in Tab. 2 is refer to the implementation of CasMVSNet [12]. Fig. E shows additional comparisons with COLMAP [40], NeuS [44], SparseNeuS [25], SparseNeuS-ft [25], VolRecon [39] and ReTR [23] on DTU dataset. We can see that our method can stably achieve superior results and exhibit finer geometry details. We further show some visual comparisons with the fast per-scene overfitting method Voxurf [49] in Fig. F. While Voxurf still requires more than 30 minutes of training time per scene, it struggles to reconstruct smooth and accurate surface from sparse inputs.

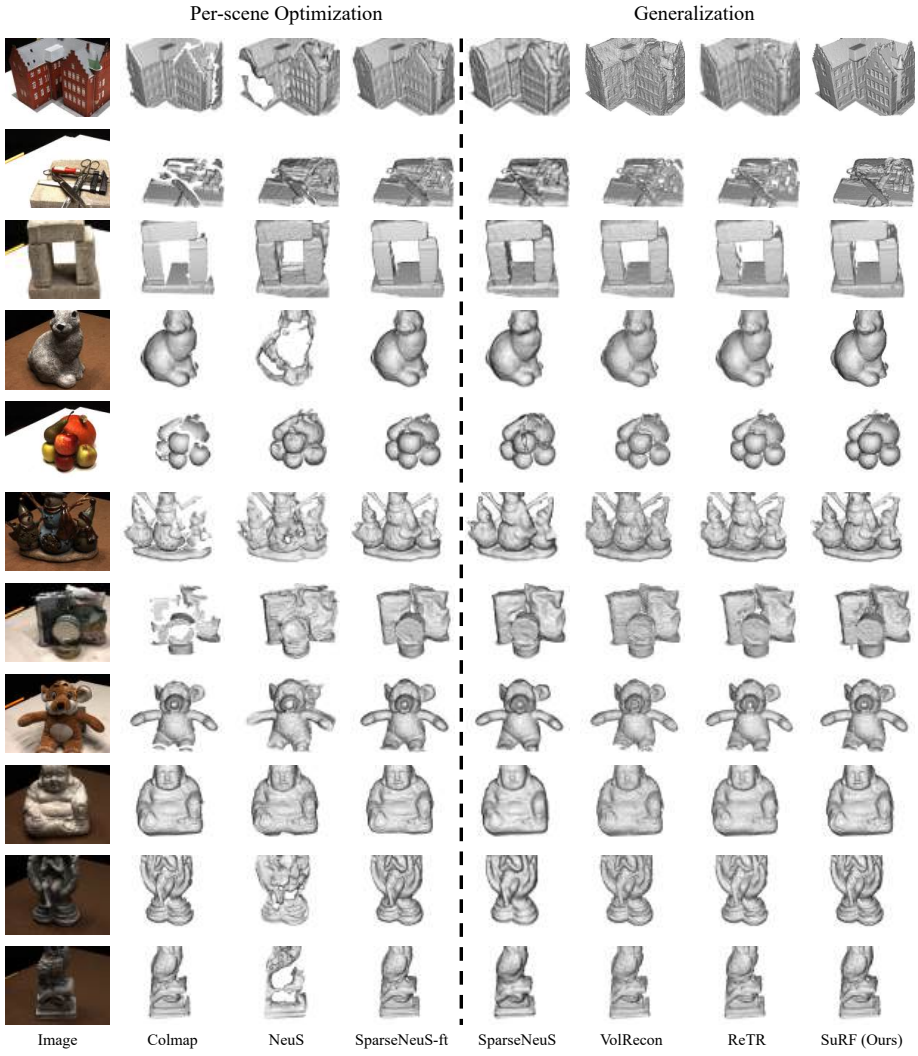


Fig. E: More qualitative comparisons on DTU dataset.

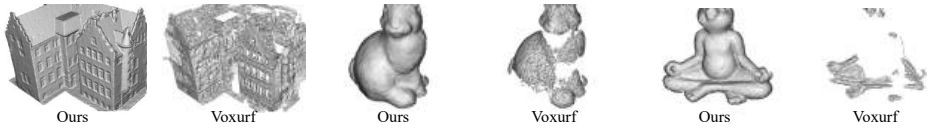


Fig. F: Comparison with Voxurf on DTU dataset with 3 inputs.

Fig. G: Visualization of the located surface region from the matching field at different scales. We visualize the depth of the middle surface for convenience.

F Visualization of the Surface Region

To understand how the surface region changes as scale increases, we show some visualization results of the surface region at different scales in Fig. G. For convenience, we show the depth of the middle position of the surface region. We can see that the located surface region at the higher resolution scale is indeed sharper, which proves the effectiveness of our design.

G Limitations and Future Work

Despite exhibiting efficiency over existing methods, our model still struggled to extract the surface in real time due to the inherent drawback of MLP-based implicit methods. In the future, we will be focusing on addressing this deficiency issue, and we have constructed a lite-version model, which will be released latter. Furthermore, we plane to train our model on more large-scale dataset like Objaverse [7] and expand the scale of the model like [14, 22].