Efficient Textured Mesh Recovery from Multiple Views with Differentiable Rendering

Lixiang Lin, Yisu Zhang, and Jianke Zhu

ZheJiang University
lxlin@zju.edu.cn, 22121124@zju.edu.cn, jkzhu@zju.edu.cn

Abstract. Despite of the promising results on shape and color recovery using self-supervision, the multi-layer perceptrons-based methods usually costs hours to train the deep neural network due to the implicit surface representation. Moreover, it is quite computational intensive to render a single image, since a forward network inference is required for each pixel. To tackle these challenges, in this paper, we propose an efficient coarse-to-fine approach to recover the textured mesh from multi-view images. Specifically, we take advantage of a differentiable Poisson Solver to represent the shape, which is able to produce topology-agnostic and watertight surfaces. To account for the depth information, we optimize the shape geometry by minimizing the difference between the rendered mesh with the depth predicted by the learning-based multi-view stereo algorithm. In contrast to the implicit neural representation on shape and color, we introduce a physically based inverse rendering scheme to jointly estimate the lighting and reflectance of the objects, which is able to render the high resolution image at real-time. Additionally, we fine-tune the extracted mesh by inverse rendering to obtain the mesh with fine details and high fidelity image. We have conducted the extensive experiments on several multi-view stereo datasets, whose promising results demonstrate the efficacy of our proposed approach. We will make our full implementation publicly available.

Keywords: multi-view stereo, differentiable rendering

1 Introduction

Recovering the textured mesh from multiple views is a fundamental problem in computer vision, which is essential to a large amount of real-world applications, including virtual reality, augmented reality, robotics and cultural heritage digitization.

Generally, the de-facto pipeline of 3D reconstruction is to firstly obtain the point cloud by multi-view stereo [1][21][27]. Then, the implicit surface [9] is fitted from the scattered points, which is further triangulated into mesh through an efficient marching cube algorithm [16].

Recently, rapid progress has been made in the area of inverse rendering [17][23], which can obtain the textured mesh with details for real-time rendering.
The photometric loss is backpropagated to the triangulated mesh through differentiable rendering [7,15,21], which makes it feasible to recover shape and reflectance by an inverse rendering optimization (or analysis-by-synthesis). As the optimization in inverse rendering is highly under-constrained, it usually makes the assumption of single light source or the known positions of all the light sources. Moreover, it is hard to deal with the complex topology changes due to the explicit mesh representation. As it is easy to stuck at the local optima, a rough initial mesh model is required, which is usually generated by MVS algorithms [25,27].

In contrast to the conventional approaches consisting of several stages, the implicit neural representation-based methods [32,34] directly learn the multi-layer perceptrons (MLP) from multiple images, which is able to recover shape and color in an end-to-end manner. The scene geometry is represented as the zero level set, and the appearance is represented as the surface light field. Moreover, the neural network is trained with self-supervision through backpropagating the color consistency loss. Similarly, the triangulated mesh is extracted by marching cube algorithm via evaluating the signed distance function. Since the implicit MLP representation is not straightforward and the gradient of the MLP decreases as the layer goes forward, it is very time-consuming to train the deep neural network. Typically, it takes several hours to reconstruct the shape and appearance using the implicit neural representation. More importantly, the color for each pixel is represented by the learned MLP, which is computational intensive to render a single image. This greatly hinders them from a large amount of real-world applications.

To address the above limitations, we propose an efficient coarse-to-fine approach to recover the textured mesh from multi-view images in this paper. Specifically, we make use of visual hull to obtain the initial mesh rather than deforming a sphere. Moreover, we take advantage of a differentiable Poisson Solver to represent the shape, which is able to produce topology-agnostic and watertight surfaces. To account for the depth information, we optimize the shape geometry by minimizing the difference between the rendered mesh with the depth predicted by the learning-based multi-view stereo algorithm. Instead of using MLP to represent the appearance and shape, we introduce a physically based inverse rendering scheme to jointly estimate the lighting and reflectance of the objects, which is able to render the high resolution image at real-time. Furthermore, we fine-tune the extracted mesh by inverse rendering to obtain the mesh with fine details and high fidelity image.

In summary, the main contributions of this paper are: (1) efficient coarse-to-fine approach to recover the textured mesh from multi-view images; (2) a differentiable Poisson Solver-based shape representation for fast mesh optimization; (3) a physically based inverse rendering scheme to extract the reflectance and fine-tune the mesh. (4) experiments on several multi-view stereo datasets, comparing against state-of-the-art methods, showing good qualitative and quantitative results.
2 Related Works

During past decades, a surge of research efforts have been spent on 3D reconstruction using multiple images. The recent approaches can be roughly divided into three categories, including Multi-view-stereo (MVS), inverse rendering and implicit neural representation. We review these methods in the following.

2.1 Multi-view Stereo

In general, most of MVS methods \cite{24,27} assume that the appearances of a surface point are consistent in all visible views. The 3D coordinates can be obtained by triangulating the correspondences through matching the image patches across all regions of interests uniformly sampled or propagated from neighboring pixels and adjacent views \cite{1,4}. Depth fusion and implicit surface reconstruction \cite{9} are required to extract the watertight mesh from point clouds, where the surface details may be smoothed in this process. Moreover, these meshes are usually texture-less. Although the color can be estimated from the input images using camera projection, the rendered images are not fidelity enough to be viewed from free perspective. There are also many some learning-based MVS methods have been developed \cite{5,30,31}. The hand-crafted features are replaced by deep features, and the overall depth quality is greatly improved.

2.2 Inverse Rendering

By taking advantage of differentiable renderer \cite{12,21,15,17,23}, \cite{7} try to estimate the object’s intrinsic color and geometry by inverse rendering, where the gradients can be back propagated to the geometry. Therefore, the geometry and appearance of objects can be recovered by minimizing the difference between the synthesized photo and input image.

The key operation in rendering pipeline is rasterization. Liu \textit{et al.} \cite{15} propose a differentiable renderer for image-based shape fitting. The rendering is treated as a differentiable aggregating process that fuses the probabilistic contributions of all mesh triangles with respect to the rendered pixels. Wang \textit{et al.} \cite{29} present a differentiable renderer for point cloud that is considered as a disk. Moreover, the discontinuous are approximated by a linear function. Jiang \textit{et al.} \cite{7} propose a differentiable renderer for shape optimization using signed distance functions (SDFs), where the gradients can be back propagated to the whole SDF grid by regularization. As the soft rasterization operation is an approximate solution, which does not provide the gradient with respect to variables other than pixel coordinates. This may lead to the inferior reconstruction results. Li \textit{et al.} \cite{14} present a Monte Carlo differentiable renderer that produces the unbiased gradients through edge sampling. Later, Luan \textit{et al.} \cite{17} propose an analysis-by-synthesis pipeline for high-quality reconstruction of geometry and spatially varying reflectance by making use of Monte Carlo differentiable renderer. As is well-known, recovering the geometry and appearance from 2D images is a highly under-constrained problem. Moreover, the pixel intensity is affected by lots of
Factors like lighting, materials, occlusions, etc. Thus, most of existing inverse rendering methods assume that there is a single light source or the directions of all the light sources are known. This may rarely happen in the real-world applications. In addition, it is difficult to deal with the complex topology changes using the inverse rendering optimization, where the gap between the initial geometry and target cannot be very large. To avoid this issue, MVS is usually employed to obtain the initial shape.

2.3 Implicit Neural Representation

Implicit neural representation directly estimates the objects’ geometry from the input image through minimizing the photometric loss, which has achieved the encouraging results in 3D reconstructions and view synthesis. Sitzmann et al. [26] employ LSTM to represent the scene by simulating the ray marching process. Mildenhall et al. [18] represent the scene in terms of volume density and view-dependent radiance, where the color of each pixel is obtained by accumulating the grid having the ray passed through. Due to the limitations on resolution in volume rendering, the geometry extracted by the marching cube algorithm [16] is usually rough. Instead of using volumetric density representation, Yariv et al. [32] employ neural networks to represent scenes through SDF and light field implicitly, where color is only calculated at the intersection. Moreover, the two implicit networks are trained by the loss of color and silhouette. Zhang et al. [35] use mixtures of spherical Gaussians to represent the bidirectional reflectance distribution function (BRDF) and environmental illumination for physically based rendering. Zhang et al. [34] leverage the feature consistency in stereo matching to optimize the implicit surface. Oechsle et al. [19] make use of both surface rendering and volume representation, which obtain the accurate reconstruction results without masks. Although the implicit representation has the merits of watertight, topology-agnostic, it tends to produce the over-smoothed surface. Since the color value of every pixel needs to be calculated by a neural network, it is quite computationally expensive to render an image.

3 Methods

In this section, we present our proposed coarse-to-fine approach to recovering the textured mesh from multi-view images.

3.1 Overview

To facilitate the efficient textured mesh recovery, we firstly obtain the coarse mesh from multi-view silhouettes. Instead of deforming a sphere like [32], we directly triangulate the visual hull [13] through the marching cube algorithm. Visual hull is the maximal object that gives the same silhouette from any possible viewpoint, which may not be suitable for all kinds of objects. Each silhouette
**Fig. 1.** Overview of our proposed coarse-to-fine approach to textured mesh recovery. The coarse mesh is obtained by visual hull. Then, the oriented point clouds are sampled and optimized by depth loss and silhouette loss. We employ physically based rendering to estimate the lighting and reflectance. The environment map is optimized jointly.

forms a cone in its corresponding camera view, and the convex hull of real object’s shape is the intersection of all these visual cones. Obviously, reconstruction results are improved with the increasing number of views. Practically, we can get roughly correct result for convex objects. When the object is concave, the results of visual hull are far from the realistic. To tackle this critical problem, we try to recover the accurate textured mesh by taking advantage of multi-view constraints with a differentiable renderer, which is able to deform the coarse mesh to the target geometry. Specifically, we sample the oriented point clouds from the coarse initial mesh as the shape representation, and a differentiable Poisson Surface Reconstruction is applied to reconstruct a mesh. Finally, the reconstructed mesh is supervised by original images and MVS depth outputs. Fig. 1 shows the overview of our proposed coarse-to-fine framework.

The overall optimization is supervised by the following loss function,

\[ \mathcal{L} = \lambda_s \mathcal{L}_s + \lambda_d \mathcal{L}_d + \lambda_c \mathcal{L}_c + \lambda_{reg} \mathcal{L}_{reg}, \]

where \( \mathcal{L}_s \) is the silhouette loss for the mask, and \( \mathcal{L}_d \) is the depth loss. \( \mathcal{L}_c \) is photometric loss to minimize the difference between the inverse rendered photo and input image. \( \mathcal{L}_{reg} \) is the regularization loss that preserves the regularity on the estimated surface. \( \lambda_s, \lambda_d, \lambda_c \) and \( \lambda_{reg} \) are the weighting coefficients to balance the different terms.

### 3.2 Silhouette Loss and Depth Loss

In order to improve the coarse mesh obtained by visual hull, we propose a point-based shape optimization framework.

Given the input mesh \( \mathcal{M}(V, F) \) with vertices \( V \) and facets \( F \), a differentiable renderer \(^{12}\) interpolates the attributes on vertices to pixels with respect to
the camera parameter $\pi$, which is a differentiable operation in deep neural network. Let $\zeta$ denote the differentiable renderer. The rendered silhouette $\hat{M}$ can be obtained by interpolating the constant value of one

$$\hat{M} = \zeta(V, F, 1; \pi).$$

(2)

As the differentiable renderer is able to backpropagate the gradient on the pixel back to the position of vertices, we impose the silhouette loss to limit the boundary of the generated mesh within the mask annotations,

$$L_s = \sum_{i=1}^{N} ||M_i - \hat{M}_i||_2^2,$$

(3)

where $|| \cdot ||_2^2$ represents $L_2$ norm. $i = 1, \cdots, N$ represents all views.

Similarly, we can render the depth map $\hat{D}$ by a differentiable renderer $\zeta$ using the camera projection matrix and current mesh prediction, which interpolates the $z$ coordinate of each vertex

$$\hat{D} = \zeta(V, F, V_z; \pi).$$

(4)

In this work, we make use of the off-the-shelf MVS method \cite{33} to estimate the depth map from the input images. Therefore, we can improve the mesh geometry by minimizing the difference between the rendered depth map and predictions from images as follows

$$L_d = \frac{1}{|P_{valid}|} \sum_{p \in P_{valid}} |D_p - \hat{D}_p|$$

(5)

where $| \cdot |$ denotes $L_1$ norm. $P_{valid}$ represents the indices, in which the estimated depth map by MVS is valid.

**Point-based Shape Representation** Instead of using the explicit mesh triangulating the visual hull, we take advantage of a differentiable Poisson Solver (SAP) \cite{20} to represent the shape, which is able to produce topology-agnostic and watertight surfaces. Moreover, it is interpretable, lightweight and fast comparing to neural implicit representation. The key step in Poisson Surface Reconstruction \cite{9} involves solving the Poisson equation, where an oriented points set can be viewed as samples of the gradient of the underlying implicit indicator function capturing the geometry. SAP solves the Poisson equation using the spectral methods. As the Fast Fourier Transform (FFT) operation are well supported for GPUs, SAP can be implemented very efficiently. Since all the computations are differentiable, the gradient can be backpropagated to points and normals directly. The mesh is generated by differentiable marching cube and the gradients can be effectively approximated by the inverse surface normal.

To this end, we obtain the oriented point cloud by uniformly sampling the points and normals $S = \{x \in \mathbb{R}^3, n \in \mathbb{R}^3\}$ from the coarse mesh produced by visual hull algorithm, which is further used to build the signed distance field
(SDF) grid through Differentiable Poisson Surface Reconstruction (DPSR)

\[ SDF = DPSR(S) \]  

where \( SDF \) represents the SDF grid obtained by solving the Poisson equation from the oriented point clouds \( S \). Then, the topology-agnostic and watertight mesh can be obtained via differentiable marching cube (DMC)

\[ \mathcal{M}(V, F) = DMC(SDF) \]

where \( V, F \) denotes the vertices and faces of the mesh \( \mathcal{M} \) triangulated from SDF grid, respectively.

### 3.3 Photometric Loss

Instead of using the MLP to represent the appearance of the mesh in the conventional approaches, we propose a physically based inverse rendering approach to jointly estimate the lighting and reflectance of the objects.

**Physically Based Rendering** In computer graphics, the color of each pixel is computed by the rendering equation based on the physical law, where we omit the radiance emitted by the object. The rendering equation is an integral equation determined by two factors, including the bidirectional reflectance distribution function (BRDF) representing the reflectance coefficient of object and the light emitted from the light source

\[ L_o(p, w_o, t) = \int_{\Omega} f_r(p, w_i, w_o, t)L_i(p, w_i)n \cdot w_i dw_i, \]  

where \( f_r(\cdot) \) is the BRDF function jointly determined by the incident light direction \( w_i \) and viewing direction \( w_o \) at the intersection point \( p \), \( L_i \) is the light intensity from direction \( w_i \), \( t \) represents the texture parameters at \( p \). To depict an object’s spatially varying reflectance, we use the Cook-Torrance BRDF model. We simulate the reflected lights by two parts, a certain amount of light in all directions and another amount in a specular way:

\[ f_r = k_d f_{\text{lambert}} + k_s f_{\text{cook-torrance}}, \]

where \( f_{\text{lambert}} = a_d \) represents the diffuse component. \( f_{\text{cook-torrance}} \) describes the specular component, which is usually quantified by microfacets theory. Besides, the two coefficients \( k_d \) and \( k_s \) with a sum of one represent the ratio of diffuse and specular part, respectively. As for Cook-Torrance specular reflectance or microfacet BRDF, it has the following form:

\[ f_{\text{cook-torrance}} = a_s \frac{DFG}{4(w_o \cdot n)(w_i \cdot n)} \]  

where \( a_s \) denotes the specular albedo. \( D \) is the normal distribution function, and \( F \) is the fresnel function and \( G \) is the geometry function. \( w_i \) and \( w_o \) represent the
incoming direction and outgoing direction, respectively. \(D\) and \(G\) are controlled by the surface roughness \(\alpha\). The overall texture parameters \(a_{tex} \in \mathbb{R}^7\) including \(a_d \in \mathbb{R}^3\), \(a_s \in \mathbb{R}^3\) and \(\alpha \in \mathbb{R}^1\). We use GGX functions \([28]\) to describe the normal distribution and geometry function. The detailed definition of the distribution function and fresnel function is described in the supplementary materials.

As for the light source of the scene, we adopt an HDR light probe image \([3]\) in the latitude-longitude format, which is simple and direct. Specifically, we use a \(4 \times 8\) resolution for our lighting environments. In our implementation, we create an environment map for each image.

Let \(V_{tex}\) represent the texture parameter for each vertex. It can be computed from the learnable texture grid by trilinear interpolation. Therefore, the interpolated texture map \(\hat{T}\) can be obtained by

\[
\hat{T} = \varsigma(V, F, V_{tex}, \pi)
\]  (11)

Let \((u, v) = \pi(p)\) denote the 2D projection of 3D intersection point \(p\). The pixel color of the rendered image \(\hat{I}\) is computed by the rendering equation.

\[
\hat{I}_{(u,v)} = L_0(p, w_o, \hat{T}_{(u,v)})
\]  (12)

From the above all, the inverse rendering optimization can be directly self-supervised by the input image through minimizing the photometric loss \(L_c\) as below

\[
L_c = \frac{1}{|P_{in}|} \sum_{p \in P_{in}} |I_p - \hat{I}_p|
\]  (13)

During the joint optimization, the probe pixels in the lighting environments are updated by the photometric loss.

### 3.4 Regularization Loss

As recovering the deformed surface is a highly under constrained problem, it is very easy to stuck at the local optima by only minimizing the loss of silhouette, depth and synthesized image. To address this issue, we introduce the regularization terms \(L_{reg}\) to penalize the surface deformations and texture consistency, which consists of a mesh loss \(L_{mesh}\) and a material loss \(L_{mat}\)

\[
L_{reg} = L_{mesh} + \lambda_{mat}L_{mat}
\]  (14)

The mesh loss \(L_{mesh}\) is made of a Laplacian loss \(L_{lap}\) and a normal loss \(L_{norm}\), which tries to preserve the topology and smoothness of the mesh

\[
L_{mesh} = \lambda_{lap}L_{lap} + \lambda_{norm}L_{norm}
\]  (15)

Let \(L\) denote the Laplacian matrix. The Laplacian loss \(L_{lap}\) of a mesh with \(n\) vertices is computed by \(L_{lap}(M) = \|LV\|^2\), where \(V\) is the \(n \times 3\) vertices matrix.

The normal loss \(L_{norm}\) enforces the normals of adjacent faces to be similar, which is calculated by \(L_{norm} = \sum_{i,j} ||1 - \cos(n_i, n_j)||^2\). The sum is computed
over all adjacent faces pairs \((i, j)\), and \(n_i\) and \(n_j\) denote their corresponding normals.

Material loss \(\mathcal{L}_{\text{mat}}\) regularizes the reflectance maps, which consists of diffuse albedo \(a_d\), specular \(a_s\) and surface roughness \(\alpha\)

\[
\mathcal{L}_{\text{mat}} = \mathcal{L}_b(a_d) + \mathcal{L}_b(a_s) + \mathcal{L}_b(\alpha),
\]

where \(\mathcal{L}_{\text{mat}}\) smooths the texture map. Let \(BF\) represents bilateral filter operation, \(\mathcal{L}_b(a) = \frac{1}{|\mathbf{a}|} \sum |BF(a) - a|\). It can be clearly observed that \(\mathcal{L}_b\) tends to smooth the texture map.

### 3.5 Implementation Details

In this paper, we firstly compute the visual hull at the resolution of \(128^3\). Secondly, we uniformly sample 10,000 oriented points from the mesh triangulated from the visual hull using marching cube, which are used to minimize the depth and silhouette differences through the point-based shape representation. We perform the optimization at a resolution of \(128^3\) for 150 epochs. To obtain the fine details, we increase the resolution to \(256^3\) for the extra 150 epochs. Moreover, we resample points and normals every 50 epochs in order to increase the robustness of the optimization process, and replace original point cloud with the resampled ones. Thirdly, the texture of the generated mesh is interpolated from a learnable texture grid which initialized at a resolution of \(128^3\), which is interpolated into \(256^3\) after 150 epochs. The texture grid and environment map are supervised by the photometric loss. Finally, we fine-tune both geometry and texture using another 100 epochs. In our implementation, we use Adam optimizer in minimization. The learning rates for the oriented point clouds and texture grid are \(5e^{-4}\) and \(1e^{-4}\), respectively. The weights for the loss term are \(\lambda_c = 5\), \(\lambda_s = 10\), \(\lambda_d = 30\), \(\lambda_{IAP} = 100\), \(\lambda_{\text{norm}} = 0.02\), and \(\lambda_{\text{mat}} = 0.1\). The optimization and inverse rendering are implemented by PyTorch. We use the off-the-shelf differentiable renderer nydiffраст [12] to obtain the silhouette and perform Cook-Torrance BRDF rendering.

### 4 Experiments

In this section, we discuss the results on the textured mesh recovery from multiple views. We compare our method against the state-of-the-art methods on multiview stereo datasets.

#### 4.1 Evaluation on DTU Dataset

We firstly evaluate our proposed approach on the DTU MVS dataset [6], which contains 128 scans. For each scan, there are 49 calibrated cameras surrounding the captured object. We evaluate both the generated mesh and rendered image using the same set of scans as the conventional method [34].
Table 1. Quantitative results on DTU dataset. Our method achieves the best mean Chamfer distance and comparable PSNR results.

| Dataset | Chamfer (mm) | PSNR  |
|---------|-------------|-------|
| PhySG   | 4.31 0.98   | 1.63 0.83 |
| Vis-MVSNet | 2.10 1.76 | 0.48 0.44 |
| IDR     | 19.18 14.71 | 23.96 19.07 |
| MVSDF   | 23.29 22.16 | 24.60 21.17 |
| Ours    | 23.09 17.55 | 23.22 21.17 |
| Mean    | 23.25 19.85 | 23.20 21.26 |

To facilitate fair comparison, Chamfer distance is employed as the evaluation metric to measure the accuracy of generated mesh, and PSNR is used to evaluate the rendered images. We compare our method against the recent state-of-the-art 3D reconstruction methods, including PhySG [35], Vis-MVSNet [33], IDR [32] and MVSDF [34]. Depth maps from Vis-MVSNet are fused into point clouds, which are further converted into meshes by the screened Poisson Surface Reconstruction (sPSR) [10]. The color is assigned from input images by back projection.

Table 2 shows the quantitative results. It can be clearly observed that our proposed approach achieves the lowest Chamfer distance on DTU dataset. It indicates that we achieve the most accurate reconstruction results. Due to the lack of GPU RAM, we use an environment map of 4 × 8 resolution, which cannot simulate the complex environment lighting with high fidelity. Our results on the quality of rendered image is a bit inferior to MVSDF [34], which makes use of depth supervision and fused feature. Fig. 2 give the example reconstruction results. It can be seen that our method can produce more accurate mesh with the realistic synthesized images.

Table 2. Computational time on DTU dataset.

| Methods  | Training time | Rendering time |
|----------|---------------|----------------|
| PhySG [35] | 5.0h          | 30s            |
| IDR [32] | 6.5h          | 30s            |
| MVSDF [34] | 5.5h          | 30s            |
| Ours     | (30 + 10) min | 0.04s          |

We also evaluate the computational time on DTU dataset. All the experiments are conducted on a single GPU of NVIDIA 2080Ti. Table 2 summarizes the computational cost for the compared methods. It can be seen that our pro-
posed approach requires 40 min to obtain the 3D reconstruction results while the implicit representation based-methods need several hours. It takes 30 minutes to perform point-based optimization, and costs 10 minutes to fine-tune the mesh. This is because the gradient of MLP decreases as the layer goes forward, which leads to longer training time. We employ the oriented point clouds as the shape representation, where the gradient can be efficiently backpropagated to points and normals. In terms of rendering time, our approach is able to run at 25Hz by taking advantage of the textured mesh. For the implicit representation based-methods, it takes dozens of seconds to render an image of resolution $1600 \times 1200$, since a forward network inference is required for each valid pixel.

![Fig. 2. Qualitative results on DTU dataset. We compare our method with Vis-MVSNet, IDR, and MVSDF.](image)

4.2 Evaluation on EPFL Dataset

To compare the performance on the scene reconstruction, we evaluate our proposed method on EPFL dataset. We conduct experiments on Fountain-P11 and Herzjesu-P8, which have ground truth meshes. We compare our method against Vis-MVSNet, IDR, and MVSDF. As EPFL dataset does not provide the mask, we generate the mask by projecting the ground truth mesh onto the image. The experimental results of Vis-MVSNet and IDR are from MVSDF.

Fig. 3 shows the reconstruction results. It can be observed that our method can produce both high-quality mesh and render the high quality images. The reconstructed mesh of IDR has the inflated surfaces, as it is difficult to recover accurate shape from only color loss. By taking advantage of the extra supervision from multi-view stereo, both MVSDF and our method can recover the correct geometry. The quantitative results are given in Table 3. We use the same evaluation metrics as the DTU dataset. It can be seen that our approach performs comparable with the state-of-the-art methods.
Fig. 3. Qualitative results on EPFL dataset. Our method is able to generate both high quality mesh and render high quality image.

Table 3. Quantitative results on EPFL dataset. Our method performs comparable to the state-of-the-art methods.

|               | Vis-MVSNet | IDR | MVSD | Ours | Vis-MVSNet | IDR | MVSD | Ours |
|---------------|------------|-----|------|------|------------|-----|------|------|
| Chamfer ($\times 10^{-2}$) |            |     |      |      |            |     |      |      |
| Fountain-P11  | 6.12       | 7.88| 6.84 | 6.93 | 24.33      | 23.43| 25.27| 27.34|
| Herzjesu-P8   | 7.47       | 32.19| 6.38 | 6.75 | 23.45      | 24.75| 28.75| 26.45|
| Mean          | 6.80       | 25.30| 6.61 | 6.84 | 23.89      | 24.67| 27.01| 26.90|

4.3 Ablation Studies

In this section, we discuss the effect of depth loss on the surface recovery. Previous methods \cite{32,35} mainly try to minimize the rendering loss, where the shape is just a by-product. Although the geometry can be recovered accurately in some cases, it is usually difficult to recover the accurate shape due to the ambiguity between geometry and appearance. We replace the depth loss with the rendering loss used in previous work to learn the shape. Specifically, we use an MLP to estimate the color of each pixel. The gradient is backpropagated to the shape. The SDF value is obtained from the SDF grid by trilinear interpolation. The results are shown in Fig. 4. The depth maps predicted by multi-view stereo provide the correct geometry information, which enable the fast convergence.

4.4 Additional Qualitative Results

Finally, we provide the qualitative results on BlendMVS, Tanks and Temples dataset. The masks of the Horse is generated by an off-the-shelf image segmentation framework \cite{11}, and the camera parameters are estimated by Colmap \cite{25}. Fig. 5 shows several examples. It demonstrates that our method is able to reconstruct the accurate geometry and texture from in-the-wild images. The results in the second row of Fig. 5 show that our proposed method is able to extract the accurate mesh and textures from multi-view images without masks as well.

As our method extracts the triangle mesh and texture explicitly, which can be rendered with the arbitrary lighting. Instead of using an MLP to represent the
surface light field, we use physically based rendering that decouples the texture and illumination. We can synthesize the images under different illuminations. Fig. 6 illustrates the qualitative results, which shows that the synthesized images are quite realistic.

Fig. 5. Qualitative results on BlendMVS dataset and Tanks and Temples dataset. The top two rows are from the BlendMVS dataset, and the third row are from Tanks and Temples dataset.

5 Conclusions

This paper proposed an efficient coarse-to-fine approach to the textured mesh recovery from multi-view images. Oriented point clouds and a differentiable Poisson Solver was used to represent the shape, which produces topology-agnostic
and watertight surfaces. We introduced a physically based inverse rendering scheme to jointly estimate the lighting and reflectance, which is able to render the high resolution image at real-time. Additionally, we fine-tuned the estimated mesh by inverse rendering. We have conducted the extensive experiments on several datasets. The encouraging results showed that our approach can efficiently reconstruct the textured mesh.

In spite of the promising results, some limitations and future work need to be addressed. Our method depends on the depth map predicted by the multi-view stereo. In future, we will incorporate the cost volume of deep feature matching into the optimization.
References

1. Bleyer, M., Rhemann, C., Rother, C.: Patchmatch stereo - stereo matching with slanted support windows. In: Hoey, J., McKenna, S.J., Trucco, E. (eds.) British Machine Vision Conference, BMVC 2011. pp. 1–11 (2011)
2. Cook, R.L., Torrance, K.E.: A reflectance model for computer graphics. In: Green, D., Lucido, T., Fuchs, H. (eds.) Proceedings of the 8th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1981. pp. 307–316 (1981)
3. Debevec, P.E.: Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In: Cunningham, S., Bransford, W., Cohen, M.F. (eds.) Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1998. pp. 189–198 (1998)
4. Furukawa, Y., Ponce, J.: Accurate, dense, and robust multiview stereopsis. IEEE Trans. Pattern Anal. Mach. Intell. 32(8), 1362–1376 (2010)
5. Huang, P., Matzen, K., Kopf, J., Ahuja, N., Huang, J.: Deepmvs: Learning multiview stereopsis. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018. pp. 2821–2830 (2018)
6. Jensen, R.R., Dahl, A.L., Vogiatzis, G., Tola, E., Aanæs, H.: Large scale multiview stereopsis evaluation. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014. pp. 406–413 (2014)
7. Jiang, Y., Ji, D., Han, Z., Zwicker, M.: Sdfdiff: Differentiable rendering of signed distance fields for 3d shape optimization. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020. pp. 1248–1258 (2020)
8. Kajiya, J.T.: The rendering equation. In: Evans, D.C., Athay, R.J. (eds.) Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1986. pp. 143–150 (1986)
9. Kazhdan, M.M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Sheffer, A., Polthier, K. (eds.) Proceedings of the Fourth Eurographics Symposium on Geometry Processing, Cagliari, Sardinia, Italy, June 26-28, 2006. ACM International Conference Proceeding Series, vol. 256, pp. 61–70 (2006)
10. Kazhdan, M.M., Hoppe, H.: Screened poisson surface reconstruction. ACM Trans. Graph. 32(3), 29:1–29:13 (2013)
11. Kirillov, A., Wu, Y., He, K., Girshick, R.B.: Pointrend: Image segmentation as rendering. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020. pp. 9796–9805 (2020)
12. Laine, S., Hellsten, J., Karras, T., Seol, Y., Lehtinen, J., Aila, T.: Modular primitives for high-performance differentiable rendering. ACM Transactions on Graphics 39(6) (2020)
13. Laurentini, A.: The visual hull concept for silhouette-based image understanding. IEEE Trans. Pattern Anal. Mach. Intell. 16(2), 150–162 (1994)
14. Li, T., Aittala, M., Durand, F., Lehtinen, J.: Differentiable monte carlo ray tracing through edge sampling. ACM Trans. Graph. 37(6), 222:1–222:11 (2018)
15. Liu, S., Chen, W., Li, T., Li, H.: Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In: IEEE/CVF International Conference on Computer Vision, ICCV 2019. pp. 7707–7716 (2019)
16. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: SIGGRAPH. pp. 163–169. ACM (1987)
17. Luan, F., Zhao, S., Bala, K., Dong, Z.: Unified shape and SVBRDF recovery using differentiable monte carlo rendering. Comput. Graph. Forum 40(4), 101–113 (2021)
18. 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: Proceedings of the European Conference on Computer Vision (ECCV) 2020. Lecture Notes in Computer Science, vol. 12346, pp. 405–421 (2020)
19. Oechsle, M., Peng, S., Geiger, A.: Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: International Conference on Computer Vision (ICCV) (2021)
20. Peng, S., Jiang, C.M., Liao, Y., Niemeyer, M., Pollefeys, M., Geiger, A.: Shape as points: A differentiable poisson solver. In: Advances in Neural Information Processing Systems (NeurIPS) (2021)
21. Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.Y., Johnson, J., Gkioxari, G.: Accelerating 3d deep learning with pytorch3d. arXiv:2007.08501 (2020)
22. Remelli, E., Lukoianov, A., Richter, S.R., Guillard, B., Bagautdinov, T.M., Baqué, P., Fua, P.: Meshsdf: Differentiable iso-surface extraction. In: Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems, NeurIPS 2020 (2020)
23. Schmitt, C., Donné, S., Riegler, G., Koltun, V., Geiger, A.: On joint estimation of pose, geometry and svbrdf from a handheld scanner. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020. pp. 3490–3500 (2020)
24. Schönhuber, J.L., Zheng, E., Frahm, J., Pollefeys, M.: Pixelwise view selection for unstructured multi-view stereo. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Proceedings of the European Conference on Computer Vision (ECCV) 2016. vol. 9907, pp. 501–518 (2016)
25. Schönhuber, J.L., Zheng, E., Frahm, J., Pollefeys, M.: Pixelwise view selection for unstructured multi-view stereo. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision - ECCV - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016. Lecture Notes in Computer Science, vol. 9907, pp. 501–518 (2016)
26. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. In: Annual Conference on Neural Information Processing Systems, NeurIPS 2019. pp. 1119–1130 (2019)
27. Tola, E., Strecha, C., Fua, P.: Efficient large-scale multi-view stereo for ultra high-resolution image sets. Mach. Vis. Appl. 23(5), 903–920 (2012)
28. Walter, B., Marschner, S.R., Li, H., Torrance, K.E.: Microfacet models for refraction through rough surfaces. In: Proceedings of the Eurographics Symposium on Rendering Techniques, Grenoble, France, 2007. pp. 195–206 (2007)
29. Wang, Y., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. ACM Trans. Graph. 38(6), 230:1–230:14 (2019)
30. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: Mvsnet: Depth inference for unstructured multi-view stereo. In: Ferrari, V., Hebert, M., Smichisi, C., Weiss, Y. (eds.) Proceedings of the European Conference on Computer Vision (ECCV) 2018. vol. 11212, pp. 785–801 (2018)
31. Yao, Y., Luo, Z., Li, S., Shen, T., Fang, T., Quan, L.: Recurrent mvsnet for high-resolution multi-view stereo depth inference. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019. pp. 5525–5534 (2019)
32. Yariv, L., Kasten, Y., Moran, D., Galun, M., Atzmon, M., Basri, R., Lipman, Y.: Multiview neural surface reconstruction by disentangling geometry and appear-
33. Zhang, J., Yao, Y., Li, S., Luo, Z., Fang, T.: Visibility-aware multi-view stereo network. In: 31st British Machine Vision Conference, BMVC 2020 (2020)
34. Zhang, J., Yao, Y., Quan, L.: Learning signed distance field for multi-view surface reconstruction. International Conference on Computer Vision (ICCV) (2021)
35. Zhang, K., Luan, F., Wang, Q., Bala, K., Snavely, N.: Physg: Inverse rendering with spherical gaussians for physics-based material editing and relighting. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021. pp. 5453–5462 (2021)