Improving Multi-View Reconstruction via Texture-Guided Gaussian-Mesh Joint Optimization

Zhejia Cai[1,†] Puhua Jiang[1,2,†] Shiwei Mao[1] Hongkun Cao[2,‡]
Ruqi Huang[1,‡]
[1] SIGS, Tsinghua University [2] Peng Cheng Laboratory † Equal Contribution ‡ Corresponding Author

Abstract

Reconstructing real-world objects from multi-view images is essential for applications in 3D editing, AR/VR, and digital content creation. Existing methods typically prioritize either geometric accuracy (Multi-View Stereo) or photorealistic rendering (Novel View Synthesis), often decoupling geometry and appearance optimization, which hinders downstream editing tasks. This paper advocates an unified treatment on geometry and appearance optimization for seamless Gaussian-mesh joint optimization. More specifically, we propose a novel framework that simultaneously optimizes mesh geometry (vertex positions and faces) and vertex colors via Gaussian-guided mesh differentiable rendering, leveraging photometric consistency from input images and geometric regularization from normal and depth maps. The obtained high-quality 3D reconstruction can be further exploit in down-stream editing tasks, such as relighting and shape deformation. Our code will be released in https://github.com/zhejia01/TexGuided-GS2Mesh

1. Introduction

Reconstruction of real-world objects from multi-view images plays a central role in a wide realm of applications, including 3D editing[19], AR/VR[2, 3], film industry[7], to name a few. Upon the recent advances on high-quality reconstruction, in this paper we investigate a relatively underexplored problem – *how to ease editing operations on both geometry and appearance of digitizations in a unified manner*? In fact, this problem is becoming increasingly critical with the rapid advancement of interactive virtual environments. For instance, one might expect to deform an object and/or change lighting condition during interaction.

The key bottleneck of the aforementioned task, in our opinion, is the separated focus of the mainstream 3D representations utilized in reconstruction. For instance, classical multi-view stereo (MVS) approaches [8, 20, 24, 29, 33, 36] primarily focus on reconstructing dense point clouds from triangulation guided by photometric consistency and leave appearance alignment to post-processing(*e.g.,* texture baking [8]). Such approaches can capture fine geometric details while suffering from oversimplified/inconsistent texture maps due to their heavy reliance on geometric priors[40, 45]. On the other hand, Neural View Synthesis (NVS) methods[1, 18, 26, 27, 46] have gained considerable popularity in computer vision, which predominantly focus on producing high-fidelity novel view renderings. Mesh reconstruction approaches(*e.g.,* [14, 41, 44, 47])based on these NVS methods essentially rely on signed distance field(SDF)[30] representation for geometry extraction and appearance association. However, SDF is not trivial to plug into existing geometry processing tools, rendering its difficulty in the geometric editing.

Perhaps the most relevant works to ours is NVdiffrec(mc) [13, 28] and NerF2Mesh [39], which both extract meshes from NVS reconstruction for consequent refinement. To attach appearance, these works train neural networks such as coordinate-field MLP [39] to address the challenging problem of mapping texture onto meshes. From this point of view, the geometry and appearance remain disentangled in optimization (or learning), hindering their utilities in the scenarios requiring simultaneously editing from both perspectives, as mentioned in the beginning.

To address this problem, our key insight is to *enhance the coherence between geometry and appearance*, in both representation and optimization. More specifically, starting from a set of multiview images, we first leverage the recent advances in 3DGS [18] to achieve appearance reconstruction and extract a coarse mesh. Crucially, we advocate to decorate this mesh with *per-vertex color*, which is also accessible from the 3DGS reconstruction. Thus, we can optimize geometry and appearance in a unified manner and easily adopt methods developed in geometry processing. In particular, we adopt the iterative, inverse-rendering-based remeshing method [31] into our framework. Unlike ContinousRemeshing [31] depending on the *ground-truth* nor-

mal and depth rendered from the given target geometry, our method can effectively refine the initial mesh via photometric consistency, weak geometric supervision from the initial mesh and some mild geometric regularization.

Though our approach seems conceptually simple, we need to overcome the disadvantage of per-vertex color encoding. More specifically, due to the linear nature of our color coding, it is prone to produce color artifacts, especially around the regions consisting of smooth geometric change but dramatic texture variation. To this end, we further propose a Texture-based Edge Length Control (TELC) scheme to robustify our remeshing pipeline.

Finally, to fully exploit the high-quality textured mesh, we further propose a vertex-Gaussian binding scheme, so that the improved geometry can be transferred to the bound Gaussian, which enables simultaneous material and geometric editing of the reconstructed object.

We conduct a rich set of experiments to verify the effectiveness and efficiency of our pipeline, highlighting its superiority in geometric accuracy, rendering fidelity, relighting precision, and deformation consistency.

2. Related Work

2.1. Surface Reconstruction with Volume Rendering

Neural Radiance Fields (NeRF) [26] represent a scene as a continuous volumetric function using a neural network that predicts the color and density for points in 3D space, enabling photo-realistic novel view synthesis. 3D Gaussian Splatting (3DGS)[18] optimizes an explicit representation through differentiable rasterization, which not only significantly enhances training speed but also improves the quality of novel view synthesis.

However, NeRF and 3DGS are not specifically designed for mesh extraction tasks, and therefore extracting meshes based on the density of sampled points leads to inaccurate reconstruction results. To address these limitations, NeuS[42] represents surfaces as the zero-level set of SDF and introduces a new volume rendering formulation to reduce geometric bias inherent in conventional volume rendering. NeuS2[43] and Neuralangelo[22] integrate multi-resolution hash encodings and accelerate training. Methods like IRON [48], NeMF [50], Neural Microfacet [25], and ROSA [16] further study object-centric inverse rendering to jointly recover geometry and materials/appearance from images, complementing scene-level surface reconstruction methods. In terms of explicit mesh extraction meshod, SuGaR[12] and Gaussian Surfels[6] regulate Gaussians and extract meshes by Poisson reconstruction[17] technique. 2D Gaussian Splatting (2DGS)[14] improves upon 3DGS by using 2D oriented planar Gaussian disks and employs TSDF fusion[5]. Furthermore, Gaussian Opacity Field (GOF)[47] provides a tetrahedron grid-based technique based on DMTet[34] instead of Poisson reconstruction and TSDF fusion. Planar-based Gaussian Splatting Reconstruction (PGSR) [4] presents a representation for efficient and high-fidelity surface reconstruction from multi-view RGB images and surpasses all existing methods. However, solely relying on parameter extraction of meshes from 3D representations can lead to a gap between 2D and 3D representations. That is, detailed information in multi-view images may be lost in the process from 2D images to 3D representations to 3D meshes. Therefore, we propose a method that optimizes meshes by simultaneously utilizing 2D images and 3D representations, enabling the meshes to have finer details.

2.2. Hybrid of Gaussian Splatting and Mesh Representations

Recent works in the field of computer graphics and geometry processing have explored hybrid methods that combine the advantages of mesh representations with the flexibility of Gaussian splatting. These approaches typically bind Gaussians to the vertices or faces of a coarse mesh, allowing the Gaussians to benefit from the geometric structure provided by the mesh. The primary goal of these methods is to enhance the rendering quality of the Gaussians, leveraging the mesh's shape to improve the appearance and coherence of the splatting process.

For instance, *Mani-GS* [11] presents a hybrid approach that binds Gaussians to a coarse mesh, aiming to refine their appearance through optimization techniques. The idea is to optimize the Gaussian parameters (such as position, scale, and opacity) while keeping the Gaussians aligned with the mesh structure. Similarly, *Gaussian Mesh Splatting* [10] also explores the fusion of Gaussian splatting with mesh representations, primarily focusing on how to deform Gaussians in accordance with mesh transformations, thereby enabling dynamic scene rendering and deformation.

However, these existing methods predominantly focus on binding Gaussians to a static or deformed mesh structure and optimizing their rendering effects. While they effectively improve the visual quality of Gaussians on the mesh, they largely neglect the reverse conversion—how to transfer learned Gaussian attributes back to the mesh for tasks such as relighting or deformation.

2.3. Reconstruction via Optimization

Recent works have sought to bridge the gap between implicit neural representations and explicit 3D meshes for practical applications. Among these, NeRF2Mesh[39] extracts a coarse mesh and iteratively refines both vertex positions and face density using re-projection errors to guide adaptive surface optimization. However, NeRF2Mesh decouples geometry and view-dependent appearance, process-

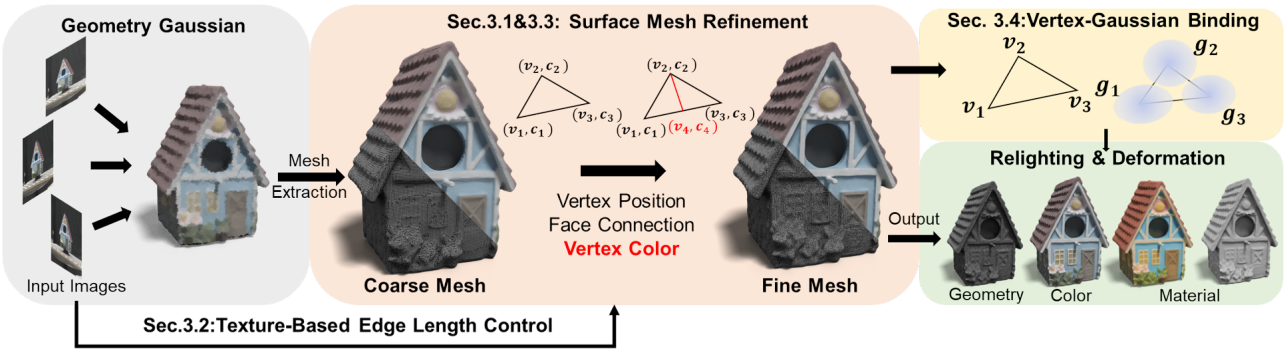

Figure 1. The schematic illustration of our pipeline.

ing them independently, which may limit the potential for joint optimization. Continuous remeshing[31] provides a tool for achieving higher-quality geometric optimization, avoiding unreasonable face patches during the refinement process. Similar to NeRF2Mesh, it also neglects the influence of appearance on the refinement process, potentially missing opportunities for enhanced reconstruction quality. Other end-to-end pipelines [23, 34, 35, 37, 38] for mesh reconstruction from multi-view images still face challenges in reconstructing fine details. Like the aforementioned methods, these approaches often disregard the role of appearance, limiting their ability to leverage joint geometry-appearance optimization for improved results.

3. Method

As shown in Fig. 1, starting from multi-view images, we first use off-the-shelf 3DGS methods [14, 18, 47] to reconstruct the scene, then compute TSDF upon the 3DGS representation, and finally obtain the initial mesh $\mathcal{M}_{ini} = (V^0, T^0, C^0)$ with marching cube algorithm. Here $V^0 = \{v_i \in \mathbb{R}^3\}_{i=1}^n$, $T^0 = \{t_j\}_{j=1}^m$, and $C^0 = \{c_i \in \mathbb{R}^3\}_{i=1}^n$, are respectively the vertex, face and per-vertex color extracted from the reconstruction. From Sec. 3.1 to Sec. 3.3, we introduce our texture-guided remeshing, which effectively refines the geometry of $\mathcal{M}_{ini}$ while preserving rendering quality. On top of the improved textured mesh, in Sec. 3.4 we propose a novel approach to bind mesh to Gaussian, which improves results in tasks such as relighting and deformation.

3.1. Geometry-Color Remeshing Operations

It is well-known that geometry is generally not well reconstructed by 3DGS on their own (see also Fig. 3). Our first goal is to refine $\mathcal{M}_{ini}$. Obviously, independently optimizing it with respect to geometric loss would fall short of preserving the rendering quality. Our key insight is to introduce the appearance attributes, namely, *color*, to join the geometric refinement.

For mesh refinement, we adopt the framework of ContinuousRemeshing [31], which leverage inverse rendering technique [21] to remesh a sphere to a target mesh. The remeshing is performed by enforcing the normal and depth image of remeshed object to approximate those computed on the target from multiple views. In particular, to accommodate the color attributes, we extend the standard remeshing operations to the following geometry-color-based ones:
Edge Split with Color Interpolation: When splitting edge $e = (v_i, v_j)$ on triangle (v_i, v_j, v_l), we create a new vertex v_k with position and color bilinearly interpolated at the midpoint of e, after that creating three edges e_1, e_2, e_3 and removing one edge e:

$$\begin{aligned} &(v_k, c_k) := \left(\frac{v_i + v_j}{2}, \frac{c_i + c_j}{2}\right), \\ &e_1 := (v_l, v_k), e_2 := (v_i, v_k), e_3 := (v_j, v_k), \\ &\text{remove } e = (v_i, v_j). \end{aligned} \tag{1}$$

Edge Collapse with Color Fusion: Collapsing edge $e = (v_i, v_j)$ propagates color information through merging the two endpoints of the edge to the midpoint.We move v_i to the midpoint and still mark it as v_i. For any edge $e_{any} = (v_{any}, v_j)$ connected to vertex v_j, we change it to (v_{any}, v_i). We define all edges between two endpoints with more than one edge as redundant and remove them:

$$\begin{aligned} &(v_i, c_i) := \left(\frac{v_i + v_j}{2}, \frac{c_i + c_j}{2}\right), \\ &e_{any} = (v_{any}, v_j) \rightarrow (v_{any}, v_i), \\ &\text{remove } e \text{ where redundant.} \end{aligned} \tag{2}$$

Edge Flip with Color Preservation (optional): For edge $e = (v_i, v_j)$ between triangles (v_i, v_j, v_k) and (v_i, v_j, v_l), flipping to (v_k, v_l) preserves color coherence through:

$$e = (v_i, v_j) \rightarrow (v_k, v_l). \tag{3}$$

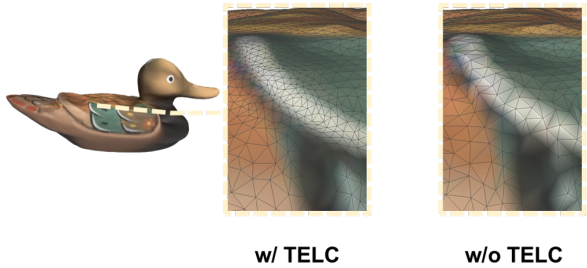

Figure 2. Remeshing results with (middle) and without (right) texture density based edge length control (*i.e.,* TELC).

To preserve color consistency during optimization, we note that edge flipping can introduce abrupt color changes at patch centroids due to interpolation, particularly when neighboring faces exhibit significant color variations. Therefore, we implement edge flipping intermittently, executing the operation every few optimization steps rather than continuously.

We defer details of our optimization goal, which involves photometric consistency and geometric regularization, to Sec. 3.3. Similarly, we refer readers to Sec. 6 of the Supp. for the details of the remesh algorithm.

3.2. Texture-Based Edge Length Control

Though the geometry-color remeshing operations presented in the last part enables flexible and efficient update on color attributes of each vertex, it can potentially introduce color artifacts due to the linear nature of color assignment. Therefore, we shall take gradients over the appearance domain into consideration of performing remeshing operations.

To see this, let us consider the mallard on the left of Fig. 2, whose wing exhibits both sharp color transition (from green to white) and smooth geometric change. Without control signal from appearance domain, we end up at the right panel of Fig. 2 – there exists large triangle face crossing the boundary, leading to color leakage in such areas. Intuitively, we would like to have smaller triangles crossing the boundary in appearance, and respectively larger triangles among flat regions from the perspectives of both geometry and appearance. To achieve such, we introduce a simple yet effective edge length control scheme, which incorporates the frequency change computed in the appearance domain.

In ContinousRemeshing [31], one computes a constant optimal edge length l_{ref}^{k+1} at each iteration based on the geometry obtained at k^{th} iteration. Subsequently, we define edge length tolerance ϵ to constrain the range of edge length. For each edge l at $(k+1)^{th}$ iteration, one performs remeshing if its length deviates from l_{ref}^{k} by a margin, namely, when

$$|length(l) - l_{ref}^{k+1}| > \epsilon \times l_{ref}^{k+1}. \tag{4}$$

Now we let $\mathcal{M}$ be the mesh at the $(k+1)^{th}$ iteration of remeshing, and computes l_{ref}^{k+1} following [31], which is based on geometry. Recall that we are given multi-view images $\mathcal{I} = \{I_1, I_2, ..., I_s\}$ as input, where each $I_i \in \mathbb{R}^{3\times H\times W}$. We proceed through the following steps:

Compute texture density map: For each pixel, p in I_i, we consider the 3×3 neighborhood around it, then perform Fast Fourier transform (FFT) and compute the magnitude of the FFT output, which is a single scalar value assigned to p, reflecting the oscillation in the regarding patch. Going through all pixels and all images, we obtain the texture density map $\mathcal{F} = \{f_1, f_2, ..., f_s\}$, where each $f_i : p \in I_i \rightarrow \mathbb{R}^+$.

Back-project texture density map to meshes and normalize: For each vertex v_p in the mesh $\mathcal{M}$, we consider the image subset where it is visible and denote by $vis(p)$ the regarding indices in $\{1, 2, ..., s\}$. We back-project v_p to a pixel in each $I_j, j \in vis(p)$, and let $f_j(p)$ be the texture density of the very pixel in f_j. The texture density of p is then defined as

$$f(v_p) = \frac{\sum_{j\in vis(p)} f_j(p)}{\#vis(j)}, \tag{5}$$

where $\#A$ returns the cardinality of set A. Finally, we normalize the per-vertex texture density as follows, so that $0 \leq f(v_p) \leq 1$ for all p

$$f(v_p) \leftarrow \frac{f(v_p) - \min\{f(v_q), v_q \in \mathcal{M}\}}{\max\{f(v_q), v_q \in \mathcal{M}\} - \min\{f(v_q), v_q \in \mathcal{M}\}}. \tag{6}$$

Compute per-edge texture density map: Now considering an edge of $\mathcal{M}$, $l = (v_p, v_q)$, we set the texture density of l as

$$F_l = (f(v_p) + f(v_q))/2. \tag{7}$$

Our adaptive edge control scheme then injects the per-edge texture density into Eqn. 4, namely, we perform remeshing on edge l whenever

$$|length(l) - l_{ref}^{k+1} \times (1 - F_l)| > \epsilon \times l_{ref}^{k+1} \times (1 - F_l). \tag{8}$$

Intuitively, when l is among a region with high frequency, $1 - F_l$ approaches 0, which makes it easier to trigger the remeshing condition.

With the above scheme, our remeshing result is shown in the middle of Fig. 2, which is clearly improved as the band region are entirely white now. Overall, our scheme allows for a more fine-grained control of mesh resolution, especially in regions where textures exhibit high-frequency details, leading to a more accurate and visually consistent result.

3.3. Mesh Optimization via Inverse Rendering

Now we proceed to describe our remeshing procedure. We first render pseudo-ground-truth depth maps $\mathcal{D} =$

Table 1. Quantitative comparison on the DTU Dataset. Our method largely improves the reconstruction accuracy on other explicit mesh reconstruction methods with a short refinement process.

Method	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Mean	Time(hours)
NeuS	1.00	1.37	0.93	0.43	1.10	0.65	0.57	1.48	1.09	0.83	0.52	1.20	0.35	0.49	0.54	0.84	> 12
Neuralangelo	0.37	0.72	0.35	0.35	0.87	0.54	0.53	1.29	0.97	0.73	0.47	0.74	0.32	0.41	0.43	0.61	> 12
3DGS	1.45	1.46	1.85	1.47	2.56	2.19	1.26	1.93	1.73	1.51	1.69	2.04	1.19	1.09	1.10	1.63	0.2
Ours + 3DGS	1.25	1.32	1.53	1.03	2.55	2.05	1.09	1.81	1.59	1.42	1.46	2.04	0.96	0.81	0.86	1.45	0.2 (+0.1)
GOF	0.47	0.82	0.40	0.36	1.28	0.83	0.76	1.19	1.24	0.75	0.74	1.12	0.49	0.69	0.57	0.78	0.3
Ours + GOF	0.42	0.76	0.35	0.33	1.22	0.74	0.66	1.13	1.23	0.70	0.65	1.14	0.43	0.57	0.46	0.72	0.3 (+0.1)
2DGS	0.49	0.82	0.34	0.42	0.95	0.86	0.82	1.29	1.24	0.66	0.64	1.44	0.41	0.67	0.50	0.77	0.2
Ours + 2DGS	0.40	0.75	0.30	0.33	0.96	0.76	0.71	1.24	1.20	0.60	0.55	1.40	0.39	0.55	0.40	0.70	0.2 (+0.1)
PGSR	0.34	0.55	0.40	0.36	0.78	0.57	0.49	1.08	0.87	0.59	0.49	0.51	0.30	0.37	0.34	0.53	0.6
Ours + PGSR	0.34	0.50	0.38	0.34	0.74	0.54	0.47	1.03	0.85	0.56	0.47	0.49	0.29	0.36	0.33	0.51	0.6 (+0.15)

Table 2. Quantitative comparison on the DTC Dataset (values multiplied by 1000). Our method improves the object reconstruction accuracy on baseline methods.

Method	**Airplane**	**BirdHouse**	**Car**	**CaramicBowl**	**Cup**	**DutchOven**	**Hammer**	**Keyboard**	**Kitchen**	**Mallard**	**Planter**	**Pottery**	**Shoe**	**Spoon**	**Teapot**	**Vase**	**Mean**
GOF	3.33	1.60	0.89	2.83	2.60	1.93	1.39	4.23	2.77	1.60	2.52	3.44	2.13	2.60	1.62	4.12	2.48
Ours + GOF	2.67	1.17	0.91	2.77	2.30	2.01	0.99	2.23	2.19	1.26	2.19	3.84	2.12	2.18	1.62	3.51	2.12
2DGS	2.24	1.26	1.06	3.28	2.84	1.90	1.20	2.49	1.67	1.99	1.64	3.79	2.20	1.86	2.29	2.93	2.17
Ours + 2DGS	2.21	1.10	0.84	3.21	2.48	1.83	0.92	2.50	1.22	1.63	1.46	3.93	2.09	1.84	2.21	2.28	1.98
PGSR	1.99	1.13	0.96	1.90	1.46	1.40	1.01	2.50	0.89	1.65	1.20	1.93	2.07	1.75	2.33	2.32	1.66
Ours + PGSR	2.01	1.05	0.91	1.78	1.35	1.40	0.88	2.50	0.80	1.41	1.06	1.82	2.06	1.63	2.22	1.98	1.55

$\{d_1, d_2, ..., d_s\}$ and normal maps $\mathcal{N} = \{n_1, n_2, ..., n_s\}$ via $\mathcal{M}_{ini}$, the initial mesh extracted from Gaussians, from the input camera views for later regularization.

At each iteration of remeshing, we denote the regarding mesh $\mathcal{M}^k = (V^k, T^k, C^k)$. Via rasterization function $\mathbf{R}$, we can compute

$$I_i^k, d_i^k, n_i^k = \mathbf{R}(V^k, T^k, C^k, MV_i, P_i), i = 1, 2, ..., s, \tag{9}$$

where MV_i is the model-view matrix of the camera, and P_i is the projection matrix of the camera, and I_i^k, d_i^k, n_i^k are respectively the RGB, depth and normal image rendered from viewpoint i.

In general, at each iteration, we enforce 1) the RGB rendering to approximate the input multiview images; 2) the rendered depth and normal images to be close to the one computed on $\mathcal{M}_{ini}$ for regularization; 3) the remeshed vertex positions and normals are smooth regarding mesh Laplacian.

It is worth noting that, though our remeshing pipeline is built on [31], our loss design differs significantly from the former as 1) We introduce photometric consistency into remeshing; 2) The optimization in [31] depends on the *ground-truth* normal and depth of the target, while our framework leverages pseudo-label obtained from multiview images; 3) [31] is primarily guided by the ground-truth geometry, therefore it applies Laplacian-based smoothness regularization on the gradient, which is too weak for our challenging task.

To conclude, our loss function is as follows:

$$\mathcal{L} = \lambda_{rgb}\mathcal{L}_{rgb} + \lambda_{geo}\mathcal{L}_{geo} + \lambda_{reg}\mathcal{L}_{reg}, \tag{10}$$

where $\mathcal{L}_{rgb}$ is the loss term for RGB images, $\mathcal{L}_{geo}$ is the loss term for depth maps and normal maps, and $\mathcal{L}_{reg}$ is the regularization term, which includes Laplacian smoothing and mesh normal consistency. The coefficients λ_{rgb}, λ_{geo}, and λ_{reg} are the respective weights for each term. We refer readers to Sec. 7 of the Supp. for the details of each loss term.

3.4. Vertex-Gaussian Binding for Relighting and Deformation

In this part, we introduce a vertex-Gaussian binding scheme, which exploit the improved geometry we obtained above to enhance down-stream editing applications

Given an optimized mesh $\mathcal{M}^* = (V^*, T^*, C^*)$, we define the transformation from the mesh to Gaussian parameters. For each vertex $v_i \in V^*$, we associate a corresponding Gaussian with the following parameters:

1. **Position**: Direct correspondence between vertex v_i and Gaussian position μ_i.
2. **Scale**: Composed of three components capturing local edge projections on the tangent plane.
3. **Rotation**: Orthonormal basis derived from vertex normal and tangent vectors.
4. **Opacity**: In our method, we assign a constant opacity value of 0.9 to each Gaussian, assuming that every point on the mesh is visible.

5. **Spherical Harmonics (SH) coefficients**: In our method, we assign the low-order SH coefficients directly from the vertex color c_i, and set the higher-order coefficients to zero.

We refer readers to Sec. 8 of the Supp. for more details. In Sec. 4.2, we feed in the above constructed 3DGS as input to R3DG [9], and demonstrate that the improved initialization directly boosts the final relighting performance.

4. Experiments

4.1. Remeshing evaluation

Baselines: Our baselines include both implicit ones – NeuS[42] and Neuralangelo[22] and explicit ones – 3DGS[18], 2DGS[14], GOF[47] and PGSR[4]. Our refinement is mainly applied on the latter four.

Benchmarks: We evaluate the performance of our method on various datasets. **DTU**[15] dataset comprises 15 scenes, each with 49 or 64 images of resolution 1600×1200. Different from the DTU dataset, which only includes partial surfaces of objects, the latest Digital Twin Catalog(**DTC**)[32] dataset provides multiview images of complete objects along with ground-truth meshes for evaluation. DTC dataset contains more than 100 objects and theirs multiview images. We consider 16 cases (*e.g.*, airplane, birdhouse) as our benchmark and down-sample all images in DTC dataset to half of their original size (*i.e.*, 1000×1000).

Implementation Details: For **DTU**[15] dataset, we initialize max edge length to $1\mathrm{e}{-3}$ and min edge length to $1\mathrm{e}{-4}$. For **DTC** [32] dataset, we initialize max edge length to $4\mathrm{e}{-3}$ and min edge length to $4\mathrm{e}{-4}$. During surface mesh refinement, we set λ_{rgb} to 3.0, λ_{reg} to 0.3, λ_{geo} to 0.1 and edge length tolerance ϵ to 0.5. We train 1000 iterations per scene and the learning rate is set to $1\mathrm{e}{-3}$. We conduct all the experiments on a single RTX3090 GPU.

Geometry Evaluation: We first compare against SOTA implicit and explicit methods on Chamfer Distance and training time using the DTU dataset in Tab. 1. Our method outperforms all compared methods in terms of Chamfer Distance. We integrated our method into 3DGS, GOF, 2DGS and PGSR, respectively, and observe consistent improvement in each case. Notably, our method requires only a short optimization time to improve the quality of surface reconstruction, making it plug-and-play for any Gaussian-based surface reconstruction method. As illustrated in Fig. 3, the surfaces reconstructed by 2DGS exhibit geometric blurring , while our approach can achieve higher quality reconstruction results. We further compare against 2DGS, GOF and PGSR on **DTC** dataset in Tab. 2. The same trend is observed – As shown in Fig. 3, our method maintains excellent performance in areas with intricate geometric details, particularly evident in the geometry of the athletic shoe at the bottom right corner of the image, where the geometric intricacies of the shoe's surface is better recovered by our refiment.

Table 3. Quantitative comparison on DTU and DTC Dataset

Method	DTU objects			DTC objects		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
GOF	24.81	0.858	0.194	25.16	0.949	0.063
Ours + GOF	<u>25.63</u>	<u>0.897</u>	<u>0.160</u>	**<u>27.12</u>**	<u>0.960</u>	<u>0.049</u>
2DGS	23.82	0.853	0.199	25.16	0.950	0.058
Ours + 2DGS	**<u>26.21</u>**	**<u>0.906</u>**	**<u>0.148</u>**	<u>26.25</u>	**<u>0.962</u>**	**<u>0.042</u>**

Rendering Evaluation: To evaluate the quality of mesh vertex color, we render the mesh to pixel space and compare rendered image with ground-truth image. As shown in Tab. 3, our method achieves a significant improvement in rendering quality compared to coarse meshes extracted from various Gaussian Splatting (GS) approaches. More specific details are illustrated in Fig. 3: the rendering results of coarse meshes exhibit blurriness and a lack of detail clarity. After our refinement, texture details such as the text on the airplane, mesh surface details of the sneakers, and window details of the house in the figure have been fully restored.

4.2. Relighting and Deformation Editing

As mentioned in Sec. 3.4, with our optimized meshes with vertex colors, we initialize Gaussian Splatting (GS) using our binding scheme and feed such as input to R3DG [9] to learn material parameters. The learned material parameters are transferred to the mesh via Gaussian binding correspondences, with a 100-iteration noise filtering applied during backpropagation to mitigate renderer discrepancies. We demonstrate the effectiveness of our method by performing relighting on the Synthetic4Relight dataset[49].

Relighting Evaluation As presented in Tab. 4, our version of initialization helps to improve relighting, albedo and roughness precision in the framework of R3DG. Benefiting from our Gaussian mesh binding method, we can effortlessly transfer parameters from R3DG to the mesh in a one-to-one correspondence. As demonstrated, the transferred metrics exhibit significant superiority in relighting over previous mesh-based approaches, while achieving these improvements with reduced computational time. Qualitatively, our method achieves visually pleasing material decomposition, facilitating a realistic relighting effect (see Fig. 4).

Last but not the least, we visualize the distributions of Gaussian points in R3DG and those of our proposed method in Fig. 5 – With a comparable number of Gaussian points, the distribution in our method is explicitly guided by the underlying mesh geometry, resulting in a more uniform spatial allocation. This structural advantage directly contributes to the superior material learning performance of our approach

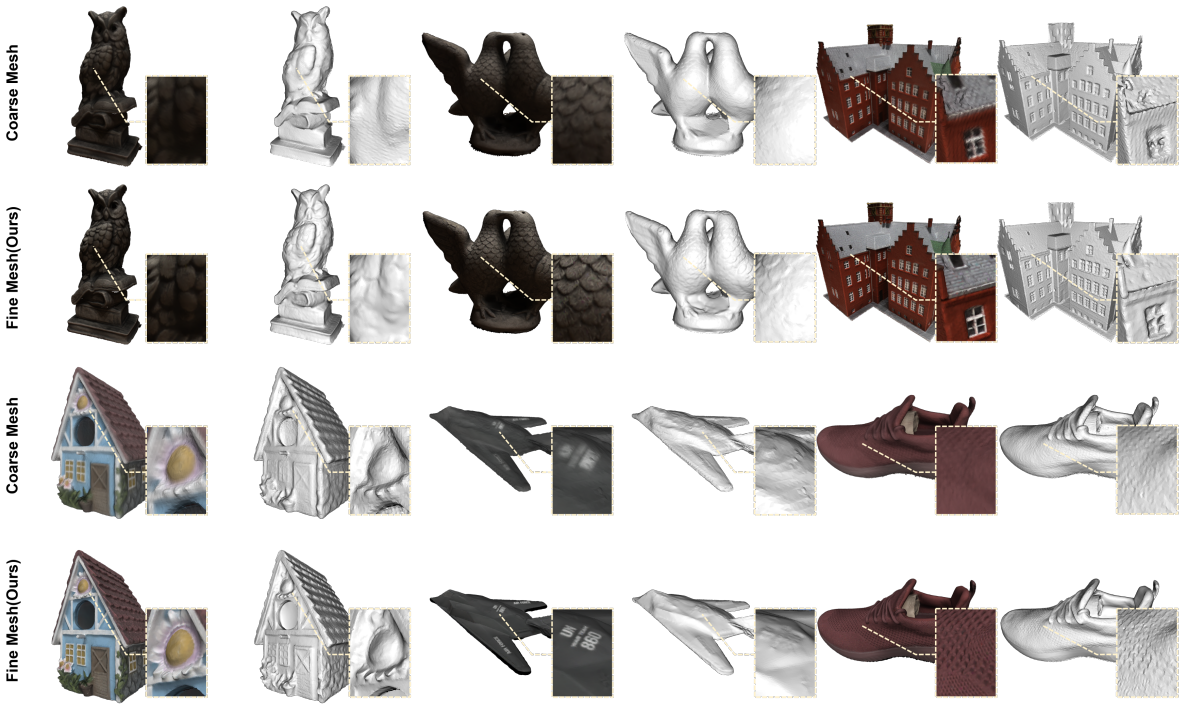

Figure 3. Qualitative results on DTU and DTC dataset

Table 4. Quantitative Results on Synthetic Dataset

Method	Novel View Synthesis			Relighting			Albedo			Roughness	Time
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	MSE ↓	(hours)
R3DG	36.80	0.982	0.028	31.00	0.964	0.050	28.31	0.951	0.058	0.013	1.5
Ours R3DG	33.44	0.969	0.052	32.87	0.965	0.054	29.20	0.948	0.065	0.009	1
Nvdiffrecmc	34.29	0.967	0.068	24.22	0.943	0.078	29.61	0.945	0.075	0.009	4.17
Ours Mesh	31.36	0.962	0.055	30.40	0.942	0.083	27.35	0.928	0.081	0.010	1+2m

compared to R3DG.

Deformation Evaluation We validate the GS-mesh binding through large-scale geometric deformation, as visualized in Fig.6. Applying a 60° X-axis twist to the jug mesh induces synchronized transformations on both the explicit surface and the bound Gaussians in R3DG. Crucially, the corresponding positional and normal adjustments of Gaussians preserve photorealistic interactions with environmental lighting: specular highlights shift coherently along the deformation path while cast shadows naturally elongate according to surface curvature changes. This parallel behavior of illumination effects demonstrates our method maintains physical consistency between mesh editing and GS manipulation. The results confirm that even under extreme topology changes, our binding mechanism successfully propagates deformations while retaining the original relighting properties of both representations.

Table 5. Ablation Study on supervision and reprojection (DTU scenes)

Config	Rendering			Geometry
	PSNR ↑	SSIM ↑	LPIPS ↓	CD ↓
Ours	26.21	0.906	0.148	0.70
w/o RGB Loss	23.66	0.843	0.206	0.89
w/o GEO Loss	25.80	0.897	0.160	0.73
w/o Length Control	25.07	0.871	0.168	0.71

4.3. Ablation Study

In this section, we systematically evaluate the impact of specific components of our approach.

We start by analyzing the loss functions, which is demonstrated in Tab. 5. First, we discover that RGB loss supervision plays a critical role in our method, whose absence leads to a significant decrease in rendering quality and Chamfer Distance, highlighting its importance in cap-

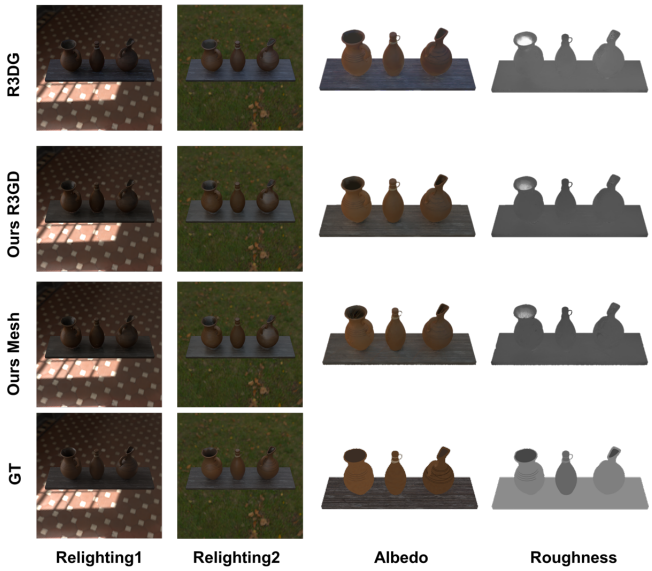

Figure 4. Qualitative results on Synthetic4Relight dataset

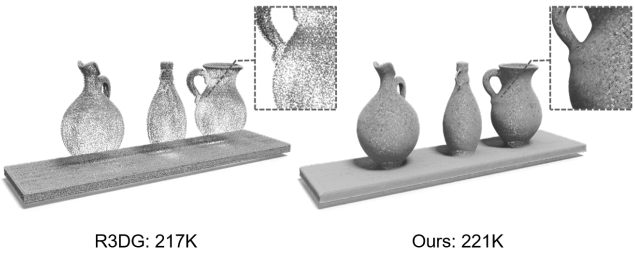

Figure 5. Comparison of points distribution between Ours and R3DG

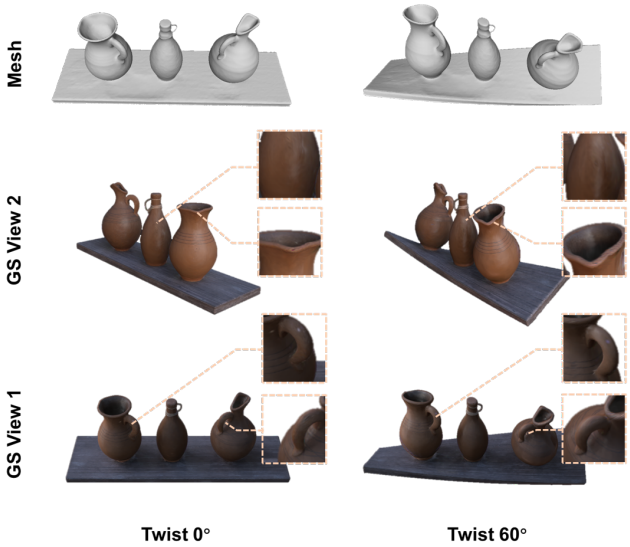

Figure 6. GS relight with mesh deform

Table 6. Ablation Results on edge length initialization (scan 65 from DTU scenes)

Metrics	(Min Length, Max Length)		
	(2e−4, 2e−3)	(4e−4, 4e−3)	(8e−4, 8e−3)
Chamfer Distance	0.76	0.76	0.82
Number of Vertices	609K	159K	46K

turing fine details and color information for accurate texture and geometric reconstruction. Second, the removal of geometry loss supervision results in a slight decrease in rendering quality and Chamfer Distance. Finally, omitting the edge length control based on texture density component also leads to a decrease in rendering quality, while Chamfer Distance remains stable. As illustrated in Fig. 2, after incorporating our length control, the mesh demonstrates more detailed representations in texture-dense regions while retaining its original configuration in texture-uniform areas.

We further validate the edge-length control parameters by varying the minimum and maximum thresholds (Tab. 6). Decreasing the thresholds produces a much denser mesh (about $4\times$more vertices) but yields essentially no accuracy gain. In contrast, increasing the thresholds markedly reduces mesh resolution and leads to a clear drop in reconstruction quality. Overall, the mid-range setting $(4\mathrm{e}{-4},\ 4\mathrm{e}{-3})$ provides a favorable trade-off between mesh complexity and reconstruction accuracy.

5. Limitations and Conclusion

We quantitatively observed that our refinement is less effective in scenarios with poor lighting conditions (see case 110 of Tab.1 and DutchOven in Tab.2). We refer readers to Sec. 9 of the Supp. for details. This work presents a unified framework for jointly optimizing geometry and appearance, mitigating the geometry–texture misalignment in generic 3DGS pipelines. By co-optimizing mesh vertices and colors under photometric and geometric constraints, we produce high-fidelity, editable textured meshes. Coupling parametric Gaussians with mesh vertices further enables synchronized material control and surface deformation, improving reconstruction quality and supporting interactive 3D editing. This advancement paves the way for more intuitive and efficient workflows in virtual environment design, digital content creation, and beyond, where cohesive geometry-appearance manipulation is essential.

Acknowledgements: This work is supported by the National Natural Science Foundation of China under contract No. 62171256, 62205178.

References

[1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5855–5864, 2021. 1

[2] Jia-Wang Bian, Wenjing Bian, Victor Adrian Prisacariu, and Philip Torr. Porf: Pose residual field for accurate neural surface reconstruction. *arXiv preprint arXiv:2310.07449*, 2023. 1

[3] S. Bullinger, C. Bodensteiner, and M. Arens. 3d surface reconstruction from multi-date satellite images. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLIII-B2-2021:313–320, 2021. 1

[4] Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *arXiv preprint arXiv:2406.06521*, 2024. 2, 6

[5] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. 2

[6] Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. High-quality surface reconstruction using gaussian surfels. *ArXiv*, abs/2404.17774, 2024. 2

[7] Peter Eisert and Anna Hilsmann. Hybrid human modeling: making volumetric video animatable. *Real VR–Immersive Digital Reality: How to Import the Real World into Head-Mounted Immersive Displays*, pages 167–187, 2020. 1

[8] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2009. 1

[9] Jian Gao, Chun Gu, Youtian Lin, Zhihao Li, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing. In *European Conference on Computer Vision*, pages 73–89. Springer, 2024. 6

[10] Lin Gao, Jie Yang, Bo-Tao Zhang, Jiali Sun, Yu-Jie Yuan, Hongbo Fu, and Yu-Kun Lai. Mesh-based gaussian splatting for real-time large-scale deformation. *ArXiv*, abs/2402.04796, 2024. 2

[11] Xiangjun Gao, Xiaoyu Li, Yi Zhuang, Qi Zhang, Wenbo Hu, Chaopeng Zhang, Yao Yao, Ying Shan, and Long Quan. Mani-gs: Gaussian splatting manipulation with triangular mesh. *ArXiv*, abs/2405.17811, 2024. 2

[12] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5354–5363, 2023. 2

[13] Jon Hasselgren, Nikolai Hofmann, and Jacob Munkberg. Shape, light, and material decomposition from images using monte carlo rendering and denoising. *Advances in Neural Information Processing Systems*, 35:22856–22869, 2022. 1

[14] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *International Conference on Computer Graphics and Interactive Techniques*, 2024. 1, 2, 3, 6

[15] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engin Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 406–413, 2014. 6

[16] Julian Kaltheuner, Patrick Stotko, and Reinhard Klein. Rosa: Reconstructing object shape and appearance textures by adaptive detail transfer. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2910–2920. IEEE, 2025. 2

[17] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, 2006. 2

[18] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42:1 – 14, 2023. 1, 2, 3, 6

[19] Zhengfei Kuang, Yunzhi Zhang, Hong-Xing Yu, Samir Agarwala, Elliott Wu, Jiajun Wu, et al. Stanford-orb: a real-world 3d object inverse rendering benchmark. *Advances in Neural Information Processing Systems*, 36:46938–46957, 2023. 1

[20] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. *International journal of computer vision*, 38:199–218, 2000. 1

[21] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020. 3

[22] Zhaoshuo Li, Thomas Muller, Alex Evans, Russell H. Taylor, M. Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8456–8465, 2023. 2, 6

[23] Zhen Liu, Yao Feng, Yuliang Xiu, Weiyang Liu, Liam Paull, Michael J. Black, and Bernhard Schölkopf. Ghost on the shell: An expressive representation of general 3d shapes. 2024. 3

[24] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004. 1

[25] Alexander Mai, Dor Verbin, Falko Kuester, and Sara Fridovich-Keil. Neural microfacet fields for inverse rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 408–418, 2023. 2

[26] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[27] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a mul-

tiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 1

[28] Jacob Munkberg, Jon Hasselgren, Tianchang Shen, Jun Gao, Wenzheng Chen, Alex Evans, Thomas Müller, and Sanja Fidler. Extracting Triangular 3D Models, Materials, and Lighting From Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8280–8290, 2022. 1

[29] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011. 1

[30] Stanley Osher, Ronald Fedkiw, and Krzysztof Piechor. Level set methods and dynamic implicit surfaces. *Appl. Mech. Rev.*, 57(3):B15–B15, 2004. 1

[31] Werner Palfinger. Continuous remeshing for inverse rendering. *Computer Animation and Virtual Worlds*, 33(5):e2101, 2022. 1, 3, 4, 5

[32] Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Yuheng (Carl) Ren. Aria digital twin: A new benchmark dataset for egocentric 3d machine perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20133–20143, 2023. 6

[33] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 1

[34] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. *Advances in Neural Information Processing Systems*, 34:6087–6101, 2021. 2, 3

[35] Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp, and Jun Gao. Flexible isosurface extraction for gradient-based mesh optimization. *ACM Transactions on Graphics (TOG)*, 42(4):1–16, 2023. 3

[36] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM siggraph 2006 papers*, pages 835–846. 2006. 1

[37] Sanghyun Son, Matheus Gadelha, Yang Zhou, Matthew Fisher, Zexiang Xu, Yi-Ling Qiao, Ming C Lin, and Yi Zhou. Dmesh++: An efficient differentiable mesh for complex shapes. *arXiv preprint arXiv:2412.16776*, 2024. 3

[38] Sanghyun Son, Matheus Gadelha, Yang Zhou, Zexiang Xu, Ming C. Lin, and Yi Zhou. Dmesh: A differentiable representation for general meshes, 2024. 3

[39] Jiaxiang Tang, Hang Zhou, Xiaokang Chen, Tianshu Hu, Errui Ding, Jingdong Wang, and Gang Zeng. Delicate textured mesh recovery from nerf via adaptive surface refinement. *arXiv preprint arXiv:2303.02091*, 2022. 1, 2

[40] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What do single-view 3d reconstruction networks learn? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3405–3414, 2019. 1

[41] Matias Turkulainen, Xuqian Ren, Iaroslav Melekhov, Otto Seiskari, Esa Rahtu, and Juho Kannala. Dn-splatter: Depth and normal priors for gaussian splatting and meshing. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025. 1

[42] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *ArXiv*, abs/2106.10689, 2021. 2, 6

[43] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3272–3283, 2022. 2

[44] Yaniv Wolf, Amit Bracha, and Ron Kimmel. GS2Mesh: Surface reconstruction from Gaussian splatting via novel stereo views. In *European Conference on Computer Vision (ECCV)*, 2024. 1

[45] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1790–1799, 2020. 1

[46] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19447–19456, 2024. 1

[47] Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. *ACM Trans. Graph.*, 43:271:1–271:13, 2024. 1, 2, 3, 6

[48] Kai Zhang, Fujun Luan, Zhengqi Li, and Noah Snavely. Iron: Inverse rendering by optimizing neural sdfs and materials from photometric images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5565–5574, 2022. 2

[49] Yuanqing Zhang, Jiaming Sun, Xingyi He, Huan Fu, Rongfei Jia, and Xiaowei Zhou. Modeling indirect illumination for inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18643–18652, 2022. 6

[50] Youjia Zhang, Teng Xu, Junqing Yu, Yuteng Ye, Yanqing Jing, Junle Wang, Jingyi Yu, and Wei Yang. Nemf: Inverse volume rendering with neural microflake field. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22919–22929, 2023. 2