Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

SDLKF: Signed Distance Linear Kernel Function for Surface Reconstruction

First Author Given Name Surname[a,*], Second Author Given Name Surname[b,1]

[a]*Address, City, Postcode, Country*
[b]*Address, City, Postcode, Country*

ARTICLE INFO

Article history:
Received July 20, 2025

Keywords: Computers and Graphics, 3D Reconstruction, Novel View Synthesis

ABSTRACT

In this paper, we introduce a novel explicit representation for surface reconstruction from multi-view images, named Signed Distance Linear Kernel Function (SDLFK), which simultaneously allows fast rendering and accurate surface reconstruction. The key insight is to use linear kernels to fit the Signed Distance Function (SDF) which has an analytic solution for volume rendering instead of numeric approximation. Specifically, the linear kernel function is defined within ellipsoids and calculated as the signed distance to the principal plane. For each ellipsoid intersected by rays, the expected depth and transmittance can be calculated through volume rendering with a closed-form solution. This procedure allows seamless switching between soft and hard surfaces, where the former facilitates optimization and the latter ensures precise reconstruction. Our evaluations demonstrate that our method improves the detailed geometry compared to state-of-the-art methods while maintaining fast and high-fidelity rendering.

1. Introduction

In recent years, the rapid advancements in computer vision and computer graphics have driven significant progress in the fields of 3D reconstruction and rendering, with applications spanning across various domains such as virtual reality, augmented reality, 3D content creation, and autonomous driving. Central to these advancements are two key tasks: novel view synthesis (NVS) and surface reconstruction. For a long time, these two tasks were treated as separate. However, the emergence of Neural Radiance Fields (NeRF) [1] has made it possible to reconstruct accurate surfaces from multi-view images using differentiable volume rendering. Since then, several works have developed differentiable volume rendering with various scene representations. Some works use neural implicit fields to fit the SDF field (e.g., NeuS [2] , VolSDF [3]). Grid-based methods replace heavy networks with voxels to speed up training time, while hash-encoding methods further accelerate rendering time. However, state-of-the-art methods, such as Neuralangelo [4], still require hundreds of hours to reconstruct a scene. More recently, 3D Gaussian Splatting (3DGS) [5] has emerged as an appealing alternative to these methods because of its photorealistic NVS results, efficient training, and real-time rendering. Thus, various methods have extended 3DGS to surface reconstruction. Notably, 2DGS [6] and PGSR [7] use surfel-based Gaussians, defining an accurate ray-Gaussian intersection point. However, the native attribute of the Gaussian function leads to soft geometry and soft boundaries, introducing multi-view inconsistency and resulting in inaccurate reconstruction and blurred images. GOF [8] fits the opacity field with Gaussian kernels, mixing volume rendering and Gaussian splatting. GOF manually defines the ray-Gaussian intersection point and the transmittance instead of performing volume rendering in individual Gaussian kernels, resulting in a bias between the geometry and radiance of the opacity field. GSDF [9] train implicit SDF field together with Gaussians, while the result quality is highly limited by the SDF field instead of Gaussians. Designing a suitable scene representation is the key to fast and accurate

*Corresponding author: Tel.: +0-000-000-0000; fax: +0-000-000-0000;
e-mail: example@email.com (Corresponding Author Name)
[1]Footnote 1.

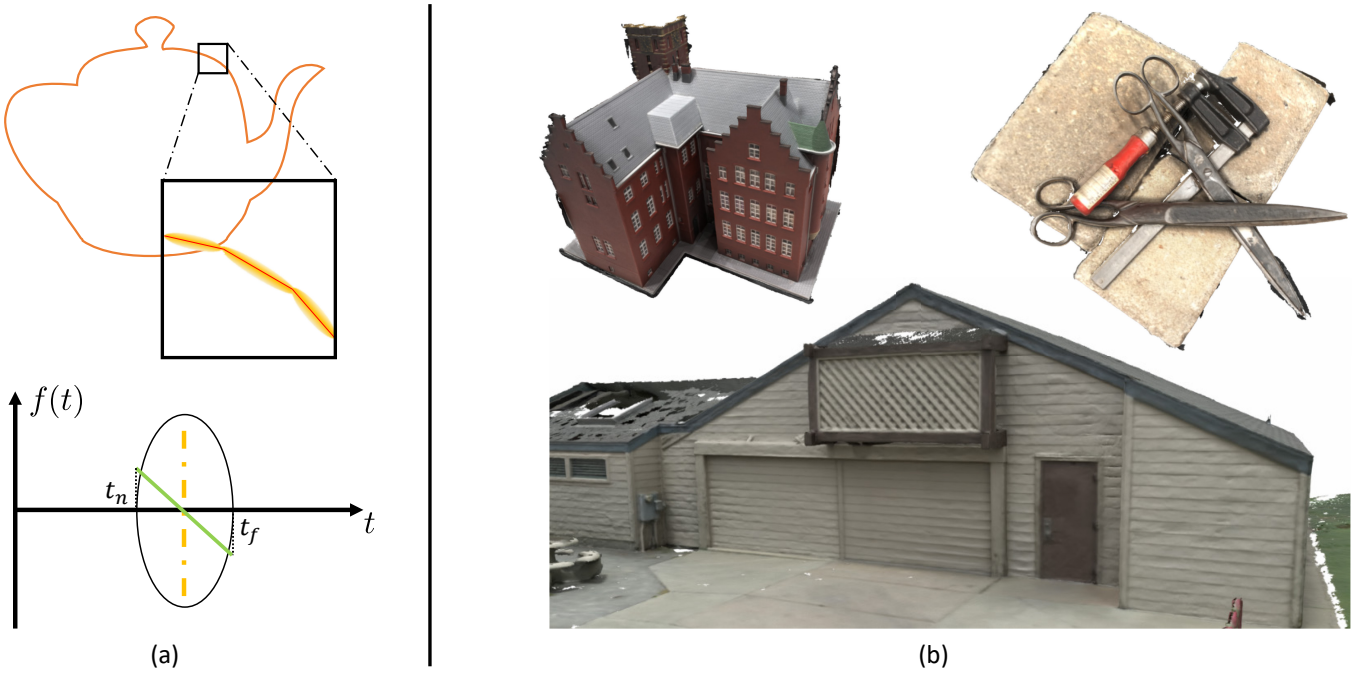

Fig. 1. SDKLF representation. (a) Signed Distance Linear Kernel Function uses several SDF kernels to represent the whole scene geometry. Each SDF is defined in the ellipsoid and computed as the distance to the principal plane of the ellipsoid. (b) SDKLF reconstructs accurate and detailed geometry in real-scene scans.

surface reconstruction.

This paper introduces a new scene representation for multi-view surface reconstruction and novel view synthesis, called Signed Distance Linear Kernel Function (SDLKF). As shown in Fig.1,our method employs a set of local SDF kernels, defined by ellipsoids, to model surface geometry . Instead of modeling opacity with Gaussian kernels, we perform volume rendering on geometry field within each kernel to establish a stronger correlation between geometry and rendered images. Specifically, each local SDF kernel is represented as an ellipsoid with a linear Signed Distance Function defined by its principal plane. When a ray intersects the ellipsoid, volume rendering is performed between the two intersection points. The transmittance and depth are quickly computed using a closed-form solution, ensuring fast volume rendering. During volume rendering, each kernel is equipped with an optimizable parameter to controls the solidity of the geometry ,balancing the ease of optimizing soft geometry with the accuracy of solid surfaces. Furthermore, achieving clean geometry requires more than photometric loss alone. To this end, we incorporate single-view and multi-view geometry constraints. For single-view constraints, we adopt the depth-normal consistency loss from [6] for local smoothness, and propose an edge-aware normal loss for capturing finer details. For multi-view constraints, we introduce normalized cross correlation(NCC) loss and reprojection loss, inspired by [7].

Our contributions can be summarized as follows:

- We propose a novel scene representation, SDLKF, equipped with a fast and high-fidelity differentiable renderer that supports accurate surface reconstruction.
- We introduce an individual learnable parameter for each kernel, ensuring smooth optimization and multi-view consistency.
- We develop a new framework incorporating single-view and multi-view constraints to train our SDLKF.
- Our approach achieves state-of-the-art geometry reconstruction results compared to other explicit and implicit representations.

2. Related Works

In recent years, the field of surface reconstruction has witnessed substantial advancements, primarily driven by innovations in multi-view stereo (MVS) [10, 11], neural implicit surface reconstruction [12, 13], and Gaussian splatting-based methods [5, 14]. However, they also face common challenges, including computational efficiency, geometric accuracy, and ensuring multi-view consistency. In this section, we review the related works, focusing on the advancements in neural implicit surface reconstruction and gaussian splatting-based surface reconstruction, which are most relevant to our approach.

2.1. Multi-view Stereo Surface Reconstruction

Multi-view Stereo(MVS) Surface Reconstruction is a well-established technique in computer vision that has been developed over several decades. It focuses on reconstructing the 3D surface of a scene from multiple 2D images captured from different viewpoints [15, 16, 17, 18, 19]. Recent developments in MVS have leveraged deep learning and neural networks to improve the accuracy and efficiency of surface reconstruction

like [20, 21, 22, 23, 24]. However, despite these advancements, MVS methods still face limitations, particularly in dealing with textureless surfaces and areas with large variations in illumination. To address these challenges, researchers have proposed Neural Radiance Fields, which model scenes as continuous radiance fields and offer more robust performance in such difficult scenarios.

2.2. Neural Implicit Surface Reconstruction

Neural implicit surface reconstruction methods leverage deep neural networks to directly predict surface models from images. These methods has gained significant attention due to their ability to model complex geometries from sparse or incomplete data. They often utilizes SDFs or other implicit representations to capture the underlying geometry of objects or scenes, allowing for high-fidelity reconstructions with smooth surface details However, these approaches are computationally expensive during inference and require extensive labeled training data, which limits their real-time performance and practical applicability.

Neural Radiance Fields(NeRF) [1], a widely used neural rendering technique, predicts density and color in continuous space from 5D ray inputs, enabling high-fidelity novel view synthesis. Despite its capabilities, NeRF falls short in capturing high-fidelity surfaces. To address this, some methods [2, 25, 26, 27] have adapted the NeRF architecture for surface reconstruction by introducing intermediate representations such as occupancy fields [28] or SDFs [29]. Subsequent research has aimed to reduce dependence on MLP layers which constraints the inference time and representation capacity by decomposing scene information into separable structures, such as points or voxels [30, 31]. Some work adapted the hybrid representation to improve the neural networks' performance like [32, 33, 34, 4, 35]

Specially, Neuralangelo [4] employs a multi-resolution hash grid encoding combined with neural SDFs to represent 3D scenes, enabling high-fidelity surface reconstruction from multi-view RGB images. The method uses numerical gradients to compute higher-order derivatives for smoothing and a coarse-to-fine optimization strategy to progressively capture details at different scales, achieving high-quality reconstruction without relying on auxiliary inputs like depth or segmentation. VolSDF [3] proposed a neural volume rendering framework that enhances geometric representation and reconstruction by modeling volume density as a transformed SDF. It provides an inductive bias for geometry learning, bounds the opacity approximation error, and enables high-fidelity sampling and disentanglement of shape and appearance, achieving superior reconstruction quality. For our work, we draw inspiration from NeuS [2], which introduces a novel approach to neural surface reconstruction by representing 3D surfaces as neural Signed Distance Functions and developing a new volume rendering method to train the implicit SDF representation. NeuS's unbiased volume rendering scheme ensures accurate surface reconstruction without the need for mask supervision, effectively handling complex structures and severe occlusions. This innovation significantly advances the field of multi-view surface reconstruction.

2.3. Gaussian Splatting-Based Surface Reconstruction

Recently, Gaussian Splatting has emerged as a powerful technique for efficient rendering and training in 3D graphics. Specifically, 3D Gaussian Splatting (3DGS) [5] optimizes the position, orientation, scale, and appearance of explicit 3D Gaussians. Combined with alpha blending for rendering, it achieves training times in minutes and rendering speeds in milliseconds. While 3DGS excels in novel view rendering and rapid training, the unstructured and irregular nature of Gaussian point clouds makes it challenging to ensure the accuracy of geometric reconstruction and multi-view consistency. In some cases, Gaussian primitive-based methods demonstrate faster rendering speeds compared to NeRF-based methods in both dynamic [36, 37, 38, 39] and large-scale scenarios [40, 41].

SuGaR [42] introduced a method to extract meshes from 3DGS by incorporating regularization terms to encourage Gaussians to fit the surface of the scene. However, the use of biased depth to constrain the density field complicates the reconstruction of smooth surfaces from discrete density fields. 2DGS [6] addresses this by collapsing 3D volumes into a series of 2D oriented Gaussian disks, thus achieving consistent geometry across views. In another way, Gaussian Surfels [43] introduced the representation that flattens 3D Gaussians into 2D ellipses, enhancing surface alignment and reconstruction quality through a self-supervised loss of depth normal consistency and volumetric cutting method. By establishing a ray-tracing-based opacity field and leveraging adaptive mesh extraction with Marching Tetrahedra, GOF [8] achieves detailed and compact surface meshes for both foreground and background regions. PGSR [7] proposes a planar-based Gaussian splatting method for efficient and high-fidelity surface reconstruction from multi-view RGB images. It achieves high reconstruction accuracy with fast training and rendering speeds by introducing unbiased depth rendering and multi-view geometric regularization. Although conceptually similar to our approach, there are significant differences in the plane rendering techniques and the utilization of planes. The main difference lies in the depth rendering, PGSR computes the depth with a blended normal, which means the depth is unbiased only when all the normal of intersected Gaussian are the same. Some methods [9, 44] train the Gaussians along with the implicit SDF field. This hybrid paradigm effectively combines the computational efficiency and photorealistic view synthesis capabilities of 3D Gaussian Splatting (3DGS) with the geometric precision inherent to implicit surface representations, achieving simultaneous optimization of rendering quality and geometry accuracy. However, this scheme imposes significant computational burdens during optimization while demonstrating limited geometric enhancement. Nevertheless, the reconstructed surface quality still depends on the convergence properties of the implicit SDF representation.

3. Method

3.1. Preliminary

3.1.1. Volume rendering

Given a density field and color field, the volume rendering process accumulates the color and depth along the rays to

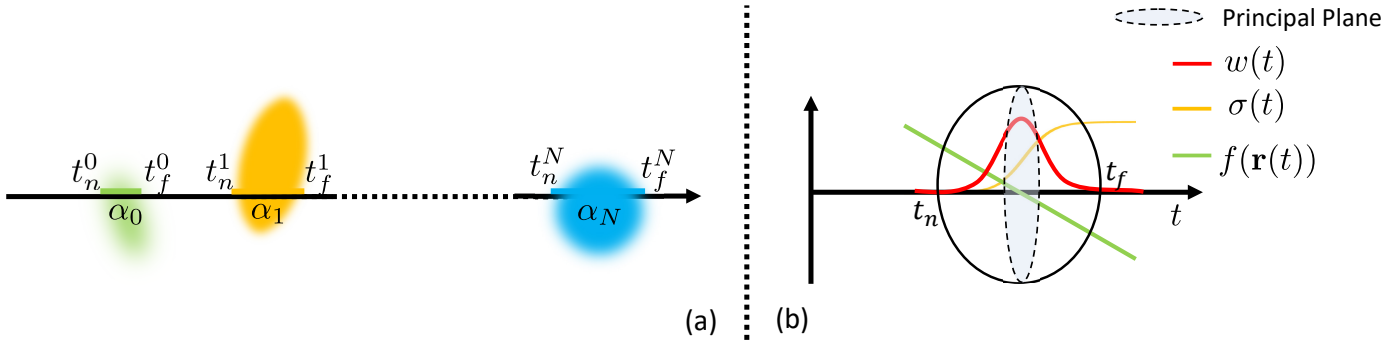

Fig. 2. Rendering Pipeline. (a) A set of SDF Kernels are intersected with a ray and sorted by intersection depth, creating a sequence of intersection intervals $[t_n^0, t_f^0], [t_n^1, t_f^1], \ldots [t_n^N, t_f^N]$. Each kernel renders an alpha value for alpha blending. (b) For a single kernel intersected with a ray, the SDF $f(\mathbf{r}(t))$ is defined as the signed distance to the principal plane of the kernel. Then the density field $\sigma(t)$ and weight field $w(t)$ are computed from the SDF. The alpha value α is obtained by integrating the weight field over the intersection interval $[t_n, t_f]$.

produce the final result. Specifically, given a ray $\mathbf{r}$ originating at $\mathbf{o}$ in the direction of $\mathbf{v} \in \mathbb{R}^3(\|\mathbf{v} = 1\|)$, parameterized as $\{\mathbf{r}(t) = \mathbf{o} + t\mathbf{v}|t \geq 0\}$. The color C and the depth D of the pixel corresponding to the ray $\mathbf{r}$ are defined by following integrals:

$$C(\mathbf{r}) = \int_0^{+\infty} T(t)\sigma(t)\mathbf{c}(\mathbf{r}(t), \mathbf{v})\mathrm{d}t, \tag{1}$$

$$D(\mathbf{r}) = \int_0^{+\infty} T(t)\sigma(t)t\mathrm{d}t, \tag{2}$$

where $\sigma(t)$ is the volume density, $T(t) = \exp(-\int_0^t \sigma(u)\mathrm{d}u)$ denotes the accumulated transmittance along the ray, and the weight function $w(t) = T(t)\sigma(t)$ defines the contribution of each point along the ray.

3.1.2. Volume rendering on SDF

Volume rendering with neural radiance field (NeRF) shows impressive novel view synthesis, yet the geometry often remains noisy and unreliable. In order to fully leverage the potential of volume rendering in surface reconstruction, it is necessary to establish a mapping Ω that converts the signed distance field $f(\mathbf{r}(t))$ into a density field $\sigma(t)$. A naive model uses a bell-shaped distribution centered at zero, which ensures that the surface-ray intersection point corresponds to the maximum density value. However, this introduces a remarkable bias between rendered depth and surface-ray intersecting depth, as the maximum of the weight function $w(t)$ does not occur precisely at the surface-ray intersection. NeuS addresses this issue by suggesting that the density function should be a monotonically increasing function, thereby ensuring that the local maximum of the weight function occurs exactly at the intersection point.

In particular, NeuS employs a Sigmoid function $\Phi_\kappa(x) = (1 + e^{-\kappa x})^{-1}$ as the convert function. The density function is then formulated as:

$$\sigma(t) = \Omega(f(\mathbf{r}(t))) = \kappa|\cos(\theta)|(1 - \Phi_\kappa(f(\mathbf{r}(t)))) \tag{3}$$

where κ is a parameter of sigmoid function, which could control the solidness of the surface and the θ is the angle between the view direction $\mathbf{v}$ and the surface normal $\mathbf{n}$

3.2. Overview

SDLKF uses local SDF kernel to represent the entire scene. Each kernel, shaped as an ellipsoid, is parameterized by the set $\Theta = \{\mu, q, s, \kappa, o, sh\}$, where:

- $\mu \in \mathbb{R}^3, q \in \mathbb{R}^4$, and $s \in \mathbb{R}^3$ are the position, orientation and scale of each kernel, respectively. These parameters define the shape and transformation of the kernel in 3D space. Notice that q is a quaternion and optimized in $\mathbb{R}^4$ space, with a normalization when converting to a rotation matrix.
- κ retains the same role as Eq.3, controlling the solidness of the kernel.
- o, and sh for opacity and view-dependent appearance, respectively, allowing the kernel to model transparency and appearance variation based on the viewing angle.

In following sections, we detail the volume renderer of Signed Distance Linear Kernel Functionin Sec.3.3, which including the color, depth and normal renderers. We also describe the training constrains in Sec.3.4, covering both multi-view and single-view constrains. Finally, the training strategies are discussed in Sec. 3.5.

3.3. Volume renderer

As illustrated in Fig.2, each kernel computes its individual alpha value, color, depth, and normal. These values are then accumulated according to the depth of each kernel. It could be proved that when the kernels do not overlap, this procedure is equivalent to computing Eq.2. Below, we detail the formulation for each attribute of the kernels.

Considering a single ray $\{\mathbf{r}(t) = \mathbf{o} + t\mathbf{v}|t \geq 0\}$ and an individual kernel, we first compute two intersection points between the ray and the ellipsoid, which serve as the upper and lower limits of the integration. For computational efficiency, the ray is transformed into the local coordinate system of the kernel, and normalized by the kernel's scale:

$$\mathbf{o}_g = \mathbf{R}(\mathbf{o} - \mu) \odot s^{-1}, \ \mathbf{v}_g = \mathbf{R}\mathbf{v} \odot s^{-1}, \ \mathbf{r}_g(t) = \mathbf{o}_g + t\mathbf{v}_g, \tag{4}$$

where $\odot$ is element wise product operation, $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the rotation matrix from quaternion q.

3.3.1. Intersection points

In this local coordinate system, the kernels are centered at origin and shaped as spheres. The intersection points could be easily computed by solving the formula $\|\mathbf{r}_g\|^2 = 1$:

$$A = \mathbf{v}_g \cdot \mathbf{v}_g,\ B = 2\mathbf{o}_g \cdot \mathbf{v}_g,\ C = \mathbf{o}_g \cdot \mathbf{o}_g - 1, \tag{5}$$

$$t_n, t_f = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A} \tag{6}$$

3.3.2. Alpha value

We define the principal plane as the plane formed by the first two main axes of the kernel. The linear signed distance function (SDF) is then determined by this principal plane:

$$f(\mathbf{r}(t)) = |\cos(\theta)|(t^* - t) \tag{7}$$

where t^* is the intersection point between ray and the principal plane. Then we derivate the accumulate transmittance T of the kernel.

$$T(t') = \exp(-\int_{t_n}^{t'} \sigma(t)\mathrm{d}t) \tag{8}$$

$$= \frac{1 + \exp(\kappa|\cos\theta|(t_n - t^*))}{1 + \exp(\kappa|\cos\theta|(t' - t^*))} \tag{9}$$

In practice, exponentiation may result in a NaN when κ is very large. To address this, we compute T in the logarithmic domain. Further details could be found in Supplementary Materials.

The weight function $w(t) = \sigma(t)T(t) = -T'(t)$, allowing the alpha value α of the kernel to be computed as follows:

$$\begin{aligned}\alpha &= \int_{t_n}^{t_f} w(t)\mathrm{d}t = -\int_{t_n}^{t_f} T'(t)\mathrm{d}t \\ &= T(t_n) - T(t_f) = 1 - T(t_f)\end{aligned} \tag{10}$$

3.3.3. Depth value

the depth d follows:

$$\begin{aligned}d &= \int_{t_n}^{t_f} w(t)t\mathrm{d}t = -\int_{t_n}^{t_f} T'(t)t\mathrm{d}t \\ &= -T(t_f)t_f + t_n + \int_{t_n}^{t_f} T(t)\mathrm{d}t\end{aligned} \tag{11}$$

The integration of $T(t)$ also has a closed-form solution, given by:

$$\begin{aligned}&\int_{t_n}^{t_f} T(t)\mathrm{d}t \\ &= \begin{cases}(1 + e^{-\kappa f(\mathbf{r}(t_n))})(t_f - t_n + \frac{\ln(T(t_f))}{\kappa|\cos\theta|}), & -\kappa f(\mathbf{r}(t_n)) < 19 \\ (1 - T(t_f))/(\kappa|\cos\theta|), & -\kappa f(\mathbf{r}(t_n)) \geq 19\end{cases}\end{aligned} \tag{12}$$

Note that the first case provides the exact solution for the integral, but may produce NaN values when $e^{-\kappa f(\mathbf{r}(t_n))}$ becomes very large. To address this, we approximate the value using a Taylor expansion, resulting in the second case. The detailed derivation can be found in the Supplementary Materials.

3.3.4. Alpha Blending

The color, normal, and the depth of a ray is rendered via alpha blending according to the primitive's depth order $1, \ldots, N$:

$$C = \sum_{k=1}^{N} c_k o_k \alpha_k \prod_{j=1}^{k-1}(1 - o_j\alpha_j), \tag{13}$$

$$D = \sum_{k=1}^{N} d_k o_k \prod_{j=1}^{k-1}(1 - o_j\alpha_j), \tag{14}$$

$$N = \sum_{k=1}^{N} n_k o_k \alpha_k \prod_{j=1}^{k-1}(1 - o_j\alpha_j), \tag{15}$$

where c_k is the view-dependent color, modeled by sh. Note that in depth rendering, we do not introduce the alpha value, as it is already encapsulated in depth value. We make an appoximation by sorting the kernels based on the depth of their locations rather than the intersection point for efficiency in rendering.

3.3.5. Discussion

As aforementioned, all attributes are computed in closed form. Compared to numerical integration methods, our approach achieves high rendering performance that is competitive with 3DGS. Moreover, unlike 3DGS, our method allows for flexible optimization of surface solidness. As illustrated in the Fig.3, when κ is small, the surface boundary is soft, enabling smooth optimization from the start. As κ increases, it provides unbiased, multi-view consistent depth.

3.4. Geomtric Regularization

3.4.1. Single View regularization

Following 2DGS [6], we employ a depth-normal consistency loss to enforce the consistency between normal computed by depth map and the rendered normal map:

$$\mathcal{L}_{\mathrm{nc}} = \sum_i (1 - N_i \cdot \hat{N}_i) \tag{16}$$

where $\hat{N}$ is the normal computed from depth map and i is the index of pixels. This loss ensures the local smoothness of depth map.

To capture finer geometric details, we introduce an edge-aware normal loss, which uses the image gradient to guide the normals of neighboring pixels. Specifically, for a pixel p located on image edge, we sample the neighbor four pixels i.e., up, left, down, and right, and obtain the corresponding rendered normal $\{N_j | j = 1, \ldots 4\}$. We then compute the inner product of the neighboring normals and minimize this value to emphasize the edges in the normal domain:

$$\begin{aligned}l_p &= (N_1^T N_3 + N_2^T N_4)/2, \\ L_{\mathrm{ne}} &= \sum_{p \in W} l_p, \text{ where } W = \{p | p \in I, \nabla I(p) > \lambda_e\}\end{aligned} \tag{17}$$

where ∇I is the image gradient normalized to the range of 0 to 1, and $\nabla I_p > \lambda_e$ indicates p is on the image edge. This loss encourages the angle of neighbor pixel surface on either side of the edge, promoting sharper edges in the final reconstruction.

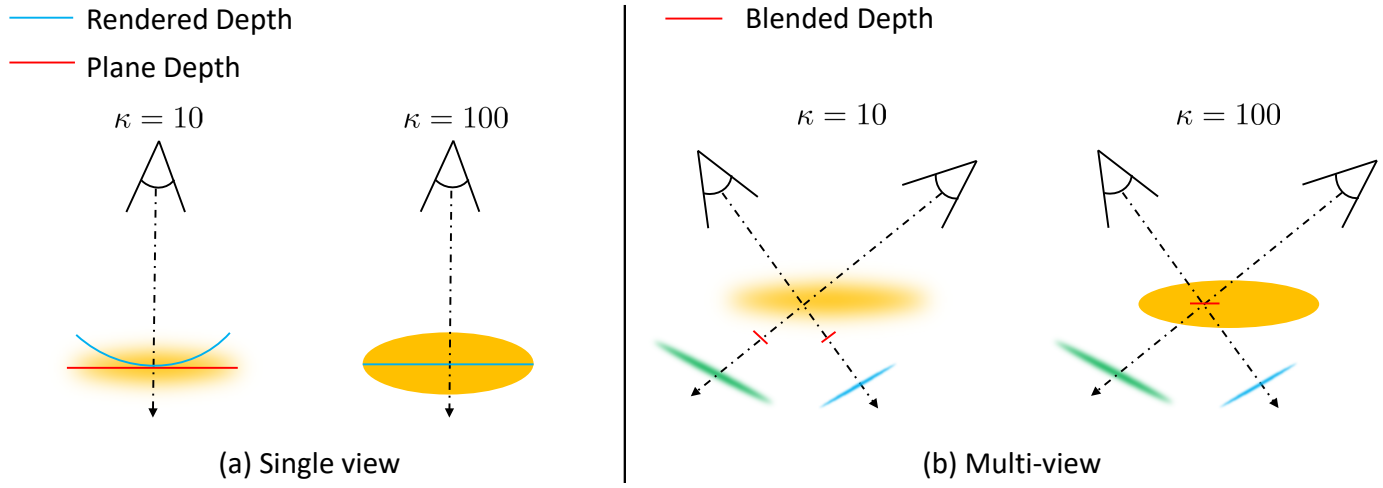

Fig. 3. illustration of different κ (a) For a single view, small kappa results in Gaussian-like soft boundaries, while a large κ yields solid boundaries with unbiased depth. (b) For multiple views, when κ is small, the blended depth varies across views. As κ increases, our method provide a multi-view consistent depth.

3.4.2. Multi-View Regularization

To constrain the multi-view consistency, we introduce both geometric and photometric constraints similar to PGSR [7]. For geometric constraint, we minimize the reprojection error. For a given pixel p_i in image I_i, we first transform it into the reference frame I_j as follows:

$$p_j = \pi_{ij}(p_i) = \pi(T_{ij}\pi^{-1}(p_i, D_i)) \tag{18}$$

where D_i is the rendered depth of pixel p_i, T_{ij} is the transformation matrix from I_i to I_j, π is the projection function which transform 3D space points to image space pixels and π^{-1} is the inverse back-projection function.

Then we minimize the error between p_i and the reprojection point of p_j:

$$\mathcal{L}_{\text{gc}} = \sum_{p \in I_i} w(p)(p - \pi_{ji}(\pi_{ij}(p))), \tag{19}$$

$$w(p) = \begin{cases} 1/\exp(p - \pi_{ji}(\pi_{ij}(p))), & p - \pi_{ji}(\pi_{ij}(p)) < 1 \\ 0, & p - \pi_{ji}(\pi_{ij}(p)) \geq 1 \end{cases} \tag{20}$$

following PGSR, we filter out the big errors, which may arise due to occlusions, and weight the projection errors using $w(p)$.

For multi-view photometric loss, we follow the PGSR [7], utilizing plane patch consistency constraints. Specially, we map a 7×7 patch P_r centered at p_r to neighbor frame patch P_n using the homography matrix $H_{rn} = K_n(R_{rn} - T_{rn}N_r^T/d_r)K_r^{-1}$, where R_{rn}, T_{rn} are the relative rotation and translation from the reference frame to neigboring frame and $d_r = D_r N_r^T \mathbf{v}_r$ is the distance to the patch plane. We then constrain the normalized cross correlation(NCC) of the patch P_r and patch P_n:

$$\mathcal{L}_{\text{pc}} = \sum_{p \in I_r} w(p)(1 - NCC(I_r(P_r), I_n(H_{rn}P_r))) \tag{21}$$

where $w(p)$ is the weight function defined in Eq.20

3.5. Training Strategy

3.5.1. Densification

Similar to 3DGS [5], we split, clone and prune the kernels in training procedure. Note that the reason we retain the opacity attribute, rather than solely relying on the accumulated alpha obtained through Eq.10, is to facilitate the removal of Gaussian points with low contribution. In contrast to 3DGS [5], we do not splat the kernel to image space, and therefore lack the position gradient in image space. Instead, we directly utilize the gradient of the 3D position to split the kernels. Furthermore, we observed that our representation struggles with growing points in texture-less areas, leading to poor fitting. To address this, we incorporate both the color gradient and the position gradient for kernel splitting.

3.5.2. Exposure Modeling

In practical scenes, different image may exhibit varying brightness for the same area due to differences in exposure time or external lighting conditions, leading to artifacts such as floating artifacts. To overcome the brightness variation across frames, we model the brightness with two exposure coefficients a and b for each image, following PGSR [7]. The images with exposure compensation could be simply computed with:

$$I_i^a = \exp(a_i)I_i^r + b_i \tag{22}$$

where I_i^r is the rendered image and I_i^a is the exposure-adjusted image. The following image loss is employed:

$$\mathcal{L}_{\text{rgb}} = (1 - \lambda)L_1(\widetilde{I} - I_i) + \lambda L_{\text{SSIM}}(I_i^r - I_i) \tag{23}$$

$$\widetilde{I} = \begin{cases} I_i^a, & L_{\text{SSIM}}(I_i^r - I_i) < 0.5 \\ I_i^r, & L_{\text{SSIM}}(I_i^r - I_i) \geq 0.5 \end{cases} \tag{24}$$

where I_i is the groud-truth image.

3.5.3. training

In summary our final training loss $\mathcal{L}$ consists of the image reconstruction loss $\mathcal{L}_{\text{rgb}}$, the depth-normal consistency loss $\mathcal{L}_{\text{nc}}$, the edge-aware normal loss $\mathcal{L}_{\text{ne}}$, the multi-view geometric loss $\mathcal{L}_{\text{gc}}$, and the multi-view photometric loss $\mathcal{L}_{\text{pc}}$:

$$\mathcal{L} = \mathcal{L}_{\text{rgb}} + \lambda_1 \mathcal{L}_{\text{nc}} + \lambda_2 \mathcal{L}_{\text{ne}} + \lambda_3 \mathcal{L}_{\text{gc}} + \lambda_4 \mathcal{L}_{\text{pc}} \tag{25}$$

where the weights are set to $\lambda_1 = 0.015$, $\lambda_2 = 0.02$, $\lambda_3 = 0.03$, and $\lambda_4 = 0.5$. Notice that in different scenes, the optimal weights are totally different, especially weights of depth-normal consistency loss and edge-aware normal loss, as they are conflict in some ways. Thus we empirically select these values.

4. Experiments

4.1. Implementation

We implement our SDKLF using custom CUDA kernels, leveraging the tile-based kernel sorting processing proposed in 3DGS [5]. The detailed computation of the rendering process can be found in the Supplementary materials. For mesh extraction, we use the Truncated Signed Distance Function (TSDF) Fusion algorithm [46] to fuse all the rendered depth maps of the training frames. We only enable exposure modeling for the Tanks and Temples dataset [47]. All experiments are conducted on a single NVIDIA RTX 4090 GPU.

4.2. Evaluation Protocol

4.2.1. Dataset

We evaluate SDKLF on various real-world datasets, including DTU [51], Tanks and Temples (TnT) [47], and Mip-NeRF360 [50]. For the first two datasets, we compare the geometry reconstruction results, and for the last dataset, we compare the novel view synthesis results. For the DTU dataset, we select 15 object-centric scenes and downsample the image resolution to 800×600 for training. For the TnT dataset, we also downsample the resolution to half for efficient training.

4.2.2. Baseline

We compare our method with both implicit and explicit geometry reconstruction methods. For implicit methods, we select VolSDF [3], NeuS [2], Geo-NeuS [45], and Neuralangelo [4]. For explicit methods, we choose state-of-the-art methods including 2DGS [6], GOF [8], GSDF [9] and PGSR [7].

4.2.3. Metrics

For the DTU dataset, we compare the Chamfer Distance (CD) and training time to the baselines, and the F1-score for the TnT dataset following the official evaluation protocol. To evaluate the rendering results, we use the peak signal-to-noise ratio (PSNR), the structural similarity index measure (SSIM), and the learned perceptual image patch similarity (LPIPS) on the Mip-NeRF360 dataset.

4.3. Comparison

4.3.1. Geometry Reconstruction

As shown in Tab. 2, Fig. 4, Fig. 6,and Tab. 4, our method achieves the highest results in most scenes of the DTU dataset and the highest mean value with a relatively fast speed. Our method fits the geometry field (Signed Distance Function) with explicit kernels, offering approximately 100× faster speed compared to implicit methods. For 3DGS-based methods, our method significantly outperforms 2DGS [6] and GOF [8], benefiting from our unbiased and multi-view consistent depth estimation. Our method also outperforms all other methods on the TnT dataset, as shown in Tab. 1 and Fig. 5. Note that although Neuralangelo [4] achieves the highest results in three scenes, it requires far more training time due to its numerical integration of volume rendering and numerical gradient computation of the normal. Our method provides competitive results with Neuralangelo and higher results than PGSR [7], which also computes unbiased depth but fails to ensure multi-view consistency. We share a similar idea with GOF [8], which fits a geometry field with kernel functions on points. However, our method transforms the geometry field into an opacity field and computes the integration of the opacity field, achieving better reconstruction.

4.3.2. Novel View Synthesis

We evaluate the render quality on the Mip-NeRF360 dataset [50] using both pure novel-view synthesis methods and 3DGS-based reconstruction methods. These methods include NeRF [1], Deep Blending [48], INGP [49], Mip-NeRF360 [50], NeuS [2], 3DGS [5], 2DGS [6], GOF [8], and PGSR [7]. As shown in Tab. 3, our method provides competitive results compared to the current state-of-the-art methods, breaking the notion that SDF-based volume rendering methods, such as NeuS [2], typically offer inferior rendering quality.

4.4. Ablations

We conduct all the quantitative ablation studies, with the results shown in Tab. 5. First, we replace the proposed SDKLF with the original 2DGS splatting algorithm, which results in a severe drop in geometry quality. This indicates that our representation provides more robust depth support and strong regularization. As shown in Tab. 5, although the Edge-aware normal loss offers a slight improvement quantitatively, it is beneficial for preserving more details. Comparing the second row with the sixth row in Tab. 5, adding color-gradient-based densification helps improve reconstruction accuracy, especially in texture-less areas. It is obvious that multi-view regularization plays a crucial role in surface reconstruction.

4.5. Limitations

Here we discuss two approximations that we have not mentioned before. First, we adopt a non-overlapping assumption, discarding the overlapping regions of each kernel. This is a common scenario in surface reconstruction, where we encourage kernels to be near the surface.

The second approximation is the global ranking strategy, following 3DGS [5]. In theory, kernels should be ranked per ray

Fig. 4. Qualititive comparison results on DTU dataset. Our method provides more geometry details

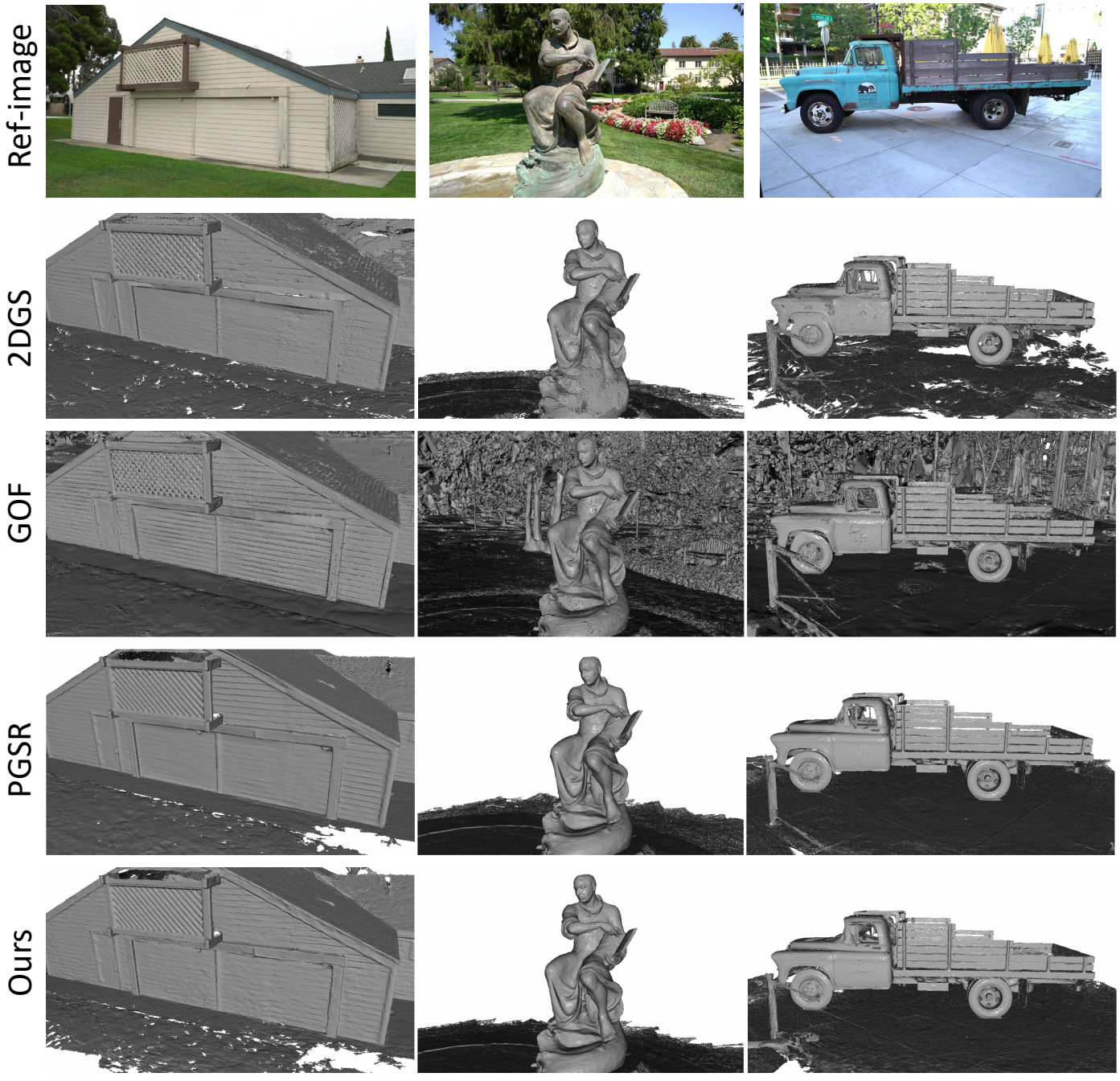

Fig. 5. Qualititive comparison results on TNT dataset.

Table 1. Quantitative results on Tanks and Temples dataset, F1-score are reported. We bold the best result and underline the second one.

	NeuS [2]	Geo-NeuS [45]	Neuralangelo [4]	2DGS [6]	GOF [8]	PGSR [7]	Ours
Barn	0.29	0.33	**0.70**	0.45	0.51	0.54	0.58
Caterpillar	0.29	0.26	0.36	0.24	0.41	**0.44**	0.42
Courthouse	0.17	0.12	**0.28**	0.13	**0.28**	0.24	0.27
Ignatius	0.83	0.72	**0.89**	0.50	0.68	0.79	0.81
Meetingroom	0.24	0.20	0.32	0.18	0.28	0.29	**0.33**
Truck	0.45	0.45	0.48	0.43	0.58	0.66	**0.68**
Mean	0.38	0.35	0.50	0.32	0.46	0.49	**0.52**

Table 2. Quantitative results on DTU dataset, Chamfer Distance are reported. We bold the best result and underline the second one.

	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	mean
NeuS [2]	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
VoLSDF [3]	1.14	1.26	0.81	0.49	1.25	0.70	0.72	1.29	1.18	0.70	0.66	1.08	0.42	0.61	0.55	0.86
Neuralangelo [4]	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
2DGS [6]	0.51	0.80	**0.33**	0.41	0.92	0.85	0.83	1.28	1.21	0.64	0.69	1.28	0.41	0.66	0.48	0.76
GOF [8]	0.50	0.82	0.37	0.37	1.12	0.74	0.73	1.18	1.29	0.68	0.77	0.90	0.42	0.66	0.49	0.74
PGSR [7]	**0.36**	0.57	0.38	**0.33**	0.78	0.58	0.50	1.08	1.08	**0.59**	**0.46**	**0.54**	**0.30**	0.38	**0.34**	0.52
GSDF [44]	1.07	0.95	0.46	0.38	1.29	0.84	0.78	1.57	1.34	0.82	0.61	1.19	0.39	0.54	0.57	0.85
Ours	0.38	**0.56**	0.35	0.35	**0.59**	0.55	**0.49**	**1.05**	**0.60**	**0.59**	**0.46**	0.62	0.32	**0.36**	0.35	**0.51**

Table 3. Quantitative results on Mip-NeRF360 dataset. Our method achieve competitive results.

	indoor scenes			outdoor scenes			Overall		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
NeRF [1]	26.84	0.790	0.370	21.46	0.458	0.515	24.15	0.624	0.443
Deep Blending [48]	26.40	0.844	0.261	21.54	0.524	0.364	23.97	0.684	0.313
INGP [49]	29.15	0.880	0.216	22.90	0.566	0.371	26.03	0.723	0.294
M-NeRF360 [50]	**31.72**	0.917	0.180	24.47	0.691	0.283	**28.10**	0.804	0.232
NeuS [2]	25.10	0.789	0.319	21.93	0.629	0.600	23.74	0.720	0.439
3DGS [5]	30.99	0.926	0.199	24.24	0.705	0.283	27.24	0.803	0.246
2DGS [6]	30.39	0.923	0.183	24.33	0.709	0.284	27.03	0.804	0.239
GOF [8]	30.80	0.928	0.167	**24.76**	0.742	0.225	27.78	**0.835**	0.196
PGSR [7]	30.36	**0.934**	**0.147**	**24.76**	**0.752**	**0.203**	27.25	0.833	**0.178**
Ours	30.12	0.915	0.191	24.36	0.721	0.252	26.92	0.808	0.225

Table 4. Training time in DTU dataset.

NeuS	VoLSDF	Neuralangelo	2DGS	GOF	PGSR	Ours
>12h	>12h	>128h	0.32h	2h	0.5h	0.5h

Table 5. Ablation study for the proposed components on the TnT dataset.

Model setting	F1-Score
w/o SDKLF	0.472
w/o color-gradient	0.473
w/o edge-aware normal loss	0.493
w/o multi-view geometric loss	0.489
w/o multi-view photometric loss	0.305
Full Model	0.518

to achieve an exact volume rendering result. However, for computational efficiency, we rank the kernels based on their centers. This approach may introduce incorrect rankings for local rays, but it significantly improves computational efficiency.

Additionally, our method does not reconstruct the background effectively, as demonstrated in the third column of Fig. 5.

5. Conclusion

In this paper, we propose a novel scene representation, SDKF, which fits SDF with linear kernels and designs a fast volume renderer for the representation. This renderer provides unbiased and multi-view consistent depth. We carefully design the training framework of SDKF, including multi-view and single-view regularization, color-position gradient-based densification, and exposure modeling. We evaluate our rendering and reconstruction quality on the Mip-NeRF360, DTU, and TnT datasets, showing that our method achieves state-of-the-art performance.

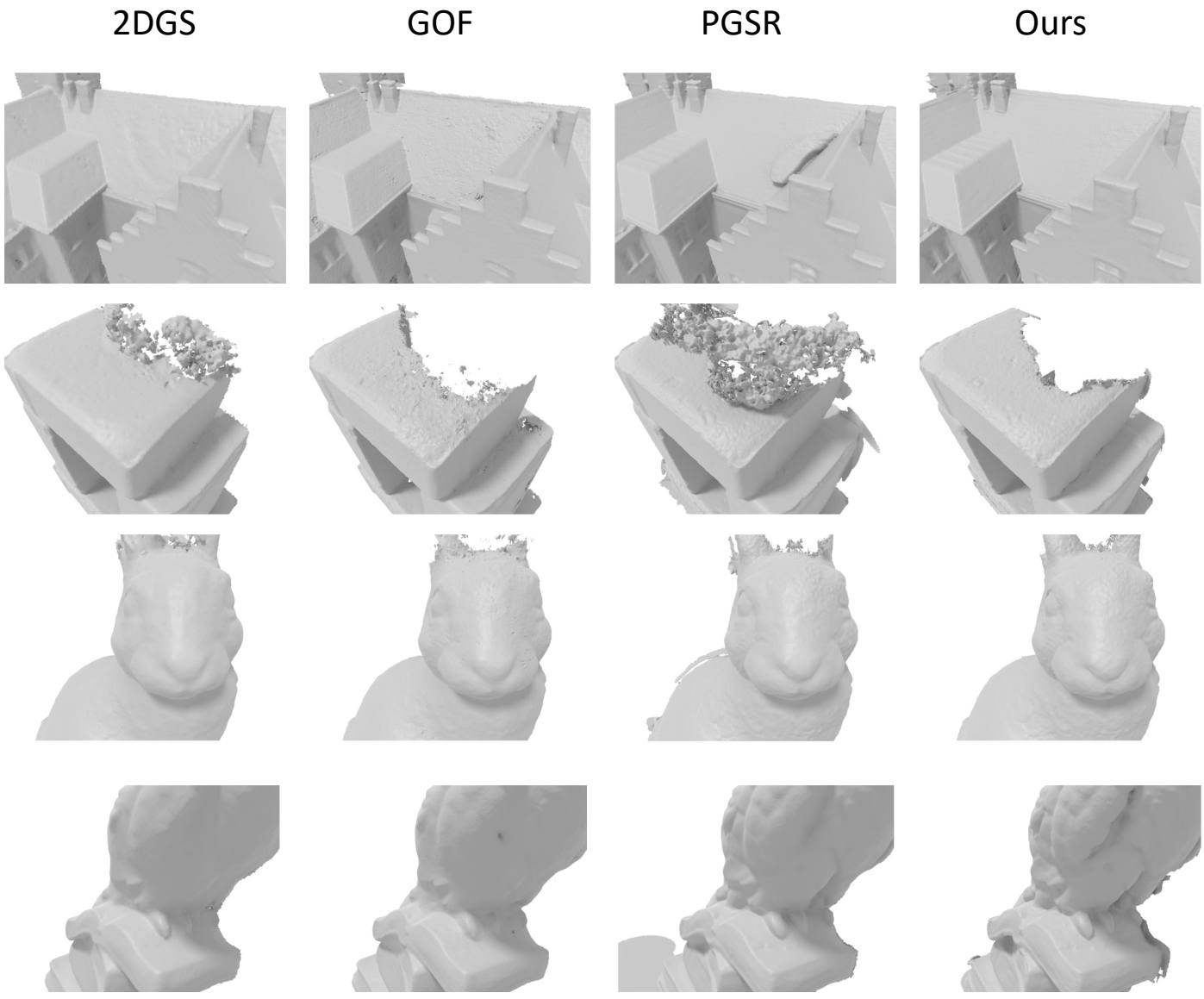

Fig. 6. Detail comparison on DTU dataset.

References

[1] Mildenhall, B, Srinivasan, PP, Tancik, M, Barron, JT, Ramamoorthi, R, Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. In: Vedaldi, A, Bischof, H, Brox, T, Frahm, J, editors. Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I; vol. 12346 of *Lecture Notes in Computer Science*. Springer; 2020, p. 405–421. URL: https://doi.org/10.1007/978-3-030-58452-8_24. doi:10.1007/978-3-030-58452-8_24.

[2] 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: Ranzato, M, Beygelzimer, A, Dauphin, YN, Liang, P, Vaughan, JW, editors. Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual. 2021, p. 27171–27183. URL: https://proceedings.neurips.cc/paper/2021/hash/e41e164f7485ec4a28741a2d0ea41c74-Abstract.html.

[3] Yariv, L, Gu, J, Kasten, Y, Lipman, Y. Volume rendering of neural implicit surfaces. In: Ranzato, M, Beygelzimer, A, Dauphin, YN, Liang, P, Vaughan, JW, editors. Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual. 2021, p. 4805–4815. URL: https://proceedings.neurips.cc/paper/2021/hash/25e2a30f44898b9f3e978b1786dcd85c-Abstract.html.

[4] Li, Z, Müller, T, Evans, A, Taylor, RH, Unberath, M, Liu, M, et al. Neuralangelo: High-fidelity neural surface reconstruction. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023. IEEE; 2023, p. 8456–8465. URL: https://doi.org/10.1109/CVPR52729.2023.00817. doi:10.1109/CVPR52729.2023.00817.

[5] Kerbl, B, Kopanas, G, Leimkühler, T, Drettakis, G. 3d gaussian splatting for real-time radiance field rendering. ACM Trans Graph 2023;42(4):139:1–139:14. URL: https://doi.org/10.1145/3592433. doi:10.1145/3592433.

[6] Huang, B, Yu, Z, Chen, A, Geiger, A, Gao, S. 2d gaussian splatting for geometrically accurate radiance fields. In: Burbano, A, Zorin, D, Jarosz, W, editors. ACM SIGGRAPH 2024 Conference Papers, SIGGRAPH 2024, Denver, CO, USA, 27 July 2024- 1 August 2024. ACM; 2024, p. 32. URL: https://doi.org/10.1145/3641519.3657428. doi:10.1145/3641519.3657428.

[7] Chen, D, Li, H, Ye, W, Wang, Y, Xie, W, Zhai, S, et al. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. arXiv preprint arXiv:240606521 2024;.

[8] Yu, Z, Sattler, T, Geiger, A. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. ACM Transactions on Graphics (TOG) 2024;43(6):1–13.

[9] Yu, M, Lu, T, Xu, L, Jiang, L, Xiangli, Y, Dai, B. Gsdf: 3dgs meets sdf for improved neural rendering and reconstruction. Advances in Neural Information Processing Systems 2024;37:129507–129530.

[10] Shen, S. Accurate multiple view 3d reconstruction using patch-based stereo for large-scale scenes. IEEE Trans Image Process 2013;22(5):1901–1914. URL: https://doi.org/10.1109/TIP.2013.2237921. doi:10.1109/TIP.2013.2237921.

[11] Schönberger, JL, Zheng, E, Frahm, J, Pollefeys, M. Pixelwise view selection for unstructured multi-view stereo. In: Leibe, B, Matas, J, Sebe, N, Welling, M, editors. Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III; vol. 9907 of *Lecture Notes in Computer Science*. Springer; 2016, p. 501–518. URL: https://doi.org/10.1007/978-3-319-46487-9_31. doi:10.1007/978-3-319-46487-9_31.

[12] Tewari, A, Fried, O, Thies, J, Sitzmann, V, Lombardi, S, Sunkavalli, K, et al. State of the art on neural rendering. Comput Graph Forum 2020;39(2):701–727. URL: https://doi.org/10.1111/cgf.14022. doi:10.1111/CGF.14022.

[13] Tewari, A, Thies, J, Mildenhall, B, Srinivasan, PP, Tretschk, E, Wang, Y, et al. Advances in neural rendering. Comput Graph Forum 2022;41(2):703–735. URL: https://doi.org/10.1111/cgf.14507. doi:10.1111/CGF.14507.

[14] Bao, Y, Ding, T, Huo, J, Liu, Y, Li, Y, Li, W, et al. 3d gaussian splatting: Survey, technologies, challenges, and opportunities. CoRR 2024;abs/2407.17418. URL: https://doi.org/10.48550/arXiv.2407.17418. doi:10.48550/ARXIV.2407.17418. arXiv:2407.17418.

[15] Hirschmüller, H. Accurate and efficient stereo processing by semi-global matching and mutual information. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA. IEEE Computer Society; 2005, p. 807–814. URL: https://doi.org/10.1109/CVPR.2005.56. doi:10.1109/CVPR.2005.56.

[16] Goesele, M, Snavely, N, Curless, B, Hoppe, H, Seitz, SM. Multi-view stereo for community photo collections. In: IEEE 11th International Conference on Computer Vision, ICCV 2007, Rio de Janeiro, Brazil, October 14-20, 2007. IEEE Computer Society; 2007, p. 1–8. URL: https://doi.org/10.1109/ICCV.2007.4408933. doi:10.1109/ICCV.2007.4408933.

[17] Hirschmüller, H. Stereo processing by semiglobal matching and mutual information. IEEE Trans Pattern Anal Mach Intell 2008;30(2):328–341. URL: https://doi.org/10.1109/TPAMI.2007.1166. doi:10.1109/TPAMI.2007.1166.

[18] Barnes, C, Shechtman, E, Finkelstein, A, Goldman, DB. Patchmatch: a randomized correspondence algorithm for structural image editing. ACM Trans Graph 2009;28(3):24. URL: https://doi.org/10.1145/1531326.1531330. doi:10.1145/1531326.1531330.

[19] Furukawa, Y, Ponce, J. Accurate, dense, and robust multiview stereopsis. IEEE Trans Pattern Anal Mach Intell 2010;32(8):1362–1376. URL: https://doi.org/10.1109/TPAMI.2009.161. doi:10.1109/TPAMI.2009.161.

[20] Yao, Y, Luo, Z, Li, S, Fang, T, Quan, L. Mvsnet: Depth inference for unstructured multi-view stereo. In: Ferrari, V, Hebert, M, Sminchisescu, C, Weiss, Y, editors. Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII; vol. 11212 of *Lecture Notes in Computer Science*. Springer; 2018, p. 785–801. URL: https://doi.org/10.1007/978-3-030-01237-3_47. doi:10.1007/978-3-030-01237-3_47.

[21] Luo, K, Guan, T, Ju, L, Huang, H, Luo, Y. P-mvsnet: Learning patch-wise matching confidence aggregation for multi-view stereo. In: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019. IEEE; 2019, p. 10451–10460. URL: https://doi.org/10.1109/ICCV.2019.01055. doi:10.1109/ICCV.2019.01055.

[22] 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, Long Beach, CA, USA, June 16-20, 2019. Computer Vision Foundation / IEEE; 2019, p. 5525–5534. URL: http://openaccess.thecvf.com/content_CVPR_2019/html/Yao_Recurrent_MVSNet_for_High-Resolution_Multi-View_Stereo_Depth_Inference_CVPR_2019_paper.html. doi:10.1109/CVPR.2019.00567.

[23] Yu, Z, Gao, S. Fast-mvsnet: Sparse-to-dense multi-view stereo with learned propagation and gauss-newton refinement. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020. Computer Vision Foundation / IEEE; 2020, p. 1946–1955. URL: https://openaccess.thecvf.com/content_CVPR_2020/html/Yu_Fast-MVSNet_Sparse-to-Dense_Multi-View_Stereo_With_Learned_Propagation_and_Gauss-Newton_Refinement_CVPR_2020_paper.html. doi:10.1109/CVPR42600.2020.00202.

[24] Wang, F, Galliani, S, Vogel, C, Speciale, P, Pollefeys, M. Patchmatchnet: Learned multi-view patchmatch stereo. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021. Computer Vision Foundation / IEEE; 2021, p. 14194–14203. URL: https://openaccess.thecvf.com/content/CVPR2021/html/Wang_PatchmatchNet_Learned_Multi-View_Patchmatch_Stereo_CVPR_2021_paper.html. doi:10.1109/CVPR46437.2021.01397.

[25] Chen, Z, Funkhouser, TA, Hedman, P, Tagliasacchi, A. Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023. IEEE; 2023, p. 16569–16578. URL: https://doi.org/10.1109/CVPR52729.2023.01590. doi:10.1109/CVPR52729.2023.01590.

[26] Rakotosaona, M, Manhardt, F, Arroyo, DM, Niemeyer, M, Kundu, A, Tombari, F. Nerfmeshing: Distilling neural radiance fields into geometrically-accurate 3d meshes. In: International Conference on 3D Vision, 3DV 2024, Davos, Switzerland, March 18-21, 2024. IEEE; 2024, p. 1156–1165. URL: https://doi.org/10.1109/3DV62453.2024.00093. doi:10.1109/3DV62453.2024.00093.

[27] Reiser, C, Garbin, SJ, Srinivasan, PP, Verbin, D, Szeliski, R, Mildenhall, B, et al. Binary opacity grids: Capturing fine geometric detail for mesh-based view synthesis. ACM Trans Graph 2024;43(4):149:1–149:14. URL: https://doi.org/10.1145/3658130. doi:10.1145/3658130.

[28] Mescheder, LM, Oechsle, M, Niemeyer, M, Nowozin, S, Geiger, A. Occupancy networks: Learning 3d reconstruction in function space. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019. Computer Vision Foundation / IEEE; 2019, p. 4460–4470. URL: http://openaccess.thecvf.com/content_CVPR_2019/html/Mescheder_Occupancy_Networks_Learning_3D_Reconstruction_in_Function_Space_CVPR_2019_paper.html. doi:10.1109/CVPR.2019.00459.

[29] Park, JJ, Florence, PR, Straub, J, Newcombe, RA, Lovegrove, S. Deepsdf: Learning continuous signed distance functions for shape representation. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019. Computer Vision Foundation / IEEE; 2019, p. 165–174. URL: http://openaccess.thecvf.com/content_CVPR_2019/html/Park_DeepSDF_Learning_Continuous_Signed_Distance_Functions_for_Shape_Representation_CVPR_2019_paper.html. doi:10.1109/CVPR.2019.00025.

[30] Xu, Q, Xu, Z, Philip, J, Bi, S, Shu, Z, Sunkavalli, K, et al. Point-nerf: Point-based neural radiance fields. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022. IEEE; 2022, p. 5428–5438. URL: https://doi.org/10.1109/CVPR52688.2022.00536. doi:10.1109/CVPR52688.2022.00536.

[31] Liu, L, Gu, J, Lin, KZ, Chua, T, Theobalt, C. Neural sparse voxel fields. In: Larochelle, H, Ranzato, M, Hadsell, R, Balcan, M, Lin, H, editors. Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual. 2020,URL: https://proceedings.neurips.cc/paper/2020/hash/b4b758962f17808746e9bb832a6fa4b8-Abstract.html.

[32] Cai, B, Huang, J, Jia, R, Lv, C, Fu, H. Neuda: Neural deformable anchor for high-fidelity implicit surface reconstruction. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023. IEEE; 2023, p. 8476–8485. URL: https://doi.org/10.1109/CVPR52729.2023.00819. doi:10.1109/CVPR52729.2023.00819.

[33] Li, H, Yang, X, Zhai, H, Liu, Y, Bao, H, Zhang, G. Vox-surf: Voxel-based implicit surface representation. IEEE Trans Vis Comput Graph 2024;30(3):1743–1755. URL: https://doi.org/10.1109/TVCG.2022.3225844. doi:10.1109/TVCG.2022.3225844.

[34] Wang, Y, Skorokhodov, I, Wonka, P. Pet-neus: Positional encoding triplanes for neural surfaces. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023. IEEE; 2023, p. 12598–12607. URL: https://doi.org/10.1109/CVPR52729.2023.01212. doi:10.1109/CVPR52729.2023.01212.

[35] Wang, Y, Han, Q, Habermann, M, Daniilidis, K, Theobalt, C, Liu, L. Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction. In: IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023. IEEE; 2023, p. 3272–3283. URL: https://doi.org/10.1109/ICCV51070.2023.00305. doi:10.1109/ICCV51070.2023.00305.

[36] Huang, Y, Sun, Y, Yang, Z, Lyu, X, Cao, Y, Qi, X. SC-GS: sparse-controlled gaussian splatting for editable dynamic scenes. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024. IEEE; 2024, p. 4220–4230. URL: https://doi.org/10.1109/CVPR52733.2024.00404. doi:10.1109/CVPR52733.2024.00404.

[37] Li, Z, Chen, Z, Li, Z, Xu, Y. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024. IEEE; 2024, p. 8508–8520. URL: https://doi.org/10.1109/CVPR52733.2024.00813. doi:10.1109/CVPR52733.2024.00813.

[38] Lu, Z, Guo, X, Hui, L, Chen, T, Yang, M, Tang, X, et al. 3d geometry-aware deformable gaussian splatting for dynamic view synthesis. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024. IEEE; 2024, p. 8900–8910. URL: https://doi.org/10.1109/CVPR52733.2024.00850. doi:10.1109/CVPR52733.2024.00850.

[39] Wu, G, Yi, T, Fang, J, Xie, L, Zhang, X, Wei, W, et al. 4d gaussian splatting for real-time dynamic scene rendering. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024. IEEE; 2024, p. 20310–20320. URL: https://doi.org/10.1109/CVPR52733.2024.01920. doi:10.1109/CVPR52733.2024.01920.

[40] Zhang, K, Bi, S, Tan, H, Xiangli, Y, Zhao, N, Sunkavalli, K, et al. GS-LRM: large reconstruction model for 3d gaussian splatting. In: Leonardis, A, Ricci, E, Roth, S, Russakovsky, O, Sattler, T, Varol, G, editors. Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part XXII; vol. 15080 of *Lecture Notes in Computer Science*. Springer; 2024, p. 1–19. URL: https://doi.org/10.1007/978-3-031-72670-5_1. doi:10.1007/978-3-031-72670-5_1.

[41] Gao, L, Yang, J, Zhang, B, Sun, J, Yuan, Y, Fu, H, et al. Real-time large-scale deformation of gaussian splatting. ACM Trans Graph 2024;43(6):200:1–200:17. URL: https://doi.org/10.1145/3687756. doi:10.1145/3687756.

[42] Guédon, A, Lepetit, V. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024. IEEE; 2024, p. 5354–5363. URL: https://doi.org/10.1109/CVPR52733.2024.00512. doi:10.1109/CVPR52733.2024.00512.

[43] Dai, P, Xu, J, Xie, W, Liu, X, Wang, H, Xu, W. High-quality surface reconstruction using gaussian surfels. In: Burbano, A, Zorin, D, Jarosz, W, editors. ACM SIGGRAPH 2024 Conference Papers, SIGGRAPH 2024, Denver, CO, USA, 27 July 2024- 1 August 2024. ACM; 2024, p. 22. URL: https://doi.org/10.1145/3641519.3657441. doi:10.1145/3641519.3657441.

[44] Lyu, X, Sun, YT, Huang, YH, Wu, X, Yang, Z, Chen, Y, et al. 3dgsr: Implicit surface reconstruction with 3d gaussian splatting. ACM Transactions on Graphics (TOG) 2024;43(6):1–12.

[45] Fu, Q, Xu, Q, Ong, YS, Tao, W. Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. In: Koyejo, S, Mohamed, S, Agarwal, A, Belgrave, D, Cho, K, Oh, A, editors. Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022. 2022,URL: http://papers.nips.cc/paper_files/paper/2022/hash/16415eed5a0a121bfce79924db05d3fe-Abstract-Conference.html.

[46] Izadi, S, Kim, D, Hilliges, O, Molyneaux, D, Newcombe, RA, Kohli, P, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In: Pierce, JS, Agrawala, M, Klemmer, SR, editors. Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, Santa Barbara, CA, USA, October 16-19, 2011. ACM; 2011, p. 559–568. URL: https://doi.org/10.1145/2047196.2047270. doi:10.1145/2047196.2047270.

[47] Knapitsch, A, Park, J, Zhou, Q, Koltun, V. Tanks and temples: benchmarking large-scale scene reconstruction. ACM Trans Graph 2017;36(4):78:1–78:13. URL: https://doi.org/10.1145/3072959.3073599. doi:10.1145/3072959.3073599.

[48] Hedman, P, Philip, J, Price, T, Frahm, J, Drettakis, G, Brostow, GJ. Deep blending for free-viewpoint image-based rendering. ACM Trans Graph 2018;37(6):257. URL: https://doi.org/10.1145/3272127.3275084. doi:10.1145/3272127.3275084.

[49] Müller, T, Evans, A, Schied, C, Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans Graph 2022;41(4):102:1–102:15. URL: https://doi.org/10.1145/3528223.3530127. doi:10.1145/3528223.3530127.

[50] Barron, JT, Mildenhall, B, Verbin, D, Srinivasan, PP, Hedman, P. Mip-

nerf 360: Unbounded anti-aliased neural radiance fields. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022. IEEE; 2022, p. 5460–5469. URL: https://doi.org/10.1109/CVPR52688.2022.00539. doi:10.1109/CVPR52688.2022.00539.

[51] Aanæs, H, Jensen, RR, Vogiatzis, G, Tola, E, Dahl, AB. Large-scale data for multiple-view stereopsis. Int J Comput Vis 2016;120(2):153–168. URL: https://doi.org/10.1007/s11263-016-0902-9. doi:10.1007/S11263-016-0902-9.