VA-GS: Enhancing the Geometric Representation of Gaussian Splatting via View Alignment

Qing Li[1] **Huifang Feng**[2*] **Xun Gong**[1] **Yu-Shen Liu**[3]
[1] Southwest Jiaotong University, Chengdu, China
[2] Xihua University, Chengdu, China [3] Tsinghua University, Beijing, China
qingli@swjtu.edu.cn fhf@xhu.edu.cn xgong@swjtu.edu.cn liuyushen@tsinghua.edu.cn

Abstract

3D Gaussian Splatting has recently emerged as an efficient solution for high-quality and real-time novel view synthesis. However, its capability for accurate surface reconstruction remains underexplored. Due to the discrete and unstructured nature of Gaussians, supervision based solely on image rendering loss often leads to inaccurate geometry and inconsistent multi-view alignment. In this work, we propose a novel method that enhances the geometric representation of 3D Gaussians through view alignment (VA). Specifically, we incorporate edge-aware image cues into the rendering loss to improve surface boundary delineation. To enforce geometric consistency across views, we introduce a visibility-aware photometric alignment loss that models occlusions and encourages accurate spatial relationships among Gaussians. To further mitigate ambiguities caused by lighting variations, we incorporate normal-based constraints to refine the spatial orientation of Gaussians and improve local surface estimation. Additionally, we leverage deep image feature embeddings to enforce cross-view consistency, enhancing the robustness of the learned geometry under varying viewpoints and illumination. Extensive experiments on standard benchmarks demonstrate that our method achieves state-of-the-art performance in both surface reconstruction and novel view synthesis. The source code is available at https://github.com/LeoQLi/VA-GS.

1 Introduction

Accurate surface reconstruction from multi-view images is a long-standing problem in computer vision, fundamental to applications such as 3D modeling, AR/VR, and robotics. Recently, 3D Gaussian Splatting (3DGS) has emerged as a powerful explicit representation for real-time novel view synthesis, demonstrating impressive rendering quality and speed by modeling scenes as collections of semi-transparent 3D Gaussian primitives. However, despite its rendering advantages, 3DGS remains limited in its ability to recover accurate and detailed geometry, especially when supervision is derived solely from RGB images. This limitation stems from the inherent discrete and unstructured nature of Gaussians, which makes it difficult to enforce global surface consistency or capture fine geometric details, particularly under complex illumination and along object boundaries.

Existing methods have attempted to enhance the geometric capabilities of Gaussian splatting. For example, SuGaR [14] constructs a density field from Gaussians and extracts meshes via level-set searching, but it struggles with large smooth surfaces and is computationally expensive. 2DGS [16] models scenes using 2D oriented planar Gaussian disks, which inherently represent surfaces and provide view-consistent geometry. However, 2DGS has difficulty reconstructing background geometry and often produces incomplete or distorted surfaces in complex or unbounded scenes. GOF [56] constructs an opacity field from Gaussians and extracts surfaces using Marching Tetrahedra [10],

[*]Corresponding author

yielding adaptive mesh resolution without volumetric fusion. Nonetheless, thin structures can be lost and strong lighting contrasts still cause artifacts. GS-Pull [60] integrates a neural signed distance field (SDF), dynamically pulling Gaussians toward the zero-level set of the learned SDF. While this improves surface completeness, it introduces additional network complexity, produces overly smooth surfaces, and primarily focuses on foreground object reconstruction. PGSR [4] fits Gaussians to local planar hypotheses and uses unbiased depth rendering to improve geometric accuracy. However, it does not fully resolve the challenges posed by complex lighting and remains sensitive to boundary ambiguities in non-planar regions. Overall, previous methods have introduced geometric regularizers or hybrid representations and achieved significant progress. However, they still struggle to address two persistent challenges: illumination-induced artifacts (*e.g.*, shadows and specular highlights) and accurate surface boundary delineation, as shown in Fig. 1. Illumination effects distort photometric losses, while ambiguous boundaries often result in geometry drift or holes.

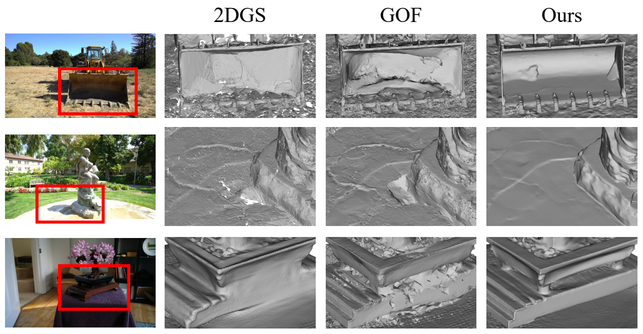

Figure 1: Our method addresses illumination and boundary artifacts that previous methods fail to resolve.

In this work, we propose a novel method for accurate and detailed surface reconstruction by enhancing the geometric representation of 3D Gaussians. We address the limitations of previous methods by incorporating multi-faceted geometric constraints and structural priors. Our approach introduces geometry-aware constraints guided by image edges, multi-view alignment that considers visibility and occlusion, and robust priors derived from surface normals and deep image features to mitigate the effects of lighting variations and boundary ambiguities. Specifically, we enhance the standard rendering loss with edge-aware image cues, which sharpen surface boundaries in the 2D projection space of Gaussian splats, resulting in clearer and more precise delineations in rendered images. To enforce geometric consistency across views, we introduce a multi-view photometric alignment loss that explicitly accounts for visibility and occlusions, encouraging accurate spatial relationships among 3D Gaussians and improving boundary localization. To further reduce ambiguity caused by lighting, we introduce normal-based alignment to constrain the spatial orientation of Gaussians, ensuring reliable surface estimation. Additionally, we leverage high-dimensional image features to enforce cross-view consistency, improving robustness to viewpoint and lighting variations. These innovations significantly reduce the impact of complex illumination and boundary ambiguity, enabling accurate surface reconstruction in challenging scenes. Experiments on standard benchmarks demonstrate that our method achieves state-of-the-art performance in both surface reconstruction and novel view synthesis. Our contributions are summarized as follows.

- Incorporating edge information and visibility-aware multi-view alignment to enhance surface boundary delineation and improve geometric consistency.
- Aligning the robust priors based on normals and deep image features to mitigate illumination-induced artifacts and increase reconstruction accuracy.
- State-of-the-art results on standard benchmarks, demonstrating the effectiveness of our method in both surface reconstruction and novel view synthesis.

2 Related Work

View Synthesis and Gaussian Splatting. Neural Radiance Fields (NeRF) [31] pioneered high-fidelity novel view synthesis by representing a scene as a continuous volumetric density and view-dependent color field, optimized via differentiable volume rendering. Subsequent works accelerated training and rendering through hybrid representations such as multi-resolution hash grids [32], explicit voxel or sparse tensor grids [40, 2], and learned feature planes [53]. However, these volumetric methods still entail high memory and computational costs. 3DGS [21] departs from dense volumes by modeling a scene as a sparse cloud of anisotropic 3D Gaussians. Follow-up work has enhanced visual fidelity through anti-aliasing and level-of-detail control [55, 38], improved training speed and robustness under sparse views using density regularization and learned radiance priors [46, 34], and extended 3DGS to dynamic scenes [29, 47], relighting [13], and animation [51]. Geometry-aware variants such as FatesGS [17], DNGaussian [23] and GeoGaussian [26] address sparse-view and textureless regions, while methods like Instantsplat [11] and Scaffold-GS [28] accelerate convergence by leveraging pretraining or hybrid implicit-explicit designs. Despite these advances, most 3DGS variants primarily emphasize appearance quality and lack mechanisms to enforce explicit surface geometry, motivating dedicated reconstruction techniques.

Surface Reconstruction with Gaussians. Extracting accurate surfaces from a 3DGS representation is challenging due to its unstructured nature and supervision based solely on RGB signals. Early approaches convert Gaussians into volumetric density or opacity fields: SuGaR [14] builds a density field and applies level-set search with Poisson reconstruction [20], but it struggles to recover large, smooth surfaces; GOF [56] accumulates per-view alpha values into an opacity volume and extracts iso-surfaces with Marching Tetrahedra [10], achieving adaptive resolution but often missing fine, thin structures under high lighting contrast. Other methods project Gaussians into oriented 2D disks (surfels) and fuse via Truncated Signed Distance Function (TSDF) fusion [33] or Poisson reconstruction. 2DGS [16] and GSurfels [8] improve local alignment but tend to introduce distortions in unbounded scenes and result in incomplete background geometry. PGSR [4] fits Gaussians to planar patches and adds multi-view photometric and geometric regularization, excelling on planar man-made scenes but remaining sensitive to non-planar boundaries. More recent works [54, 60, 30, 1, 24, 25] integrate Signed Distance Fields (SDF) to guide Gaussian placement. GSDF [54] and 3DGSR [30] jointly optimize a neural SDF branch alongside Gaussian parameters using volume-rendered depth and normal supervision, which improves surface smoothness but requires additional network branches. GS-Pull [60] leverages SDF gradients to pull Gaussians toward the zero-level set, enhancing alignment at the cost of limiting object-level reconstruction and producing overly smooth results. Methods that incorporate depth or normal estimators [5, 57, 41, 45, 43] impose priors on Gaussians but rely on TSDF fusion's fixed resolution or Poisson reconstruction's sensitivity to noisy inputs, and often struggle under varying illumination or around complex geometric boundaries. Our approach enforces view consistency through multi-faceted constraints during Gaussian optimization, enabling high-fidelity mesh extraction even under challenging lighting and boundary conditions.

3 Preliminaries

3DGS [21] explicitly represents a scene as a collection of anisotropic 3D Gaussians, which can be rendered to images from arbitrary viewpoints using a splatting-based rasterization technique [61]. Specifically, each 3D Gaussian $\mathbb{G}$ is defined as:

$$\mathbb{G}(x) = \exp\left(-0.5(x-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(x-\boldsymbol{\mu})\right), \tag{1}$$

where $\boldsymbol{\mu}$ is the Gaussian center and $\boldsymbol{\Sigma}$ is its covariance matrix. For novel-view rendering, the color at pixel $\boldsymbol{p}$ is obtained by compositing K ordered Gaussian splats using point-based α-blending, *i.e.*,

$$\boldsymbol{C}(\boldsymbol{p}) = \sum_{i=1}^{K} \boldsymbol{c}_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{2}$$

where α_i denotes the pixel translucency determined by the learned opacity of the i-th Gaussian kernel and its projected footprint at pixel $\boldsymbol{p}$. The view-dependent color $\boldsymbol{c}_i$ is encoded using spherical harmonics associated with each Gaussian. In addition to color, Eq. (2) is similarly used to render per-pixel normals and depths by replacing $\boldsymbol{c}_i$ with the corresponding normal or depth value.

Normal and Depth Estimation from Gaussians. The covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$ of a 3D Gaussian can be decomposed into a rotation matrix $\boldsymbol{R}$ and a scaling matrix $\boldsymbol{S}$, *i.e.*, $\boldsymbol{\Sigma} = \boldsymbol{R}\boldsymbol{S}\boldsymbol{S}^\top\boldsymbol{R}^\top$, where $\boldsymbol{R}$ contains the three orthogonal eigenvectors, and $\boldsymbol{S}$ encodes the scale along these directions. This decomposition resembles an ellipsoid representation: the eigenvectors define the axes of the ellipsoid, while the scale values correspond to the axis lengths. As optimization progresses, the initially spherical Gaussian flattens and approaches a plane [19]. We take the direction corresponding to the smallest scale factor as the normal $\boldsymbol{n}$ of the Gaussian. The distance from the local plane to the camera center is then computed as $\boldsymbol{d} = (\boldsymbol{R}_c^\top(\boldsymbol{\mu} - \boldsymbol{T}_c))^\top(\boldsymbol{R}_c^\top\boldsymbol{n})$, where $\boldsymbol{R}_c$ is the rotation from the camera to the world frame, and $\boldsymbol{T}_c$ is the camera center in world coordinates. Given the normal and distance, the depth is obtained by intersecting the viewing ray with the local plane: $\boldsymbol{z} = \boldsymbol{d}/(\boldsymbol{R}_c^\top\boldsymbol{n}\,\boldsymbol{K}^{-1}\bar{\boldsymbol{p}})$, where $\bar{\boldsymbol{p}}$ is the homogeneous coordinate of the pixel (we use $\boldsymbol{p}$ to denote both the homogeneous and 2D pixel coordinates for simplicity), and $\boldsymbol{K}$ is the intrinsic matrix of

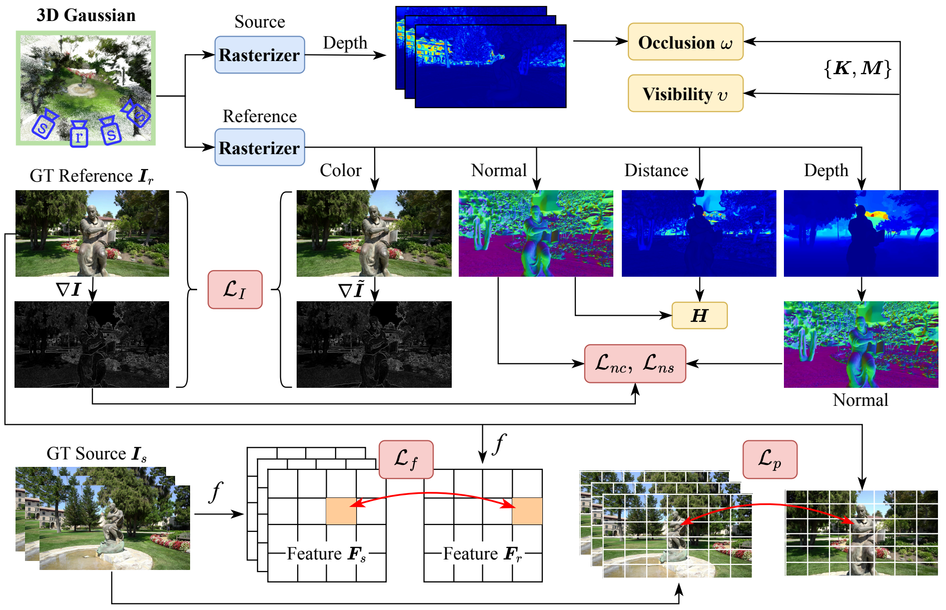

Figure 2: Overview of our method. The training includes five loss functions: $\mathcal{L}_I$, $\mathcal{L}_{nc}$, $\mathcal{L}_{ns}$, $\mathcal{L}_p$ and $\mathcal{L}_f$. The occlusion weight ω, visibility item υ and homography matrix $\boldsymbol{H}$ are involved in $\mathcal{L}_p$ and $\mathcal{L}_f$. The image features $\boldsymbol{F}_s$ and $\boldsymbol{F}_r$ are extracted using a pretrained network f. $\{\boldsymbol{K}, \boldsymbol{M}\}$ is the intrinsic/extrinsic parameter matrix of the camera view.

the camera. Finally, the per-pixel distance, depth, and normal maps under the current viewpoint are rendered using α-blending as defined in Eq. (2), where the attribute color $\boldsymbol{c}_i$ is replaced with the corresponding Gaussian attributes.

4 Method

Fig. 2 illustrates the overall framework of our approach. Given a set of posed RGB images, our goal is to learn a bunch of 3D Gaussian functions with associated attributes, such as color, opacity, position and shape, to represent the geometry of a 3D scene. We introduce novel constraints to enable accurate surface reconstruction while preserving high-quality novel view synthesis.

4.1 Single-View Alignment

Edge-aware Image Reconstruction. The original 3DGS [21] and its variants typically employ a color rendering loss, which combines the L1 reconstruction error with a D-SSIM term. While effective for overall image quality, this loss alone is insufficient for accurately capturing object boundaries during surface reconstruction, and it tends to overly smooth high-frequency regions and complex structures. To address this limitation, we propose an edge-aware image reconstruction loss that encourages the model to better preserve sharp structures and boundary details:

$$\mathcal{L}_I = (1-\beta_1)\boldsymbol{L}_1(\tilde{\boldsymbol{I}} - \boldsymbol{I}) + \beta_1 \boldsymbol{L}_{SSIM}(\tilde{\boldsymbol{I}} - \boldsymbol{I}) + \beta_2 \boldsymbol{L}_1(\nabla\tilde{\boldsymbol{I}} - \nabla\boldsymbol{I}), \tag{3}$$

where $\tilde{\boldsymbol{I}}$ is the rendered image, $\boldsymbol{I}$ is the ground-truth image, and $\nabla\boldsymbol{I}$ denotes the image gradient normalized to the range $[0, 1]$. β_1 and β_2 are weight factors. The incorporation of gradient-based supervision leads to better preservation of object contours and improves reconstruction quality in boundary and texture-rich regions.

Normal-based Geometry Alignment. 2DGS [16] introduces a normal consistency loss that aligns the normals of Gaussian primitives with those derived from the rendered depth map, ensuring that each 2D splat locally approximates the underlying object surface. However, in boundary regions, the Gaussian primitives often exhibit ambiguous normal directions due to insufficient local support, which can lead to inaccurate geometry reconstruction across different surfaces. To address this issue, we utilize image edges as proxies for geometric edges, assuming that areas with strong image gradients are likely to correspond to surface discontinuities. Thus, we adopt an edge-aware normal consistency loss defined as:

$$\mathcal{L}_{nc} = \frac{1}{\mathcal{I}} \sum_{\boldsymbol{p} \in \mathcal{I}} \delta \cdot \| \hat{\boldsymbol{N}} - \tilde{\boldsymbol{N}} \|_1 , \tag{4}$$

where $\delta = (1 - \nabla \boldsymbol{I})^2$ serves as a per-pixel weight [4] that downweights loss contributions from edge regions, and $\mathcal{I}$ denotes the set of image pixels. $\tilde{\boldsymbol{N}}$ is the rendered normal, and $\hat{\boldsymbol{N}}$ is the normal estimated from the depth map gradient [16]. To compute normal $\hat{\boldsymbol{N}}$, we first project four neighboring depth samples into 3D points in the camera coordinate system. We then estimate the surface normal at pixel $\boldsymbol{p}$ by computing the cross product of vectors formed from these projected points, effectively fitting a local plane.

While the above loss enforces the global alignment of Gaussian primitives with the actual surface, noisy primitives can still appear in flat or texture-less regions, leading to abrupt and unnatural changes in surface normals. Moreover, illumination changes, such as shadows shown in Fig. 1, may introduce false edges during reconstruction. To address these, we use a normal smoothing loss that encourages local continuity of surface normals by penalizing large discrepancies between adjacent pixels:

$$\mathcal{L}_{ns} = \frac{1}{\mathcal{I}} \sum_{i,j,k} \delta_k \cdot \mathcal{R}\left(|\hat{\boldsymbol{N}}_k - \hat{\boldsymbol{N}}_{(i,j)}| - \tau^2 \right) \cdot \left[|\tilde{\boldsymbol{N}}_k - \tilde{\boldsymbol{N}}_{(i,j)}| - \tau \right] , \tag{5}$$

where $\hat{\boldsymbol{N}}_{(i,j)}$ and $\tilde{\boldsymbol{N}}_{(i,j)}$ denote the normals at pixel location (i,j), and $k \in \{(i+1,j),(i,j+1)\}$ refers to its neighboring pixels in the horizontal and vertical directions. $\mathcal{R}(\cdot)$ is the ReLU function, and $[\cdot]$ denotes the Iverson bracket, which evaluates to 1 if the condition inside is true and 0 otherwise. The threshold τ and weight δ help distinguish surface edges and prevents over-smoothing in high-frequency regions. This loss promotes smoother local geometry while preserving meaningful structural edges, thereby improving the overall surface fidelity.

4.2 Multi-View Alignment

Multi-View Photometric Alignment. While image reconstruction and geometry alignment losses help reduce artifacts and preserve coarse geometry, they often fail to capture fine details. To address this, we draw inspiration from traditional multi-view stereo (MVS) methods [37, 3, 12], which refine surfaces by enforcing photometric consistency across views. Specifically, they project 3D points derived from depth maps onto multiple views and compare their colors to evaluate consistency. By introducing a photometric consistency loss based on plane patches, we leverage multi-view observations to resolve geometric ambiguities, particularly at object boundaries, and enhance reconstruction accuracy.

As shown in Fig. 2, let $\boldsymbol{I}_r$ be the reference view image, and $\boldsymbol{I}_s \in \{\boldsymbol{I}_{s,i} \mid i = 1, 2, \ldots, N\}$ denote its neighboring source views. For a pixel $\boldsymbol{p}_r$ in the reference view, we define its corresponding plane by normal $\boldsymbol{n}_r$ and distance $\boldsymbol{d}_r$. Using a homography matrix $\boldsymbol{H}_{rs}$, $\boldsymbol{p}_r$ is projected to $\boldsymbol{p}_s^r$ in the source view as follows:

$$\boldsymbol{p}_s^r = \boldsymbol{H}_{rs}\, \boldsymbol{p}_r, \;\; \boldsymbol{H}_{rs} = \boldsymbol{K}_s \left(\boldsymbol{R}_{rs} - \frac{\boldsymbol{T}_{rs} \boldsymbol{n}_r^\top}{\boldsymbol{d}_r} \right) \boldsymbol{K}_r^{-1} \;\; , \tag{6}$$

where $\boldsymbol{R}_{rs}$ and $\boldsymbol{T}_{rs}$ are the relative rotation and translation from the reference to the source view. Assuming local planarity, we warp a reference patch $\mathcal{P}_r$ centered at $\boldsymbol{p}_r$ to its corresponding source patch $\mathcal{P}_s$ using $\boldsymbol{H}_{rs}$. We enforce multi-view photometric alignment by encouraging consistency between $\mathcal{P}_r$ and $\mathcal{P}_s$:

$$\mathcal{L}_p = \sum_{\boldsymbol{I}_s \in \{\boldsymbol{I}_{s,i}\}} \frac{1}{V} \sum_{\boldsymbol{p}_r \in \boldsymbol{I}_r} \upsilon_{rs}(\boldsymbol{p}_r) \cdot \omega(\boldsymbol{p}_r) \cdot \left(1 - \mathcal{C}\big(\mathcal{P}_r(\boldsymbol{p}_r),\; \mathcal{P}_s(\boldsymbol{p}_s^r)\big) \right) , \;\; i = 1, 2, \ldots, N \; , \tag{7}$$

where $\mathcal{C}(\cdot)$ is the normalized cross-correlation [52], and V is the number of visible pixels. The visibility term $\upsilon_{rs}(\boldsymbol{p}_r)$ indicates whether $\boldsymbol{p}_r$ is visible in the source view, and $\omega(\boldsymbol{p}_r)$ is a weight accounting for geometric occlusion. Note that we aggregate the losses from all source views by summation, not averaging. The definitions of $\upsilon_{rs}(\boldsymbol{p}_r)$ and $\omega(\boldsymbol{p}_r)$ are detailed in the following.

• Due to viewpoint changes, a 2D pixel $\boldsymbol{p}_r$ in the reference view may fall outside the field of view when projected into a source view. We define a visibility term $\upsilon_{rs}(\boldsymbol{p}_r)$ to indicate whether $\boldsymbol{p}_r$ is visible from the source viewpoint. Given a pixel $\boldsymbol{p}_r$ with rendered depth $\boldsymbol{z}_r$, its corresponding 3D point $\boldsymbol{x}_r$ and projected pixel coordinate $\boldsymbol{p}'_s$ in the source view are computed as:

$$\boldsymbol{p}'_s = \pi(\boldsymbol{K}\boldsymbol{M}_s\boldsymbol{M}_r^{-1}\boldsymbol{x}_r), \;\; \boldsymbol{x}_r = \boldsymbol{z}_r\boldsymbol{K}^{-1}\bar{\boldsymbol{p}}_r \;, \tag{8}$$

where $\boldsymbol{M}$ is the extrinsic matrix of the camera, $\pi(\cdot)$ converts 3D coordinates to 2D pixels. The pixel $\boldsymbol{p}_r$ is considered visible in the source view if its projection $\boldsymbol{p}'_s$ lies within the image bounds. Thus the visibility term is defined as:

$$\upsilon_{rs}(\boldsymbol{p}_r) = \big[(0,0) < \boldsymbol{p}'_s < (W,H)\big], \tag{9}$$

where (W,H) is the image resolution, and $[\cdot]$ denotes the Iverson bracket.

• During projection via the homography matrix, some pixels may be occluded or exhibit significant geometric error [4]. To avoid the influence of such outliers, we exclude them from the multi-view alignment loss using an occlusion-aware weight. Given a reference 3D point $\boldsymbol{x}_r$ and its corresponding rendered (or interpolated) depth $\boldsymbol{z}_s$ in the source view, we first compute the projection error at $\boldsymbol{p}_r$ as:

$$\varphi(\boldsymbol{p}_r) = ||\, \boldsymbol{p}_r - \boldsymbol{p}'_r \,||_2 \;, \tag{10}$$

$$\boldsymbol{p}'_r = \pi(\boldsymbol{K}\boldsymbol{M}_r\boldsymbol{M}_s^{-1}\boldsymbol{x}_s), \;\; \boldsymbol{x}_s = \ddot{\boldsymbol{x}}'_s \cdot \boldsymbol{z}_s, \;\; \boldsymbol{x}'_s = \boldsymbol{M}_s\boldsymbol{M}_r^{-1}\boldsymbol{x}_r, \tag{11}$$

where $\boldsymbol{p}'_r$ is the reprojected pixel in the reference view, $\ddot{\boldsymbol{x}}'_s$ denotes the depth normalized version of $\boldsymbol{x}'_s$. We then define the occlusion weight as $\omega(\boldsymbol{p}_r) = 1/\exp(\varphi(\boldsymbol{p}_r))$ if $\varphi(\boldsymbol{p}_r) < 1$, and otherwise 0. A small projection error indicates reliable geometry, resulting in a higher weight, while a large error implies occlusion or misalignment, thus being downweighted or discarded.

Multi-View Feature Alignment. The previously introduced image reconstruction and photometric alignment losses help preserve the shape and structure of the objects. However, image-based losses are susceptible to noise, blur, and low-texture regions. Additionally, due to lighting variations, the color of the same surface point may differ across views, making photometric consistency unreliable. To address these limitations, we introduce a multi-view feature alignment loss. We extract image features using a pretrained network f [58], *i.e.*, $\boldsymbol{F} = f(\boldsymbol{I})$. Let $\boldsymbol{F}_r$ denote the reference view's feature map, and $\boldsymbol{F}_s$ be one of the source view features, with $\boldsymbol{F}_s \in \{\boldsymbol{F}_{s,i} \mid i = 1,2,\dots,N\}$. Then the pixel-wise feature alignment loss is defined as:

$$\mathcal{L}_f = \frac{1}{N}\sum_{\boldsymbol{F}_s\in\{\boldsymbol{F}_{s,i}\}}\frac{1}{V}\sum_{\boldsymbol{p}_r\in\boldsymbol{I}_r}\upsilon_{rs}(\boldsymbol{p}_r)\cdot\omega(\boldsymbol{p}_r)\cdot\big|1-\cos\big(\boldsymbol{F}_r(\boldsymbol{p}_r),\;\boldsymbol{F}_s(\boldsymbol{p}'_s)\big)\big|\,,\;\; i=1,2,\dots,N. \tag{12}$$

where $\cos(\cdot)$ denotes the cosine similarity between feature vectors. This feature-level loss improves robustness under challenging conditions such as appearance variation and poor lighting consistency.

Final loss. To summarize, the final training objective integrates five components:

$$\mathcal{L} = \mathcal{L}_I + \lambda_1\mathcal{L}_{nc} + \lambda_2\mathcal{L}_{ns} + \lambda_3\mathcal{L}_p + \lambda_4\mathcal{L}_f \;, \tag{13}$$

where λ_1, λ_2, λ_3 and λ_4 are weighting factors determined based on validation performance.

5 Experiments

Evaluation Protocols. We evaluate our surface reconstruction performance on the DTU [18] and Tanks and Temples (TNT) [22] datasets. Following prior works [16, 56, 4, 57], we use 15 scenes from the DTU dataset and 6 scenes from the TNT dataset for evaluation. Depth maps are rendered for all training views, and a TSDF [7] is constructed for mesh extraction. For novel view synthesis, we use the Mip-NeRF 360 dataset [2], which contains large-scale indoor and outdoor scenes with complex lighting and fine-grained geometric details. Following 3DGS [21], one out of every eight images is used for evaluation, while the remaining seven are used for training. We employ COLMAP [36] to generate a sparse point cloud from the original dataset images for initializing the 3D Gaussians. All images are downsampled to a lower resolution to facilitate training. Following established protocols [16, 56, 4, 57], we report Chamfer distance for surface reconstruction on the DTU dataset and F1-score for the TNT dataset. For novel view synthesis, we evaluate using three widely adopted image quality metrics: PSNR, SSIM, and LPIPS.

Implementation Details. Our overall pipeline, training strategy, and hyperparameter settings generally follow 3DGS [21]. We set the number of source views to $N = 3$, the threshold in $\mathcal{L}_{ns}$

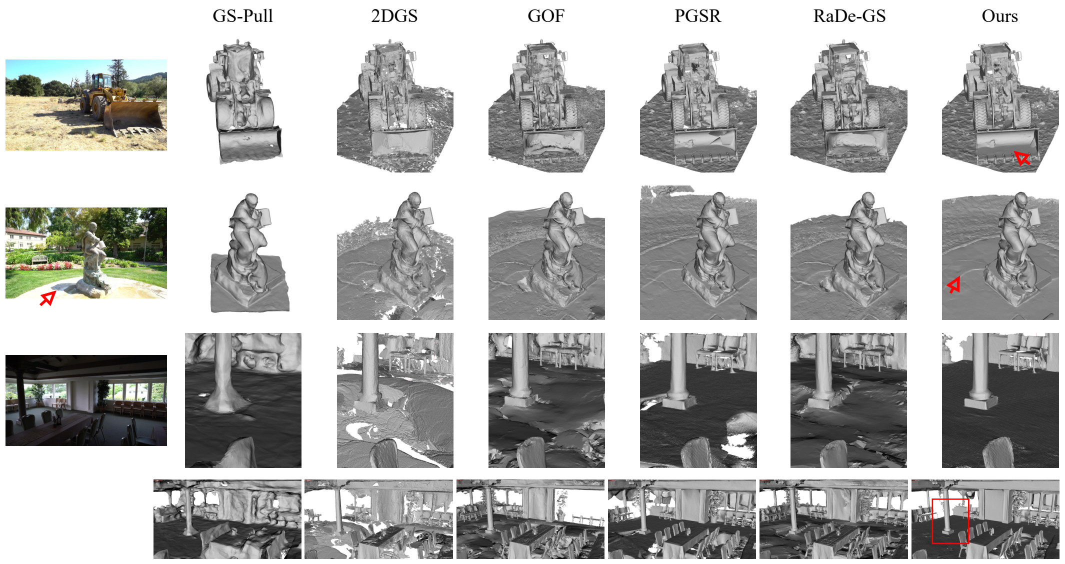

Figure 3: Visual comparison of surface reconstruction results on the TNT dataset. Our method can handle shadows and large indoor flat regions. GS-Pull reconstructs only the foreground objects.

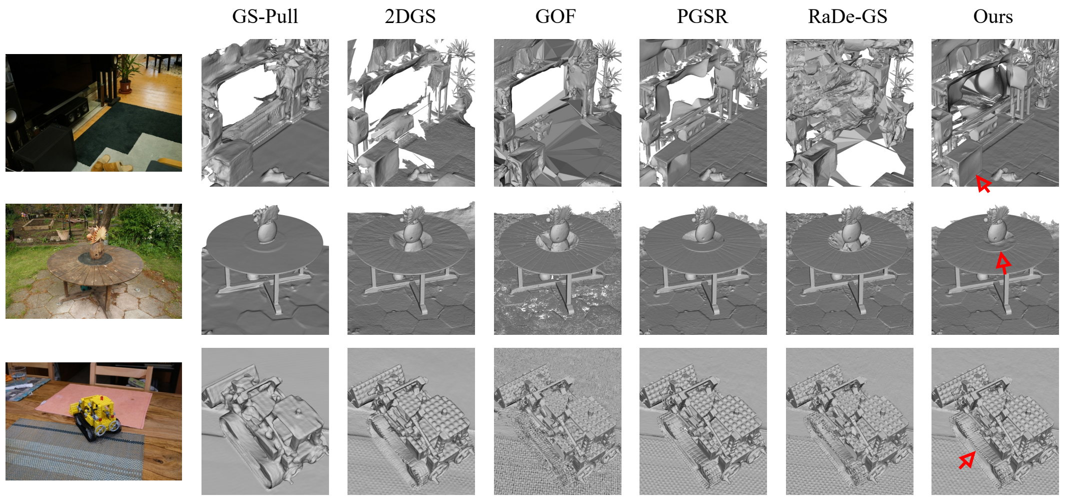

Figure 4: Visual comparison of surface reconstruction results on the Mip-NeRF 360 dataset. Our approach effectively handles the challenges posed by cluttered lighting and boundaries.

to $\tau = 0.01$, and the patch size in $\mathcal{L}_p$ to 7×7. The loss weight factors are set as follows: $\beta_1 = 0.2$, $\beta_2 = 0.03$, $\lambda_1 = 0.015$, $\lambda_2 = 0.3$, $\lambda_3 = 0.15$, and $\lambda_4 = 1.0$. The model is trained for 20,000 iterations for surface reconstruction and 30,000 iterations for novel view synthesis. We first pretrain the model using only the color loss for 7,000 steps to obtain a coarse geometric initialization, which provides a stable foundation for subsequent geometry refinement. Then, we incorporate our image edge item and normal-based geometry alignment into the training. To further refine geometry, we sequentially apply our multi-view photometric alignment for 8,000 iterations, followed by 5,000 iterations of multi-view feature alignment. For novel view synthesis, we continue training for an additional 10,000 steps to optimize rendering quality. All experiments are conducted on a single NVIDIA RTX 4090 GPU.

5.1 Performance Evaluation

Comparisons on DTU. We first compare our method with state-of-the-art implicit and explicit surface reconstruction approaches on the DTU dataset [18]. Following standard protocol, reconstructions are clipped using the provided mask, and evaluations are performed only on foreground objects, as the ground truth point clouds exclude background regions. As shown in Table 1, our method achieves the lowest average Chamfer distance and ranks best across most scenes. Compared to implicit approaches

Table 1: Quantitative comparison of Chamfer distances on the DTU dataset. The best results are highlighted as 1st, 2nd and 3rd. ∗ means that the source code is not available.

		24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	**Mean**	Time
Implicit	NeRF [31]	1.90	1.60	1.85	0.58	2.28	1.27	1.47	1.67	2.05	1.07	0.88	2.53	1.06	1.15	0.96	1.49	>12h
	VolSDF [48]	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	>12h
	NeuS [44]	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	>12h
	NeuralWarp [9]	0.49	0.71	0.38	0.38	0.79	0.81	0.82	1.20	1.06	0.68	0.66	0.74	0.41	0.63	0.51	0.68	>10h
	Neuralangelo [27]	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	>12h
	PSDF* [39]	0.36	0.60	0.35	0.36	0.70	0.61	0.49	1.11	0.89	0.60	0.47	0.57	0.30	0.40	0.37	0.55	-
Explicit	3DGS [21]	2.14	1.53	2.08	1.68	3.49	2.21	1.43	2.07	2.22	1.75	1.79	2.55	1.53	1.52	1.50	1.96	3.4m
	SuGaR [14]	1.47	1.33	1.13	0.61	2.25	1.71	1.15	1.63	1.62	1.07	0.79	2.45	0.98	0.88	0.79	1.33	1h
	GaussianSurfels[8]	0.66	0.93	0.54	0.41	1.06	1.14	0.85	1.29	1.53	0.79	0.82	1.58	0.45	0.66	0.53	0.88	4.5m
	2DGS [16]	0.48	0.91	0.39	0.39	1.01	0.83	0.81	1.36	1.27	0.76	0.70	1.40	0.40	0.76	0.52	0.80	5.8m
	GS-Pull [60]	0.51	0.56	0.46	0.39	0.82	0.67	0.85	1.37	1.25	0.73	0.54	1.39	0.35	0.88	0.42	0.75	5.6m
	GOF [56]	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	32m
	RaDe-GS [57]	0.46	0.73	0.33	0.38	0.79	0.75	0.76	1.19	1.22	0.62	0.70	0.78	0.36	0.68	0.47	0.68	6.5m
	PGSR [4]	0.34	0.58	0.29	0.29	0.78	0.58	0.54	1.01	0.73	0.51	0.49	0.69	0.31	0.37	0.38	0.53	15m
	GausSurf* [43]	0.35	0.55	0.34	0.34	0.77	0.58	0.51	1.10	0.69	0.60	0.43	0.49	0.32	0.40	0.37	0.52	-
	Ours	0.32	0.49	0.32	0.30	0.77	0.68	0.43	1.05	0.61	0.57	0.36	0.52	0.28	0.33	0.30	0.49	15.5m

Table 2: Quantitative comparison of F1-scores on the TNT dataset. The best results are highlighted as 1st, 2nd and 3rd. ∗ means that the source code is not available.

		Barn	Caterpillar	Courthouse	Ignatius	Meetingroom	Truck	**Mean**	Time
Implicit	NeuS [44]	0.29	0.29	0.17	0.83	0.24	0.45	0.38	>12h
	Geo-Neus [12]	0.33	0.26	0.12	0.72	0.20	0.45	0.35	>12h
	Neuralangelo [27]	0.70	0.36	0.28	0.89	0.32	0.48	0.50	>12h
	PSDF* [39]	0.62	0.39	0.42	0.79	0.47	0.53	0.53	-
Explicit	3DGS [21]	0.13	0.08	0.09	0.04	0.01	0.19	0.09	7.5m
	DN-Splatter [42]	0.15	0.11	0.07	0.18	0.01	0.20	0.12	20m
	SuGaR [14]	0.14	0.16	0.08	0.33	0.15	0.26	0.19	2h
	GaussianSurfels [8]	0.24	0.22	0.07	0.39	0.12	0.24	0.21	5m
	2DGS [16]	0.41	0.23	0.16	0.51	0.17	0.45	0.32	7.5m
	GS-Pull [60]	0.60	0.37	0.16	0.71	0.22	0.52	0.43	18m
	GOF [56]	0.51	0.41	0.28	0.68	0.28	0.59	0.46	40m
	RaDe-GS [57]	0.43	0.32	0.21	0.69	0.25	0.51	0.40	9m
	PGSR [4]	0.66	0.44	0.20	0.81	0.33	0.66	0.52	25.5m
	GausSurf* [43]	0.50	0.42	0.30	0.73	0.39	0.65	0.50	-
	Ours	0.71	0.45	0.21	0.82	0.40	0.64	0.54	20.6m

such as NeuS [44] and Neuralangelo [27], our method delivers significantly better reconstruction accuracy while being much more efficient in terms of runtime. It is worth noting that most implicit methods [44, 27] only reconstruct foreground geometry, whereas our approach can produce detailed and complete meshes, including background regions, which is an essential feature for mesh-based rendering. Although our method is slightly slower than 3DGS [21] and 2DGS [16] due to the use of multi-view alignment, it achieves significant improvements in reconstruction quality over these earlier Gaussian-based methods.

Comparisons on TNT. We further evaluate our method on the TNT dataset [22], comparing it against both implicit and explicit surface reconstruction baselines. Since the ground-truth point clouds do not include background regions, the evaluation is restricted to foreground objects. As shown in Table 2, our method achieves the best reconstruction performance among all competing approaches, including both implicit and explicit methods. Notably, while several Gaussian-based methods require less optimization time, they tend to produce results with much lower accuracy. In contrast, our method reaches a better balance between efficiency and reconstruction quality. For example, GS-Pull [60] only reconstructs foreground objects and often generates overly smooth surfaces. Fig. 3 provides a qualitative comparison. Our method produces more accurate and detailed reconstructions for both foreground and background regions. It also effectively mitigates the impact of shadows, whereas baseline methods often yield noisy meshes or fail to capture geometric details. The use of geometry, photometric, and feature-based alignment from multiple views provides strong guidance, enabling the Gaussian primitives to converge more accurately to the true surface geometry.

Table 3: Quantitative comparison on the Mip-NeRF 360 dataset. The best results are highlighted as 1st, 2nd and 3rd.

	Outdoor scenes			Indoor scenes			Average on all scenes		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF [31]	21.46	0.458	0.515	26.84	0.790	0.370	23.85	0.606	0.451
Deep Blending [15]	21.54	0.524	0.364	26.40	0.844	0.261	23.70	0.666	0.318
Instant NGP [32]	22.90	0.566	0.371	29.15	0.880	0.216	25.68	0.706	0.302
MERF [35]	23.19	0.616	0.343	27.80	0.855	0.271	25.24	0.722	0.311
BakedSDF [49]	22.47	0.585	0.349	27.06	0.836	0.258	24.51	0.697	0.309
Mip-NeRF 360 [2]	24.47	0.691	0.283	31.72	0.917	0.180	27.69	0.791	0.237
3DGS [21]	24.64	0.731	0.234	30.41	0.920	0.189	27.20	0.815	0.214
SuGaR [14]	22.93	0.629	0.356	29.43	0.906	0.225	25.82	0.752	0.298
2DGS [16]	24.34	0.717	0.246	30.40	0.916	0.195	27.03	0.805	0.223
GS-Pull [60]	23.76	0.703	0.278	30.78	0.925	0.182	26.88	0.802	0.235
GOF [56]	24.82	0.750	0.202	30.79	0.924	0.184	27.47	0.827	0.194
RaDe-GS [57]	25.17	0.764	0.199	30.74	0.928	0.165	27.65	0.837	0.184
PGSR [4]	24.45	0.730	0.224	30.41	0.930	0.161	27.10	0.819	0.196
GausSurf [43]	25.09	0.753	0.212	30.05	0.920	0.183	27.29	0.827	0.199
Ours	25.00	0.760	0.191	30.63	0.933	0.153	27.50	0.837	0.174

Comparisons on Mip-NeRF 360. We also evaluate our approach on the Mip-NeRF 360 dataset [2] for novel view synthesis. Table 3 reports quantitative comparisons against state-of-the-art Gaussian-based and other neural rendering baselines. Our method outperforms competitors on most metrics, demonstrating superior image fitting and generalization to unseen viewpoints. This evidences that our enhanced geometry representation yields higher visual fidelity. Notably, the Mip-NeRF 360 itself achieves the highest average PSNR on indoor scenes but lags on SSIM and LPIPS. Among Gaussian-based methods, 2DGS [16], SuGaR [14], and GS-Pull [60] perform worse than vanilla 3DGS [21], suggesting that their planar Gaussian constraints degrade performance in complex environments. Our ablation results in Table 4 further confirm that flattening 3D Gaussians into planar Gaussian disks is ineffective for our framework. Our method preserves the full 3D Gaussian representation and delivers high-quality surfaces without sacrificing novel-view rendering quality. Fig. 4 provides a qualitative comparison of reconstructed meshes. Consistent with our observations on the TNT dataset, our method recovers more accurate and complete surfaces in both foreground and background regions, whereas other methods suffer from noise, oversmoothing, or missing details, especially in challenging indoor scenes.

5.2 Ablation Studies

To quantify the contributions of our alignment constraints, we perform ablations by selectively removing loss terms and report reconstruction quality on the TNT dataset. In addition to the *F1-score*, we also report *Precision* and *Recall* to provide a more comprehensive evaluation. The base color rendering loss from 3DGS is always retained in the following experiments. We provide quantitative results in Table 4.

Table 4: Ablations on the TNT dataset.

	Precision ↑	Recall ↑	F1-score ↑
Only $\mathcal{L}_I$	0.09	0.23	0.13
w/o edge item	0.49	0.59	0.53
w/o weight δ	0.50	0.59	0.53
w/o $\mathcal{L}_{nc}$	0.48	**0.60**	0.52
w/o $\mathcal{L}_{ns}$	0.47	0.58	0.51
w/o $\mathcal{L}_{nc} + \mathcal{L}_{ns}$	0.40	0.57	0.46
w/o $\mathcal{L}_p$	0.46	0.56	0.50
w/o $\mathcal{L}_f$	0.49	**0.60**	0.53
w/o $\mathcal{L}_p + \mathcal{L}_f$	0.33	0.40	0.36
w/ scale loss	**0.51**	**0.60**	**0.54**
$N = 1$	0.49	0.58	0.52
$N = 2$	0.49	0.59	0.53
$N = 4$	**0.51**	**0.60**	**0.54**
Ours	**0.51**	**0.60**	**0.54**

(1) *Only image reconstruction loss ($\mathcal{L}_I$)*: Removing all alignment losses yields the worst results, with an average F1-score of 0.13, but still better than the vanilla 3DGS's score of 0.09.

(2) *Edge-aware term in $\mathcal{L}_I$*: Omitting the image edge-based component slightly degrades performance, confirming its role in preserving boundary detail.

(3) *Edge-aware weight δ*: In boundary regions, Gaussian primitives often exhibit ambiguous or noisy normal directions, which can lead to incorrect supervision signals. The weight δ in loss $\mathcal{L}_{nc}$ reduces the loss contribution from these areas, allowing the network to focus learning on more reliable surface

regions. While the improvement is modest, it reflects the fact that shape boundaries constitute a relatively small proportion of the scene, and thus affect only a small number of sampled points during evaluation.

(4) *Normal-based alignment ($\mathcal{L}_{nc}$, $\mathcal{L}_{ns}$)*: The normal consistency ($\mathcal{L}_{nc}$) and smoothing ($\mathcal{L}_{ns}$) losses are critical. Excluding either term causes a noticeable drop in Precision and F1-score, and removing both leads to a dramatic performance collapse.

(5) *Multi-view alignment ($\mathcal{L}_{p}$, $\mathcal{L}_{f}$)*: Enforcing photometric and feature consistency across views consistently improves reconstruction accuracy. Each multi-view alignment term contributes positively, validating the benefit of cross-view geometric constraints.

(6) *Scale regularization*: The scaling matrix $\boldsymbol{S}$ represents the stretching of a spherical Gaussian along the three axes. Different from previous works [4, 6, 60], incorporating the widely used scale penalty into our method to flatten the 3D Gaussian disks provides no performance gains, and even degrades novel-view rendering quality on the Mip-NeRF 360 dataset.

(7) *Number of source views (N)*: Our method takes both a reference view and N source views. Increasing the number of source views used in the alignment losses improves reconstruction quality. However, setting $N = 4$ yields no additional performance gains but increases the computational cost. We therefore choose $N = 3$ to balance accuracy and efficiency.

Overall, these ablations demonstrate that each of our proposed alignment constraints plays a distinct and essential role in achieving high-fidelity surface reconstruction.

6 Conclusion

In this paper, we address the limitations of existing 3D Gaussian Splatting approaches in recovering accurate and detailed surface geometry, especially under challenging conditions such as complex lighting and ambiguous object boundaries. We propose a novel method that improves geometric fidelity by integrating edge-aware supervision, visibility-aware multi-view alignment, and robust geometric constraints based on surface normals and deep visual features. These components jointly enforce cross-view consistency, enhance boundary sharpness, and mitigate the impact of illumination-induced artifacts. Extensive experiments demonstrate that our method achieves state-of-the-art performance in both surface reconstruction and novel view synthesis, underscoring its effectiveness and robustness in complex real-world scenarios. The main limitation of our approach is its relatively slower training speed compared to earlier 3DGS variants. In future work, we aim to explore adaptive Gaussian pruning and learned covariance regularization to accelerate training and further improve robustness in large-scale and dynamic scenes.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (62402401), the Sichuan Provincial Natural Science Foundation of China (2025ZNSFSC1462) and the Fundamental Research Funds for the Central Universities.

References

[1] Xu Baixin, Hu Jiangbei, Li Jiaze, and He Ying. GSurf: 3D reconstruction via signed distance fields with direct gaussian supervision. *arXiv preprint arXiv:2411.15723*, 2024.

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5470–5479, 2022.

[3] Neill DF Campbell, George Vogiatzis, Carlos Hernández, and Roberto Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part I 10*, pages 766–779. Springer, 2008.

[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. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

[5] Hanlin Chen, Fangyin Wei, Chen Li, Tianxin Huang, Yunsong Wang, and Gim Hee Lee. VCR-GauS: View consistent depth-normal regularizer for gaussian surface reconstruction. *arXiv preprint arXiv:2406.05774*, 2024.

[6] Kai Cheng, Xiaoxiao Long, Kaizhi Yang, Yao Yao, Wei Yin, Yuexin Ma, Wenping Wang, and Xuejin Chen. GaussianPro: 3D gaussian splatting with progressive propagation. In *Forty-first International Conference on Machine Learning*, 2024.

[7] 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.

[8] Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. High-quality surface reconstruction using gaussian surfels. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024.

[9] François Darmon, Bénédicte Bascle, Jean-Clément Devaux, Pascal Monasse, and Mathieu Aubry. Improving neural implicit surfaces geometry with patch warping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6260–6269, 2022.

[10] Akio Doi and Akio Koide. An efficient method of triangulating equi-valued surfaces by using tetrahedral cells. *IEICE TRANSACTIONS on Information and Systems*, 74(1):214–224, 1991.

[11] Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, et al. Instantsplat: Unbounded sparse-view pose-free gaussian splatting in 40 seconds. *arXiv preprint arXiv:2403.20309*, 2024.

[12] Qiancheng Fu, Qingshan Xu, Yew Soon Ong, and Wenbing Tao. Geo-Neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. *Advances in Neural Information Processing Systems*, 35:3403–3416, 2022.

[13] Jian Gao, Chun Gu, Youtian Lin, Hao Zhu, Xun Cao, Li Zhang, and Yao Yao. Relightable 3D gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing. *arXiv preprint arXiv:2311.16043*, 2023.

[14] Antoine Guédon and Vincent Lepetit. SuGaR: Surface-aligned gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5354–5363, 2024.

[15] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018.

[16] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024.

[17] Han Huang, Yulun Wu, Chao Deng, Ge Gao, Ming Gu, and Yu-Shen Liu. FatesGS: Fast and accurate sparse-view surface reconstruction using gaussian splatting with depth-feature consistency. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[18] 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.

[19] Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. GaussianShader: 3D gaussian splatting with shading functions for reflective surfaces. *arXiv preprint arXiv:2311.17977*, 2023.

[20] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics Symposium on Geometry Processing*, 2006.

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023.

[22] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017.

[23] Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. Dngaussian: Optimizing sparse-view 3D gaussian radiance fields with global-local depth normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20775–20785, 2024.

[24] Kunyi Li, Michael Niemeyer, Zeyu Chen, Nassir Navab, and Federico Tombari. MonoGSDF: Exploring monocular geometric cues for gaussian splatting-guided implicit surface reconstruction. *arXiv preprint arXiv:2411.16898*, 2024.

[25] Shujuan Li, Yu-Shen Liu, and Zhizhong Han. GaussianUDF: Inferring unsigned distance functions through 3D gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[26] Yanyan Li, Chenyu Lyu, Yan Di, Guangyao Zhai, Gim Hee Lee, and Federico Tombari. GeoGaussian: Geometry-aware gaussian splatting for scene rendering. In *European Conference on Computer Vision*, pages 441–457. Springer, 2024.

[27] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8456–8465, 2023.

[28] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-GS: Structured 3D gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024.

[29] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3D gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023.

[30] Xiaoyang Lyu, Yang-Tian Sun, Yi-Hua Huang, Xiuzhe Wu, Ziyi Yang, Yilun Chen, Jiangmiao Pang, and Xiaojuan Qi. 3DGSR: Implicit surface reconstruction with 3D gaussian splatting. *arXiv preprint arXiv:2404.00409*, 2024.

[31] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, pages 405–421, 2020.

[32] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022.

[33] 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.

[34] Michael Niemeyer, Fabian Manhardt, Marie-Julie Rakotosaona, Michael Oechsle, Daniel Duckworth, Rama Gosula, Keisuke Tateno, John Bates, Dominik Kaeser, and Federico Tombari. Radsplat: Radiance field-informed gaussian splatting for robust real-time rendering with 900+ fps. *arXiv preprint arXiv:2403.13806*, 2024.

[35] Christian Reiser, Rick Szeliski, Dor Verbin, Pratul Srinivasan, Ben Mildenhall, Andreas Geiger, Jon Barron, and Peter Hedman. Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023.

[36] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[37] 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.

[38] Xiaowei Song, Jv Zheng, Shiran Yuan, Huan-ang Gao, Jingwei Zhao, Xiang He, Weihao Gu, and Hao Zhao. SA-GS: Scale-adaptive gaussian splatting for training-free anti-aliasing. *arXiv preprint arXiv:2403.19615*, 2024.

[39] Wanjuan Su, Chen Zhang, Qingshan Xu, and Wenbing Tao. PSDF: Prior-driven neural implicit surface learning for multi-view reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

[40] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022.

[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. *arXiv preprint arXiv:2403.17822*, 2024.

[42] 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 *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2421–2431. IEEE, 2025.

[43] Jiepeng Wang, Yuan Liu, Peng Wang, Cheng Lin, Junhui Hou, Xin Li, Taku Komura, and Wenping Wang. GausSurf: Geometry-guided 3D gaussian splatting for surface reconstruction. *arXiv preprint arXiv:2411.19454*, 2024.

[44] 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. *Advances in Neural Information Processing Systems*, 34:27171–27183, 2021.

[45] Yaniv Wolf, Amit Bracha, and Ron Kimmel. Surface reconstruction from gaussian splatting via novel stereo views. *arXiv preprint arXiv:2404.01810*, 2024.

[46] Runyi Yang, Zhenxin Zhu, Zhou Jiang, Baijun Ye, Xiaoxue Chen, Yifei Zhang, Yuantao Chen, Jian Zhao, and Hao Zhao. Spectrally pruned gaussian fields with neural compensation. *arXiv preprint arXiv:2405.00676*, 2024.

[47] Zeyu Yang, Hongye Yang, Zijie Pan, Xiatian Zhu, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *International Conference on Learning Representations (ICLR)*, 2024.

[48] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. *Advances in Neural Information Processing Systems*, 34:4805–4815, 2021.

[49] Lior Yariv, Peter Hedman, Christian Reiser, Dor Verbin, Pratul P Srinivasan, Richard Szeliski, Jonathan T Barron, and Ben Mildenhall. BakedSDF: Meshing neural sdfs for real-time view synthesis. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–9, 2023.

[50] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No pose, no problem: Surprisingly simple 3d gaussian splats from sparse unposed images. *arXiv preprint arXiv:2410.24207*, 2024.

[51] Keyang Ye, Tianjia Shao, and Kun Zhou. Animatable 3D gaussians for high-fidelity synthesis of human motions. *arXiv preprint arXiv:2311.13404*, 2023.

[52] Jae-Chern Yoo and Tae Hee Han. Fast normalized cross-correlation. *Circuits, Systems and Signal Processing*, 28:819–843, 2009.

[53] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. PlenOctrees for real-time rendering of neural radiance fields. In *IEEE International Conference on Computer Vision*, 2021.

[54] Mulin Yu, Tao Lu, Linning Xu, Lihan Jiang, Yuanbo Xiangli, and Bo Dai. GSDF: 3dgs meets sdf for improved rendering and reconstruction. *arXiv preprint arXiv:2403.16964*, 2024.

[55] 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.

[56] Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes. *ACM Transactions on Graphics (TOG)*, 43(6):1–13, 2024.

[57] Baowen Zhang, Chuan Fang, Rakesh Shrestha, Yixun Liang, Xiaoxiao Long, and Ping Tan. RaDe-GS: Rasterizing depth in gaussian splatting. *arXiv preprint arXiv:2406.01467*, 2024.

[58] Jingyang Zhang, Shiwei Li, Zixin Luo, Tian Fang, and Yao Yao. Vis-mvsnet: Visibility-aware multi-view stereo network. *International Journal of Computer Vision*, 131(1):199–214, 2023.

[59] Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. Flare: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. *arXiv preprint arXiv:2502.12138*, 2025.

[60] Wenyuan Zhang, Yu-Shen Liu, and Zhizhong Han. Neural signed distance function inference through splatting 3D gaussians pulled on zero-level set. *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[61] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. EWA volume splatting. In *Proceedings Visualization, 2001. VIS'01.*, pages 29–538. IEEE, 2001.

This supplementary document is organized as follows: (1) We first present an overview of the baseline methods used in our evaluation experiments. (2) We then provide additional qualitative and quantitative experimental results to complement those in the main paper. (3) Further ablation studies are included to analyze the impact of key components in our method. (4) More discussion is provided to give a precise understanding of our approach. (5) We discuss the current limitations of our approach and outline potential directions for future research. (6) Finally, we consider the potential negative societal impacts associated with our work.

A Baseline Methods

We compare our approach with state-of-the-art surface reconstruction and novel view synthesis methods based on 3D Gaussian Splatting, such as 3DGS [21], 2DGS [16], GOF [56], and PGSR [4]. In addition, we evaluate against implicit NeRF-based methods, such as NeuS [44] and Neuralangelo [27], which utilize Signed Distance Functions (SDFs) to represent scenes and convert them into opacity fields for volume rendering via ray tracing.

In our main paper, we report baseline results using values provided in the respective original publications whenever available. For visual comparisons shown in our figures, we generate the results using the official implementations released by the authors.

B More Evaluation Results

B.1 Comparisons on the Deep Blending Dataset

In order to further test the performance of the algorithm on different real-world data, we evaluate both novel view synthesis and surface reconstruction on two commonly used indoor scenes, Dr Johnson and Playroom, from the Deep Blending dataset [15]. Table 5 presents quantitative comparisons against state-of-the-art Gaussian-based methods on novel view synthesis. All results, unless otherwise noted, are obtained from our own re-implementations using the official code released by the respective authors. For 3DGS [21], we use the pretrained models provided by the authors. For SuGaR [14], we report numbers directly from the original paper to avoid potential inconsistencies. Our method consistently outperforms all baselines across all metrics, demonstrating superior rendering quality and generalization to unseen viewpoints. These improvements can be attributed to our enhanced geometric representation, which yields better visual fidelity. Consistent with observations in the evaluation on the Mip-NeRF 360 dataset [2] presented in the main paper, methods such as 2DGS [16], SuGaR [14], and GS-Pull [60] underperform compared to vanilla 3DGS [21], suggesting that imposing planar Gaussian constraints may hinder performance in complex scenes.

Fig. 5 shows qualitative comparisons of reconstructed meshes on the Deep Blending dataset. Our method recovers more accurate and complete surfaces, handling both dark and bright regions effectively. Competing methods often exhibit noise, oversmoothing, or missing geometry, especially near object boundaries. Results from GS-Pull are omitted due to its poor mesh quality and potential need for significant parameter tuning. For other baselines, we use the same default parameters as used in their evaluations on the Mip-NeRF 360 dataset.

B.2 More Experimental Results

Novel view synthesis. To provide a more comprehensive evaluation of novel view synthesis, we report per-scene quality metrics (PSNR, SSIM, and LPIPS) for Gaussian-based methods on the Mip-NeRF 360 dataset [2], as shown in Table 6. All results are obtained from our own runs using the official implementations of prior methods, except for 3DGS, where we use the pretrained models provided by the authors to avoid training inconsistencies. Our method consistently achieves the highest PSNR and SSIM scores on most scenes and has the best average scores. Notably, it yields significant improvements in LPIPS across all scenes, highlighting its ability to capture high-frequency details and produce perceptually superior renderings.

Qualitative comparisons are shown in Fig. 6, where we visualize novel view synthesis results on the Mip-NeRF 360 dataset. As highlighted in the boxed regions, our method produces sharper and more detailed renderings, particularly in complex areas such as grass and foliage, where competing methods often yield blurry outputs.

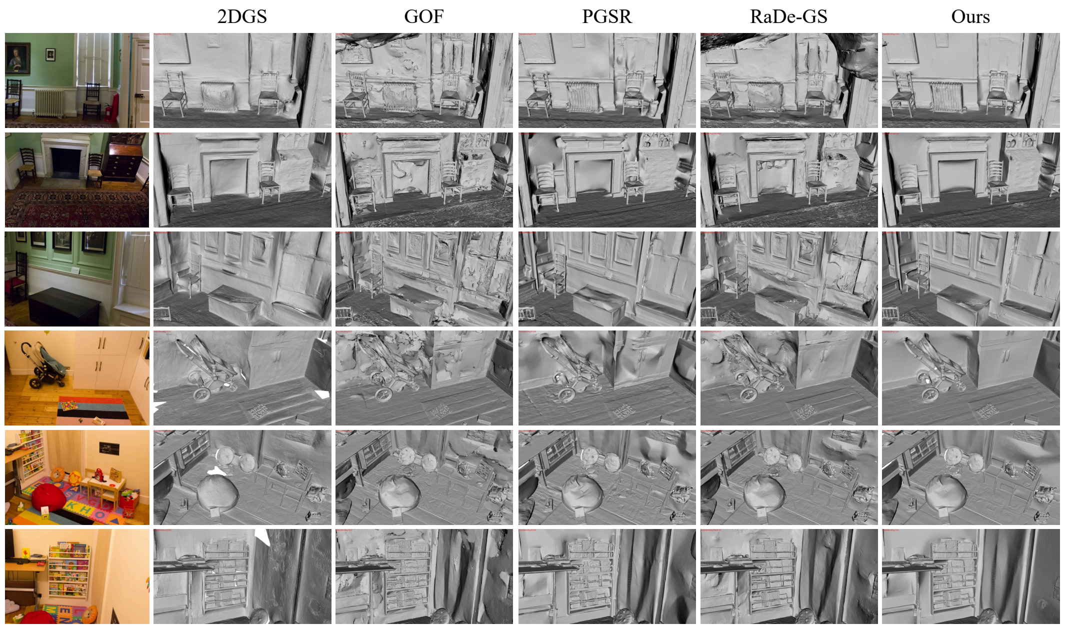

Figure 5: Visual comparison of surface reconstruction results on the Deep Blending dataset. Our method effectively handles the challenges posed by complex lighting conditions and ambiguous boundaries. GS-Pull is omitted as it fails to produce reasonable reconstructions.

Table 5: Quantitative comparison on the Deep Blending dataset. The best results are highlighted as 1st , 2nd and 3rd . $*$ indicates results copied from the original paper.

	Dr Johnson			Playroom			**Mean**		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
3DGS [21]	28.94	0.896	0.248	29.93	0.901	0.244	29.43	0.898	0.246
SuGaR* [14]	28.71	0.889	0.273	30.12	0.898	0.261	29.41	0.893	0.267
2DGS [16]	28.89	0.898	0.259	29.88	0.901	0.259	29.38	0.899	0.259
GS-Pull [60]	25.69	0.830	0.387	25.89	0.838	0.375	25.79	0.834	0.381
GOF [56]	27.85	0.893	0.257	30.16	0.904	0.242	29.01	0.899	0.250
RaDe-GS [57]	27.83	0.896	0.257	30.04	0.905	0.243	28.94	0.901	0.250
PGSR [4]	28.61	0.891	0.251	29.92	0.903	0.243	29.27	0.897	0.247
Ours	29.10	0.900	0.241	30.22	0.905	0.241	29.66	0.903	0.241

Surface reconstruction. Fig. 7 presents additional qualitative comparisons of reconstructed surfaces on real-world indoor and outdoor scenes from the TNT [22] and Mip-NeRF 360 [2] datasets. Compared to baseline methods, our approach produces more complete and continuous meshes with well-preserved high-frequency details. It also effectively avoids local minima, maintaining fine structures such as holes and sharp edges. By contrast, 2DGS [16] and GOF [56] often yield non-manifold meshes with broken topology, while GS-Pull [60], which extracts surfaces from a learned SDF, tends to produce overly smooth geometry.

Additional comparisons of surface reconstruction on the DTU dataset [18] are shown in Fig. 8. Given the relatively simple geometry of the target objects, most methods achieve visually complete reconstructions. However, the differences lie in the surface quality and detail. Our method demonstrates superior performance in handling reflective surfaces and preserves fine-scale features, producing reconstructions that are both smoother and more accurate, thereby improving visual fidelity and geometric realism.

C More Ablation Results

To thoroughly evaluate the effectiveness and individual contributions of the components proposed in our method, we further conducted a series of ablation studies. These experiments systematically

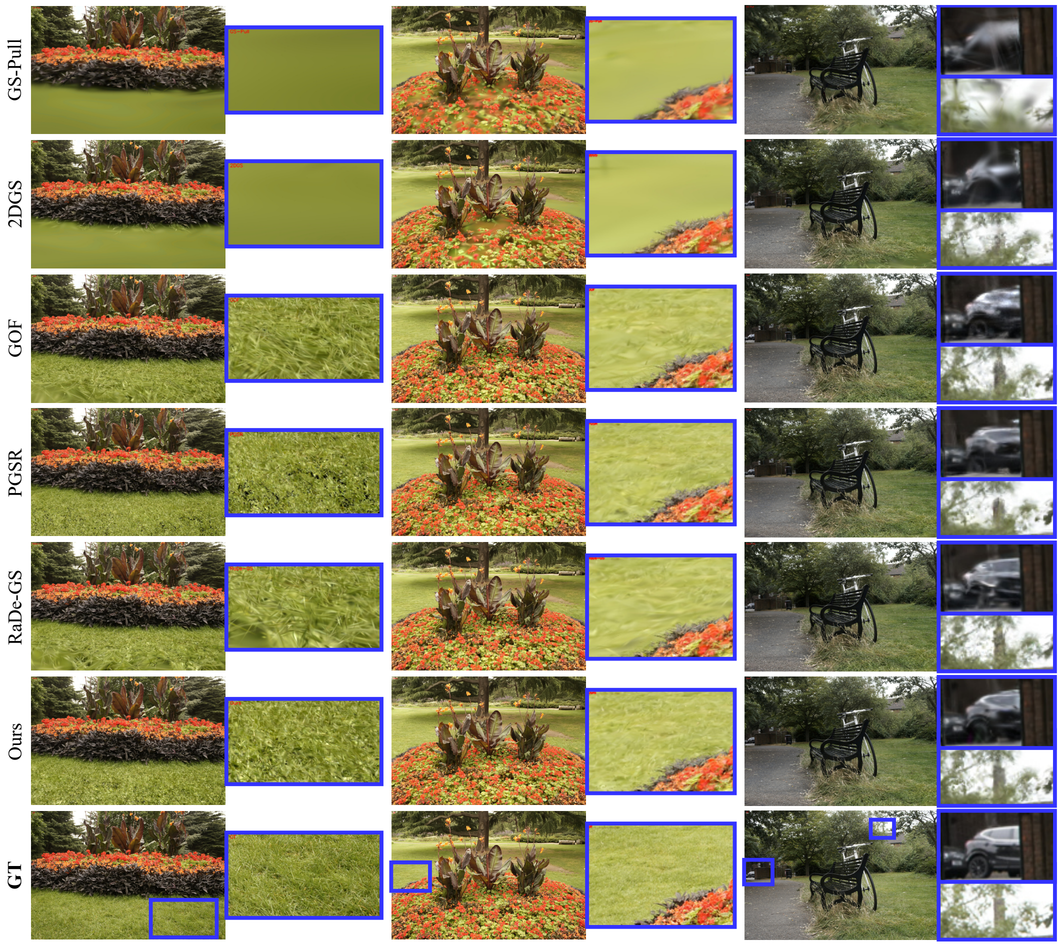

Figure 6: Comparison of our method with prior Gaussian-based approaches on novel view synthesis on the Mip-NeRF 360 dataset. Our method produces high-quality renderings with clear details, while previous methods generate blurry results in complex regions such as grass, vegetation, and cars, as highlighted in the framed areas.

replace or remove specific modules and vary key hyperparameters to assess their impact on overall performance. The quantitative results are reported in Table 7.

(1) *Patch size in* $\mathcal{L}_p$. In our multi-view photometric alignment loss $\mathcal{L}_p$, we use a default patch size of 7×7. We also evaluate alternative patch sizes, including 3×3, 5×5, and 9×9. As shown in Table 7, smaller patches lead to suboptimal performance due to limited spatial context. Larger patches, such as 9×9, offer no significant improvement while increasing computational cost. This confirms that our chosen patch size achieves a good balance between accuracy and efficiency.

(2) *Gaussian Initialization.* Similar to prior works, we use COLMAP [36] to generate a sparse point cloud from the input images, which serves as the initialization for our 3D Gaussians. To evaluate the impact of initialization quality, we experiment with randomly generated point clouds using two alternative strategies: (i) uniform sampling within a cube, and (ii) sampling points on a spherical surface. As shown in Table 7, both alternatives degrade the final reconstruction quality compared to COLMAP initialization. However, the spherical initialization performs better than the cube sampling, likely due to its more uniform coverage and approximate enclosure of the scene geometry.

(3) *Threshold in* $\mathcal{L}_{ns}$. Our normal-based geometric alignment module incorporates a smoothing term $\mathcal{L}_{ns}$, with a threshold set to 0.01 by default. This threshold helps preserve sharp edges while reducing noise in low-curvature regions. We conduct ablations by testing alternative values, including 0.001 and 0.03. The results show that excessively small values (*e.g.*, 0.001) overly constrain normal variation, leading to surface oversmoothing, while large values (*e.g.*, 0.03) reduce the loss's regularization effect. Our default setting (0.01) strikes a good balance and achieves the best performance.

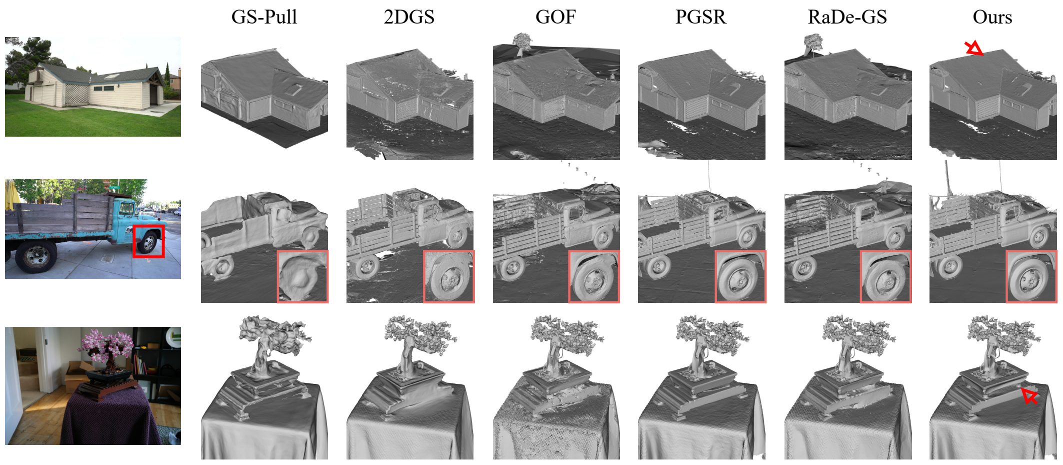

Figure 7: Visual comparison of surface reconstruction results on the TNT (first two rows) and Mip-NeRF 360 (third row) datasets. Our method produces more complete and accurate surfaces with clearer object boundaries and fewer artifacts, effectively handling challenges such as complex lighting conditions and ambiguous geometric structures.

Table 6: Quantitative results on the Mip-NeRF 360 dataset. We report PSNR, SSIM, and LPIPS for each scene. All results are from our own runs using official code, except for 3DGS, where we use the authors' pre-trained models. The best results are highlighted as 1st, 2nd and 3rd.

	bicycle	flowers	garden	stump	treehill	room	counter	kitchen	bonsai	**Mean**
PSNR $\uparrow$										
3DGS [21]	25.17	21.45	27.18	26.56	22.30	31.34	28.89	30.71	31.98	27.29
2DGS [16]	24.71	21.03	26.61	26.11	22.27	30.72	28.08	30.30	31.24	26.79
GS-Pull [60]	24.19	20.56	26.08	25.24	22.60	30.84	26.41	26.06	29.10	25.68
RaDe-GS [57]	25.56	21.67	27.39	27.10	22.34	30.84	28.73	31.26	31.73	27.40
GOF [56]	25.44	21.59	27.27	26.93	22.40	30.42	28.62	30.64	31.50	27.20
PGSR [4]	25.66	21.52	27.49	26.98	22.29	30.06	28.31	30.80	31.55	27.18
Ours	25.89	21.71	27.69	27.33	22.38	30.73	28.77	31.09	31.93	27.50
SSIM $\uparrow$										
3DGS [21]	0.762	0.602	0.861	0.770	0.631	0.916	0.905	0.923	0.938	0.812
2DGS [16]	0.729	0.569	0.838	0.753	0.614	0.906	0.892	0.915	0.929	0.794
GS-Pull [60]	0.660	0.498	0.776	0.661	0.617	0.896	0.846	0.838	0.897	0.743
RaDe-GS [57]	0.793	0.640	0.870	0.801	0.648	0.916	0.905	0.924	0.939	0.826
GOF [56]	0.786	0.634	0.864	0.790	0.641	0.911	0.900	0.915	0.935	0.820
PGSR [4]	0.793	0.635	0.873	0.798	0.660	0.927	0.914	0.932	0.945	0.831
Ours	0.803	0.648	0.877	0.810	0.660	0.930	0.919	0.934	0.948	0.837
LPIPS $\downarrow$										
3DGS [21]	0.216	0.341	0.115	0.219	0.328	0.223	0.204	0.130	0.208	0.220
2DGS [16]	0.276	0.380	0.149	0.263	0.381	0.243	0.230	0.146	0.228	0.255
GS-Pull [60]	0.331	0.443	0.223	0.362	0.414	0.239	0.269	0.248	0.242	0.308
RaDe-GS [57]	0.176	0.286	0.103	0.190	0.279	0.218	0.205	0.130	0.204	0.199
GOF [56]	0.182	0.282	0.109	0.197	0.282	0.221	0.205	0.137	0.200	0.202
PGSR [4]	0.186	0.264	0.103	0.192	0.273	0.180	0.172	0.113	0.169	0.184
Ours	0.169	0.258	0.098	0.177	0.254	0.175	0.164	0.109	0.163	0.174

The run-time cost of each loss. We provide the runtime of each loss ablation on the TNT dataset in Table 8. With all losses enabled, the training takes 20.9 minutes on RTX 4090. Removing $\mathcal{L}_f$ (feature alignment) reduces training time to 15.9 minutes, yielding a $\sim$ 5-minute saving, while still preserving strong performance. By comparison, removing $\mathcal{L}_p$ (photometric alignment) saves more time (8.3 min) but sacrifices more reconstruction quality. When only $\mathcal{L}_I$ (image reconstruction) is used, our method has a lower time cost than the original 3DGS (5.9 min *vs.*7.5 min) and provides better F1-score results than 3DGS (0.13 *vs.*0.09).

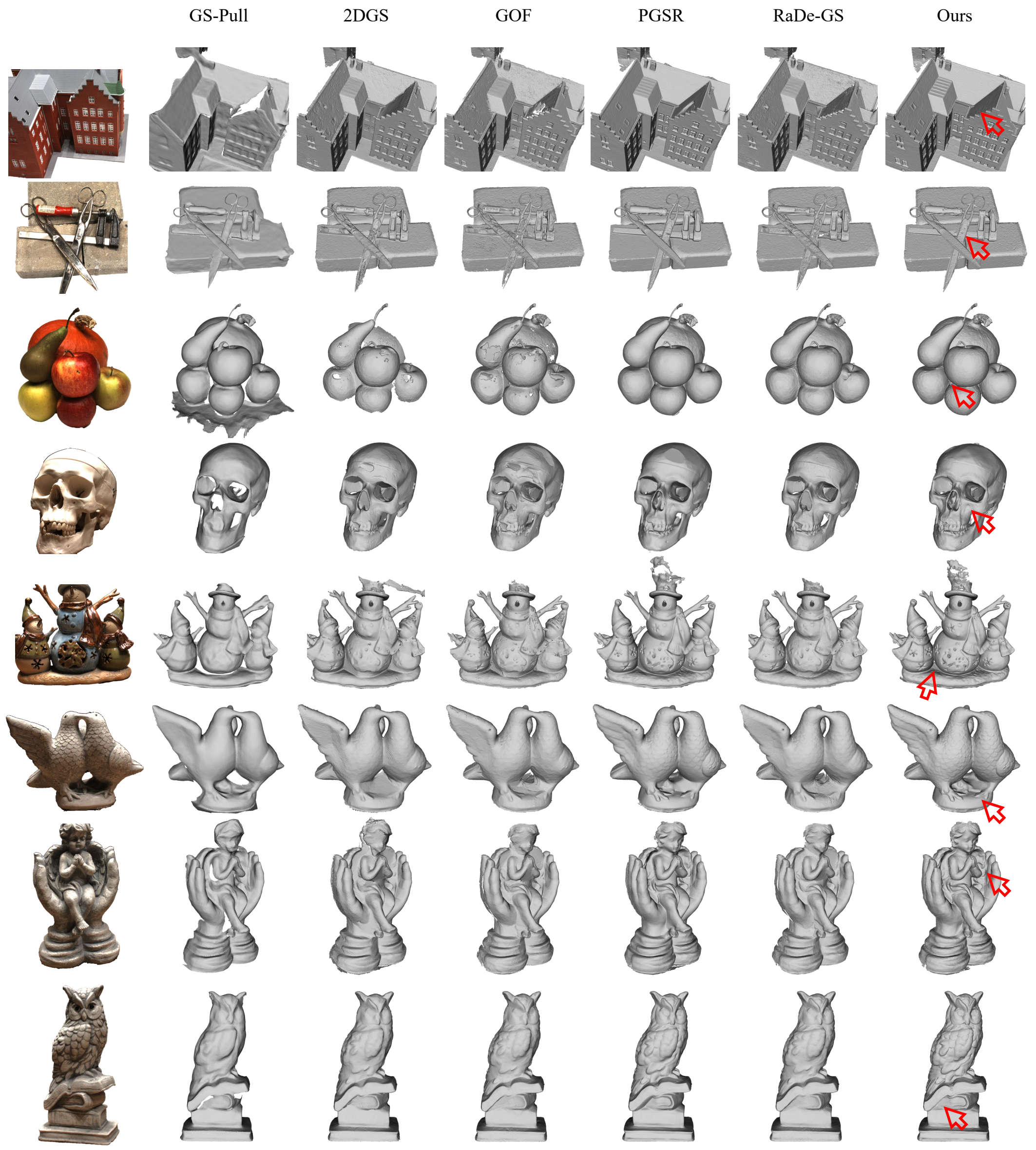

Figure 8: Visual comparison of surface reconstruction results on the DTU dataset. Our method demonstrates superior capability in handling reflective materials and recovering fine-grained surface geometry. Please zoom in on the digital version for detailed inspection.

D More Discussion

More clarification of loss $\mathcal{L}_f$. We agree that $\mathcal{L}_f$ brings modest improvements on the TNT dataset. $\mathcal{L}_f$ improves general robustness and is beneficial in scenes with consistent lighting or texture. It improves robustness under low-texture or lighting-variant scenes, which is not fully reflected by the F1-score metric alone. Removing it often leads to over-smoothed meshes in challenging regions. Moreover, the contribution of the feature alignment loss $\mathcal{L}_f$ is more significant on the DTU dataset: removing $\mathcal{L}_f$ increases Chamfer distance from 0.49 to 0.52. We attribute this to dataset characteristics. DTU scenes benefit more from learned feature-level consistency due to cleaner lighting and smoother structure. TNT contains diverse indoor/outdoor scenes and severe lighting variation. The expressive power of off-the-shelf image features is limited, which may partially underutilize $\mathcal{L}_f$. We plan to explore stronger feature extractors in future work.

Table 7: Ablations on the TNT dataset.

	Precision ↑	Recall ↑	F1-score ↑
3×3	0.49	0.58	0.52
5×5	0.50	**0.60**	**0.54**
9×9	**0.51**	**0.60**	**0.54**
Random cube init	0.44	0.59	0.49
Random sphere init	0.47	0.59	0.51
$\tau = 0.001$	0.50	**0.60**	0.53
$\tau = 0.03$	0.50	0.59	0.53
Ours	**0.51**	**0.60**	**0.54**

Table 8: Ablation study on different loss terms.

	3DGS	Only $\mathcal{L}_I$	w/o $\mathcal{L}_{nc}$	w/o $\mathcal{L}_{ns}$	w/o $\mathcal{L}_p$	w/o $\mathcal{L}_f$	Full
F1-score ↑	0.09	0.13	0.52	0.51	0.50	0.53	**0.54**
Time (min) ↓	7.5	**5.9**	20.3	17.1	12.6	15.9	20.9
Time Gap	-	15.0	0.6	3.8	8.3	5.0	0.0

Clarification of performance gains and training time. (a) While our training time is longer than that of vanilla 3DGS, the added cost stems from the newly introduced geometry supervision and view-alignment mechanisms, which directly improve surface accuracy. These components are essential for multi-view reconstruction, especially in challenging scenes. While the gains over the latest baselines may appear modest in some scenes, these differences represent noticeable geometric improvements. (b) Moreover, increased time is a common trade-off among geometry-enhanced 3DGS variants, where introducing geometric priors and view constraints improves reconstruction quality at the cost of runtime (see Tables 1 and 2 of main paper). The original 3DGS remains the fastest due to its lightweight constraints, but also exhibits the least accurate geometry. In future work, we plan to explore adaptive Gaussian pruning to further reduce training cost without sacrificing accuracy. (c) Our goal of this work is not to accelerate or compress 3DGS, but to enhance its geometric representation. On DTU, we reduce Chamfer distance from 1.96 to 0.49; on TNT, we raise F1-score from 0.09 to 0.54, a substantial improvement over 3DGS; on Mip-NeRF 360, we also observe consistent gains across all metrics. As shown in Table 8, when only using edge-aware image reconstruction loss $\mathcal{L}_I$ on TNT, our method runs faster than 3DGS (5.9 min *vs.*7.5 min) and achieves a higher F1-score (0.13 *vs.*0.09). (d) In terms of practicality, our training time on TNT ($\sim$ 20 min) is production-viable for many offline applications. In contrast, neural implicit SDF methods often require $10+$ hours of training. We believe that accuracy is often prioritized over marginal runtime gains in such scenarios, with the rapid advancement of hardware.

More explanation of the ablations. In all ablation studies, we adopt a consistent and controlled protocol to ensure fair comparisons: (a) When ablating a loss term (*e.g.*, $\mathcal{L}_p$ or $\mathcal{L}_f$), we drop the corresponding term entirely from the optimization objective. (b) The remaining loss weights are kept unchanged, and we do not re-normalize or re-scale other terms to preserve consistent training dynamics. (c) All experiments are conducted with the same number of training iterations and identical optimization settings (*e.g.*, learning rate, batch size, data split). (d) For the hyperparameter ablations, we similarly vary only the parameter under study (*e.g.*, source view count, threshold value), while keeping all other parameters and training settings fixed. This ensures that any performance change can be attributed directly to the presence or absence of the specific loss being tested.

The difference from PGSR. Firstly, we would like to clarify that our method is not a direct extension of PGSR with feature alignment added on top. Instead, our framework introduces a new combination of single-view alignment (including edge-aware image reconstruction and normal-based geometry supervision) and multi-view alignment (photometric and feature alignment), which are structurally and conceptually different from PGSR. As shown in the ablations of Table 4 of main paper, even without feature alignment, our method already achieves better performance than PGSR. This demonstrates that our core design, particularly single-view and multi-view photometric supervision, is effective. The additional feature alignment further improves the results, validating our design choices. Even when the improvement appears numerically small on the TNT dataset, reaching the same or slightly higher performance than PGSR still represents a state-of-the-art level. Moreover, on the DTU dataset, the contribution of the feature alignment loss $\mathcal{L}_f$ is more significant. Removing $\mathcal{L}_f$

increases Chamfer distance from 0.49 to 0.52, which is still better than PGSR's reported 0.53. This shows that our method offers consistent geometric improvement across datasets. Finally, our method also outperforms PGSR on the Mip-NeRF 360 dataset for novel view synthesis, as shown in Table 3 of main paper. We achieve better scores on all metrics, indicating that our method is not merely comparable to PGSR but surpasses it in reconstruction/synthesis quality and generalizability across varied benchmarks.

Our Multi-View Photometric Alignment differs from PGSR's in several key aspects: (a) Optimization complexity: PGSR couples its photometric consistency loss with geometric consistency regularization, which minimizes forward-backward reprojection error of neighboring views, resulting in more variables participating in network backpropagation and a more complex gradient computation. In contrast, our method simplifies the backpropagation path by aligning Gaussian orientations first and then applying image/feature-level constraints, which leads to better optimization efficiency. (b) Multi-view formulation: PGSR computes photometric consistency between one reference view and one source view. Our framework uses one reference view and multiple source views (three by default, varied in ablation), which introduces richer multi-view constraints and improves geometric supervision. (c) Gaussian flattening: PGSR flattens 3D Gaussians using scale regularization of the Gaussian ellipsoid to mimic planar surfaces. In our method, we found this strategy ineffective through ablation in Table 4 of main paper, and therefore designed our supervision differently. (d) Efficiency and performance: As shown in Table 2 of main paper, our method achieves better reconstruction quality across more scenes on real outdoor scenes of the TNT dataset while requiring less training time than PGSR. (e) Occlusion modeling: We explicitly define a visibility term and occlusion weight to mitigate the effect of outlier pixels caused by occlusion or misalignment. We give their motivation and clearly show the derivation of their formulas in the method section.

E Limitations and Future Works

Despite achieving strong performance in both surface reconstruction and novel view synthesis, our method has several limitations that suggest promising directions for future work. First, our model assumes known camera poses as input, which may not always be available in real-world scenarios, especially when the number of input views is extremely sparse. Removing this requirement by exploring pose-free approaches [50, 59] is a compelling future direction that would increase the applicability of our method in unconstrained settings. Moreover, our method assumes a moderate number of input views. For extremely sparse-view settings, the multi-view consistency constraints may become less effective, reducing reconstruction quality. Second, the number of Gaussians grows significantly with the number of input views, potentially limiting the scalability of our method in large-scale or densely captured scenes. Designing more compact or adaptive Gaussian representations, or incorporating efficient pruning and regularization strategies, could help improve scalability without compromising reconstruction quality. Third, our method currently relies on a pre-trained feature extraction network to guide geometric alignment and ensure robustness under varying lighting and viewpoints. This dependence introduces limitations, particularly when the pre-trained features are suboptimal for specific domains or scenes. Exploring more reliable, possibly self-supervised, feature learning strategies or reducing reliance on external networks altogether would further improve robustness and generalization. Addressing these limitations can pave the way for more flexible, efficient, and generalizable Gaussian-based surface reconstruction frameworks.

F Potential Negative Social Impacts

While our method advances the state-of-the-art in surface reconstruction and novel view synthesis using 3D Gaussian Splatting, it is important to consider its potential negative social impacts. First, high-fidelity 3D reconstruction from multi-view imagery could be misused to reconstruct environments or individuals without consent. For instance, when applied to personal photos or public surveillance footage, our method may enable unauthorized digital replication of private spaces or identities. We strongly advocate for the ethical and consensual collection and use of input data. Second, enhanced geometry reconstruction and photorealistic novel view synthesis could be integrated into pipelines for generating synthetic scenes or manipulating real-world data. This could potentially contribute to the creation of deceptive media, raising concerns about misinformation, impersonation, or forgery. Preventative measures such as watermarking and provenance tracking should be considered in downstream applications. Third, although more efficient than dense volumetric methods, our approach still requires significant GPU resources for training and inference. As the field moves toward real-time and scalable 3D vision, energy consumption and environmental sustainability must remain part of the design considerations.