MVGSR: Multi-View Consistency Gaussian Splatting for Robust Surface Reconstruction

Chenfeng Hou[1] Qi Xun Yeo[2] Mengqi Guo[2] Yongxin Su[1]
Yanyan Li[2,✉] Gim Hee Lee[2]
[1]Beihang University [2]National University of Singapore
https://mvgsr.github.io

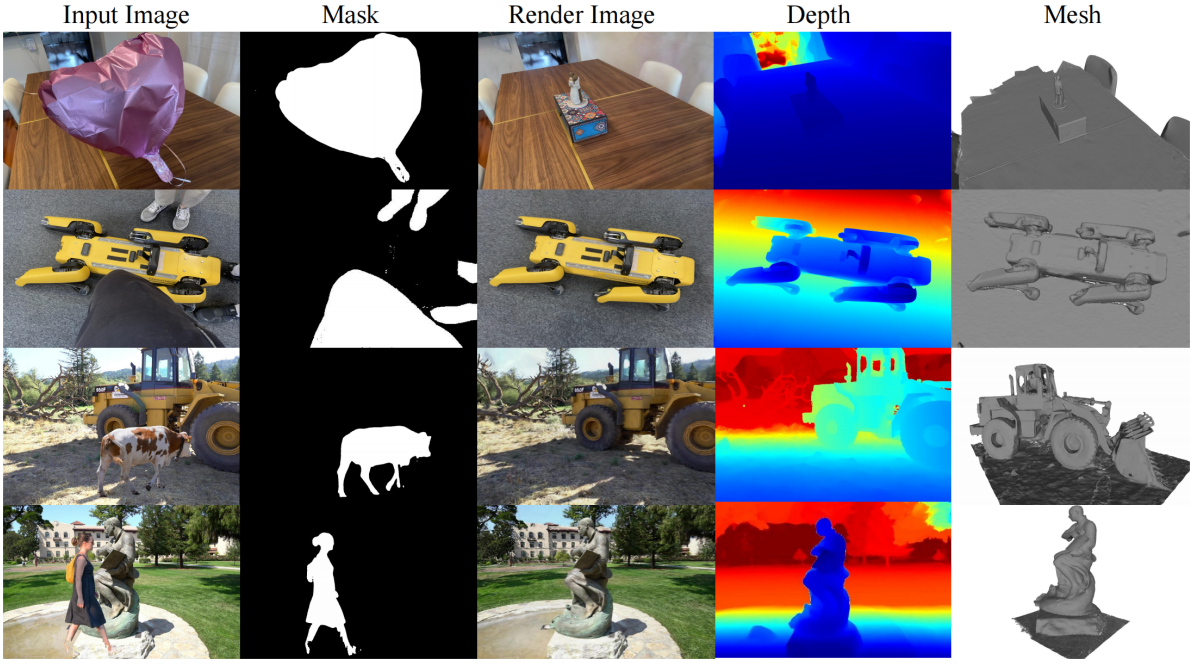


Figure 1. MVGSR Performance Across Datasets: RobustNeRF[27] (line 1), On-the-Go[24] (line 2), and TnT [15] (lines 3-4). When input images contain distractors, MVGSR generates distractor masks through multi-view feature contrast, effectively preventing gradient leakage and thereby achieving high-quality surface reconstruction with photorealistic rendering fidelity

Abstract

*3D Gaussian Splatting (3DGS) has gained significant attention for its high-quality rendering capabilities, ultra-fast training, and inference speeds. However, when we apply 3DGS to surface reconstruction tasks, especially in environments with dynamic objects and distractors, the method suffers from floating artifacts and color errors due to inconsistency from different viewpoints. To address this challenge, we propose Multi-View Consistency Gaussian Splatting for the domain of Robust Surface Reconstruction (**MVGSR**), which takes advantage of lightweight Gaussian models and a heuristics-guided distractor masking strategy for robust surface reconstruction in non-static environments. Compared to existing methods that rely on MLPs for distractor segmentation strategies, our approach separates distractors from static scene elements by comparing multi-view feature consistency, allowing us to obtain precise distractor masks early in training. Furthermore, we introduce a pruning measure based on multi-view contributions to reset transmittance, effectively reducing floating artifacts. Finally, a multi-view consistency loss is applied to achieve high-quality performance in surface reconstruction tasks. Experimental results demonstrate that MVGSR achieves competitive geometric accuracy and rendering fidelity compared to the state-of-the-art surface reconstruction algorithms. More information is available on our project page (this url).*

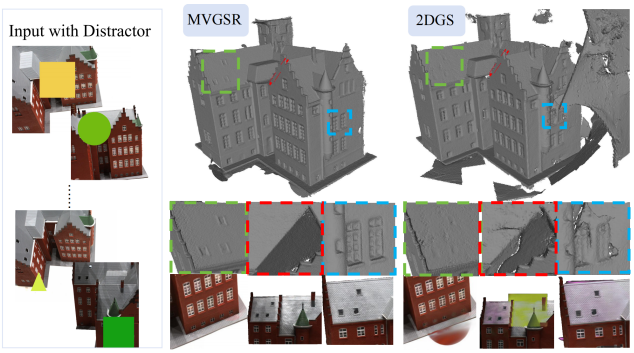


Figure 2. Surface reconstruction in scenes with distractors.

1. Introduction

In recent years, neural network-based implicit representations [5, 19, 26] of 3D scenes have gained significant popularity due to their impressive modeling capabilities and generalization power. More recently, 3D Gaussian-based representations [7, 13, 16] have emerged as a promising alternative, offering a new approach to addressing this challenge. Compared to neural implicit representations, 3D Gaussian-based scene representations exhibit faster processing and higher rendering quality.

Although 3D Gaussian-based scene representations achieve impressive visual results, they have limitations [4, 16] in accurately describing a scene's geometry. This deficiency, particularly in the context of robust surface reconstruction, significantly impacts the overall quality of 3D reconstruction. In casually captured scenes [39, 40], moving pedestrians, vehicles, and other objects in the images are unavoidable distractors. These distractors occlude the central objects of interest. As shown in Fig. 2, when there are viewpoint-dependent distractors, 3DGS-based surface reconstruction algorithms may model them as artifacts clustered in front of the camera lens or as viewpoint-dependent color representations attached to the object's surface. To address the issue of distractors during reconstruction, we propose a surface reconstruction algorithm that avoids these distractions, preventing artifacts while achieving competitive geometric accuracy and enhanced rendering capabilities.

Different to previous work [27, 29] using photometric residual decomposition to obtain distractor masks, our core insight is that distractors appearing in only a few images lack semantic consistency across different views, resulting in noticeable differences in features extracted by pre-trained base models. We leverage this observation to separate distractors from static scenes the early stage of training. Compared to iterative masks learned by multi-layer perceptrons (MLP) during training, masks obtained through multi-view comparison have two advantages: first, they distinguish distractors from high-frequency details, allowing continuous optimization of high-quality surface reconstruction. Second, strict mask boundaries prevent gradient leakage during Gaussian splitting, avoiding artifacts and viewpoint-dependent color errors in distractor regions.

To address the floating artifacts and ghosting caused by uncertain geometric relationships, we design a pruning measure based on multi-view contributions. We reset the transmittance of objects occluding the camera's view to mitigate the impact of occlusion on gradient flow. This strategy can remove floating artifacts while compressing the number of point clouds, with only a minimal decrease in rendering quality.

For high-precision optimization of surface reconstruction, we use a multi-view consistency loss function to enhance the reconstruction capability of Gaussian splats. This is achieved through structural and color consistency constraints on corresponding points from different views. In scenes with distractors, our method achieves optimal rendering quality and competitive reconstruction results. Our contributions can be summarized as follows:

- We propose a method using multi-view consistency to distinguish distractors from static objects before surface reconstruction, achieving strict distractor masking and significantly reducing floating artifacts and color errors.
- We introduce a new Gaussian pruning technique based on multi-view contributions to remove floating artifacts with minimal reduction in rendering capability.
- In scenes with distractors, we achieve high-fidelity rendering quality and high-precision reconstruction results. Our code and generated dataset will be released to the community.

2. Related Work

2.1. Traditional 3D Reconstruction

Traditional Surface Reconstruction methods [30, 31, 33, 38] can be roughly grouped into different classes based on their intermediate representations , such as point clouds [30], volumes [38], and depth maps[9]. These methods typically decompose the entire multi-view stereo (MVS) problem into several stages. First, a dense point cloud is extracted from the multi-view images via patch-based matching [20]. Then, the surface structure is constructed through either feature triangulation [17] or implicit surface fitting [8]. These methods are often affected by mismatching or noise introduced during the reconstruction pipeline.

2.2. Neural Surface Reconstruction

NeRF-based methods [1, 19] take 5D rays as input to predict density and color sampled in implicit space, leading to more realistic rendering results. Despite NeRF's impressive performance in 3D reconstruction, its implicit function

representation through volume rendering results in poor geometric accuracy and susceptibility to noise. To address these issues, methods such as NeuS [34], BakedSDF [35], and UNISURF [21] represent surfaces using signed distance functions (SDF) to achieve more accurate scene geometry. Meanwhile, Nerf2Mesh [32] introduces an iterative mesh refinement algorithm that adaptively adjusts vertex positions and volume density based on reprojection rendering error.However, while the NeRF-based framework exhibits powerful surface reconstruction capabilities, the stacked MLP layers impose limitations on inference time and representation power.

2.3. GS-based Surface Reconstruction

Different from neural surface reconstruction methods, GS-based approaches optimize point-based radiance fields, including 3D Gaussians, 2D Gaussians, and Gaussian surfels. SuGaR [6] extracts meshes from the positions of 3D Gaussians and introduces a regularization term to encourage Gaussians to align with the scene surface based on sampled 3D point clouds. While this alignment improves geometric reconstruction accuracy, the irregular shapes of 3D Gaussians pose challenges in modeling smooth geometric surfaces. 2DGS [7] achieves view-consistent geometry by collapsing 3D volumes into a set of 2D Gaussian. GOF [37] constructs Gaussian opacity fields and extracts 3D models directly from them. However, these 3DGS-based methods typically produce biased depth estimates and struggle with maintaining multi-view geometric consistency. To address the inconsistent issue in scene reconstruction, PGSR [2] flattens the Gaussian function into a planar shape based on the assumption that scenes compose of several smaller planar regions. Leveraging these 3D relationships, PGSR introduces a new depth computation strategy to accurately extract geometric parameters from Gaussian surfels. Instead of assuming planar surfaces, our method proposes a more general architecture for object reconstruction, which incrementally transforms inconsistent ellipsoids to Gaussian surfels during the densification process.

2.4. Distractor Removal

For reconstruction in non-static scenes, distractor removal is one of the most important task. There are two main strategies in dealing with this problem namely segmentation-based methods and heuristics-based methods. For the former, deep semantic or segmentation models are used to detect object with potential dynamic characteristics or to recognize static scenarios. Based on pre-trained models, DynaMoN [12] further utilized semantic maps to reconstruct dynamic scenes via Motion-Aware Fast and Robust Camera Localization for in dynamic NeRF. However, for the latter, approaches [3, 18, 27] make use of heuristics generated from multi-view geometry algorithms to separate dynamic objects from static scenes. Specifically, NeRF-W [18] assumes that most of transient objects are generally small during the NeRF training process, while RobustNeRF [27] tries to define transient objects from static background based on the photometric residuals during the optimization module. Compared to RobustNeRF, NeRF-HUGS [3] leverages on masks estimated based pre-trained model with residual mask obtained from RobustNeRF to achieve more accurate distractor masking performance.

3. Preliminary of Gaussian Splatting

3DGS [13] models 3D scenes by employing a set of 3D Gaussians $\{\mathcal{G}_i\}$. A Gaussian function defines each of these Gaussians at point $\boldsymbol{p}_i \in \mathcal{P}$:

$$\mathcal{G}_i(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma_i}) = e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)},$$

Each point $\boldsymbol{p}_i \in \mathcal{P}$ is centered at $\boldsymbol{\mu}_i \in \mathbb{R}^3$ with a corresponding 3D covariance matrix $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3\times3}$. This covariance matrix $\boldsymbol{\Sigma}_i$ is factorized into a scaling matrix $\boldsymbol{S}_i \in \mathbb{R}^{3\times3}$ and a rotation matrix $\boldsymbol{R}_i \in \mathbb{S}O(3)$:

$$\boldsymbol{\Sigma}_i = \boldsymbol{R}_i \boldsymbol{S}_i \boldsymbol{S}_i^\top \boldsymbol{R}_i^\top.$$

3DGS facilitates quick α-blending for rendering views. The transformation matrix W and intrinsic matrix $\boldsymbol{K}$ allow $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ to be converted into camera coordinates related to W, and subsequently projected into 2D coordinates using the following functions:

$$\boldsymbol{\mu}_i^{'} = \boldsymbol{K}\boldsymbol{W}[\boldsymbol{\mu}_i, 1]^\top, \quad \boldsymbol{\Sigma}_i^{'} = \boldsymbol{J}\boldsymbol{W}\boldsymbol{\Sigma}_i\boldsymbol{W}^\top\boldsymbol{J}^\top,$$

where $\boldsymbol{J}$ represents the Jacobian of the affine approximation for the projective transformation. The color $\boldsymbol{C} \in \mathbb{R}^3$ of a pixel $\boldsymbol{u}$ can be rendered using α-blending.:

$$\boldsymbol{C} = \sum_{i\in N} T_i \alpha_i \boldsymbol{c}_i, \quad T_i = \prod_{j=1}^{i-1}(1-\alpha_i),$$

Here, α_i is determined by evaluating $\mathcal{G}_i(\boldsymbol{u}|\boldsymbol{\mu}_i^{'}, \boldsymbol{\Sigma}_i^{'})$ and multiplying it with a learnable opacity associated with $\mathcal{G}_i$. The view-dependent color $\boldsymbol{c}_i$ is expressed using spherical harmonics (SH) from the Gaussian $\mathcal{G}_i$. T_i represents the cumulative opacity, and N denotes the number of Gaussians the ray intersects.

The center $\boldsymbol{\mu}_i$ of a Gaussian $\mathcal{G}_i$ can be transformed into the camera coordinate system as:

$$\left[x_i, y_i, z_i, 1\right]^\top = [\boldsymbol{W}|\boldsymbol{t}][\boldsymbol{\mu}_i, 1]^\top,$$

And previous methods [4, 11] in depth rendering under the current viewpoint can be denoted as:

$$\boldsymbol{D} = \sum_{i\in N} T_i \alpha_i z_i.$$

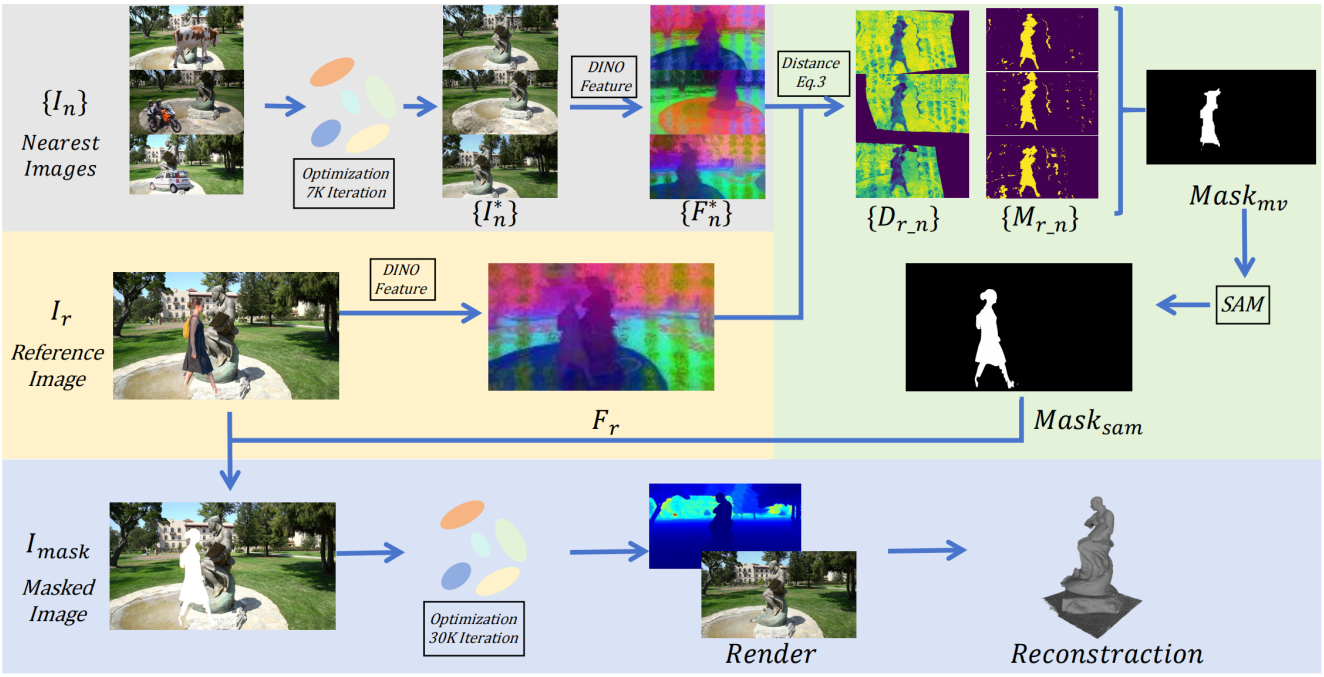


Figure 3. The detailed architecture of our MVGSR Framework. Images with distractors are fed to the system that makes use of multi-view consistency Gaussian Splatting algorithm to achieve robust surface reconstruction for non-static environments.

For 3D Gaussians, the direction of the minimum scale factor corresponds to the normal n_i of the Gaussian. The normal map under the current viewpoint is achieved through α-blending:

$$\boldsymbol{N} = \sum_{i \in N} T_i \alpha_i \boldsymbol{R}_c^T \boldsymbol{n}_i$$

where R_c is the rotation from the camera to the global world.

4. Methodology

Given a set of images of a scene containing outliers, the goal of our method is to achieve high-precision surface reconstruction while maintaining robust view rendering performance. In existing work on outlier masking [27, 29], mask estimation typically relies on photometric errors, but these methods cannot accurately distinguish between outliers and reconstruction details, leading to potential inaccuracies. To address this issue, we propose a novel method that leverages feature similarity across multiple views to differentiate outliers. Specifically, we utilize features extracted from a self-supervised 2D foundation model [22] for feature extraction. With fewer training iterations, we can generate a rough scene representation to compute the mapping between the original view and adjacent views. By comparing feature similarities, we can derive an initial mask that helps identify outliers. To refine the mask boundaries and enhance its accuracy, we sample points from the mask and apply a basic segmentation model to obtain precise mask results. Retraining with this refined mask helps mitigate gradient leakage during the optimization process. Additionally, to further reduce artifacts, we apply pruning based on multi-view contributions, ensuring the removal of unreliable information from the reconstruction. The process of obtaining the initial mask will be detailed in Sec. 4.1, while Sec. 4.2 will discuss multi-view pruning, and Sec. 4.3 will describe the loss function used in the optimization.

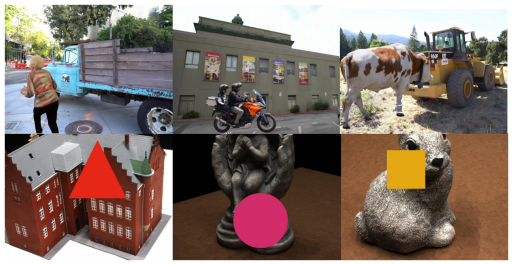


Figure 4. Example of distractors of DTU-Robust and TnT-Robust dataset.

4.1. Distractor Detection

Similar to the traditional 3D Gaussian Splatting methods [13, 16], scenes containing distractors are processed in the same initialization stage. This stage leverages sparse point clouds and camera parameters generated from structure-from-motion (SfM) techniques. Using this initial

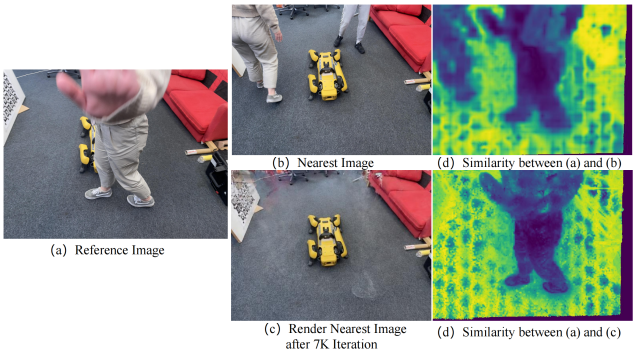


Figure 5. Reference image (a), nearest image after 7000 iteration (b), rendered image (c), feature similarity map between reference and nearest viewpoints (d), and feature similarity map between reference and rendered images (e). The rendered image can remove some distracting objects, preventing interference with the features of the reference image.

setup, a rough scene reconstruction is performed, resulting in the computation of the initial depth map D, surface normals N, and an initial rendered image I^*. This initialization provides a foundational representation of the scene, which is then refined in subsequent stages of the method to improve accuracy and handle outliers effectively.

Then, three steps are proposed to achieve an effective distractor detection strategy. First, the Gaussian surfels observed from the viewpoint I_r are associated with the corresponding surfels detected from neighboring viewpoints I_n, based on the relative camera pose $\mathbf{R}, \mathbf{t}$ between the two viewpoints. For a pixel $\mathbf{p}_r$ on image I_r, the corresponding pixel $\mathbf{p}_n$ in a neighboring viewpoint I_n can be calculated using the homographic relationship in Eqn. 1.

$$\mathbf{H}_{nr} = \mathbf{K}_n \mathbf{R}_{nr} \left(\mathbf{I} + \frac{1}{d_r} \cdot \mathbf{t}_{rn} \mathbf{n}_r^\top \right) \mathbf{K}_r^{-1} \tag{1}$$

$$\mathbf{p}_n = \mathbf{H}_{nr} \mathbf{p}_r \tag{2}$$

This allows us to establish correspondences between pixels across different viewpoints, which is essential for identifying and handling potential distractors in the scene.

For a given reference image I_r and an initially rendered neighboring viewpoint I^*, we use DINOv2[22] to extract image features following the Nerf-on-the-Go[25] method, resulting in feature maps F_r and F_n^*. To remove the transient distracting objects that may also be present in the neighboring viewpoints which misleads the feature similarity calculation, we use the initially rendered image I^* instead of the original neighboring viewpoint image I_n. This issue is illustrated in Fig. 5, where distracting objects in the original neighboring viewpoint image I_n interferes with accurate feature matching and distractor detection.

Based on the two associated points, we estimate the feature distance between them shown in Eqn. 3. The number of channels in the feature map is 384. To compute the feature vectors for two points on the corresponding feature maps, we first apply bilinear interpolation to obtain the features $F_r(\mathbf{p}_r), F_n^*(\mathbf{p}_n)$, respectively. This is necessary because the feature map is downsampled by a factor of 14 from the original image, and bilinear interpolation allows us to extract the feature values at specific pixel locations. Pixels with a distance lower than a certain threshold δ_{near} are used as a mask for the adjacent view. Then, the computation process of the distance is denoted as:

$$distance(\mathbf{p}_r, \mathbf{p}_n) = abs\left(\frac{F_r(\mathbf{p}_r) \cdot F_n^*(\mathbf{p}_n)}{|F_r(\mathbf{p}_r)||F_n^*(p_n)|}\right) \tag{3}$$

where M_n is defined based on $distance_{rn}(\mathbf{p}_r) < \delta_{near}$.

Second, each mask M_n obtained from the first step contains many incorrect segmentations due to rough geometric estimates and noise. Using the multi-view mapping relationships, we calculate the visibility of the 3D spatial points corresponding to the pixels classified as clutter in the mask and the current view. If a pixel is identified as clutter by at least two visible adjacent views, it is retained as the final segmentation result $Mask_{mv}$.

Third, the quality of segmentation in some boundary regions of the coarse mask is not very accurate. Therefore, we continue to refine these regions based on the segment anything model (SAM) [14] which is a prompt-based segmentation model. We perform uniform sampling on the $Mask_{mv}$ to obtain positive sample points on the clutter and negative sample points in the background region. These are computed with the original image using the outputs of SAM, resulting in a clutter mask $Mask_{\text{sam}}$ with precise edges.

$$Mask_{sam} = SAM(I_r, \mathcal{P}, \mathcal{N}) \tag{4}$$

4.2. Multi-view Pruning

In the process of reconstructing clutter, the optimization often causes Gaussians to cluster near the camera. These floaters are typically incorrect models of viewpoint-dependent phenomena for clutter that only exists for a few frames. Once these floaters appear during optimization, they force the line of sight to reach the transmittance limit prematurely, preventing gradient propagation. 3DGS is to reset the opacity of all Gaussians every few iterations, using it as a control mechanism. This allows gradients to flow again and prunes Gaussians that cannot regain higher opacity.

However, in regions with clutter, resetting opacity is not fully effective. First, due to the presence of masks, this area tends to split into more Gaussians after opacity reset. For masks that cannot be perfectly segmented, this will lead to further aggravation of floaters. Therefore, we propose multi-view contribution-based pruning (MV-Prune). We de-

fine multi-view contribution as:

$$\mathbf{C}_{MV}(p) = \sum_{\mathbb{V}_k \in \mathbb{V}} \left(\sum_{p \in \mathbb{V}_k} \alpha_{i(p)} \prod_{j=1}^{i(p)-1} (1 - \alpha_j) > \delta_{opacity} \right) \tag{5}$$

where V_k is the training viewpoint, and p is the Gaussian for which contribution needs to be calculated. The contribution of Gaussian p for viewpoint V_k is measured by the cumulative transmittance of the pixel associated with Gaussian p. When the cumulative transmittance along the ray exceeds the threshold δ_{opacity}, the contribution of Gaussian p in this viewpoint is incremented by one.

When the opacity of a Gaussian in a certain view exceeds a threshold, the contribution is raised by one. Once $C_{\text{MV}}(p)$ exceeds the threshold δ_{prune}, its transmittance is reset to a lower value to allow gradients to flow again. Experimental results show that MV-Prune effectively handles floaters, compressing by 60% with comparable rendering quality.

4.3. Multi-view Consistency

To enhance geometric consistency, we use photometric consistency constraints based on neighboring patches, similar to MVS algorithms. We use a homography matrix to compute the 11×11 pixel patch P_r around pixel $\mathbf{p}_r$, mapping it to the corresponding region P_n in the neighboring view. To measure consistency, we use the normalized cross correlation (NCC) [36] coefficient as a loss metric:

$$\boldsymbol{L}_{mv} = \frac{1}{V} \sum_{\boldsymbol{p}_r \in V} \left(1 - NCC\left(\boldsymbol{I}_r\left(\boldsymbol{p}_r\right), \boldsymbol{I}_n\left(\boldsymbol{p}_n\right)\right)\right). \tag{6}$$

To focus on these areas with reconstruction errors, we calculate per-pixel geometric reconstruction accuracy weights. This is done by reprojecting the corresponding pixel $\mathbf{p}_n$ from the neighboring view back using the homography matrix to obtain $\mathbf{p}_n^{'}$, then calculating the reprojection error and weight. This weight can also handle out-of-bounds and potential occlusion issues in different views.

$$\boldsymbol{E}_{repro} = \frac{1}{V} \sum_{\boldsymbol{p}_r \in V} \left\| \boldsymbol{p}_r - \boldsymbol{H}_{rn} \boldsymbol{p}_n \right\| \tag{7}$$

$$\boldsymbol{w}_{repro} = \frac{1}{1 + \boldsymbol{E}_{repro}} \tag{8}$$

Unlike MVS algorithms, we do not directly optimize the reprojection error, as the lack of neighborhood information can impair the gradient propagation of Gaussian errors. Using color error alone is sufficient for depth optimization.

Another loss function is the regularization term L_s that minimizes the shortest axis, modeling the Gaussian as a thin surface. The image reconstruction loss is L_{rgb}. In summary, the loss function is:

$$\boldsymbol{L} = L_{rgb} + \lambda_1 L_s + \lambda_2 w_{repro} L_{mv} \tag{9}$$

For the surface loss ,We set λ_1 = 100. For the geometric loss, we set λ_2 = 0.2.

5. Experiments

In this section, we present both qualitative and quantitative comparisons of our method with state-of-the-art Gaussian Splatting reconstruction approaches [2, 7] on public datasets.

5.1. Implementation

All experiments were conducted on a single NVIDIA 4090 GPU. The maximum number of nearest images in multi-view scenarios was set to 8, with a maximum angular difference of 60 degrees between adjacent view cameras and a maximum distance of 1.5. The novel view rendering task was performed using the training results from $7,000$ iterations, and the corresponding relationships between adjacent views were calculated using Eqn. 3. When computing the multi-view masks in Sec. 4.1, the cosine similarity threshold δ_{near} is set to 0.5. For SAM prompt points, 20 segmentation prompt points and 1 exclusion prompt point are used, with the segmentation results being considered valid if the top 10 votes exceeded 2, serving as the refined mask segmentation results. The masks are then used for re-training, resulting in high-quality surface reconstruction after $30,000$ iterations. In the multi-view pruning phase, the transmittance contribution was set to 0.5, and only views with contributions exceeding the threshold of 8 were retained.

5.2. Datasets and Metrics

Datasets. To evaluate reconstruction and rendering performance, we employ the DTU [10] and TnT [15] datasets with ground truth 3D models, containing 15 object-centric scenes and 6 large-scale outdoor scenarios respectively. To address their limitation in dynamic elements, we introduce DTU-Robust and TnT-Robust (publicly released), incorporating distractors as shown in Fig. 4.

In DTU-Robust, random geometric shapes (squares, circles, triangles) of varying sizes are superimposed on training images at rates $r = 0.3, 0.5, 0.8, 1.0$ to control distractor density. TnT-Robust enhances real-world fidelity by injecting distractors (pedestrians, vehicles, animals) from DAVIS 2017 [23] into static scenes. Both extended datasets enable comprehensive robustness testing under distractors while preserving original reconstruction benchmarks.

Metrics for rendering and reconstruction evaluations. In our experiments, rendering quality is measured via Peak Signal-to-Noise Ratio for novel views. Surface reconstruction uses Chamfer Distance (for DTU) and F1 score (for TnT), as is commonly done following the methods in this area. The bidirectional CD evaluates point cloud alignment,

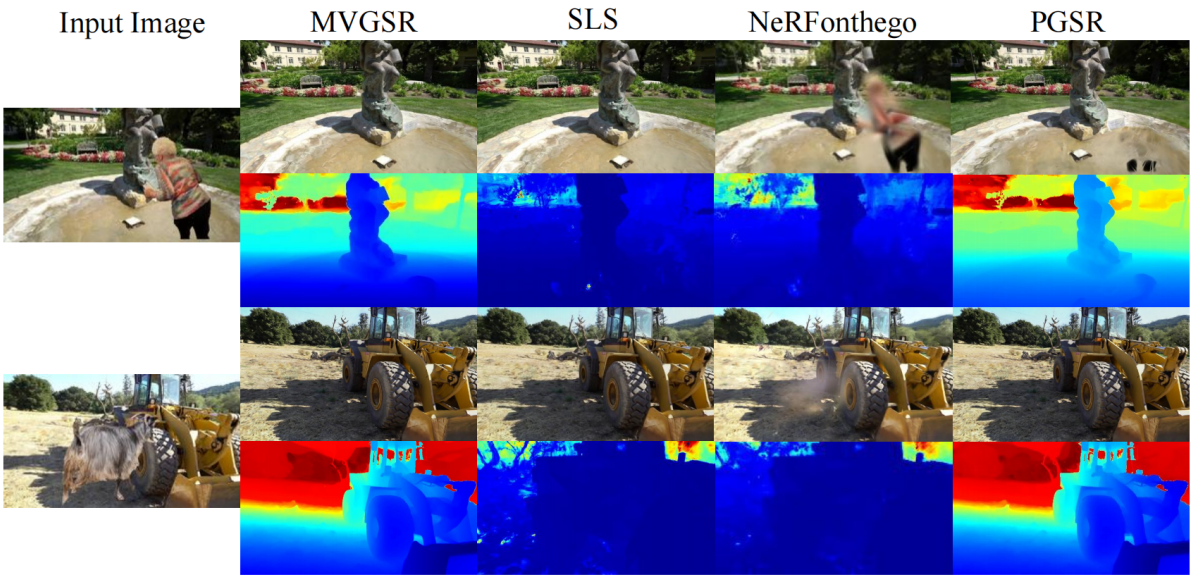


Figure 6. Comparison of view rendering on the TnT-Robust dataset.

	scan	Truck	Caterpillar	Barn	Meetingroom	Ignatius	Courthouse	Avg.
F1↑	PGSR [2]	0.57	0.37	0.57	0.30	0.64	0.18	0.44
	SLS [29]	0.48	0.30	0.43	0.24	0.57	0.15	0.36
	NeRFonthego [24]	0.37	0.22	0.28	0.17	0.49	0.11	0.27
	MVGSR	0.60	0.42	0.59	0.34	0.73	0.18	0.48

Table 1. Quantitative results of reconstruction performance on the TnT-Robust dataset.

while F1 quantifies geometric consistency by balancing precision and recall through threshold-based surface distance analysis.

5.3. Reconstruction

In this section of the paper, we evaluated the reconstruction capabilities of the distractor-free algorithms SLS[28] and NeRF-Onthego[24], as well as the surface reconstruction algorithm PGSR[2], on the TnT-Robust dataset. As shown in Table 1, on TnT-Robust, we outperformed PGSR in F1 scores by 0.03 (Truck), 0.05 (Caterpillar), and 0.09 (Ignatius), highlighting robustness against dynamic interference. On DTU-Robust, our method achieved a mean CD score of 0.42, surpassing 2DGS and PGSR by 76.45% and 60.31% on average, with consistent performance across occlusion rates and scenes.

Next, we present a qualitative comparison with the baselines on the DTU-Robust dataset Our method demonstrates superior reconstruction quality across both DTU-Robust and TnT-Robust datasets compared to baseline models (2DGS, PGSR, SLS, NeRFonthego). On DTU-Robust (Fig. 8), our outputs feature smoother surfaces with fewer holes/distractors, notably resolving complex geometries like the rabbit neck (scan 55) without fragmentation. In contrast, 2DGS loses fine details, while PGSR introduces scattered artifacts. For TnT-Robust (Fig. 6), our multi-view consistent representation ensures geometric continuity, outperforming SLS (distractor-free but artifact-prone) and PGSR (discontinuous surfaces). Key advantages include detail preservation, artifact suppression, and robustness to challenging scenes. This highlights our method's ability to balance structural accuracy with surface coherence under diverse real-world conditions.

5.4. View rendering

We demonstrate the rendering performance on the DTU-Robust dataset and compare it with PGSR and 2DGS. As shown in Table 2, our method consistently achieves higher PSNR across different scenes. Please note that the values here are the average results at different rates, verifying the stability of the performance in random scenes. Particularly in the scan83 scene, our method outperforms 2DGS and PGSR by 9.67 dB and 8.88 dB in PSNR, respectively. On average, our method achieves 35.58 dB in PSNR and demonstrates superior performance compared to the baseline methods. The higher results across scenes indicate that our method exhibits robustness and strong performance in novel view synthesis.

	scan	24	37	40	55	63	65	69	83	97	105	106	110	114	118	122	Avg.
CD↓	PGSR [2]	0.53	1.00	0.51	0.36	1.14	0.61	0.49	0.91	0.62	0.59	0.47	0.46	0.32	0.37	0.35	0.61
	2DGS [7]	0.52	0.86	0.60	0.45	1.12	0.99	0.82	1.37	1.19	0.69	0.71	0.70	0.41	0.70	0.56	0.77
	MVGSR	**0.34**	**0.51**	**0.30**	**0.30**	**0.44**	**0.52**	**0.45**	**0.63**	**0.59**	**0.56**	**0.35**	**0.38**	**0.29**	**0.34**	**0.35**	**0.42**
PSNR↑	PGSR [2]	31.34	26.72	29.88	31.81	32.66	31.27	30.94	32.02	30.37	32.76	34.94	33.58	32.16	36.41	35.71	32.15
	2DGS [7]	30.31	28.73	27.87	26.65	33.52	33.66	31.09	31.23	33.22	30.99	30.89	32.31	32.16	33.58	35.87	31.47
	MVGSR	**33.06**	**29.22**	**32.52**	**34.29**	**37.01**	**34.11**	**32.86**	**40.90**	**34.37**	**38.02**	**37.96**	**35.81**	**33.21**	**39.40**	**39.82**	**35.58**

Table 2. Quantitative results of reconstruction performance on the DTU-Robust dataset. **Bold** indicates the best results.

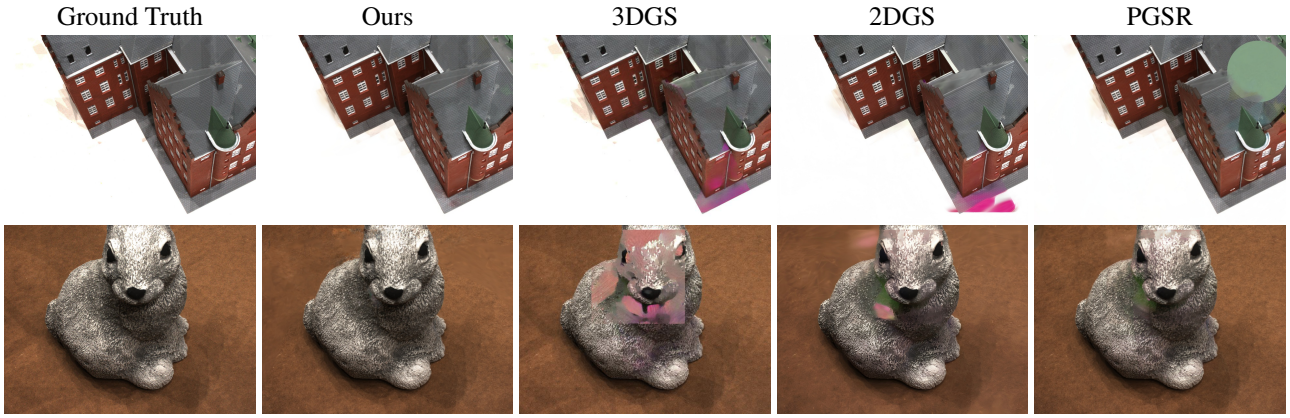


Figure 7. Comparison of view rendering on the DTU-Robust dataset. We select scan scene 24 (building), 55 (rabbit)

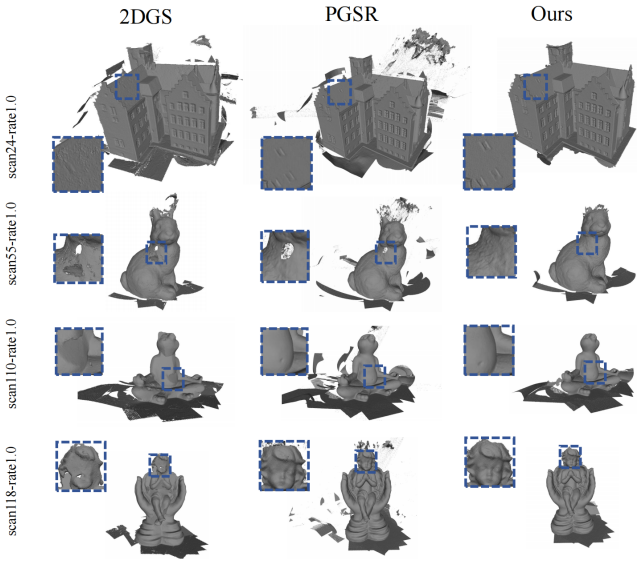


Figure 8. Comparison of reconstruction on DTU-Robust dataset.

We evaluated our method qualitatively in comparison to other competing methods as shown in Fig. 7. Our method clearly removes most distractors and distortion compared to the other methods. Our method clearly resembles the ground truth the closest without distractors and this validates the effectiveness of our pruning strategy in alias removing. For results on additional datasets, please refer to the supplementary materials.

5.5. Ablation Study

To demonstrate the contribution of each component, ablation experiments were conducted on DTU-Robust scan24 in Table 3. The multi-view Loss L_{mv} significantly improves surface reconstruction quality by reducing the Chamfer Distance from 1.63 to 0.37, despite a slight decrease in rendering performance. Mask and Mv-Prune effectively reduce artifacts, ultimately achieving high-fidelity rendering with superior quality preservation. Also, we compare the runtime and resource consumption of different algorithms on scan24 of DTU-Robust dataset. For more details, please refer to the supplementary materials.

mask_mv	mask_sam	L_mv	MV-Prnue	PSNR↑	CD↓
✓	✓			32.20	1.63
✓		✓		30.01	0.56
✓	✓	✓		30.75	0.37
✓	✓	✓	✓	33.06	0.34

Table 3. Ablation Study

6. Discussion and Conclusion

In this paper, we introduced Multi-View Consistency Gaussian Splatting for Robust Surface Reconstruction (MVGSR), a novel approach that addresses the common issues of floating artifacts and color errors in 3D Gaussian Splatting when applied to surface reconstruction. While 3DGS has gained popularity due to its high-quality rendering and fast training speeds, it is prone to distortions from distractors across different viewpoints,

which can severely affect the accuracy and quality of surface reconstruction. Our method mitigates these issues by focusing on multi-view consistency to identify and separate distractors from the static elements in the scene.

References

[1] 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. 2

[2] Danpeng Chen, Hai Li, Weicai Ye, Yifan Wang, Weijian Xie, Shangjin Zhai, Nan Wang, Haomin Liu, Hujun Bao, and Guofeng Zhang. Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *arXiv preprint arXiv:2406.06521*, 2024. 3, 6, 7, 8, 11, 12, 14, 15, 16

[3] Jiahao Chen, Yipeng Qin, Lingjie Liu, Jiangbo Lu, and Guanbin Li. Nerf-hugs: Improved neural radiance fields in non-static scenes using heuristics-guided segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19436–19446, 2024. 3

[4] 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. 2, 3

[5] Yan Di, Chenyangguang Zhang, Chaowei Wang, Ruida Zhang, Guangyao Zhai, Yanyan Li, Bowen Fu, Xiangyang Ji, and Shan Gao. Shapematcher: Self-supervised joint shape canonicalization segmentation retrieval and deformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21017–21028, 2024. 2

[6] 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. 3

[7] 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. 2, 3, 6, 8, 11, 12, 15, 16

[8] Slobodan Ilic and Pascal Fua. Implicit meshes for surface reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(2):328–333, 2005. 2

[9] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568, 2011. 2

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

[11] 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. 3

[12] Mert Asim Karaoglu, Hannah Schieber, Nicolas Schischka, Melih Görgülü, Florian Grötzner, Alexander Ladikos, Daniel Roth, Nassir Navab, and Benjamin Busam. Dynamon: Motion-aware fast and robust camera localization for dynamic nerf. *arXiv preprint arXiv:2309.08927*, 2023. 3

[13] 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. 2, 3, 4, 12, 16

[14] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 5

[15] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 36(4), 2017. 1, 6

[16] Yanyan Li, Chenyu Lyu, Yan Di, Guangyao Zhai, Gim Hee Lee, and Federico Tombari. Geogaussian: Geometry-aware gaussian splatting for scene rendering. *arXiv preprint arXiv:2403.11324*, 2024. 2, 4

[17] Fangchang Ma and Sertac Karaman. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4796–4803. IEEE, 2018. 2

[18] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7210–7219, 2021. 3

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

[20] Raúl Mur-Artal and Juan D Tardós. Probabilistic semi-dense mapping from highly accurate feature-based monocular slam. In *Robotics: Science and Systems*. Rome, 2015. 2

[21] Michael Oechsle, Songyou Peng, and Andreas Geiger. Unisurf: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5589–5599, 2021. 3

[22] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 4, 5

[23] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alexander Sorkine-Hornung, and Luc Van Gool. The 2017 DAVIS challenge on video object segmentation. *CoRR*, abs/1704.00675, 2017. 6

[24] Weining Ren, Zihan Zhu, Boyang Sun, Jiaqi Chen, Marc Pollefeys, and Songyou Peng. Nerf on-the-go: Exploiting uncertainty for distractor-free nerfs in the wild. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 1, 7, 14

[25] Weining Ren, Zihan Zhu, Boyang Sun, Jiaqi Chen, Marc Pollefeys, and Songyou Peng. Nerf on-the-go: Exploiting uncertainty for distractor-free nerfs in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8931–8940, 2024. 5, 11

[26] Antoni Rosinol, John J Leonard, and Luca Carlone. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3437–3444. IEEE, 2023. 2

[27] Sara Sabour, Suhani Vora, Daniel Duckworth, Ivan Krasin, David J Fleet, and Andrea Tagliasacchi. Robustnerf: Ignoring distractors with robust losses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20626–20636, 2023. 1, 2, 3, 4

[28] Sara Sabour, Lily Goli, George Kopanas, Mark Matthews, Dmitry Lagun, Leonidas Guibas, Alec Jacobson, David J Fleet, and Andrea Tagliasacchi. Spotlesssplats: Ignoring distractors in 3d gaussian splatting. *arXiv preprint arXiv:2406.20055*, 2024. 7

[29] Sara Sabour, Lily Goli, George Kopanas, Mark Matthews, Dmitry Lagun, Leonidas Guibas, Alec Jacobson, David J. Fleet, and Andrea Tagliasacchi. SpotLessSplats: Ignoring distractors in 3d gaussian splatting. *arXiv:2406.20055*, 2024. 2, 4, 7, 14

[30] 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. 2

[31] Steven M Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE computer society conference on computer vision and pattern recognition (CVPR'06)*, pages 519–528. IEEE, 2006. 2

[32] Jiaxiang Tang, Hang Zhou, Xiaokang Chen, Tianshu Hu, Errui Ding, Jingdong Wang, and Gang Zeng. Delicate textured mesh recovery from nerf via adaptive surface refinement. *International Conference on Computer Vision (ICCV)*, 2023. 3

[33] Shimon Ullman. The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153):405–426, 1979. 2

[34] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 3

[35] 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. 3

[36] Jae-Chern Yoo and Tae Hee Han. Fast normalized cross-correlation. *Circuits, Systems and Signal Processing*, 28: 819–843, 2009. 6

[37] Zehao Yu, Torsten Sattler, and Andreas Geiger. Gaussian opacity fields: Efficient and compact surface reconstruction in unbounded scenes. *arXiv preprint arXiv:2404.10772*, 2024. 3

[38] Raza Yunus, Yanyan Li, and Federico Tombari. Manhattanslam: Robust planar tracking and mapping leveraging mixture of manhattan frames. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6687–6693. IEEE, 2021. 2

[39] Dongbin Zhang, Chuming Wang, Weitao Wang, Peihao Li, Minghan Qin, and Haoqian Wang. Gaussian in the wild: 3d gaussian splatting for unconstrained image collections. In *European Conference on Computer Vision*, pages 341–359. Springer, 2025. 2

[40] Ruida Zhang, Chengxi Li, Chenyangguang Zhang, Xingyu Liu, Haili Yuan, Yanyan Li, Xiangyang Ji, and Gim Hee Lee. Street gaussians without 3d object tracker, 2024. 2

In the supplementary materials, we analyze the effects of different masking strategies (section A), visualize the performance of MV-Pruned (section A), and report its quantitative metrics. We additionally provide ablation studies on key parameters and compare the computational overhead across algorithms (section B). Extended experimental results on the DTU-Robust (section C), TnT-Robust (section D) and NeRF-onthego (section E)datasets are also included for comprehensive evaluation.

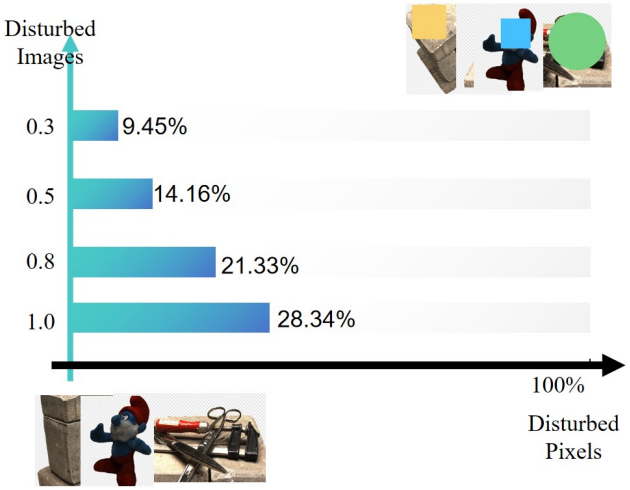


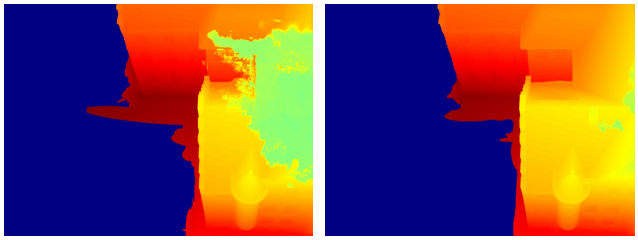
Figure 9. The average ratio of disturbed pixels to all object pixels under different disturbed image proportions on the DTU-Robust dataset.

Dataset	Metric	2DGS [7]	PGSR [2]	Ours
fountain	PSNR↑	20.74	20.69	21.82
	SSIM↑	0.583	0.657	0.659
	LPIPS↓	0.424	0.353	0.355
corner	PSNR↑	26.36	25.17	25.84
	SSIM↑	0.778	0.799	0.799
	LPIPS↓	0.369	0.325	0.338
spot	PSNR↑	22.57	22.72	22.99
	SSIM↑	0.576	0.593	0.586
	LPIPS↓	0.415	0.369	0.379
Avg.	PSNR↑	23.22	22.86	23.55
	SSIM↑	0.645	0.683	0.681
	LPIPS↓	0.402	0.349	0.357

Table 4. Comparison of novel view synthesis for Gaussian Splatting reconstruction methods on NeRF-on-the-go [25].

A. Pruning Analysis

Quantitatively, we compare the effectiveness of our MV-Prune in Tab. 5. With MV-Prune, the storage requirement of the model is decreased by 30.19% on average while maintaining 99.4% of the original performance according to PSNR↑ and a 0.01 decrease in CD↓.



(a) Before MV-prune (b) After MV-prune

Figure 10. Proposed pruning strategy MV-prune improves artifact removal, comparing results before and after MV-prune application.

	MVPSR			MVGSR(+w MV-Prune)		
	PSNR↑	CD↓	SIZE(MB)	PSNR↑	CD↓	SIZE(MB)
0.0	35.71	0.37	27.28	38.49	0.35	28.00
0.3	35.85	0.37	55.78	38.29	0.35	33.02
0.5	35.95	0.36	56.74	38.59	0.35	35.45
0.8	36.16	0.39	60.78	37.99	0.35	35.57
1.0	35.98	0.39	105.63	36.98	0.37	34.90
Avg.	35.93	0.38	73.40	38.07	0.35	33.38

Table 5. The average metrics on DTU-Robust scan106 before and after MV-Prune, including PSNR, CD, and Gaussian file size.

Qualitatively, when we compare the images visualized in Fig. 10, we observe that the image on the right is cleaner with fewer distractors. Firstly, the green artifact present on the right of Fig. 10a is largely removed in Fig. 10b with only a few small specks remaining. Moreover, the dark red jagged edges in the middle of Fig. 10a are smoothed to become less pronounced in Fig. 10b. Lastly, the details in the scene such as the structure of the object in yellow to orange are preserved in both pictures. These three observations show the efficacy of pruning with fewer distractors and better outlier removal whilst maintaining the high-frequency details of the scene.

Mask Analysis

We show the analysis of the effectiveness of our proposed masking strategies in Fig. 11. The baseline 'w/o mask' renders the scene with a noticeable artifact - the distractor as shown in Fig. 11a, and '$Mask_{mv}$' improves the geometry consistency and renders the details behind the distractor with small artifacts, as shown in Fig. 11b. The '$Mask_{sam}$' achieves further enhancement by combining the results from SAM, and provides clear and accurate rendering results, as shown in Fig 11c. The analysis proves that our integrated masking approach effectively mitigates visual artifacts caused by the distractor and improves the overall rendering quality.

As shown in Fig. 12, the semantic features of disturbed images can visually distinguish the distractors. By comparing features from multiple views, it is possible to obtain hints of disturbances before training. SAM is then used to refine the distractor masks. These optimized masks can enhance the model's accuracy in handling complex scenes. This approach not only allows us to identify potential disturbances before training but also enables dynamic adjustments during the training process, improving overall performance and robustness. Experiments on the DTU-Robust dataset demonstrate that this method achieves significant improvements across various scenarios.

(a) w/o mask (b) $Mask_{mv}$ (c) $Mask_{sam}$

Figure 11. Comparison of different mask strategies.

B. Ablation Study and Runtime

	delta_near=0.3	delta_near=0.5	delta_near=0.8
PSNR↑	32.56	33.06	33.10
CD↓	0.54	0.34	0.60

Table 6. Ablation Study in the setting of delta_near.

	CD↓	PSNR↑	Time/min	Memory/GB	Size/MB	Points
NeRF on-the-go	1.05	17.26	240	60.0	103.4	-
SpotLessSplats	0.90	27.27	9.95	3.06	80.97	400k
PGSR	0.53	31.31	33.3	4.11	89.8	379k
2DGS	0.57	30.96	17.1	3.85	2.7	11k
MVGSR w/o MV-Prune	0.37	32.33	39.8	4.45	133.0	560k
MVGSR	0.34	33.06	39.9	4.45	59.9	260k

MVGSR	train_7k	render	extract_feature	mask_mv	mask_sam	train	total
Time	123.2s	19.8s	41.8s	91.7s	89.9s	2029.0s	39.93min

Table 7. The Runtime metrics on DTU-Robust scan24

To demonstrate the contribution of each component, ablation experiments were conducted on DTU-Robust scan24 in this section. Also, as shown in Tab. 7, we compare the runtime and resource consumption of different algorithms on scan24 of the DTU-Robust dataset. Our approach achieves optimal results with lightweight resource utilization.

C. More Results on DTU-Robust Dataset

C.1. Dataset Analysis

The DTU-Robust dataset introduces distractors with random positions, colors, and shapes to a given proportion (rate) of DTU training images. The relationship between the number of disturbed images and the number of disturbed pixels is shown in Fig. 9. Specifically, at a *rate 0.3*, the distractor regions occupy 9.45% of the image, while this increases to 28.34% at *rate 1.0*. Therefore, from the perspective of the occupied pixels in the distractor areas, the challenges increase from *rate 0.3* to *rate 1.0* for the same sequence.

The DTU-Robust dataset effectively demonstrates the impact of occlusions, highlighting reconstruction errors and novel view rendering errors in the presence of these distractions.

C.2. Reconstruction

For surface quality evaluation, we use two other metrics, in addition to the bidirectional Chamfer distance CD used in the main paper, to assess the similarity between point clouds. We denote the Chamfer distance from the destination to the source as d2s, and from the source to the destination as s2d.

We compare the reconstruction performance of our proposed method to PGSR [2] and 2DGS [7] on the DTU-Robust dataset, across different scans and varying occlusion rates. As shown in Tab. 9, our method achieves superior and comparable performance under various evaluation protocols (CD, d2s, and s2d metrics) for the additional scans in the dataset.

Notably, our method outperforms both PGSR and 2DGS significantly in all scans. Specifically for *scan 24*, at an occlusion rate of 0.8, our method achieves a CD score that is more than twice as good as PGSR, improving from 0.83 to 0.34. Compared to 2DGS and PGSR, our method exhibits greater robustness across all scenes, with fewer performance fluctuations. We achieve comparable performance to PGSR for all other settings.

On average across all settings, our method outperforms 2DGS and PGSR by 76.4% and 60.3%, with an average CD score of 0.42. These results demonstrate that our method maintains robust reconstruction quality across varying occlusion rates and different scenes.

C.3. Rendering

To evaluate the novel view rendering performance of our proposed method, we compare with PGSR [2], 2DGS [7] and, additionally, 3DGS [13] for the novel view rendering task on the same DTU-Robust dataset. As shown in Tab. 10, our method is largely able to outperform the above competing methods. When comparing PSNR, our method achieves best performance on average under all settings for scenes that are evaluated on.

Similar to the task of reconstruction, our method exhibits greater robustness across all scenes, with fewer performance fluctuations compared to other competing methods. On average, we outperform 3DGS most significantly

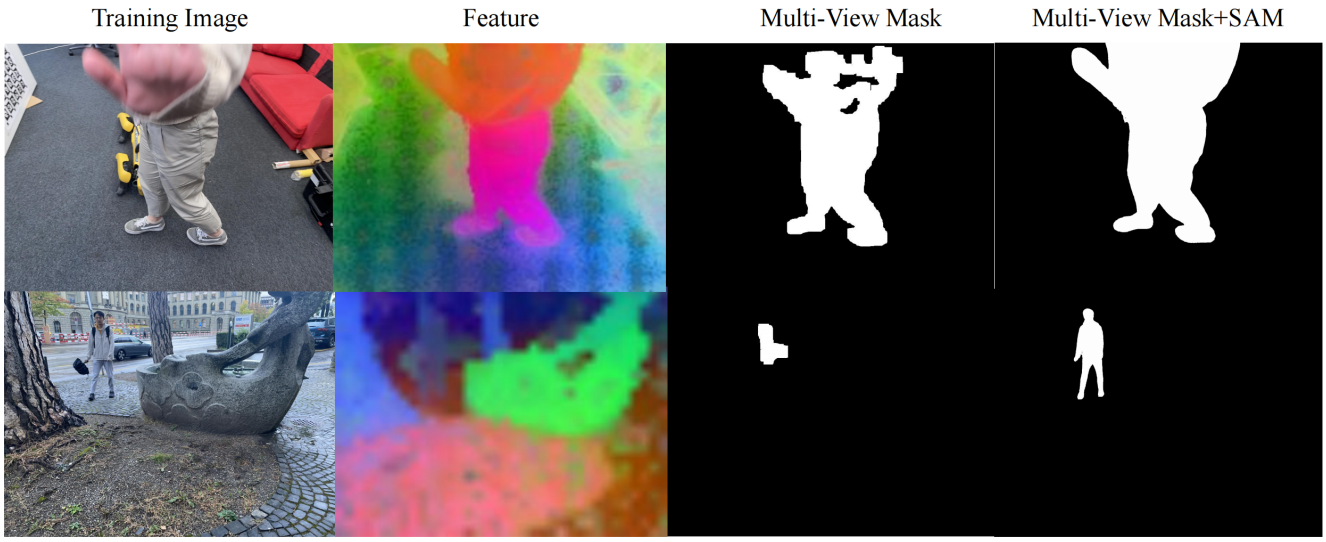


Figure 12. Disturbed images, corresponding semantic features, and multi-view mask results on NeRF onthego Dataset

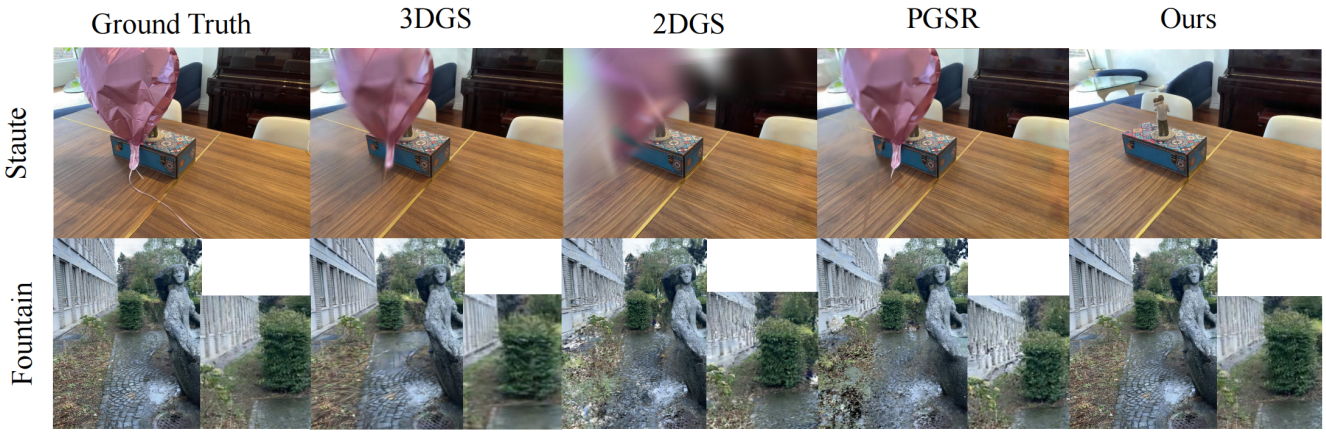


Figure 13. Visualization results of the t_balloon_statue on the dataset provided by RobustNeRF and the fountain on the NeRF-On-the-GO

by close to 6.82 dB with a PSNR of 34.42 dB while outperforming other methods by a smaller margin. These results demonstrate that our method can improve the quality of rendered images across varying occlusion rates and different scenes.

D. More Results on TnT-Robust Dataset

Tab. 8 and Fig. 14 show more results on TnT-Robust Datset.

E. More Results on NeRF on-the-go Dataset

As shown in Figures 12 and 13, we present additional visualizations on the NeRF onthego and RobustNerf datasets, demonstrating clearer segmentation effects and artifact removal. Quantitative visualization metrics on the onthego dataset are further reported in Tab. 4.

scan		Truck	Caterpillar	Barn	Meetingroom	Ignatius	Courthouse	Avg.
F1↑	PGSR [2]	0.57	0.37	0.57	0.30	0.64	0.18	0.44
	SLS [29]	0.48	0.30	0.43	0.24	0.57	0.15	0.36
	NeRFonthego [24]	0.37	0.22	0.28	0.17	0.49	0.11	0.27
	MVGSR	0.60	0.42	0.59	0.34	0.73	0.18	0.48
PSNR↑	PGSR [2]	21.67	20.35	23.21	21.94	18.79	20.74	20.95
	SLS [29]	20.66	21.14	21.77	21.88	20.67	20.44	21.09
	NeRFonthego [24]	19.33	19.10	23.46	21.95	17.36	20.97	20.36
	MVGSR	21.64	19.79	23.46	21.96	18.95	20.97	21.13

Table 8. Quantitative results of reconstruction performance on the TnT-Robust dataset.

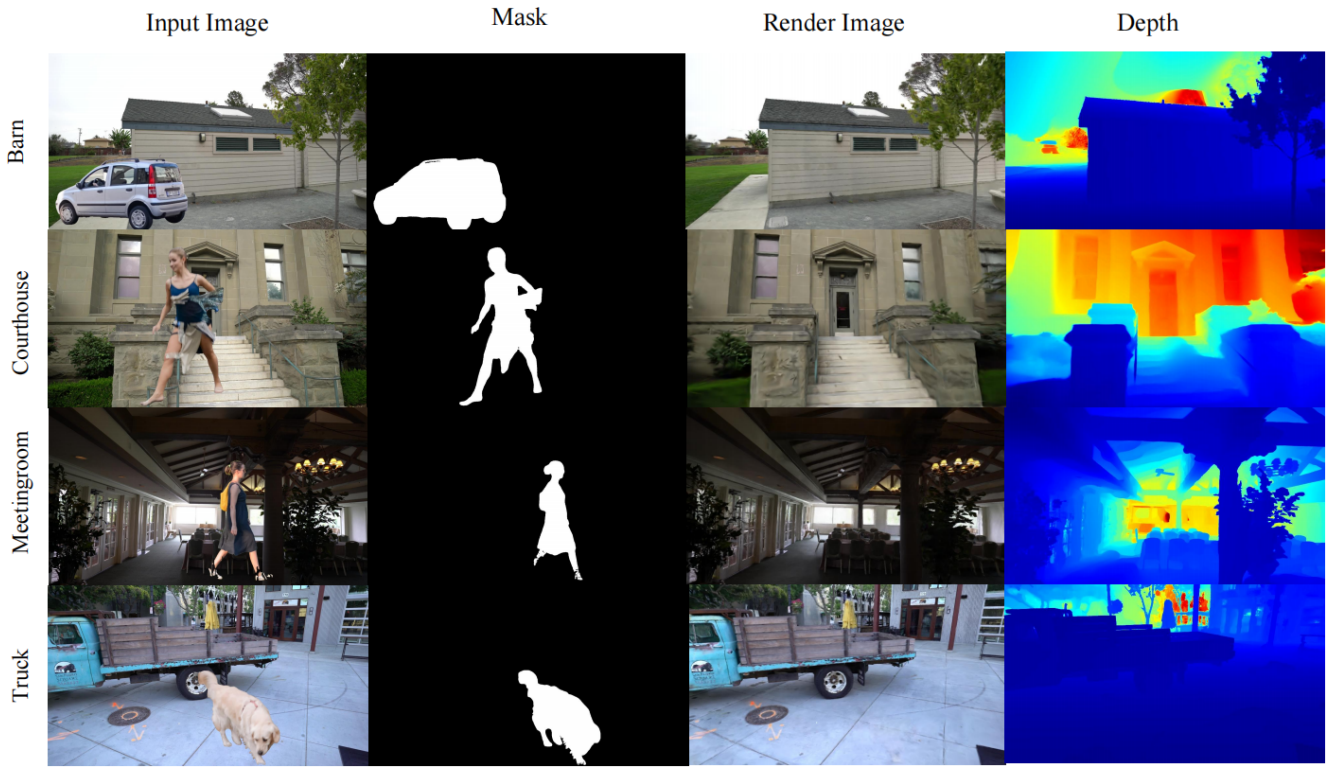


Figure 14. More results on TnT-Robust dataset

scan		24				37				40				55			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
CD↓	PGSR [2]	0.39	0.37	0.83	0.53	0.66	0.96	1.22	1.16	0.44	0.51	0.46	0.62	0.36	0.36	0.35	0.36
	2DGS [7]	0.50	0.52	0.48	0.57	0.83	0.85	0.85	0.92	0.34	0.69	0.92	0.43	0.38	0.39	0.45	0.56
	MVGSR	0.34	0.34	0.34	0.33	0.50	0.50	0.51	0.52	0.30	0.30	0.30	0.31	0.30	0.30	0.30	0.30
d2s↓	PGSR [2]	0.38	0.36	1.23	0.70	0.78	1.39	1.91	1.79	0.51	0.66	0.54	0.87	0.33	0.32	0.29	0.31
	2DGS [7]	0.50	0.54	0.47	0.58	0.99	1.02	1.03	1.18	0.35	0.48	0.59	0.53	0.35	0.36	0.46	0.65
	MVGSR	0.31	0.32	0.32	0.31	0.51	0.52	0.54	0.55	0.30	0.30	0.30	0.32	0.30	0.29	0.3	0.30
s2d↓	PGSR [2]	0.40	0.37	0.44	0.37	0.54	0.54	0.54	0.53	0.37	0.37	0.38	0.38	0.38	0.40	0.41	0.40
	2DGS [7]	0.50	0.51	0.49	0.57	0.67	0.67	0.66	0.66	0.33	1.24	0.90	0.34	0.41	0.43	0.44	0.48
	MVGSR	0.36	0.36	0.36	0.35	0.48	0.48	0.48	0.49	0.30	0.30	0.30	0.31	0.31	0.30	0.31	0.31

scan		63				65				69				83			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
CD↓	PGSR [2]	0.75	1.17	1.17	1.46	0.60	0.64	0.61	0.59	0.48	0.49	0.49	0.51	0.82	0.85	1.02	0.96
	2DGS [7]	1.07	1.02	1.16	1.22	0.97	1.09	0.92	1.00	0.79	0.81	0.85	0.81	1.32	1.33	1.39	1.42
	MVGSR	0.43	0.43	0.44	0.44	0.50	0.53	0.52	0.51	0.45	0.45	0.45	0.45	0.62	0.62	0.62	0.63
d2s↓	PGSR [2]	1.09	1.93	1.91	2.52	0.58	0.58	0.55	0.57	0.50	0.50	0.51	0.51	0.55	0.84	0.90	1.07
	2DGS [7]	1.18	1.13	1.25	1.30	0.89	1.08	0.91	1.02	0.79	0.77	0.84	0.79	1.00	0.95	1.00	1.04
	MVGSR	0.52	0.51	0.53	0.52	0.52	0.55	0.55	0.54	0.47	0.46	0.47	0.47	0.56	0.55	0.56	0.56
s2d↓	PGSR [2]	0.41	0.42	0.43	0.41	0.62	0.70	0.67	0.61	0.47	0.47	0.48	0.51	1.09	0.86	1.14	0.84
	2DGS [7]	0.96	0.91	1.07	1.14	1.06	1.11	0.93	0.97	0.80	0.84	0.85	0.84	1.63	1.71	1.78	1.81
	MVGSR	0.34	0.35	0.36	0.35	0.47	0.50	0.50	0.48	0.43	0.43	0.44	0.44	0.68	0.69	0.69	0.71

scan		97				105				106				110			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
CD↓	PGSR [2]	0.63	0.63	0.62	0.63	0.58	0.59	0.61	0.73	0.47	0.47	0.47	0.49	0.51	0.46	0.45	0.44
	2DGS [7]	1.22	1.19	1.22	1.13	0.68	0.68	0.69	0.69	0.70	0.69	0.70	0.73	0.70	0.69	0.70	0.73
	MVGSR	0.59	0.59	0.60	0.59	0.55	0.56	0.56	0.56	0.35	0.35	0.35	0.37	0.38	0.37	0.36	0.38
d2s↓	PGSR [2]	0.63	0.64	0.61	0.65	0.49	0.50	0.54	0.79	0.36	0.37	0.36	0.39	0.63	0.55	0.52	0.51
	2DGS [7]	1.09	1.01	1.08	0.96	0.60	0.59	0.61	0.60	0.50	0.48	0.50	0.58	1.44	1.60	1.53	1.63
	MVGSR	0.55	0.57	0.57	0.56	0.45	0.46	0.46	0.46	0.34	0.33	0.33	0.37	0.39	0.38	0.36	0.39
s2d↓	PGSR [2]	0.62	0.62	0.62	0.61	0.68	0.68	0.69	0.68	0.58	0.57	0.57	0.58	0.38	0.38	0.38	0.38
	2DGS [7]	1.35	1.36	1.36	1.30	0.77	0.78	0.77	0.78	0.90	0.90	0.90	0.88	1.23	1.56	1.60	1.57
	MVGSR	0.62	0.62	0.63	0.62	0.65	0.66	0.66	0.66	0.36	0.36	0.36	0.37	0.38	0.37	0.36	0.37

scan		114				118				122				Avg.			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
CD↓	PGSR [2]	0.32	0.31	0.32	0.36	0.37	0.37	0.44	0.38	0.35	0.34	0.37	0.37	0.52	0.57	0.63	0.64
	2DGS [7]	0.38	0.39	0.48	0.39	0.68	0.71	0.69	0.71	0.51	0.54	0.56	0.64	0.74	0.77	0.80	0.80
	MVGSR	0.29	0.29	0.30	0.29	0.34	0.34	0.34	0.35	0.34	0.34	0.35	0.35	0.42	0.43	0.44	0.40
d2s↓	PGSR [2]	0.33	0.33	0.33	0.33	0.40	0.40	0.42	0.43	0.38	0.38	0.38	0.41	0.53	0.65	0.73	0.79
	2DGS [7]	0.33	0.35	0.43	0.34	0.55	0.57	0.60	0.60	0.49	0.50	0.56	0.61	0.74	0.76	0.79	0.83
	MVGSR	0.26	0.26	0.26	0.26	0.32	0.34	0.33	0.34	0.32	0.32	0.33	0.33	0.44	0.46	0.48	0.43
s2d↓	PGSR [2]	0.32	0.31	0.32	0.36	0.37	0.37	0.44	0.38	0.35	0.34	0.37	0.37	0.51	0.50	0.53	0.49
	2DGS [7]	0.44	0.43	0.53	0.44	0.81	0.86	0.78	0.81	0.54	0.59	0.55	0.67	0.83	0.93	0.91	0.88
	MVGSR	0.33	0.33	0.33	0.32	0.35	0.35	0.35	0.37	0.36	0.36	0.36	0.37	0.43	0.45	0.47	0.44

Table 9. Quantitative results of reconstruction performance on the DTU-Robust dataset.

scan		24				37				40				55			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
PSNR↑	3DGS [13]	26.21	25.46	25.26	24.74	24.56	24.92	23.84	22.25	22.86	23.55	22.84	22.05	30.34	29.35	28.07	25.99
	PGSR [2]	31.86	31.49	30.68	31.31	27.51	27.07	26.56	25.73	30.83	30.56	29.48	28.66	32.95	32.11	31.73	30.44
	2DGS [7]	33.47	28.94	27.87	30.96	28.96	29.07	28.86	27.94	29.94	27.89	32.77	21.02	21.02	23.92	30.44	33.22
	MVGSR	33.61	32.95	32.41	32.28	30.05	29.89	29.27	27.68	33.43	33.03	32.3	31.33	35.19	35.33	33.99	32.66
SSIM↑	3DGS [13]	0.926	0.921	0.919	0.916	0.955	0.959	0.952	0.939	0.935	0.943	0.942	0.927	0.963	0.961	0.953	0.944
	PGSR [2]	0.946	0.944	0.941	0.943	0.942	0.940	0.936	0.931	0.941	0.940	0.935	0.932	0.923	0.920	0.919	0.913
	2DGS [7]	0.958	0.909	0.906	0.950	0.950	0.958	0.956	0.951	0.951	0.944	0.952	0.657	0.657	0.882	0.940	0.927
	MVGSR	0.956	0.954	0.952	0.951	0.963	0.962	0.959	0.953	0.954	0.953	0.95	0.945	0.981	0.982	0.979	0.974
LPIPS↓	3DGS [13]	0.086	0.094	0.095	0.098	0.050	0.046	0.053	0.068	0.098	0.093	0.095	0.115	0.060	0.063	0.072	0.082
	PGSR [2]	0.068	0.073	0.076	0.075	0.105	0.107	0.113	0.119	0.118	0.121	0.128	0.133	0.138	0.144	0.146	0.153
	2DGS [7]	0.073	0.140	0.146	0.082	0.082	0.096	0.098	0.103	0.103	0.112	0.111	0.448	0.448	0.189	0.128	0.168
	MVGSR	0.042	0.046	0.048	0.05	0.038	0.04	0.042	0.05	0.081	0.084	0.089	0.095	0.031	0.029	0.034	0.042

scan		63				65				69				83			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
PSNR↑	3DGS [13]	26.91	26.78	24.56	23.20	29.32	27.69	25.53	24.57	28.64	27.29	25.54	24.38	25.96	25.70	25.06	23.39
	PGSR [2]	33.33	32.77	32.54	32.00	32.94	31.88	30.44	29.80	31.79	30.99	30.96	30.02	32.79	32.05	32.00	31.22
	2DGS [7]	33.22	32.99	31.47	30.37	35.38	33.86	32.52	30.82	31.82	33.57	31.94	31.04	31.04	29.93	32.34	32.16
	MVGSR	38.68	38.06	36.37	34.94	36.07	34.42	33.72	32.24	33.68	33.27	33	31.47	41.78	41.47	40.9	39.44
SSIM↑	3DGS [13]	0.968	0.965	0.957	0.947	0.968	0.961	0.953	0.951	0.947	0.939	0.919	0.925	0.965	0.959	0.958	0.947
	PGSR [2]	0.950	0.948	0.946	0.943	0.917	0.915	0.910	0.911	0.917	0.914	0.912	0.908	0.909	0.905	0.905	0.902
	2DGS [7]	0.927	0.925	0.921	0.914	0.965	0.962	0.959	0.955	0.955	0.930	0.927	0.924	0.924	0.922	0.929	0.930
	MVGSR	0.978	0.977	0.974	0.972	0.979	0.977	0.976	0.973	0.965	0.965	0.963	0.958	0.987	0.987	0.986	0.984
LPIPS↓	3DGS [13]	0.038	0.045	0.056	0.069	0.046	0.055	0.068	0.069	0.072	0.084	0.103	0.100	0.048	0.055	0.057	0.068
	PGSR [2]	0.106	0.110	0.112	0.116	0.163	0.165	0.174	0.171	0.145	0.149	0.151	0.158	0.216	0.229	0.228	0.234
	2DGS [7]	0.168	0.170	0.177	0.188	0.101	0.108	0.111	0.119	0.119	0.172	0.176	0.181	0.181	0.185	0.152	0.152
	MVGSR	0.024	0.026	0.03	0.034	0.039	0.042	0.045	0.048	0.057	0.058	0.062	0.067	0.022	0.024	0.025	0.027

scan		97				105				106				110			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
PSNR↑	3DGS [13]	23.35	25.67	24.89	24.19	24.21	24.36	23.33	22.89	33.98	34.21	33.24	31.19	35.31	34.57	34.50	33.98
	PGSR [2]	30.80	30.89	30.30	29.49	33.63	33.09	32.38	31.92	35.40	35.43	35.05	33.86	34.35	32.71	33.96	33.28
	2DGS [7]	32.16	31.74	30.59	33.58	33.58	32.99	19.52	18.21	31.01	30.68	30.33	29.84	29.84	33.80	33.42	32.31
	MVGSR	35.21	34.99	34.44	32.83	39.3	38.66	37.74	36.38	38.29	38.59	37.99	36.98	36.35	35.93	35.61	35.34
SSIM↑	3DGS [13]	0.936	0.953	0.942	0.940	0.952	0.952	0.945	0.941	0.959	0.959	0.957	0.950	0.961	0.959	0.958	0.957
	PGSR [2]	0.910	0.909	0.909	0.905	0.916	0.914	0.912	0.909	0.933	0.933	0.933	0.928	0.923	0.917	0.919	0.918
	2DGS [7]	0.930	0.927	0.922	0.918	0.918	0.916	0.789	0.766	0.919	0.918	0.916	0.914	0.914	0.925	0.923	0.920
	MVGSR	0.974	0.973	0.973	0.968	0.977	0.976	0.974	0.972	0.977	0.978	0.977	0.974	0.974	0.973	0.971	0.971
LPIPS↓	3DGS [13]	0.072	0.055	0.068	0.069	0.061	0.063	0.074	0.079	0.072	0.073	0.078	0.086	0.074	0.076	0.078	0.080
	PGSR [2]	0.185	0.185	0.189	0.192	0.185	0.187	0.190	0.196	0.186	0.185	0.188	0.193	0.192	0.199	0.198	0.203
	2DGS [7]	0.152	0.155	0.161	0.235	0.235	0.236	0.427	0.453	0.196	0.195	0.200	0.201	0.201	0.193	0.199	0.202
	MVGSR	0.037	0.038	0.039	0.044	0.038	0.041	0.043	0.047	0.045	0.044	0.047	0.05	0.052	0.054	0.056	0.058

scan		114				118				122				Avg.			
rate		0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0	0.3	0.5	0.8	1.0
PSNR↑	3DGS [13]	31.59	27.46	28.19	29.55	35.13	34.89	32.51	32.60	32.69	31.54	29.48	27.09	28.74	28.23	27.12	26.33
	PGSR [2]	32.47	32.54	32.02	31.61	37.23	37.04	35.71	35.67	36.71	36.01	35.67	34.46	32.97	32.44	31.97	31.30
	2DGS [7]	32.73	32.47	26.54	31.52	37.30	37.12	35.56	35.18	36.74	35.89	35.52	35.32	32.08	31.86	30.84	30.43
	MVGSR	33.71	33.49	32.95	32.67	40.24	40.13	38.47	38.75	40.82	40.76	39.27	38.44	35.58	35.09	34.06	32.98
SSIM↑	3DGS [13]	0.965	0.947	0.953	0.959	0.968	0.966	0.960	0.962	0.972	0.967	0.958	0.948	0.956	0.953	0.948	0.943
	PGSR [2]	0.924	0.925	0.922	0.923	0.933	0.933	0.928	0.930	0.931	0.928	0.927	0.925	0.928	0.926	0.924	0.921
	2DGS [7]	0.931	0.929	0.872	0.927	0.935	0.934	0.930	0.929	0.931	0.930	0.930	0.927	0.918	0.927	0.918	0.900
	MVGSR	0.97	0.969	0.967	0.967	0.982	0.982	0.978	0.98	0.986	0.986	0.984	0.982	0.970	0.970	0.967	0.965
LPIPS↓	3DGS [13]	0.048	0.066	0.060	0.058	0.053	0.055	0.065	0.062	0.033	0.041	0.050	0.061	0.061	0.064	0.071	0.078
	PGSR [2]	0.170	0.170	0.176	0.176	0.194	0.192	0.205	0.199	0.196	0.210	0.206	0.202	0.158	0.162	0.165	0.168
	2DGS [7]	0.194	0.196	0.291	0.201	0.229	0.232	0.238	0.239	0.236	0.242	0.237	0.246	0.182	0.175	0.190	0.215
	MVGSR	0.044	0.045	0.049	0.05	0.034	0.035	0.04	0.038	0.022	0.023	0.026	0.03	0.042	0.044	0.047	0.051

Table 10. Quantitative results of novel view synthesis (PSNR↑) on DTU-Robust.