Binocular-Guided 3D Gaussian Splatting with View Consistency for Sparse View Synthesis

Anonymous Author(s)
Affiliation
Address
email

Abstract

Learning to synthesize novel views from sparse inputs is a vital yet challenging task in 3D computer vision. Previous methods explore 3D Gaussian Splatting with neural priors (e.g. depth priors) as the additional supervisions, demonstrating promising quality and efficiency compared to the NeRF based methods. However, the neural priors from 2D pretrained models are often noisy and blurred, which struggle in precisely guiding radiance field learning. In this paper, we propose a novel prior-free method for sparse view Gaussian Splatting by exploring the self supervisions inherent in the binocular stereo consistency between each pair of binocular images constructed with disparity-guided image warping. We additionally introduce a Gaussian Opacity constraint which regularizes the Gaussians locations and avoids Gaussian redundancy for improving the robustness and efficiency of sparse-view Gaussian Splatting. Extensive experiments on the LLFF, DTU, and Blender datasets demonstrate that our method significantly outperforms the state-of-the-art methods.

1 Introduction

3D reconstruction technologies [22, 17] have demonstrated significant advances in synthesizing realistic novel views given a set of dense input views. To explore the challenging task of sparse view synthesis for harsh real-world situations where only sparse inputs are available, some researches train NeRF[22] with specially designed constraints [15, 36, 35, 38] and regularizations [24, 41, 39] on the view scarcity. However, NeRF-based methods often suffer from slow training and inference speeds, leading to high computational costs that restrict their practical applications.

3D Gaussian Splatting (3DGS) [17] has achieved notable advantages in rendering quality and efficiency. However, 3DGS faces significant challenges with sparse views, where the unstructured 3D Gaussians with limited constraints are prone to overfitting to the input views, resulting in incorrectly learned scene geometry. Some recent works [25, 46, 20, 40] on sparse view synthesis based on 3DGS employ the commonly-used depth priors from pre-trained models as additional constraints on the Gaussians geometries. However, the neural priors are often noisy and blurred, which struggle in precisely guiding radiance field learning.

In this paper, we aim at designing a prior-free approach which directly exploring the self supervisions from the limited input views for improving the quality and efficiency of 3DGS for sparse views. We justify that the key factors in achieving these goals are 1) learning correct scene geometry of Gaussians which leads to consistent views synthesis, and 2) avoiding redundant Gaussians near the surface for better efficiency and filtering noises.

For learning correct scene geometry of Gaussians, we propose a novel prior-free method for sparse view Gaussian Splatting. We explore the self supervisions inherent in the binocular stereo consistency

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

to constrain on the rendered depth of 3DGS solely based on existing input views and synthesized novel views. The key insight of our method lies in the observation that binocular image pairs implicitly involve the property of view consistency, as demonstrated in the binocular stereo vision works [11, 10, 44]. Specifically, we first translate the camera of one input view slightly to the left or right to obtain a translational view, from which we render the image and depth from 3DGS. The rendered image and the input one thus form a left-right view pair as in binocular stereo vision. We then leverage the rendered depth and the known camera intrinsics to compute the disparity of the view pair. We conduct the supervisions by warping the rendered image of the translational view to the viewpoint of the input image using the disparity, and constrains on the consistency between the warped and input images.

To further reduce redundant Gaussians near the scene surface and enhance the quality and efficiency of novel view synthesis, we propose a decay schema for the opacity of Gaussians. Specifically, we simply apply a decay coefficient to the opacity property of the Gaussians, penalizing the opacity during training. To this end, Gaussians with lower opacity gradients are pruned due to the continuous reduction of their opacity, while those with higher opacity gradients are retained. This results in cleaner and more robust Gaussians by filtering out the redundant ones and guiding the remaining Gaussians to be closer to the scene surface. This opacity decay strategy significantly reduces artifacts in novel views and decreases the number of Gaussians, improving both the rendering quality and optimization efficiency of 3DGS.

Additionally, to achieve better geometry initialization for improving 3DGS quality when conducting optimization on sparse views, we use pre-trained keypoints matching network [34] to generate a dense initialization point cloud. The dense point cloud describes the geometry of the scene more accurately, preventing Gaussians from appearing far from the scene surface, especially in low-texture areas where the distribution of Gaussians is subject to limited constraints.

In summary, our main contributions are as follows:

- We propose a novel prior-free method for sparse view Gaussian Splatting. We explore the self supervisions inherent in the binocular stereo consistency to constrain on the rendered depth of 3DGS solely based on existing input views and synthesized views.
- We propose an opacity decay strategy which significantly regularizes the Gaussians and reduces Gaussian redundancy, leading to better rendering quality and optimization efficiency for sparse view Gaussian Splatting.
- Extensive experiments on widely-used forward-facing and 360-degree scene datasets demonstrate that our method achieves state-of-the-art results compared to existing sparse novel view synthesis methods.

2 Related Works

2.1 Neural Radiance Field

Detailed and realistic 3D scene representation has always been the research goal in the field of computer vision, and the emergence of neural radiance fields (NeRFs) [22] has brought fundamental innovation to this domain. NeRF can reconstruct high-quality 3D scenes from sparse 2D images and generate realistic images from arbitrary viewpoints by representing scenes as continuous volume radiance functions.

However, NeRF requires a large number of views as input during the training stage and exhibits limitations in both training and inference speed. Consequently, the following researches have primarily focused on addressing these bottlenecks by improving computational efficiency [9, 23, 32, 17, 14, 4, 5, 8] or reducing the number of input views [24, 15, 36, 35, 6, 38, 30, 19, 7, 31, 29, 33, 39], while continuously striving for enhanced rendering quality [1, 2, 3].

Notably, recent approach 3D Gaussian Splatting [17] has shown promising results in achieving real-time rendering capabilities without compromising rendering quality.

3D Gaussian Splatting [17] is an emerging method for novel view synthesis, which reconstructs scenes rapidly and with high quality by utilizing a set of 3D Gaussians to represent the radiance field in the scene. This method performs excellently when dealing with real-world scenes, particularly excelling

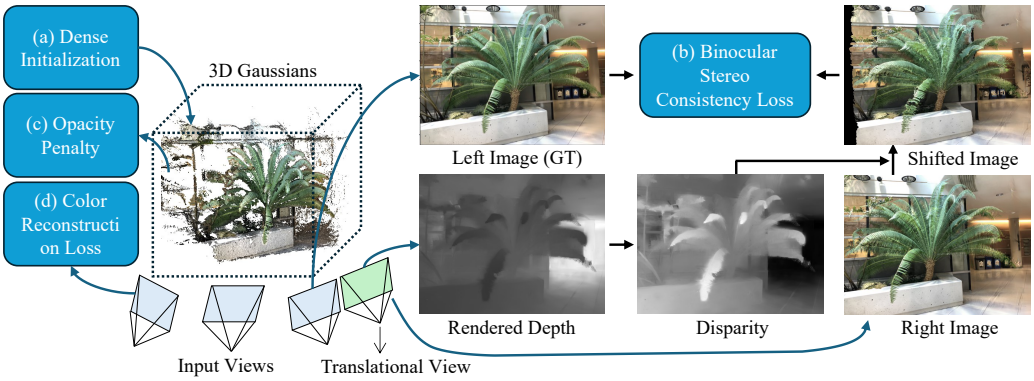


Figure 1: The overview of our method. (a) We leverage dense initialization for achieving Gaussian locations, and optimize the locations and Gaussian attributes with three constraints or strategies: (b) Binocular Stereo Consistency Loss. We construct a binocular view pair by translating an input view with camera positions, where we constrain on the view consistency of binocular view pairs in a self-supervised manner. (c) Opacity Penalty Strategy is designed to decay the Gaussian opacity during training for regularizing them. (d) The commonly-used Color Reconstruction Loss.

in handling high-frequency details. Moreover, 3D Gaussian Splatting demonstrates significant advantages in inference speed and offers more intuitive editing and interpretability capabilities.

2.2 Novel View Synthesis from Sparse Views

Recent researches [24, 15, 36, 35, 6, 38, 30, 19, 7, 31, 29, 33, 39] in sparse novel view synthesis have explored various approaches to generate novel views from sparse input images, with focus on enhancing both rendering quality and efficiency. Methods such as NeRF[22] and 3DGS [17] have been refined through various techniques, resulting in continuous improvement in the quality of novel view synthesis.

Using depth priors obtained from pre-trained networks [42, 26, 27] to supervise neural radiance fields is one of the effective techniques [31, 36]. Some methods [24, 41, 39, 7] introduce regularization terms in NeRF to address the problem, such as frequency regularization [41] and ray entropy regularization [18]. Additionally, there are also some methods [39, 38, 15] leverage pre-trained models such as Diffusion model to enhance the rendering quality of novel views. However, most methods incur high costs during training and inference. While some methods have improved inference efficiency through generalizable models [5, 43] or by using voxel grids [33], they often sacrifice rendering quality.

Currently, with the advent of 3D Gaussian splatting, some methods use 3DGS for sparse view synthesis, such as FSGS [46], SparseGS [40], DNGaussian [20], and CoherentGS [25]. These methods utilize depth priors obtained from pre-trained models [26] as supervision. Depth constraints enable the unstructured Gaussians to approximately distribute along the scene surface, thereby enhancing the quality of novel view images. However, the depth priors from pre-trained models often contain significant errors and cannot make the Gaussians distribute to optimal positions. In contrast, our method employs view consistency constraints based on binocular vision, resulting in more accurate depth information and thereby achieving a more optimal distribution of Gaussians.

3 Methods

The pipeline of our method is depicted in Figure 1. In this section, we first review the 3D information representation method based on 3D Gaussians. Then, we explain how to construct stereo view pair and utilize it to enforce view consistency in a self-supervised manner. Next, we introduce the opacity penalty strategy and the dense initialization method for 3D Gaussians. Finally, we present the overall loss function used for optimization.

3.1 Review of 3D Gaussian Splatting

Gaussian splatting [17] represents scene with a set of 3D Gaussians. Each 3D Gaussian is defined by a central location $\mu \in \mathbb{R}^3$, and a covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$. Formally, it is defined as follows

$$G_i(x) = e^{-\frac{1}{2}(x-\mu_i)^T\Sigma^{-1}(x-\mu_i)}, \tag{1}$$

The covariance matrix Σ have physical meaning only when they are positive semi-definite. Therefore, it is decomposed into $\Sigma = RSS^TR^T$, $S \in \mathbb{R}^3$ is a diagonal scaling matrix with 3 parameters, $R \in \mathbb{R}^4$ is a rotation matrix analytically expressed with quaternions. In addition, for rendering the image, each Gaussian also stores an opacity value $\alpha \in \mathbb{R}$ and a color feature $f \in \mathbb{R}^k$.

The 3D Gaussian is projected into the 2D image space when rendering an image, the projected 2D Gaussian is sorted by its depth value, and then Alpha Blending is used to calculate the color of each pixel is calculated with α-blending.

$$c = \sum_{i=1}^{n} c_i \alpha_i' \prod_{j=1}^{i-1}(1-\alpha_j'), \tag{2}$$

where c_i is the color computed from feature f and α' is the opacity of the 2D Gaussian, which is obtained by multiplying the covariance Σ' of the 2D Gaussian by the opacity α of the corresponding 3D Gaussian. The 2D covariance matrix Σ' is calculated by $\Sigma' = JW\Sigma W^TJ^T$, where J is the Jacobian of the affine approximation of the projective transformation. W is the view transformation matrix.

The optimization process of Gaussian splatting begins with creating a set of 3D Gaussians from a sparse SfM point clouds. Subsequently, the density of the Gaussian set is optimized and adaptively controlled. During optimization, a fast tile-based renderer is employed, allowing competitive training and inference times compared to the fast radiance field methods.

3.2 Binocular Stereo Consistency Constraint

The key factor in improving the rendering quality of 3DGS is to guide the Gaussians to be distributed close to the exact scene surfaces. To this end, we aim to design a constraint on the rendered depth of 3DGS which directly represents the Gaussian geometry. Previous works commonly adopt the depth priors from pretrained models to guide the depth of 3DGS. However, the depth prior are often noisy and blurred, especially on complex scenes, which fails in providing accurate guidance.

In this paper, we propose a novel prior-free method for sparse view Gaussian Splatting by exploring the self supervisions inherent in the binocular stereo consistency. We constrain on the rendered depth of 3DGS solely based on the existing input views and the synthesized novel views rendered from 3DGS itself. Specifically, we first obtain the translated image of the input view by translating the camera position corresponding to the input view, thus constructing a pair of binocular vision images. We then shift the novel view to the perspective of the input image using disparity. Finally, we constrain on the consistency between the input image and the warped image.

Given an image view I_l as input, we translate its corresponding camera position C_l to the right by a distance d_{cam} and obtain the translational camera position O_r. We then obtain a novel rendered image I_r by rendering 3DGS from O_r. The images I_l and I_r form a binocular stereo image pair, where I_l is the left image and I_r is the right image. Simultaneously, we can obtain the rendered depth D_r corresponding to the right image I_r by rendering 3DGS from O_r. According to the geometric theory of binocular stereo vision, the relationship between disparity d and depth D_r is given by $d = f \cdot d_{cam}/D_r$, where $d \in \mathbb{R}^{h\times w}$, each value d_{ij} in d indicates the horizontal shift required to align a pixel in the right image with its counterpart in the left image. f is the focal length of the camera. Then, we can use the disparity d to move each pixel of the right image I_r, obtaining the reconstructed left image $I_{shifted}$.

$$I_{shifted}[i,j] = I_r[i-d_i, j-d_j] \tag{3}$$

In practice, to make the reconstructed left image $I_{shifted}$ differentiable, we use a bilinear sampler to interpolate on the right image I_r to obtain each pixel of $I_{shifted}$. The sampling coordinates for each pixel are directly obtained from the disparity. Formally, we define the loss function as follows

$$L_{consis} = \frac{1}{N}\sum_{i,j}\left|I_{ij}^{l} - I_{ij}^{shifted}\right| \tag{4}$$

To achieve better optimization with the loss function, we do not use a fixed camera position shift in practice. Instead, we randomly translate the input camera to the left or right by a distance within a range $[-d_{max}, d_{max}]$.

Theoretically, it is also possible to implement the consistency constraint between input images by only using the input images and the corresponding rendered depths. However, due to the large camera pose variation among the sparse input views, the warped image can differ greatly from the target image, making it difficult to achieve convergence during training.

3.3 Opacity Penalty Strategy

We further justify that relying solely on the depth constraints does not always lead to correct Gaussian geometries that are closely aligned with the exact scene surfaces. The reason is that the rendered depth varies with changes in the scale and opacity of the Gaussians, rather than being solely determined by their positions. While 3DGS flexibly optimizes the scale and opacity during training, which leads to deviations in the novel views. To address this issue, we design a simple strategy by applying a decay coefficient λ to the opacity α of the Gaussians, penalizing the opacity during training.

$$\hat{\alpha} = \lambda\alpha, 0 < \lambda < 1 \tag{5}$$

We illustrate the opacity penalization strategy in Figure 2. The Gaussians close to the scene surface often contain larger gradients due to constraints from various views, while the Gaussians far from the surface are often with much lower gradients due to weaker constraints. We aim to improve and stabilize the optimization of 3DGS by filtering out the far Gaussians which indicates incorrect geometry while remain the ones close to the exact scene surfaces. This is achieved by applying the opacity penalty strategy. We justify that the strategy does not lead to a decrease in opacity for all Gaussians. As illustrated in Figure 2, the opacity of Gaussians with lower opacity gradients gradually decreases until they are pruned. Conversely, the increase in opacity for Gaussians near the scene surface exceeds the penalty magnitude, ultimately achieving a balance between the opacity increase and the penalty, thereby preserving Gaussians close to the surface.

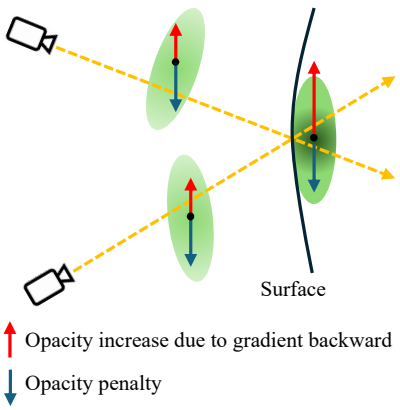


Figure 2: Illustration of the Gaussian opacity penalty strategy.

3.4 Initialization from dense point clouds

Previous 3DGS methods [17, 46, 40] usually utilize a sparse point cloud generated by Structure from Motion (SfM) [28] to initialize 3D Gaussians. However, the point cloud produced by sparse views is too sparse to adequately describe the scene to be reconstructed. Although the splitting strategy in 3DGS can replicate new Gaussians to cover the under-reconstructed area, they are subject to limited geometric constraints and cannot adhere well to the scene surfaces, especially in low-texture areas where the distribution of Gaussians may be arbitrary. We therefore seek a robust approach to achieve better geometry initialization for improving 3DGS quality when optimizing from sparse views.

To achieve this, we use a pre-trained keypoints matching network [34] to generate a dense initialization point cloud. Specifically, we arbitrarily select two images from the input images, input them into the matching network, and obtain matching points. We then leverage the triangulation method [13], along with the camera parameters corresponding to these images, to project the matching points into 3D space. This forms a dense point cloud, providing a more robust initialization for the Gaussians.

Compared with the sparse point cloud, the dense point cloud describes the geometry of the scene more accurately, preventing Gaussians from appearing far from the scene surface and ultimately leading to improved quality in novel view synthesis.

3.5 Training Loss

The final loss function consists of two parts: the proposed binocular stereo consistency loss L_{consis} as introduced in Eq.4 and the commonly-used color reconstruction loss L_{color} of 3DGS [17]. We

Table 1: Quantitative comparison on LLFF. We evaluate the NeRF-based and the 3DGS-based methods, our method achieves the best results in all metrics under different input-view settings.

Methods	PSNR↑			SSIM↑			LPIPS↓		
	3-view	6-view	9-view	3-view	6-view	9-view	3-view	6-view	9-view
DietNeRF [15]	14.94	21.75	24.28	0.370	0.717	0.801	0.496	0.248	0.183
RegNeRF [24]	19.08	23.10	24.86	0.587	0.760	0.820	0.336	0.206	0.161
FreeNeRF [41]	19.63	23.73	25.13	0.612	0.779	0.827	0.308	0.195	0.160
SparseNeRF [36]	19.86	23.26	24.27	0.714	0.741	0.781	0.243	0.235	0.228
ReconFusion [38]	21.34	24.25	25.21	0.724	0.815	0.848	0.203	0.152	0.134
3DGS [17]	15.52	19.45	21.13	0.405	0.627	0.715	0.408	0.268	0.214
FSGS [46]	16.49	19.67	20.45	0.483	0.621	0.662	0.367	0.286	0.267
DNGaussian [20]	19.12	22.18	23.17	0.591	0.755	0.788	0.294	0.198	0.180
Ours	21.44	24.87	26.17	0.751	0.845	0.877	0.168	0.106	0.090

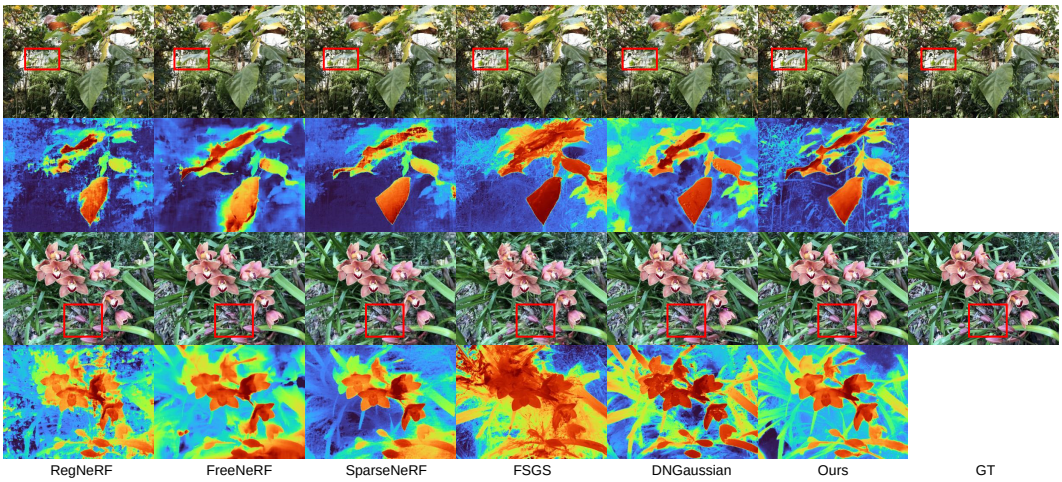


Figure 3: Visual comparison on LLFF dataset.

define the overall loss function as follows:

$$L = L_{consis} + L_{color}, \tag{6}$$

Where L_{color} is composed of an L_1 loss and a structural similarity loss L_{D-SSIM}, defined as

$$L_{color} = (1 - \beta)L_1 + \beta L_{D-SSIM} \tag{7}$$

4 Experiments

4.1 Datasets

We conducted experiments on three public datasets, including the LLFF dataset [21], the DTU dataset [16], and the NeRF Blender Synthetic dataset (Blender) [22]. Following prior works [24, 41, 15], we used 3, 6, and 9 views as training sets for the LLFF and DTU datasets, and 8 images for training on the Blender dataset. The selection of test images remained consistent with previous works [15, 24, 41]. The downsampling rates for the LLFF, DTU, and Blender datasets were 8, 4, and 2, respectively.

4.2 Implementation details

Since The LLFF and DTU are datasets of forward-facing scenes, they cannot be constrained by views from other directions during the optimization process. On the other hand, Blender is a dataset of 360-degree scenes, 8 images from different viewpoints are used as input during training, providing stronger constraints. Therefore, for the LLFF and DTU datasets, we utilized a pre-trained matching network PDC-Net+ [34] to obtain keypoints from input images, which were then used to generate

Table 2: Quantitative comparison on DTU. We evaluate the NeRF-based and the 3DGS-based methods, our method achieves the best results in most metrics under different input-view settings.

Methods	PSNR↑			SSIM↑			LPIPS↓		
	3-view	6-view	9-view	3-view	6-view	9-view	3-view	6-view	9-view
DietNeRF [15]	11.85	20.63	23.83	0.633	0.778	0.823	0.214	0.201	0.173
RegNeRF [24]	18.89	22.20	24.93	0.745	0.841	0.884	0.190	0.117	0.089
FreeNeRF [41]	19.52	23.25	25.38	0.787	0.844	0.888	0.173	0.131	0.102
SparseNeRF [36]	19.47	-	-	0.829	-	-	0.183	-	-
ReconFusion [38]	20.74	23.62	24.62	0.875	0.904	0.921	0.124	0.105	0.094
3DGS [17]	10.99	20.33	22.90	0.585	0.776	0.816	0.313	0.223	0.173
FSGS [46]	17.34	21.55	24.33	0.818	0.880	0.911	0.169	0.127	0.106
DNGaussian [20]	18.91	22.10	23.94	0.790	0.851	0.887	0.176	0.148	0.131
Ours	20.71	24.31	26.70	0.862	0.917	0.947	0.111	0.073	0.052

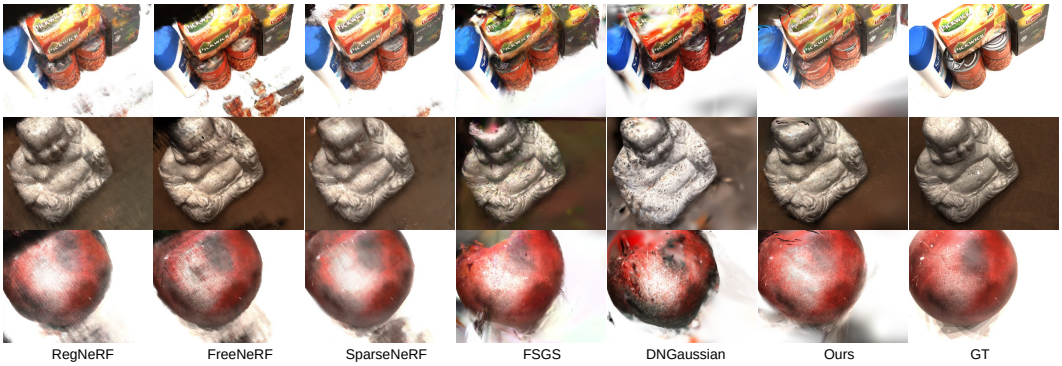


Figure 4: Visual comparison on DTU dataset.

dense initialization point clouds. For the Blender dataset, we adopted random initialization as in the original 3DGS [17].

Moreover, based on the analysis above, we train the LLFF and DTU datasets for 30,000 iterations, while the Blender dataset is trained for 7,000 iterations. Since the view consistency constraint based on binocular stereo vision needs to be performed on the basis of the training views that can already be rendered with high quality, we add the view consistency loss at 20,000 iterations for the LLFF and DTU datasets, and at 4,000 iterations for the Blender dataset. The maximum distance d_{max} for camera shift is set to 0.4, the opacity decay coefficient λ is set to 0.995, and the β in the loss function is set to 0.2 as in the original 3DGS [17].

4.3 Baseline

We select some state-of-the-art NeRF-based and 3DGS-based sparse view synthesis methods to contrast with our method. NeRF-based methods include DietNeRF [15], RegNeRF [24], FreeNeRF [41], SparseNeRF [36], and ReconFusion [38]. 3DGS-based methods include DNGaussian [20] and FSGS [46]. Additionally, we compared against the original 3DGS [17].

4.4 Comparisons

LLFF The Table 1 shows the quantitative results of the LLFF dataset with 3, 6, and 9 input views respectively. Our method surpasses all baseline methods in terms of PSNR, SSIM [37], and LPIPS [45] scores under different number of input views. When the number of input views increases to 9, it is almost adequate to provide sufficient color constraints. However, the evaluation scores of DNGaussian [20] do not show a significant improvement, indicating that it does not perform well with an increased number of input views. This is because errors in the depth prior negatively affect the optimization process.

Figure 3 presents the visual comparison of novel view synthesis and depth rendering for the *leaves* and *orchids* scenes in the LLFF dataset. From the rendered results of novel views, several state-of-the-art methods are all perceptually acceptable. However, regarding the depth maps, RegNerf [24], FreeNerf [41], and FSGS [46] exhibit poor quality, indicating significant errors in their geometric estimation. SparseNeRF [36] and DNGaussian [20] indirectly utilize relative depth information from depth priors to supervise geometry estimation, hence achieving adequate rendering depth. Although FSGS [46] also exploits depth priors, its attempt to directly employ depth prior information by using depth correlation loss. It does not yield accurate scene geometry, instead, inaccurate depth prior information exacerbates scene geometry. In contrast, our method utilizes self-supervised view consistency loss to obtain more precise geometry, thereby recovering more structural details in novel views.

Table 3: Quantitative comparison on Blender for 8 input views. We evaluate the NeRF-based and the 3DGS-based methods, our method achieves comparable results to the state of the art methods.

Methods	PSNR↑	SSIM↑	LPIPS↓
DietNeRF [15]	22.50	0.823	0.124
RegNeRF [24]	23.86	0.852	0.105
FreeNeRF [41]	24.26	0.883	0.098
SparseNeRF [36]	22.41	0.861	0.199
3DGS [17]	22.23	0.858	0.114
FSGS [46]	22.76	0.829	0.157
DNGaussian [20]	24.31	0.886	0.088
Ours	24.71	0.872	0.101

DTU The Table 2 shows the quantitative results of the DTU dataset under 3, 6, and 9 input views respectively. Figure 4 illustrates the visual comparison of novel view synthesis for three scenes in the DTU dataset. To distinctly discern the differences among various methods, we select the synthesized images far from the training views for comparison in each scene. From the rendered results of novel views, it can be observed that NeRF-based methods produce blurry results, while FSGS [46] and DNGaussian [20] exhibit numerous artifacts due to insufficient constraints on Gaussians. Our method outperforms previous state-of-the-art methods on most evaluation metrics and achieves better visual quality as well.

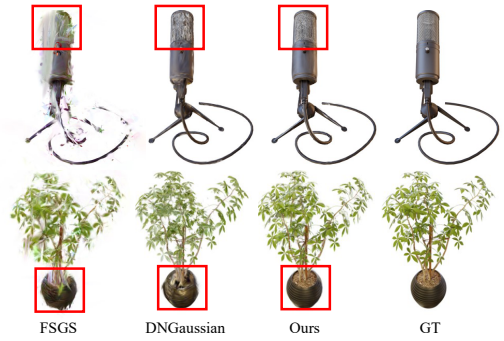


Figure 5: Visual comparison on Blender dataset.

Blender We evaluate our method on 360-degree scenes using the Blender dataset. Table 3 presents the quantitative results under 8 input views. Our approach outperforms all baselines in the PSNR score, and achieves comparable SSIM and LPIPS. Figure 5 illustrates the visual comparison results of several 3DGS-based methods. It can be observed that FSGS [46] exhibits noticeable artifacts in the synthesized images due to insufficient Gaussian constraints, and the rendering results of DNGaussian [20] also has some distortion. In contrast, our method performs better in terms of detail preservation.

4.5 Ablation Study

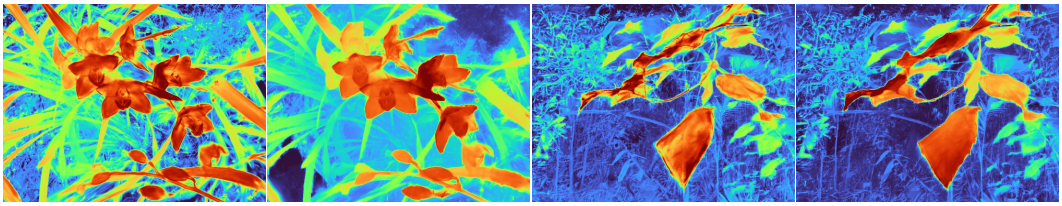
To verify the effectiveness of the view consistency loss, opacity penalty strategy, and the dense initialization for Gaussians, we isolate each of these modules separately while keeping the other modules unchanged. We then evaluate the metrics and compared the visual results. As shown in Table 4, the performance decreases when any of our proposed modules is not used.

Table 4: Ablation studies on LLFF dataset with 3 input views.

Dense Init	L_{consis}	Opacity Penalty	PSNR ↑	SSIM ↑	LPIPS ↓
			16.67	0.494	0.384
✓			19.30	0.651	0.238
✓	✓		20.48	0.715	0.218
	✓	✓	20.82	0.716	0.189
✓		✓	21.13	0.738	0.180
✓	✓	✓	21.44	0.751	0.168

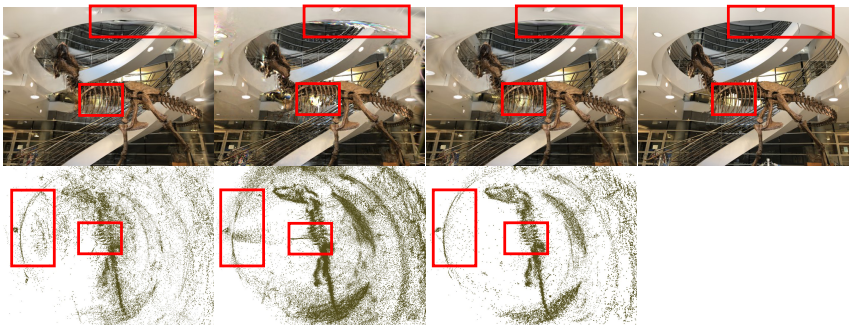
Effectiveness of view consistency loss. To verify the effectiveness of the view consistency loss, we compare the depth maps from novel views. Figure 6 presents the visual comparison of rendered depth maps. The comparison in the figure shows that it is evident that the view consistency loss significantly improves the estimation of scene geometry.

Effectiveness of opacity penalty. Figure 7 (b) and (c) present the comparison of novel view synthesis results and Gaussian point cloud visualizations for the *trex* scene in LLFF before and after



w/o L_{consis} w/ L_{consis} w/o L_{consis} w/ L_{consis}

Figure 6: Visual comparison of depth maps before and after using view consistency loss.



(a) Sparse Init (b) w/o Opacity Penalty (c) w/ Opacity Penalty (d) GT

Figure 7: Visual comparison of novel view images and Gaussian point clouds.

applying the opacity penalty. It can be seen that without the opacity penalty, Gaussians appear far from the surfaces and there is noise near the surfaces. After applying the opacity penalty strategy, the Gaussians far from the scene surfaces are eliminated, and the noisy point clouds are significantly reduced, thereby further improving the rendering quality of novel views.

Effectiveness of dense initialization. Figure 7 (a) and (c) present the novel view images and the final Gaussian point clouds obtained using different initialization point cloud. It can be seen that without dense point cloud initialization, the spatial positions of the Gaussians become arbitrary in some low-texture regions or areas occluded in the training views due to insufficient constraints, resulting in reduced quality of the novel view images.

5 Conclusion

In this paper, we propose a novel method for the task of novel view synthesis from sparse views based on the 3DGS framework. We construct a self-supervised multi-view consistency constraint using the synthetic and input images, and introduce a Gaussian opacity penalty and a dense point cloud initialization strategy. These constraints ensure that the Gaussians are distributed as closely as possible to the scene surfaces and filter out those far from the surfaces. Our approach enables the unstructured Gaussians to accurately represent scene geometry even with sparse input views, resulting in high-quality novel rendering images. Extensive experiments on the LLFF, DTU, and Blender datasets demonstrate that our method outperforms existing state-of-the-art sparse view synthesis methods.

Limitation. Due to the utilization of view consistency constraints to estimate scene depth in our method, some areas with low texture in the scene cannot been accurately estimate the depth, thus failing to constrain the corresponding Gaussians, such as the white background areas in the DTU dataset. This results the white Gaussians to potentially occlude the object in the novel views. In contrast, DNGaussian [20] uses the depth prior estimated by the pre-trained network to constrain the Gaussians, preventing this scenario from happening. The visual comparisons are presented in the appendix.

References

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

[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] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19697–19705, 2023.

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022.

[5] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14124–14133, 2021.

[6] Di Chen, Yu Liu, Lianghua Huang, Bin Wang, and Pan Pan. Geoaug: Data augmentation for few-shot nerf with geometry constraints. In *European Conference on Computer Vision*, pages 322–337. Springer, 2022.

[7] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022.

[8] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023.

[9] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, 2022.

[10] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 270–279, 2017.

[11] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3828–3838, 2019.

[12] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *arXiv preprint arXiv:2311.12775*, 2023.

[13] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[14] Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19774–19783, 2023.

[15] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5885–5894, 2021.

[16] 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.

[17] 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.

[18] Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12912–12921, 2022.

[19] Min-Seop Kwak, Jiuhn Song, and Seungryong Kim. Geconerf: Few-shot neural radiance fields via geometric consistency. *arXiv preprint arXiv:2301.10941*, 2023.

[20] 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. *arXiv preprint arXiv:2403.06912*, 2024.

[21] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Transactions on Graphics (TOG)*, 38(4):1–14, 2019.

[22] 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.

[23] 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.

[24] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5480–5490, 2022.

[25] Avinash Paliwal, Wei Ye, Jinhui Xiong, Dmytro Kotovenko, Rakesh Ranjan, Vikas Chandra, and Nima Khademi Kalantari. Coherentgs: Sparse novel view synthesis with coherent 3d gaussians. *arXiv preprint arXiv:2403.19495*, 2024.

[26] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021.

[27] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637, 2020.

[28] 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.

[29] Seunghyeon Seo, Yeonjin Chang, and Nojun Kwak. Flipnerf: Flipped reflection rays for few-shot novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22883–22893, 2023.

[30] Nagabhushan Somraj and Rajiv Soundararajan. Vip-nerf: Visibility prior for sparse input neural radiance fields. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023.

[31] Jiuhn Song, Seonghoon Park, Honggyu An, Seokju Cho, Min-Seop Kwak, Sungjin Cho, and Seungryong Kim. Därf: Boosting radiance fields from sparse input views with monocular depth adaptation. *Advances in Neural Information Processing Systems*, 36, 2024.

[32] 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.

[33] Jiakai Sun, Zhanjie Zhang, Jiafu Chen, Guangyuan Li, Boyan Ji, Lei Zhao, Wei Xing, and Huaizhong Lin. Vgos: Voxel grid optimization for view synthesis from sparse inputs. *arXiv preprint arXiv:2304.13386*, 2023.

[34] Prune Truong, Martin Danelljan, Radu Timofte, and Luc Van Gool. Pdc-net+: Enhanced probabilistic dense correspondence network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[35] Prune Truong, Marie-Julie Rakotosaona, Fabian Manhardt, and Federico Tombari. Sparf: Neural radiance fields from sparse and noisy poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4190–4200, 2023.

[36] Guangcong Wang, Zhaoxi Chen, Chen Change Loy, and Ziwei Liu. Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9065–9076, 2023.

[37] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[38] Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P Srinivasan, Dor Verbin, Jonathan T Barron, Ben Poole, et al. Reconfusion: 3d reconstruction with diffusion priors. *arXiv preprint arXiv:2312.02981*, 2023.

[39] Jamie Wynn and Daniyar Turmukhambetov. Diffusionerf: Regularizing neural radiance fields with denoising diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4180–4189, 2023.

[40] Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. Sparsegs: Real-time 360 {\deg} sparse view synthesis using gaussian splatting. *arXiv preprint arXiv:2312.00206*, 2023.

[41] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8254–8263, 2023.

[42] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. *arXiv preprint arXiv:2401.10891*, 2024.

[43] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[44] Ning Zhang, Francesco Nex, George Vosselman, and Norman Kerle. Lite-mono: A lightweight cnn and transformer architecture for self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18537–18546, 2023.

[45] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[46] Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. Fsgs: Real-time few-shot view synthesis using gaussian splatting. *arXiv preprint arXiv:2312.00451*, 2023.

A More Visualizations

A.1 Additional Results on LLFF

Figure 8 shows the visualization comparison of our method and DNGaussian [20] across all scenes in the LLFF [21] dataset. Each scene is trained using the same 3 input views. Although both methods achieve perceptually acceptable results, the novel view rendering results of our method are closer to the Ground Truth.

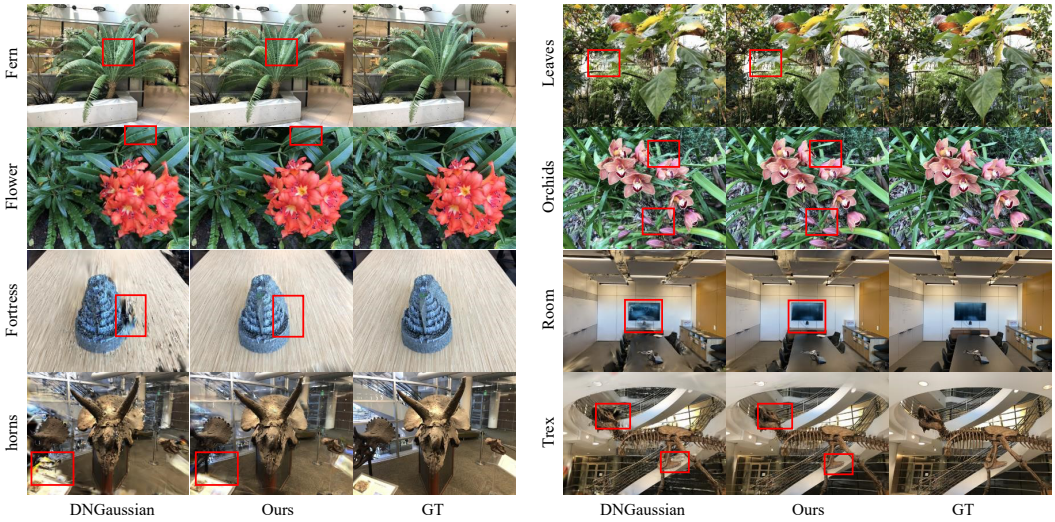


Figure 8: Visual comparison on LLFF dataset with 3 input views.

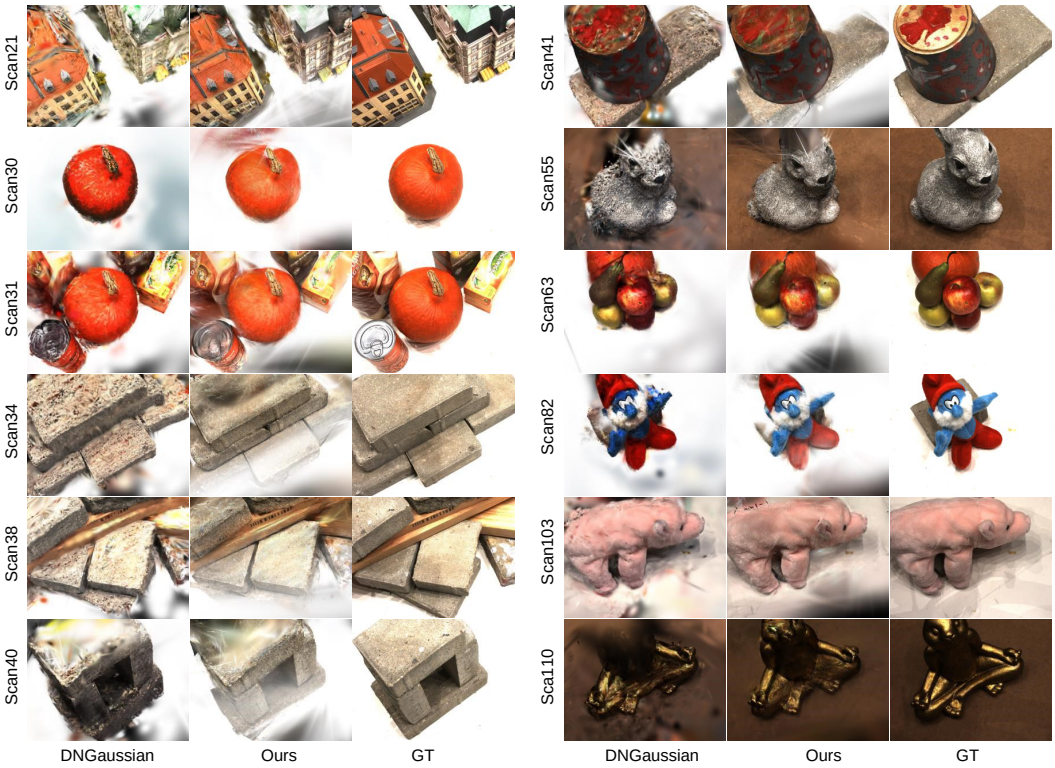
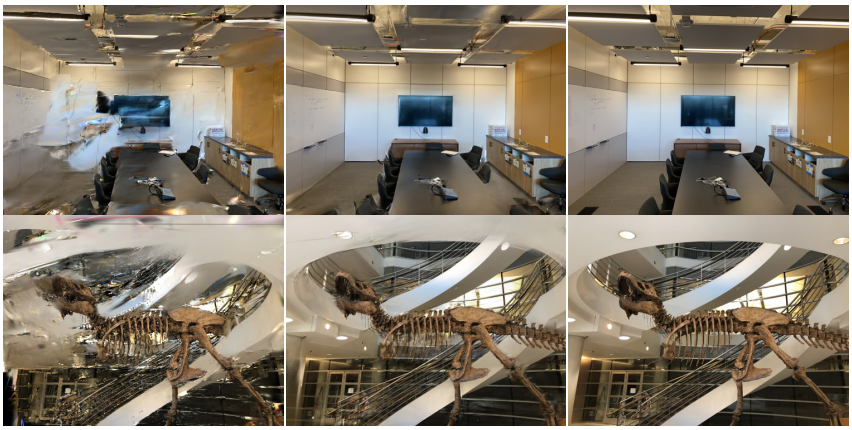


Figure 9: Visual comparison on DTU dataset with 3 input views.

Components	PSNR↑	SSIM↑	LPIPS↓
Opacity Entropy Reg.	15.75	0.480	0.361
Opacity Penalty	21.44	0.751	0.168

Table 5: Quantitative comparison of using different regularization components for opacity.



w/ Opacity Entropy Reg. w/ Opacity Penalty GT

Figure 10: Visual comparison of using different regularization components for opacity.

A.2 Additional Results on DTU

Figure 9 shows additional visualization comparison of our method and DNGaussian [20] on the DTU [16] dataset. Each scene is trained using the same 3 input views. To clearly distinguish the differences between the two methods, we select a novel view that is far from the training views for comparison. As analyzed in the limitations section of the main text, our method cannot effectively constrain the white background in training views, resulting in object being occluded by white Gaussians in some scenes. Even so, the rendering quality of the novel views from our method is still better than the results from DNGaussian.

B More Experiments

B.1 Entropy regularization instead of Opacity Penalty

The Sugar [12] method applies entropy regularization to the opacity of Gaussians to bring Gaussians closer to the surface, similar to our proposed opacity penalty strategy. Therefore, we also evaluated the performance of opacity entropy regularization on the LLFF dataset. Following Sugar [12], the opacity threshold for Gaussians pruning is set to 0.5. Table 5 shows the metrics with opacity entropy regularization. It can be seen that our opacity penalty strategy has advantages. Figure 10 shows the novel view synthesis using different opacity regularization strategies. It can be seen that entropy regularization leads to noticeable overfitting with sparse view inputs, resulting in lower quality novel view rendering.

B.2 Training the DTU with Mask

To eliminate the undesirable effects of the solid color background on novel view rendering results in the DTU dataset, we perform additional experiments by using mask to filter out the background in the training views. The evaluation results are shown in Table 6. Figure 11 illustrates the visual comparison between our method and DNGaussian [20] when using masks in the training views. It can be seen that without the solid color background, our method shows a significant improvement, while the performance of DNGaussian decreases.

Methods	PSNR↑	SSIM↑	LPIPS↓
DNGaussian [20]	18.91	0.790	0.176
DNGaussian$_{mask}$ [20]	14.94	0.699	0.237
Ours	20.71	0.862	0.111
Ours$_{mask}$	22.03	0.875	0.098

Table 6: Quantitative comparison of using background masks for input views on DTU dataset.

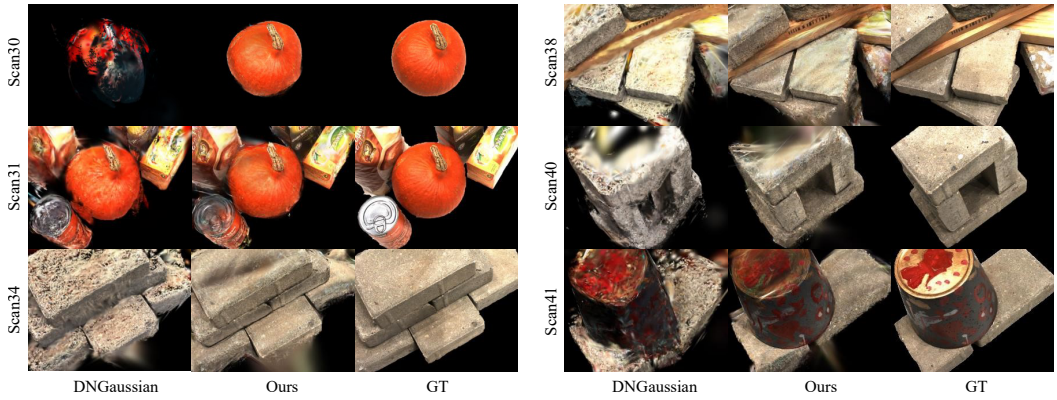


Figure 11: Visual comparison of using background masks for input views on DTU dataset.

C Implementations Details

Since our method uses an opacity penalty strategy, we do not use the opacity reset strategy from the original 3DGS, and we also remove the scale threshold for Gaussians pruning from the original 3DGS.

We train our model on RTX 3090 GPU, running 30,000 iterations per scene on the LLFF and DTU datasets, which takes approximately 9 minutes each. For the Blender dataset, we train for 7,000 iterations per scene, which takes approximately 3 minutes each. The storage space required for the Gaussian point clouds is about one-third of that required by the original 3DGS.

D Dataset Details

LLFF. the LLFF [21] is a forward-facing dataset with 8 scenes. Following previous works [24, 41], we take every 8th image as the novel views for evaluation. The input views are evenly sampled from the remaining views. During training and evaluation, the images are downsampled by a factor of 8, resulting in a resolution of 378×504 for each image.

DTU. The DTU [16] dataset consists of 124 scenes. We follow previous works [24, 41] to train and evaluate our method on 15 test scenes. The test scene IDs are: 8, 21, 30, 31, 34, 38, 40, 41, 45, 55, 63, 82, 103, 110, and 114. In each scene, we use images with the following IDs as input views: 25, 22, 28, 40, 44, 48, 0, 8, 13. The first 3 and 6 image IDs correspond to the input views in 3-view and 6-view settings respectively. We use 25 images as novel views for evaluation, with the following IDs: 1, 2, 9, 10, 11, 12, 14, 15, 23, 24, 26, 27, 29, 30, 31, 32, 33, 34, 35, 41, 42, 43, 45, 46, 47. During training and evaluation, the images are downsampled by a factor of 4, resulting in a resolution of 300 × 400 for each image.

Blender. The Blender [22] dataset consists of 8 synthetic scenes. We follow previous works [24, 41] and use 8 images as input views for each scene, with the following IDs: 26, 86, 2, 55, 75, 93, 16, 73, 8. The 25 test views are sampled evenly from the testing images for evaluation. During training and evaluation, the images are downsampled by a factor of 2, resulting in a resolution of 400 × 400 for each image.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the contributions and scope of the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Our paper includes a dedicated "Limitation" paragraph where potential limitations are discussed.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: Our paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We clearly describe all the information needed to reproduce the experimental results in the paper, including our method, hyperparameter values, and datasets.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We use the publicly datasets and provided sufficient instructions on how to use them for our method in the supplementary materials. The source code will be publicly available.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We specify hyperparameters in the "Implementation Details" section of the paper, with any unspecified hyperparameters being the same as the baselines. The supplementary materials include detailed information on data splits.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We report the average performance in the experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the information of GPU resource we used in the supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research conform with the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: Our paper proposes a method for synthesizing novel view images from sparse input views. The aim is to observe the objects in the input images from new perspectives. There is no generation of new or fake data involved.

    Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original papers of the code and datasets we used.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Our paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.