

MVPGS: Excavating Multi-view Priors for Gaussian Splatting from Sparse Input Views

Wangze Xu¹, Huachen Gao¹, Shihe Shen¹, Rui Peng¹, Jianbo Jiao², and Ronggang Wang^{1,3}

¹ School of Electronic and Computer Engineering, Peking University

² School of Computer Science, University of Birmingham

³ Peng Cheng Laboratory

xuwangze@stu.pku.edu.cn rgwang@pkusz.edu.cn

Abstract. Recently, the Neural Radiance Field (NeRF) advancement has facilitated few-shot Novel View Synthesis (NVS), which is a significant challenge in 3D vision applications. Despite numerous attempts to reduce the dense input requirement in NeRF, it still suffers from time-consuming training and rendering processes. More recently, 3D Gaussian Splatting (3DGS) achieves real-time high-quality rendering with an explicit point-based representation. However, similar to NeRF, it tends to overfit the train views for lack of constraints. In this paper, we propose **MVPGS**, a few-shot NVS method that excavates the multi-view priors based on 3D Gaussian Splatting. We leverage the recent learning-based Multi-view Stereo (MVS) to enhance the quality of geometric initialization for 3DGS. To mitigate overfitting, we propose a forward-warping method for additional appearance constraints conforming to scenes based on the computed geometry. Furthermore, we introduce a view-consistent geometry constraint for Gaussian parameters to facilitate proper optimization convergence and utilize a monocular depth regularization as compensation. Experiments show that the proposed method achieves state-of-the-art performance with real-time rendering speed. Project page: <https://zezeaaa.github.io/projects/MVPGS/>

Keywords: NeRF · Gaussian Splatting · Multi-view Stereo

1 Introduction

Given a set of posed images, novel view synthesis(NVS) aims to render images of unseen views, which has shown great importance in 3D vision applications. Recently, the neural radiance field(NeRF) [24] brought large advancements to the NVS field, which implicitly represents a scene through MLPs and applies volume rendering to generate target views. Various subsequent methods have explored enhancements to NeRF [1,2,25], yielding remarkable progress in practice. More recently, 3D Gaussian Splatting (3DGS) [17] employs a set of point-based Gaussians as scene representations, enabling efficient differentiable splatting for

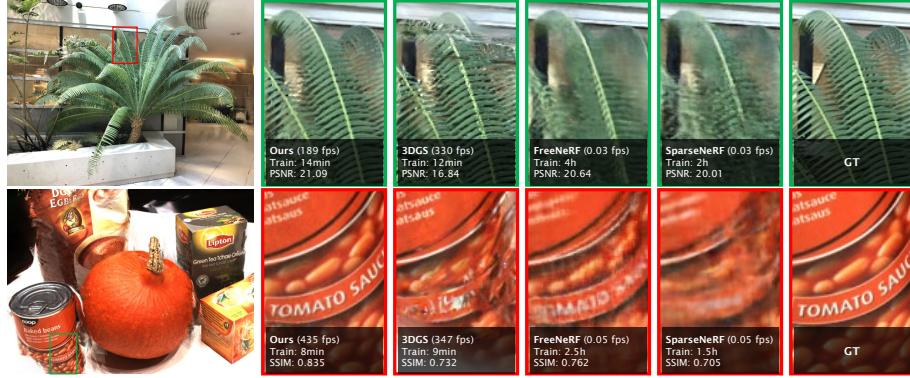


Fig. 1: Qualitative Results on LLFF and DTU Datasets under High-Resolution Setting with 3-view Inputs. Compared with NeRF, the proposed method maintains high-fidelity quality in high-frequency regions and meanwhile achieves competitive training and rendering speed for few-shot NVS.

optimization and achieving high-quality, real-time rendering. While NeRF and 3DGS show impressive results in NVS, both methods require a substantial number of images for training. However, in real-world applications such as robotics and VR/AR, available views are often much sparser. In few-shot situations, these methods struggle to learn accurate 3D structures without enough constraints across views and the performance degrades drastically due to overfitting on training views [8]. To achieve higher quality and efficiency in few-shot NVS, numerous methods have been developed and most of them are built upon NeRF for improvements. Some methods propose additional constraints [13, 18, 20, 26] as regularization to prevent overfitting. Some introduce priors [8, 32, 35, 36, 40, 41], such as depth, to guide NeRF to learn proper scene geometry. However, the rendering quality of these methods remains to be improved and they still suffer from time-consuming training and rendering processes due to the implicit representation and volume rendering.

In this paper, we propose MVPGS, a few-shot NVS method based on 3D Gaussian Splatting [17], which leverages recent learning-based Multi-view Stereo [11, 48, 49] to excavate more cues from sparse input views, achieving both high-quality and high-speed rendering. Similar to NeRF, 3DGS encounters overfitting problems during optimization under limited view constraints. However, unlike NeRF’s blank MLPs, 3DGS’s initial points contain part of the appearance and geometry priors of scenes. The proposed method explores the potential information from this explicit initialization to alleviate overfitting and preserve accurate geometries of scenes during optimization.

Specifically, we use MVSFormer [4] to estimate dense view-consistent depths as initialization for 3D Gaussians. Our experiments (see in Tab. 1) show that the optimization of 3DGS benefits well from such richer initial information compared with original COLMAP’s [33, 34] outputs, especially in few-shot situations.

Furthermore, to mitigate overfitting, we utilize a forward-warping method [15] to conduct additional appearance constraints conforming to scenes for other unseen observations, which can be seamlessly integrated with the initial positions of 3DGS derived from MVS. Since the warped locations might be floating-point numbers and not exactly aligned with an image grid, we utilize reversed bilinear sampling [42] to distribute colors to the local regions. Our approach differs from previous methods [6, 20] (as illustrated in Fig.3) where the appearance for constraints is fetched using the rendered depth of target view, which is called 'backward warping' as discussed in [42]. While these methods enhance the consistency between views, converging for large variations of camera poses in sparse views is difficult. Besides, compared to approaches [22, 26, 44] that rely on plausible signals from other models, e.g. normalizing flow model [9] or diffusion model [12], to supervise the rendering of unseen views, our method provides appearance information that conforms to scenes through geometric solutions. More importantly, our approach naturally adapts to 3DGS's initialization and broadens the scope of available training data, even if it is derived from known views.

Furthermore, the position parameters in 3DGS are directly updated by the back-propagation gradient, which may lead to deviations from accurate geometry during few-shot optimization (see leaves in Fig. 7). To facilitate convergence during optimization, we introduce a loss between 3DGS's geometry and the confident geometric structure computed from MVS. Additionally, MVS may have poor performance in areas such as textureless and low overlap [49], we incorporate monocular depth priors [31] to further constrain the global geometry of scenes and mitigate the influence of inaccurate warped appearance priors caused by imprecise MVS depths.

Our contributions include

- We propose a novel paradigm for few-shot Novel View Synthesis (NVS) which utilizes an efficient point-based Gaussian representation and is seamlessly integrated with geometric priors from Multi-View Stereo (MVS).
- We introduce forward warping for Gaussian initialization obtained from MVS to extract potential appearance information as constraints for unseen views. Additionally, we employ geometry constraints to ensure the proper convergence of 3D Gaussians.
- Experiments on *LLFF*, *DTU*, *NVS-RGBD*, and *Tanks and Temples* datasets demonstrate our method's state-of-the-art performance while maintaining real-time rendering speed.

2 Related Work

Neural Representations for 3D Reconstruction. Neural Radience Field (NeRF) [24] has shown great power in novel view synthesis. Different from previous explicit scene representations such as point clouds, voxels, and meshes, NeRF is a compact paradigm that represents scenes with MLPs implicitly. Let a NeRF be a function f that maps 3D coordinates (x, y, z) and view direction d to corresponding color c and density σ . With the volume rendering formulation

$C = \sum_i T_i \sigma_i c_i$ [24], we can derive each pixel’s color by integrating colors and densities along a ray from the camera to the scene. NeRF brought a new paradigm for scene representations and tremendous follow-ups emerged focusing on improving the rendering quality [1, 2], accelerating [25, 37], 3D generation [21, 46, 51], dynamic scenes [10, 29], and generalizing to unseen scenes [5, 7, 27, 52]. Among them, Mip-NeRF [1] replaces point-based ray tracing with cone tracing to alleviate aliasing and Mip-NeRF360 [2] then extends Mip-NeRF to unbounded scenes. Though these methods achieve excellent rendering results, the training and rendering time (several days to hours) is unbearable. For accelerating, Instant-NGP [25] proposed a multi-resolution hash encoding to compress the training time to seconds. Recently, 3D Gaussian Splatting (3DGS) [17] has shown a great prospect for real-time rendering. Instead of time-consuming volume rendering, 3DGS takes point clouds as initialization and utilizes an alpha blending method for the rasterization. Benefiting from this explicit representation, 3DGS is highly efficient compared to NeRF. However, its performance degrades drastically with insufficient inputs due to the similar overfitting in NeRF.

Few-shot Novel View Synthesis. There are numerous pieces of research to improve the performance of NeRF with sparse input views. Some methods add additional constraints to alleviate over-fitting. RegNeRF [26] applies appearance regularization from normalizing flow model [9] and conducts depth smooth loss on unseen views. FreeNeRF [47] proposes a frequency regularization to solve the overfitting problem caused by high-frequency components. ViPNeRF [35] revises the framework of the original NeRF to output the visibility of points and regularize visibilities and depths on input views. Another branch introduces priors such as depths or semantics to supervise NeRF to learn proper information about scenes [32, 40]. DS-NeRF [8] firstly uses sparse depth generated from COLMAP [33, 34] to supervise the depth distribution along a ray. SparseNeRF [41] utilizes an effective scale-invariant loss to distill geometric cues from monocular depth estimation. Diet-NeRF [13] uses CLIP [30] to add a semantic consistency between views. Different from previous few-shot NVS methods, our method is built on explicit point-based representation and achieves both real-time rendering and competitive rendering quality.

Muti-View Stereo Reconstruction. Recent learning-based MVS [4, 11, 28, 45, 48, 49, 54] has demonstrated remarkable advancements in 3D reconstruction. Typically, it constructs a cost volume by warping 2D image features into different front-parallel planes of the reference camera. Subsequently, 3D CNNs are employed for cost regularization and depth regression [49]. MVS exhibits better robustness to noise and occlusions, achieving higher accuracy compared to traditional methods [33, 34] that rely on handcrafted features for matching. Our approach leverages the MVS method [4] enhanced with Vision Transformers (ViTs) to thoroughly excavate clues from few-shot inputs and facilitate the optimization of 3DGS.

3 Method

Fig. 2 illustrates the overall framework of MVPGS. Our approach is built upon the highly efficient 3D Gaussian splatting [17], which employs point-based explicit representation for scenes (Sec. 3.1). To address overfitting in few-shot scenarios, we utilize a forward warping method to extract appearance information as constraints for unseen views based on geometry computed from MVS (Sec. 3.2). Moreover, we conduct view-consistent geometric regularization to preserve the accurate structure of Gaussian parameters and apply monocular depth priors as compensation during optimization (Sec. 3.3).

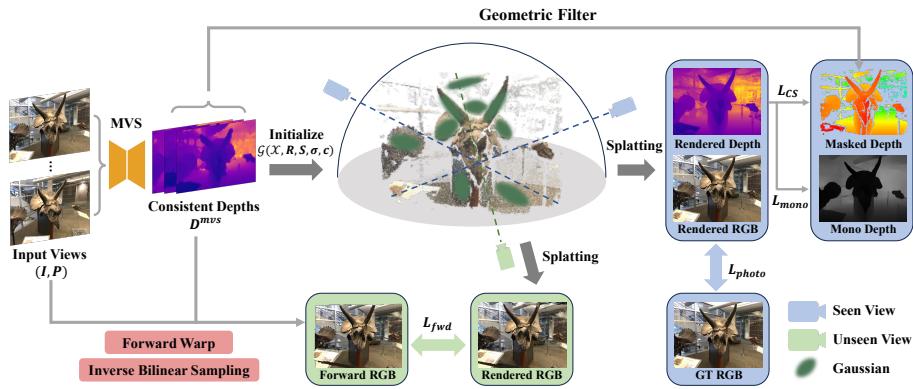


Fig. 2: Framework Overview. MVPGS leverages learning-based MVS to estimate dense view-consistent depth D^{mvs} and construct a point cloud \mathcal{P} for the initialization of Gaussians \mathcal{G} . We excavate the computed geometry from MVS through forward warping to generate appearance priors for the supervision of unseen views. To regularize the geometry update during optimization, we introduce L_{CS} from MVS depth and L_{mono} from monocular depth priors to guide Gaussians to converge to proper positions.

3.1 Preliminary for Gaussian Splatting

3D Gaussian Splatting [17] represents scenes explicitly with a set of point-based Gaussian parameters. Each Gaussian distribution is defined by a 3D covariance matrix Σ and a center position at point (mean) \mathcal{X} in world space [17]:

$$G(\mathcal{X}) = e^{-\frac{1}{2}\mathcal{X}^T \Sigma^{-1} \mathcal{X}}. \quad (1)$$

To reduce the difficulty of optimizing, Σ is decomposed into two learnable parameters with physical significance:

$$\Sigma = \mathbf{R} \mathbf{S} \mathbf{S}^T \mathbf{R}^T, \quad (2)$$

where \mathbf{R} is a quaternion rotation matrix and \mathbf{S} is a scaling matrix. Besides, each Gaussian also has attributions of opacity σ and radiance c represented by spherical harmonics (SH) for rendering. The complete Gaussian parameter is defined by $\mathcal{G} = \{(\mathcal{X}_i, \mathbf{S}_i, \mathbf{R}_i, \sigma_i, c_i)\}_{i=0}^{i=n}$.

When rendering, a splatting technique [17] is utilized and each 3D Gaussian is projected to 2D image space for rasterization. With a viewing transform W , we can obtain the covariance Σ' in camera coordinate:

$$\Sigma' = JW\Sigma W^T J^T, \quad (3)$$

where J is the Jacobian of the affine approximation of the projective transformation. After projecting Gaussians into 2D, all Gaussians covering a pixel are sorted and each pixel's color C is computed by blending the overlapping Gaussians:

$$C = \sum_{i \in N} T_i \alpha_i c_i, \quad (4)$$

where $T_i = \prod_{j=1}^{j=i-1} (1 - \alpha_j)$ is the transmittance, α_i is σ_i multiplied by a 2D Gaussian with covariance Σ' , and c_i is i -th Gaussian's color based on spherical harmonics. The photometric loss for optimizing parameters is given by a combination of L_1 and SSIM [43] loss:

$$L_{photo} = \lambda_1 L_1(\hat{I}, I) + (1 - \lambda_1)(1 - SSIM(\hat{I}, I)) \quad (5)$$

where \hat{I} and I are the rendered image and the ground-truth, respectively. λ_1 is a controlling weight which is set to 0.8 in experiments.

3.2 Multi-view Stereo Priors for Optimization

3D Gaussian Initialization. We leverage MVSFormer [4], a learning-based MVS method enhanced by Vision Transformers (ViTs), to excavate more cues from sparse inputs. Given a set of input views $\{V_i^{train} = (I_i, P_i)\}_{i=0}^{i=N}$ (where I, P denote images and corresponding camera pose, respectively), we consider each view V_i as reference view and the remaining as source views. For each reference view, we warp all source views' 2D features to construct a 3D cost volume and regress the depth map D_i^{mvs} . Then we employ geometric filter [49] based on depth reproject errors and areas with high consistency are indicated by masks $\{M_i\}_{i=0}^{i=N}$. With $\{D_i^{mvs}\}$ and $\{M_i\}$, we use method in [49] to merge them into a set of 3D positions $\{(x_i, y_i, z_i)\}_{i=0}^{i=n}$, which is regarded as the mean of initial 3D Gaussian's \mathcal{X} . More details are in the supplementary material (Supp. Mat.).

Forward Warping Appearance for Unseen Views. In contrast to NeRF's lack of scene knowledge, the initialization of Gaussian parameters incorporates potential priors about scenes. Through the ViT-based MVS model, we can acquire richer scene information compared to the original COLMAP outputs, particularly in sparse input scenarios. We further exploit these priors to alleviate

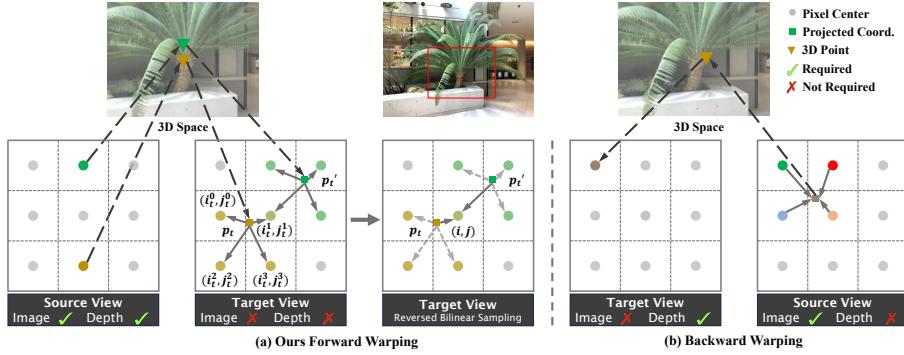


Fig. 3: Forward Warping and Backward Warping. Forward warping (a) takes $(I_{src}, D_{src}^{mvs}, P_{src}, P_{tgt})$ as input and output the appearance I_{tgt} in pose P_{tgt} based on the known geometry D_{src}^{mvs} . Owning to the warped location p_t is not necessarily lie in the center of the grid, we adopt reversed bilinear sampling [42] to determine the color of each pixel. This is different from backward warping (b) which uses bilinear sampling to sample color from the source view according to the target view's geometry.

overfitting in few-shot situations. The key idea is to leverage the computed geometry from MVS to obtain the appearance of unseen views around training views by forward warping, which can be seamlessly integrated into 3DGS's pipeline. Our approach differs from previous methods [6, 20], which is termed 'backward warping' as discussed in [42] and is used widely in self-supervised monocular depth estimation [55]. Backward warping utilizes the target view's geometry to collect appearance from other views, while the proposed method infers the target view's appearance based on the known view's geometry and appearance.

Given a known training view V_{src} with corresponding image I_{src} , MVS depth D_{src}^{mvs} , camera pose P_{src} and target view's pose P_{tgt} , the forward warping is defined as

$$I_{tgt} = fwd(I_{src}, D_{src}^{mvs}, P_{src}, P_{tgt}). \quad (6)$$

where I_{tgt} is the image in target view V_{tgt} warped from V_{src} . To establish a more robust mapping relationship between pixels in two views, we combine forward warping with reversed bilinear sampling [42]. The process is concluded as follows: 1) find the corresponding locations in the target view for each pixel in V_{src} , 2) find the nearest pixels influenced by the warped location, 3) compute the color contribution to the target view's pixels [16]. Specifically, the coordinate relationships between V_{src} and V_{tgt} is

$$p_{tgt} \sim KP_{tgt}^{-1}P_{src}D_{src}^{mvs}K^{-1}p_{src}. \quad (7)$$

where K is the camera intrinsic. p_{src} and p_{tgt} denote the pixel coordinates in V_{src} and the warped coordinates in V_{tgt} . Given a pixel $p_s = (x_s, y_s) \in p_{src}$ and its depth d_s^{mvs} in V_{src} , p_s can be backproject to a 3D point τ and then projected to V_{tgt} to get the coordinate $p_t = (x_t, y_t)$. The color c_t at position p_t can be viewed as the corresponding color $I_{src}(p_s)$ in V_{src} , under the assumption that

the color remains constant across different views. In practice, p_t is fractional and may not religiously lie in the pixel center. To address this problem, we utilize reversed bilinear sampling [42], which distributes the weight of the projected pixel to its nearest neighbors. Specifically, we choose the 4 nearest pixel centers $\{(i_t^m, j_t^m)\}_{m=0}^{m=3}$ around p_t as the areas influenced by p_t (illustrated in Fig. 3). The color's contribution to these 4 pixels from the warped point is determined by the distance [42]:

$$w_t(i, j) = (1 - |i - x_t|)(1 - |j - y_t|) \quad (8)$$

With Eq. 8, pixels in target views can be colored with corresponding projected points p_t . Another problem is that different points may map to the same location which causes occlusion. To solve this, we align higher weights for points that are closer to the screen. Following [16], the depth weight is computed as $w_t^d = \frac{1}{(1+d_t)^\gamma}$, where d_t is p_t 's depth in V_{tgt} . γ is a hyperparameter given by $\gamma = \frac{50}{\log(1+d^{max})}$, where d^{max} is the maximum of projected depths in V_{tgt} for all $p_t \in p_{tgt}$ [16]. The final color of the target view pixel is

$$I_{tgt}(i, j) = \frac{\sum_{p_t \in \Omega} w_t(i, j) w_t^d c_t}{\sum_{p_t \in \Omega} w_t(i, j) w_t^d}, \quad (9)$$

where Ω denotes all mapped points that contributes to pixel (i, j) .

For additional supervision in unseen views, we follow [26] to sample random pose P' and generate its forward warping appearance I' according to the Eq. 9. During optimization, we render the image \hat{I}' with P' from Gaussian parameters. We use a combination of L_1 and SSIM loss for supervision:

$$L_{fwd} = \lambda_2 L_1(\hat{I}', I') + (1 - \lambda_2)(1 - SSIM(\hat{I}', I')) \quad (10)$$

where λ_2 is set to 0.2 in our experiments.

3.3 Geometric Regularization

The geometry of scenes can be reflected by the Gaussian's mean parameter \mathcal{X} , which is updated directly through back-propagate gradients during optimization. In practice, these parameters tend to have difficulty converging to correct positions when constraints from input views are insufficient. To facilitate convergence, we introduce two geometric constraints derived from MVS outputs and recent monocular depth priors [31]. It is difficult to directly control the geometry of Gaussians through regularizing these 3D positions. Therefore, similar to NeRF [8, 41, 56], we render the depth map as a proxy by blending the z-buffer of the sorted Gaussians contributing to pixels:

$$d = \sum_{i \in N} d_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (11)$$

where d_i is the depth of i -th Gaussian and α_i is identical to that in Eq. 4.

Consistent Structure Regularization. The MVS depths computed in Sec. 3.2 are checked by geometric consistency and serve as the initialization for Gaussian parameters. These reliable geometric structures are expected to remain consistent throughout the optimization process. Therefore, we render the depth D_i^r from Gaussian parameters with Eq. 11 for V_i^{train} and impose a L_1 loss with MVS depths in areas with high confidence:

$$L_{CS} = \sum |D^r - D^{mvs}| \odot M \quad (12)$$

where D^{mvs} is the MVS depth map and M is the mask computed in 3.2. While the geometry in D^{mvs} is equivalent to the initialization positions \mathcal{X} and does not provide additional information about scenes, this term serves to regularize the shape to maintain consistency during optimization.

Global Geometry Regularization. MVS depth may not be consistent in certain areas. For regions lacking consistent structure regularization, we use monocular depth priors as compensation. Specifically, we utilize the ViT-based DPT [31], which is trained on large-scale datasets and demonstrates strong generalization capabilities on other scenes, to predict depth map D_i^{mono} for each view V_i^{train} . Inspired by [38, 41], we use a random rank loss to avoid the unknown scales of monocular depths and distill the relative position relationships for whole scenes including regions without MVS depth supervision. Specifically, we random sample two disjoint batches of pixel positions \mathcal{S}_1 and \mathcal{S}_2 with the same size $n_{\mathcal{S}}$. The monocular rank loss is formulated as

$$L_{mono} = \sum M' \max(0, D^r(\mathcal{S}_1) - D^r(\mathcal{S}_2)) + \neg M' \max(0, D^r(\mathcal{S}_2) - D^r(\mathcal{S}_1)) \quad (13)$$

where $M' = \mathbb{I}_{D^{mono}(\mathcal{S}_1) < D^{mono}(\mathcal{S}_2)}$ and $n_{\mathcal{S}}$ is set to 512 in experiments.

3.4 Training

The total loss for optimizing 3D Gaussian parameters is

$$L_{total} = L_{photo} + L_{fwd} + \beta_1 L_{CS} + \beta_2 L_{mono} \quad (14)$$

where β_1 and β_2 are hyper-parameters. In practice, we sample unseen view every \mathcal{E} iteration in which only L_{fwd} is computed for optimization. Additionally, we employ the adaptive pruning and densification strategy [17] to dynamically adjust Gaussians during training. The training details are shown in Supp. Mat.

4 Experiments

4.1 Experimental Setup

Datasets. We report results on four real-world datasets, LLFF [23], DTU [14], NVS-RGBD [41] and T&T (Tanks and Temples) [3, 19]. LLFF consists of 8

complex forward-facing scenes. We use every 8-th image as the held-out test set and evenly sample from the remaining views for training. We report the performance at two different resolutions of $1/8$ (504×378) and $1/4$ (1008×756). DTU is a multi-view dataset that contains different objects placed on a table. We use the same 15 scenes and dataset split in previous methods [26, 41, 47] and conduct experiments at resolutions of $1/4$ (400×300) and $1/2$ (800×600). NVS-RGBD is an RGBD dataset containing three branches, each branch consists of 8 scenes. We adopt Kinect and ZED2 for experiments with 3 given views for training and the rest for evaluation. T&T is a complex large-scale dataset containing both outdoor and indoor scenes. We use the first 50 frames of each scene for experiments and adhere to LLFF’s dataset-split scheme.

Metrics. We adopt the mean of PSNR, SSIM [43], and LPIPS [53] as quantitative metrics. For DTU, we follow previous few-shot methods [26, 41, 47] to remove the background with object masks in evaluation.

Implementation Details. Our method is built upon 3DGS [17] and implemented with Pytorch and CUDA. We use pre-trained MVSFormer [4] to get the view-consistent depth maps of training views. MVS depths are filtered according to geometric consistency and merged into point clouds [49] as the initial Gaussian positions. Considering the black background is irrelevant to objects, we use the filtered MVS depth maps with masks M for DTU to only warp foreground regions. For LLFF, NVS-RGBD, and T&T datasets, we use lama [39] to inpaint the warped image for border pixels. The monocular depths for input views are estimated by DPT [31]. We report the performance of our methods with 10k (*Ours*[†]) or 20k (*Ours*) training iterations. We densify and prune Gaussians every 100 iterations from 500 until 5k and 10k iterations, respectively. We set $\beta_1 = 0.1$, $\beta_2 = 0.005$ ($\beta_2 = 0.01$ for DTU) in experiments. FPS metrics in Tab. 1 are tested on a single V100 GPU. For more details please refer to Supp. Mat.

4.2 Comparison

We compare performances with state-of-the-art per-scene optimization methods including RegNeRF [26], FreeNeRF [47], and SparseNeRF [41]. We also compare the performance of original Mip-NeRF [1] and 3DGS with sparse inputs. The training set and testing set are retained to be the same across all methods.

Comparison on DTU. We report the quantitative results of the proposed method compared to state-of-the-art NVS methods. As shown in Tab. 1, our approach outperforms the compared methods in terms of rendering quality and achieves real-time rendering speed of 366 FPS (res. $1/2$), which is faster than previous NeRF methods. Fig. 4 shows the rendered images in novel views under the $1/4$ resolution setting. We observed that our method performs better in capturing the overall object structure and rendering finer details in local regions. Furthermore, the point-based representation enables our approach to preserve fine details even in high-resolution settings (see Fig. 1). The overall quantitative results in Tab. 1 also indicate that our approach possesses better adaptability to high-resolution rendering in sparse input scenarios. More rendering results on DTU can be found in Supp. Mat.

Table 1: Quantitative Results on LLFF and DTU with 3 Input Views. Our approach achieves competitive performances across different resolutions. 3DGS* denotes 3DGS initialized with our MVS point clouds. We mark the **best**, **second best**, and **third best** methods in cells, respectively.

Methods	LLFF								DTU							
	res. 8 ×				res. 4 ×				res. 4 ×				res. 2 ×			
	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑	PSNR ↑ SSIM ↑ LPIPS ↓ FPS ↑
3DGS	14.87	0.438	0.383	309	14.40	0.415	0.428	343	14.90	0.783	0.183	326	14.07	0.764	0.209	366
3DGS*	16.16	0.519	0.331	315	15.92	0.504	0.370	351	15.45	0.814	0.158	359	15.18	0.801	0.177	355
MipNeRF	14.62	0.351	0.495	0.07	15.53	0.416	0.490	0.03	8.68	0.571	0.353	0.18	8.86	0.607	0.344	0.05
RegNeRF	19.08	0.587	0.336	0.07	18.40	0.545	0.405	0.03	18.89	0.745	0.190	0.18	18.62	0.750	0.232	0.05
FreeNeRF	19.63	0.612	0.308	0.07	19.12	0.568	0.393	0.03	19.92	0.787	0.182	0.18	19.84	0.770	0.240	0.05
SparseNeRF	19.86	0.620	0.329	0.07	19.30	0.565	0.413	0.03	19.46	0.766	0.204	0.18	19.63	0.762	0.242	0.05
Ours†	20.54	0.727	0.194	209	19.91	0.696	0.229	193	20.65	0.877	0.099	531	20.24	0.858	0.124	425
Ours	20.39	0.715	0.203	132	19.70	0.681	0.241	170	20.50	0.871	0.106	365	20.26	0.851	0.130	366

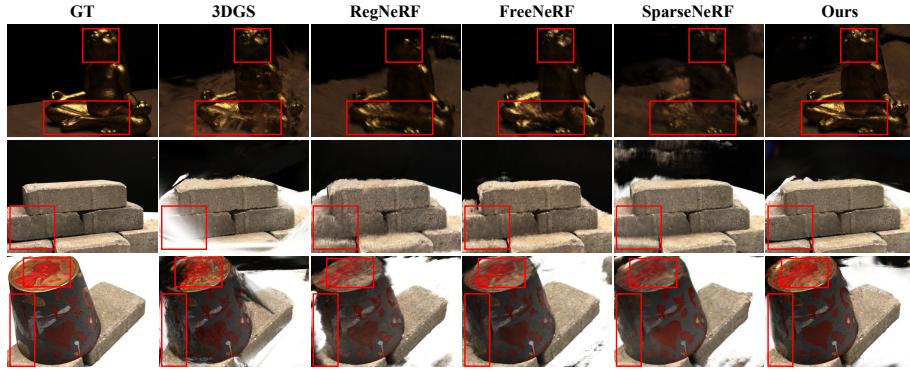


Fig. 4: Qualitative Results on DTU with 3 Input Views.

Comparison on LLFF. Tab. 1 presents the quantitative results of our method on the more complex scene-level LLFF dataset. Our method achieves the highest scores in all rendering quality metrics across different resolutions. In particular, the higher SSIM metrics indicate that our rendering quality aligns more closely with subjective human perception. Moreover, our approach is more practical for real-world applications for it maintains a real-time rendering speed of 132 FPS (res. 1/4). Fig. 5 illustrates the visual quality of novel views in LLFF under 1/8 resolution. In the horns and trex scenes, the proposed method, benefiting from constraints of multi-view consistent priors, preserves more accurate and complete structures compared to other methods. Quantitative metrics in Tab. 1 also demonstrate our approach achieves competitive rendering quality under high resolution, particularly in high-frequency areas (see fern in Fig. 1). For more visualizations on LLFF please refer to Supp. Mat.

Comparison on T&T and NVS-RGBD. Results in Tab.2 and Fig. 6 show that the proposed method also achieves better performance in large-scale scenes and in-the-wild situations. Compared to 3DGS and previous NeRF-based methods, our approach adopts view-consistent constraint and is capable of rendering better details in edge areas where scene geometry changes abruptly (see visual

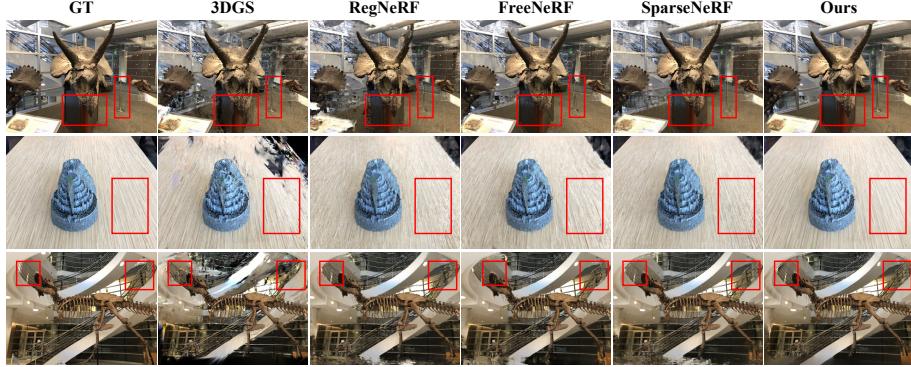


Fig. 5: Qualitative Results on LLFF with 3 Input Views.

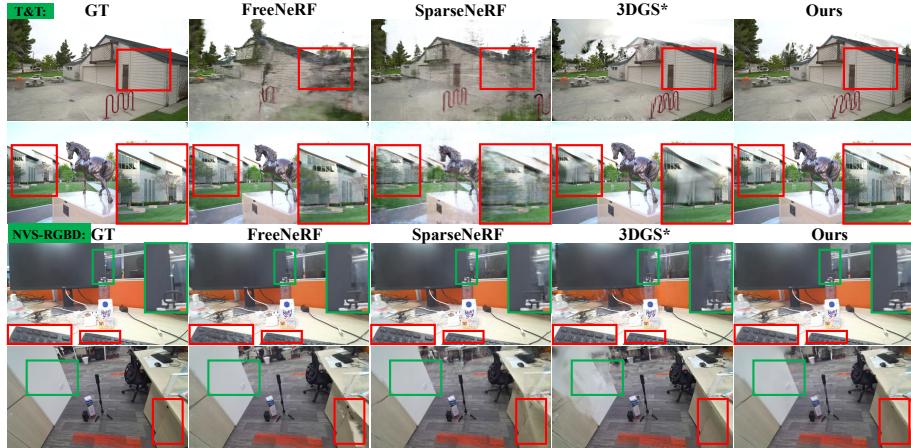


Fig. 6: Qualitative Results on T&T and NVS-RGBD with 3 Input Views.

quality on NVS-RGBD in Fig. 6). Besides, the performance on T&T demonstrates our method’s better robustness in large-scale outdoor scenes where the geometry of scenes is more complex and the variation of camera positions is quite large. Previous methods suffer from fitting the accurate geometry due to insufficient view constraints even with high-quality initialization of Gaussians (3DGS* in Tab. 2). Our methods leverage multi-view priors for additional supervision of unseen views, which improves the quality of scene reconstruction.

Gaussian Initialization. In Tab. 1 and Tab. 2, we report quantitative results of the original 3DGS equipped with our MVS point clouds (3DGS*). 3DGS with our MVS initialization outperforms its counterpart with COLMAP points in all metrics. This shows that the learning-based MVS can provide richer initialization (please refer to Supp. Mat.), which contributes to the subsequent optimization. However, the improvement brought by only replacing better initialization is limited since the overfitting problem during few-shot optimization still exists.

Table 2: Quantitative Results on T&T and NVS-RGBD with 3 Input Views.

Methods	NVS-RGBD(ZED2)			NVS-RGBD(Kinect)			T&T		
	PSNR ↑ SSIM ↑ LPIPS ↓								
3DG*(<i>MVS init.</i>)	21.95	0.749	0.252	20.59	0.770	0.274	22.05	0.769	0.207
FreeNeRF	24.74	0.780	0.269	26.47	0.856	0.206	18.49	0.533	0.463
SparseNeRF	26.26	0.805	0.261	26.30	0.848	0.232	20.62	0.611	0.414
Ours	26.62	0.841	0.185	27.04	0.887	0.151	25.57	0.846	0.139

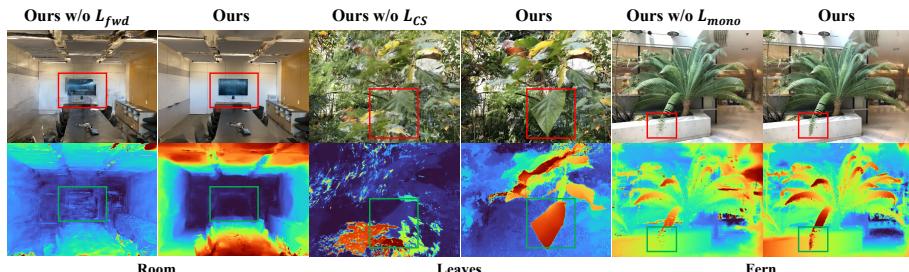
Table 3: Quantitative Results with Different Input Views on LLFF. The proposed method achieves the best in most metrics under different input-view settings.

Methods	2 view			3 view			4 view			5 view		
	PSNR ↑ SSIM ↑ LPIPS ↓											
3D-GS	13.39	0.322	0.459	14.87	0.438	0.383	16.78	0.541	0.321	18.21	0.606	0.284
Mip-NeRF	12.44	0.205	0.581	14.62	0.351	0.495	19.41	0.628	0.311	20.84	0.667	0.284
RegNeRF	16.32	0.404	0.442	19.08	0.587	0.336	21.10	0.692	0.271	21.81	0.712	0.258
FreeNeRF	16.99	0.473	0.366	19.63	0.612	0.308	21.28	0.703	0.255	22.57	0.725	0.247
SparseNeRF	17.44	0.447	0.423	19.86	0.620	0.329	21.42	0.696	0.283	21.89	0.695	0.288
Ours	18.53	0.607	0.280	20.39	0.715	0.203	21.28	0.750	0.180	22.18	0.773	0.164

Training Views. Tab. 3 shows the performance of the proposed methods in different input-view settings compared to other methods on LLFF under 1/8 resolution. Our method excels in most metrics, demonstrating robust performance across various training view selections. Moreover, the proposed method maintains competitive performance even in the 2-view extremely sparse setting.

4.3 Ablation Studies

In this section, we ablate our design choices on the LLFF dataset with the 3-view input setting under 1/8 resolution.

**Fig. 7: Ablation Study of Visual Quality.** We compare the RGB and depth of novel views to verify the effectiveness of components in our method.

Forward Warping Prior. Tab. 4 shows that the performance degrades significantly when the forward-warping appearance constraint for unseen views is

Table 4: Quantitative Ablation **Table 5: Impact of pre-trained MVS Results on the LLFF Dataset.** **models on LLFF Dataset.**

Methods	PSNR ↑ SSIM ↑ LPIPS ↓
Baseline	14.87 0.438 0.383
Baseline w/ <i>mvs pc</i>	16.16 0.519 0.331
Ours w/o L_{fwd}	17.32 0.563 0.306
Ours w/o L_{CS}	20.06 0.699 0.218
Ours w/o L_{mono}	20.17 0.708 0.209
Ours	20.39 0.715 0.203

LLFF	PSNR ↑ SSIM ↑ LPIPS ↓
Baseline	14.87 0.438 0.383
DTU	20.39 0.715 0.203
BlendedMVS	20.17 0.709 0.213
DTU and BlendedMVS ft	20.34 0.722 0.202

absent. In the room scene in Fig. 7, the quality of the novel view appears blurry without L_{fwd} , indicating poor geometries of the scene caused by overfitting. Leveraging appearance priors computed from MVS, our method learns more reasonable Gaussian parameters and achieves higher-quality rendering results.

Consistent Structure Regularization. The leaves scene in Fig. 7 illustrates that the proposed consistent depth loss contributes to maintaining proper positions during optimization. Though Gaussian parameters \mathcal{X} are initialized with the corresponding point clouds in the green box, they tend to overfit to improper positions in complex scenes. Our approach corrects geometric errors during optimization by utilizing confident view-consistent depths for regularization. Quantitative results in Tab. 4 also demonstrate the effectiveness of this constraint.

Global Geometry Regularization. Tab. 4 and Fig. 7 also demonstrates the improvement from monocular depth priors. In regions where MVS depths are filtered out due to insufficient view consistency (Sec. 3.2), Gaussians may encounter difficulty in fitting the geometry and appearance of scenes (as observed in the fern scene in Fig. 7). With the compensatory constraint of monocular depth, our method can learn more reasonable geometric structures.

Different Pre-trained MVS Models. We test the performance with different pre-trained models. Specifically, we test models trained on BlendedMVS [50], DTU [14] and also trained on DTU then finetuned on BlendedMVS. Tab. 5 demonstrates our approach is robust to the pre-trained data of MVS model.

5 Conclusion

In this paper, we propose a novel paradigm for real-time few-shot novel view synthesis based on 3D Gaussian Splatting. The key insight is to excavate the multi-view cues derived from the data-driven MVS method for limited input views. The geometry computed from MVS serves as the initialization of Gaussians for capturing more information about scenes. To alleviate overfitting, we introduce a forward warping method to generate appearance priors for unseen views. To facilitate convergence during optimization, we propose a view-consistent depth loss in regions with confident geometry and also a monocular depth loss for additional geometry constraints. Experiments show that our method achieves state-of-the-art performance in the task of few-shot novel view synthesis.

Limitation. The proposed method mitigates the overfitting problem during optimization with the view-consistent appearance priors, but it does not deal with the light variation of non-Lambertian surfaces.

Acknowledgments

This work is financially supported by Outstanding Talents Training Fund in Shenzhen, Shenzhen Science and Technology Program-Shenzhen Cultivation of Excellent Scientific and Technological Innovation Talents project (Grant No. RCJC20200714114435057), Shenzhen Science and Technology Program-Shenzhen Hong Kong joint funding project (Grant No. SGDX20211123144400001), Guangdong Provincial Key Laboratory of Ultra High Definition Immersive Media Technology, National Natural Science Foundation of China U21B2012, R24115SG MIGU-PKU META VISION TECHNOLOGY INNOVATION LAB. Jianbo Jiao is supported by the Royal Society Short Industry Fellowship (SIF\R1\231009).

References

1. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5855–5864 (2021)
2. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5470–5479 (2022)
3. Bian, W., Wang, Z., Li, K., Bian, J.W., Prisacariu, V.A.: Nope-nerf: Optimising neural radiance field with no pose prior. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4160–4169 (2023)
4. Cao, C., Ren, X., Fu, Y.: Mvsformer: Learning robust image representations via transformers and temperature-based depth for multi-view stereo. arXiv preprint arXiv:2208.02541 (2022)
5. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14124–14133 (2021)
6. Chen, D., Liu, Y., Huang, L., Wang, B., Pan, P.: Geoaug: Data augmentation for few-shot nerf with geometry constraints. In: European Conference on Computer Vision. pp. 322–337. Springer (2022)
7. Chibane, J., Bansal, A., Lazova, V., Pons-Moll, G.: Stereo radiance fields (srif): Learning view synthesis for sparse views of novel scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7911–7920 (2021)
8. Deng, K., Liu, A., Zhu, J.Y., Ramanan, D.: Depth-supervised nerf: Fewer views and faster training for free. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12882–12891 (2022)
9. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. arXiv preprint arXiv:1605.08803 (2016)
10. Fridovich-Keil, S., Meanti, G., Warburg, F.R., Recht, B., Kanazawa, A.: K-planes: Explicit radiance fields in space, time, and appearance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12479–12488 (2023)

11. Gu, X., Fan, Z., Zhu, S., Dai, Z., Tan, F., Tan, P.: Cascade cost volume for high-resolution multi-view stereo and stereo matching. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2495–2504 (2020)
12. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)
13. Jain, A., Tancik, M., Abbeel, P.: Putting nerf on a diet: Semantically consistent few-shot view synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5885–5894 (2021)
14. Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., Aanæs, H.: Large scale multi-view stereopsis evaluation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 406–413 (2014)
15. Johnson, J., Cao, A., Rockwell, C.: Real-time novel view synthesis with forward warping and depth (Nov 9 2023), uS Patent App. 17/739,572
16. Kanchana, V., Somraj, N., Yadwad, S., Soundararajan, R.: Revealing disocclusions in temporal view synthesis through infilling vector prediction. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3541–3550 (2022)
17. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)
18. Kim, M., Seo, S., Han, B.: Infonerf: Ray entropy minimization for few-shot neural volume rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12912–12921 (2022)
19. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics (ToG) **36**(4), 1–13 (2017)
20. Kwak, M.S., Song, J., Kim, S.: Geconerf: Few-shot neural radiance fields via geometric consistency. arXiv preprint arXiv:2301.10941 (2023)
21. Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9298–9309 (2023)
22. Liu, X., Kao, S.h., Chen, J., Tai, Y.W., Tang, C.K.: Deceptive-nerf: Enhancing nerf reconstruction using pseudo-observations from diffusion models. arXiv preprint arXiv:2305.15171 (2023)
23. Mildenhall, B., Srinivasan, P.P., Ortiz-Cayon, R., Kalantari, N.K., Ramamoorthi, R., Ng, R., Kar, A.: Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG) **38**(4), 1–14 (2019)
24. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
25. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022)
26. Niemeyer, M., Barron, J.T., Mildenhall, B., Sajjadi, M.S., Geiger, A., Radwan, N.: Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5480–5490 (2022)
27. Peng, R., Gu, X., Tang, L., Shen, S., Yu, F., Wang, R.: Gens: Generalizable neural surface reconstruction from multi-view images. Advances in Neural Information Processing Systems **36**, 56932–56945 (2023)

28. Peng, R., Wang, R., Wang, Z., Lai, Y., Wang, R.: Rethinking depth estimation for multi-view stereo: A unified representation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8645–8654 (2022)
29. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-nerf: Neural radiance fields for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10318–10327 (2021)
30. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International conference on machine learning. pp. 8748–8763. PMLR (2021)
31. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 12179–12188 (2021)
32. Roessle, B., Barron, J.T., Mildenhall, B., Srinivasan, P.P., Nießner, M.: Dense depth priors for neural radiance fields from sparse input views. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12892–12901 (2022)
33. Schönberger, J.L., Frahm, J.M.: Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4104–4113 (2016)
34. Schönberger, J.L., Zheng, E., Frahm, J.M., Pollefeys, M.: Pixelwise view selection for unstructured multi-view stereo. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part III 14. pp. 501–518. Springer (2016)
35. Somraj, N., Soundararajan, R.: Vip-nerf: Visibility prior for sparse input neural radiance fields. arXiv preprint arXiv:2305.00041 (2023)
36. Song, J., Park, S., An, H., Cho, S., Kwak, M.S., Cho, S., Kim, S.: Därf: Boosting radiance fields from sparse input views with monocular depth adaptation. Advances in Neural Information Processing Systems **36** (2024)
37. Sun, C., Sun, M., Chen, H.T.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5459–5469 (2022)
38. Sun, L., Bian, J.W., Zhan, H., Yin, W., Reid, I., Shen, C.: Sc-depthv3: Robust self-supervised monocular depth estimation for dynamic scenes. IEEE Transactions on Pattern Analysis and Machine Intelligence (2023)
39. Suvorov, R., Logacheva, E., Mashikhin, A., Remizova, A., Ashukha, A., Silvestrov, A., Kong, N., Goka, H., Park, K., Lempitsky, V.: Resolution-robust large mask inpainting with fourier convolutions. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 2149–2159 (2022)
40. Uy, M.A., Martin-Brualla, R., Guibas, L., Li, K.: Scade: Nerfs from space carving with ambiguity-aware depth estimates. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16518–16527 (2023)
41. Wang, G., Chen, Z., Loy, C.C., Liu, Z.: Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. arXiv preprint arXiv:2303.16196 (2023)
42. Wang, Y., Yang, Y., Yang, Z., Zhao, L., Wang, P., Xu, W.: Occlusion aware unsupervised learning of optical flow. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4884–4893 (2018)
43. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing **13**(4), 600–612 (2004)

44. Wynn, J., Turmukhambetov, D.: Diffusionerf: Regularizing neural radiance fields with denoising diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4180–4189 (2023)
45. Xiong, K., Peng, R., Zhang, Z., Feng, T., Jiao, J., Gao, F., Wang, R.: Cl-mvsnet: Unsupervised multi-view stereo with dual-level contrastive learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3769–3780 (2023)
46. Xu, J., Wang, X., Cheng, W., Cao, Y.P., Shan, Y., Qie, X., Gao, S.: Dream3d: Zero-shot text-to-3d synthesis using 3d shape prior and text-to-image diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20908–20918 (2023)
47. Yang, J., Pavone, M., Wang, Y.: Freenerf: Improving few-shot neural rendering with free frequency regularization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8254–8263 (2023)
48. Yang, J., Mao, W., Alvarez, J.M., Liu, M.: Cost volume pyramid based depth inference for multi-view stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4877–4886 (2020)
49. Yao, Y., Luo, Z., Li, S., Fang, T., Quan, L.: Mvsnet: Depth inference for unstructured multi-view stereo. In: Proceedings of the European conference on computer vision (ECCV). pp. 767–783 (2018)
50. Yao, Y., Luo, Z., Li, S., Zhang, J., Ren, Y., Zhou, L., Fang, T., Quan, L.: Blended-mvs: A large-scale dataset for generalized multi-view stereo networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 1790–1799 (2020)
51. Ye, J., Wang, P., Li, K., Shi, Y., Wang, H.: Consistent-1-to-3: Consistent image to 3d view synthesis via geometry-aware diffusion models. arXiv preprint arXiv:2310.03020 (2023)
52. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelnerf: Neural radiance fields from one or few images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4578–4587 (2021)
53. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)
54. Zhang, Z., Peng, R., Hu, Y., Wang, R.: Geomvsnet: Learning multi-view stereo with geometry perception. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21508–21518 (2023)
55. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1851–1858 (2017)
56. Zhu, Z., Fan, Z., Jiang, Y., Wang, Z.: Fsgs: Real-time few-shot view synthesis using gaussian splatting. arXiv preprint arXiv:2312.00451 (2023)

Supplementary Material

A More Results

A.1 Visual Quality

Fig. 8, Fig. 9 illustrate the qualitative results of our approach on DTU and LLFF datasets under high-resolution settings with three input views. These visualizations demonstrate our method’s ability to capture finer details in high-frequency regions compared to NeRF-based approaches. Fig. 10 shows more rendering results on T&T and NVS-RGBD datasets, demonstrating our method’s robustness in complex large-scale scenes and in-the-wild scenarios.

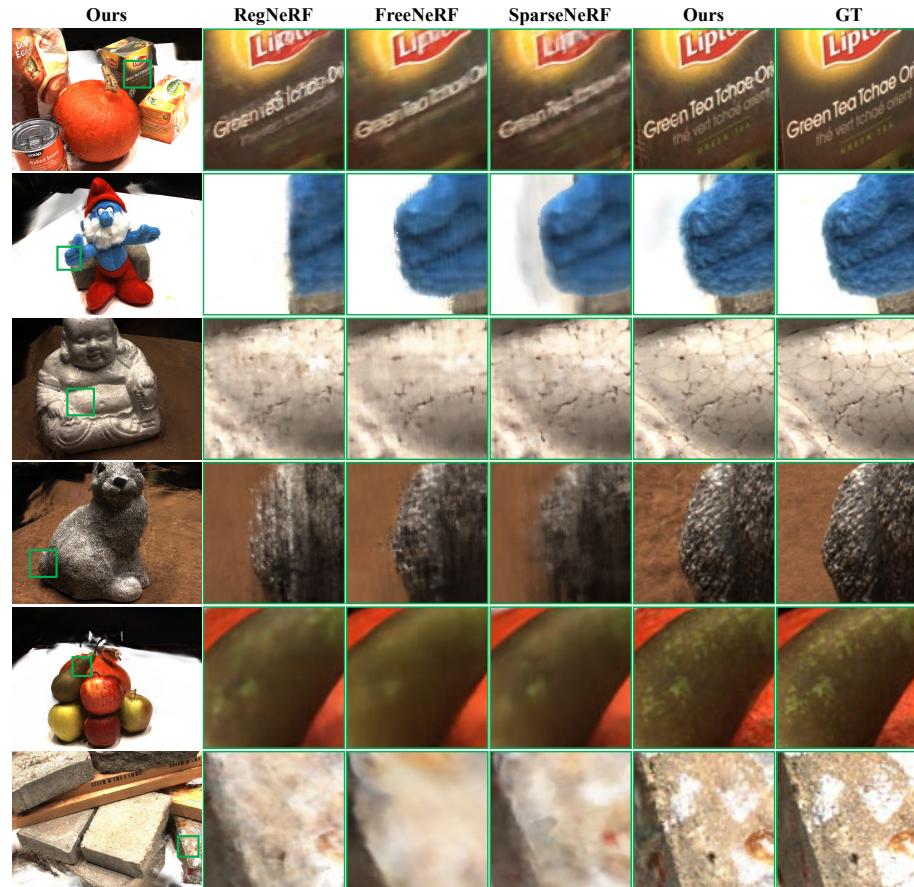


Fig. 8: Qualitative Results on DTU with 3 Input Views (res. 1/2).

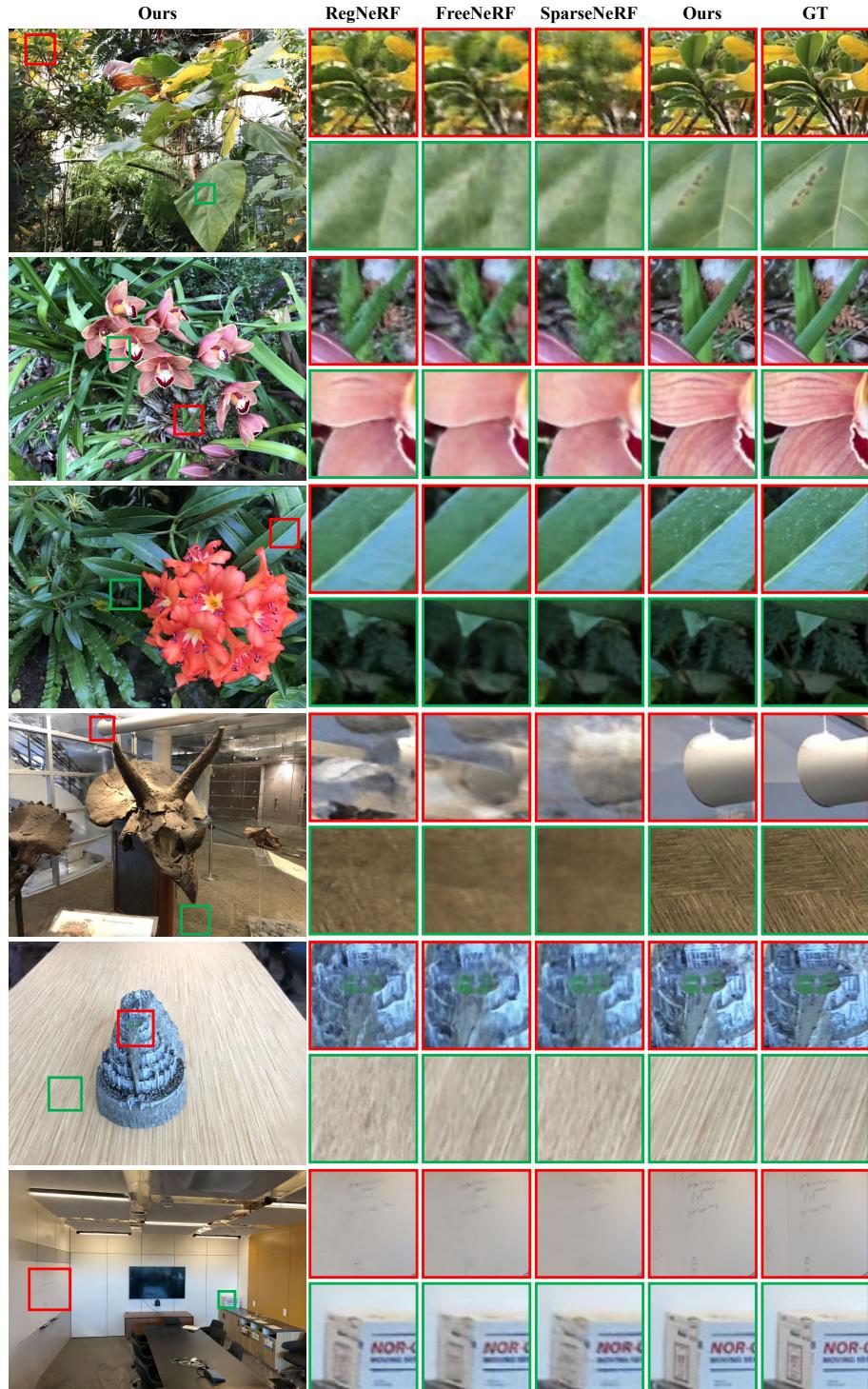


Fig. 9: Qualitative Results on LLFF with 3 Input Views (res. 1/4).



Fig. 10: Qualitative Results on T&T (the first two rows) and NVS-RGBD (the last two rows) with 3 Input Views.

A.2 Different Input Views

Tab. 6 shows the quantitative results with different input views on DTU (res. 1/4). Similar to the results in Tab. 3, our method achieves competitive results in most metrics on DTU, which demonstrates our robustness in different training views. Fig. 12 and Fig. 13 illustrate the visual quality with only 2-view inputs. Our method can still render reasonable results, even in such an extremely sparse situation. The visualizations of rendered depths in novel views (Fig. 11) suggest that our approach can catch accurate geometric structures of scenes in few-shot situations. Fig. 14 gives the qualitative results with more input views (5 views). It shows that our method still outperforms in finer details regions.

Table 6: Quantitative Results with Different Input Views on DTU.

Methods	2 view			3 view			4 view			5 view		
	PSNR ↑ SSIM ↑ LPIPS ↓											
3DGS	12.91	0.728	0.226	14.90	0.783	0.183	18.01	0.840	0.132	21.62	0.886	0.092
Mip-NeRF	8.39	0.561	0.365	8.68	0.571	0.353	15.57	0.692	0.245	18.35	0.758	0.202
RegNeRF	16.51	0.711	0.236	18.89	0.745	0.190	19.89	0.783	0.177	21.91	0.829	0.147
FreeNeRF	15.47	0.694	0.247	19.92	0.787	0.182	21.04	0.802	0.185	22.35	0.819	0.175
SparseNeRF	17.21	0.715	0.238	19.46	0.766	0.204	20.69	0.800	0.190	22.05	0.820	0.178
Ours	17.55	0.823	0.147	20.50	0.871	0.106	21.57	0.887	0.091	22.87	0.899	0.080

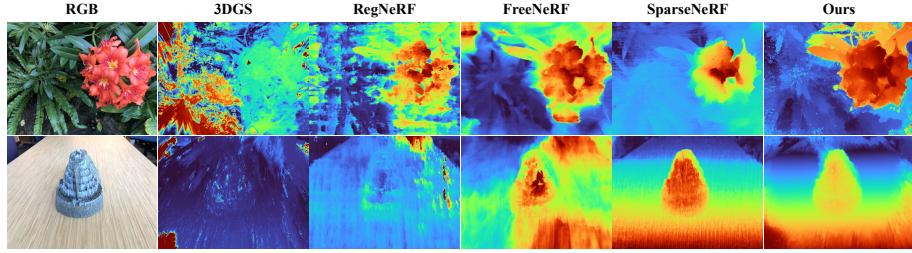


Fig. 11: Depths Visualization on LLFF with 2 Input Views (res. 1/8).

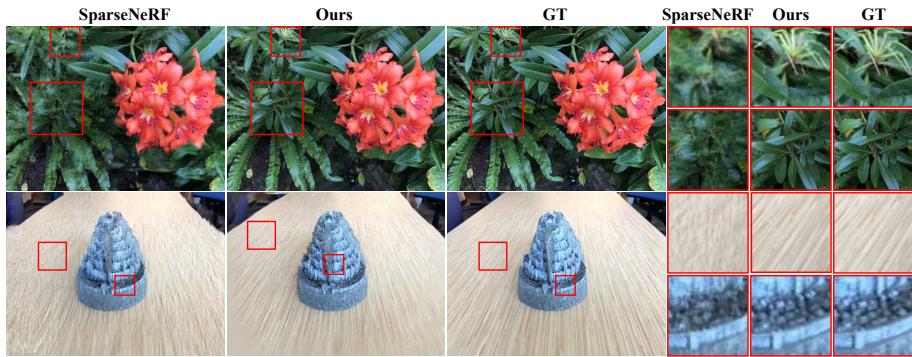


Fig. 12: Qualitative Results on LLFF with 2 Input Views (res. 1/8).

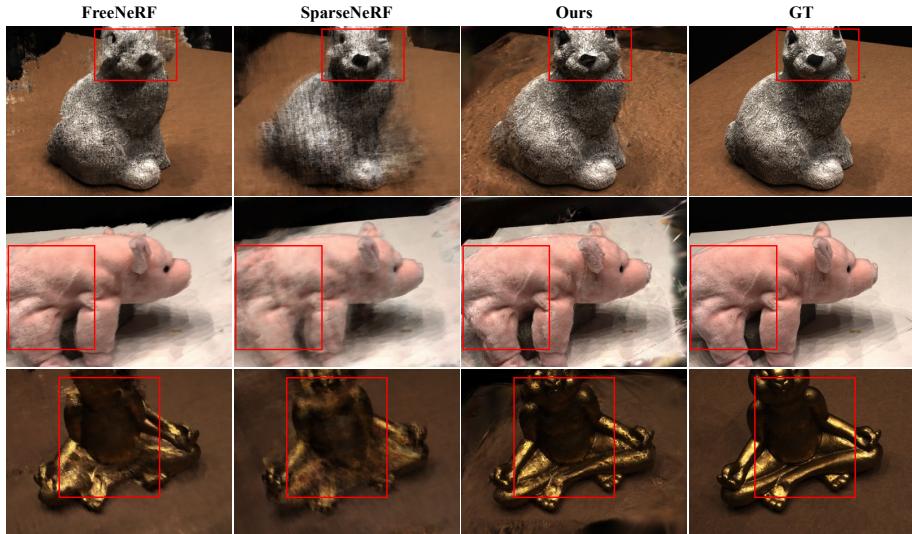


Fig. 13: Qualitative Results on DTU with 2 Input Views (res. 1/4).

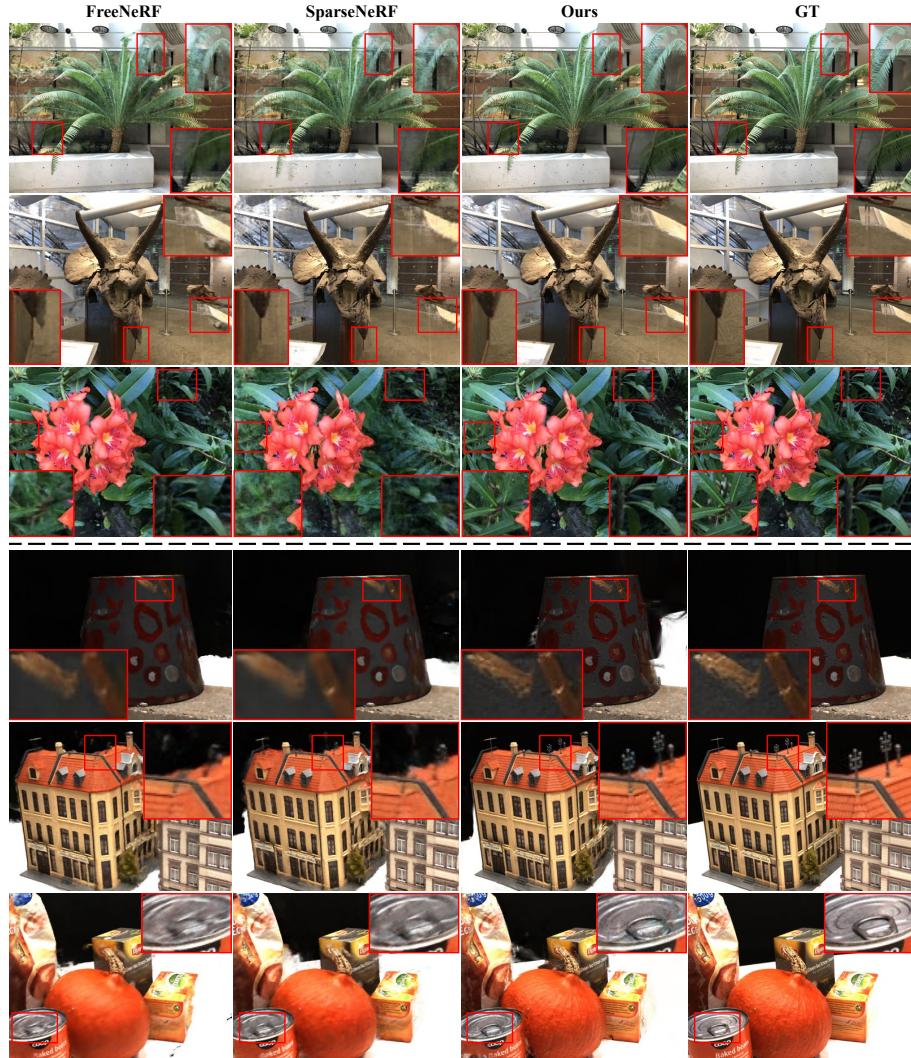


Fig. 14: Qualitative Results on LLFF (res. 1/8, the first three rows) and DTU (res. 1/4, the last three rows) with 5 Input Views.

A.3 Visualization of Different Gaussian Initialization

Fig. 15 compares visualizations of COLMAP points and our points derived from MVSFormer [4] on LLFF and DTU datasets. It shows that learning-based methods can offer more comprehensive initial positions in few-shot scenarios, thereby facilitating subsequent optimization of Gaussian parameters, as indicated by the metrics in Tab. 1.

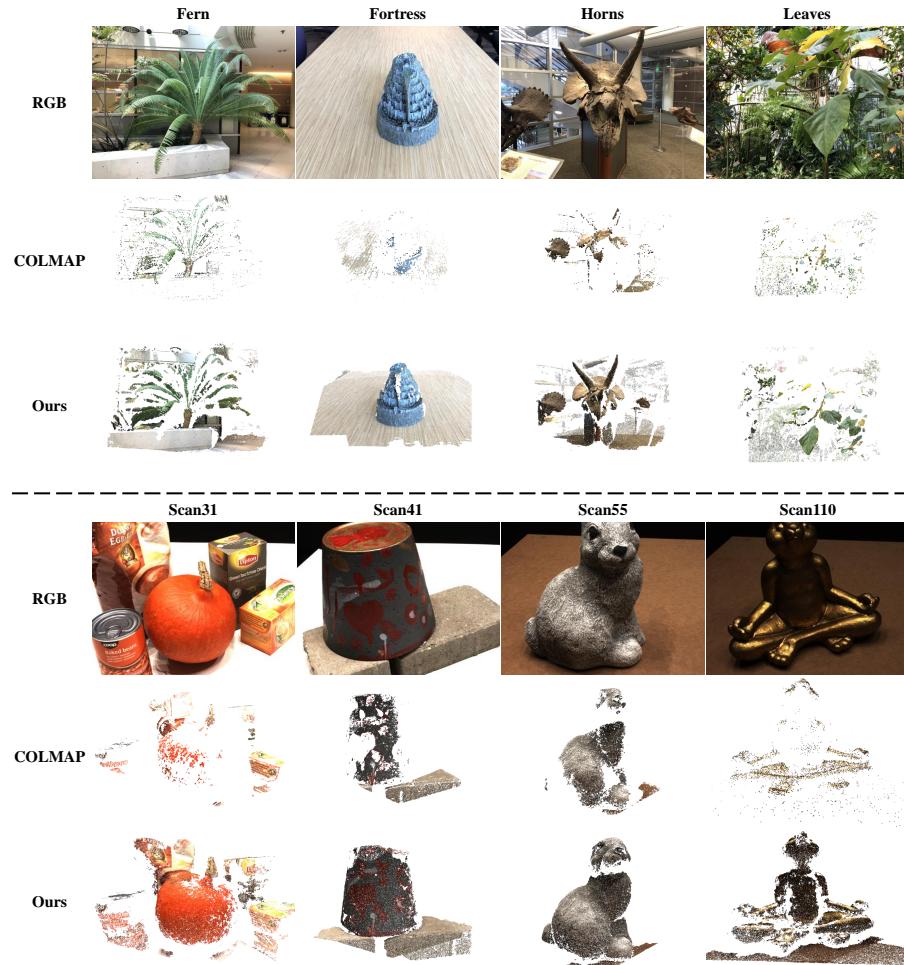


Fig. 15: Visualizations of Initial Points for LLFF (the first three rows) and DTU (the last three rows) with 3 Input Views.

B Experiment Details

B.1 Datasets

We follow previous methods [26, 41, 47] for the splits of LLFF and DTU datasets. For LLFF, we use every 8-th image as the held-out test set, and evenly sample from the remaining views for training. For DTU, we adhere to [26, 41, 47] to evaluate the proposed method on 15 selected scenes. The scan IDs include 8, 21, 30, 31, 34, 38, 40, 41, 45, 55, 63, 82, 103, 110, and 114. The images with IDs (all image IDs start from 0): 25, 22, 28, 40, 44, 48, 0, 8, 13 are used for training. We select the first 2,3,4,5 image IDs as input views in our experiments. Images with 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 are used as the novel views in evaluation. For NVS-RGBD, the 3 training views are fixed and we use images with the resolution of 960×540 and 640×360 for ZED2 and Kinect branches, respectively. For T&T, we conduct experiments on images with a resolution of 960×540 .

B.2 Metrics

We use the formula $-10 \cdot \log_{10}(MSE)$ to compute the metric PSNR. We uniformly adopt the scikit-image⁴ python package to compute the SSIM metrics for comparison. We uniformly use lpips package⁵ with VGG model to compute LPIPS metrics in experiments.

B.3 Trainging Details

Alg. 1 illustrates the details of the training procedure. We set the unseen-view sampling interval $\mathcal{E} = 3$ in our experiments. We follow the setting in SparseNeRF [41] to predict monocular depth maps by DPT Hybrid model.

B.4 Details for MVS

Basic Settings We use the official codebase⁶ of MVSFormer [4] to estimate each view’s depth D^{mvs} with default settings. The input images are resized to the default resolution 1152×1536 for MVSFormer. We adopt scene bounds provided by datasets as depth ranges. The number of depth hypothesis planes is set to the default 192 for all datasets.

Pretrained Models We use official MVSFormer model pre-trained on DTU for our experiments on LLFF. For NVS-RGBD and T&T, we adopt the model pre-trained on DTU and fine-tuned on BlendedMVS. For our experiments on DTU, we exclude DTU scenes from the training data of MVSFormer that overlap with the selected scenes mentioned in Sec. B.1 and then fine-tuned on BlendedMVS.

⁴ <https://scikit-image.org/>

⁵ <https://pypi.org/project/lpips/>

⁶ <https://github.com/ewrfcas/MVSFormer>

Algorithm 1: Optimizing on a single scene

Data: N input views $\{V_i^{train}\}_{i=1}^N$, ground-truth images $\{I_i\}_{i=1}^N$, camera intrinsics K , camera poses $\{P_i\}_{i=1}^N$

Result: Optimized Gaussian parameters \mathcal{G}

Estimate view-consistent depths $\{D_i^{mvs}\}_{i=1}^N$ for $\{V_i^{train}\}_{i=1}^N$ by MVSFormer;

Compute high consistency masks $\{M_i\}_{i=1}^N$ based on reprojected errors of $\{D_i^{mvs}\}_{i=1}^N$, estimate monocular depths $\{D_i^{mono}\}_{i=1}^N$ by DPT ;

Merge masked depths into point clouds \mathcal{P} and initialize \mathcal{G} with \mathcal{P} ;

Sample unseen views $\{V_i^{unseen}\}_{i=1}^M$ randomly according to $\{V_i^{train}\}_{i=1}^N$;

for it from 1 to num_iters **do**

- if** $it \% \mathcal{E} \neq 0$ **then**
 - Choose a V_i^{train} randomly ;
 - Render image \hat{I}_i and depth \hat{D}_i for V_i^{train} based on \mathcal{G} by splatting ;
 - $\mathcal{L} \leftarrow L_{photo}(\hat{I}_i, I_i)$;
 - $\mathcal{L} \leftarrow \mathcal{L} + L_{CS}(\hat{D}_i, D_i^{mvs}, M_i)$;
 - $\mathcal{L} \leftarrow \mathcal{L} + L_{mono}(\hat{D}_i, D_i^{mono})$;
- else**
 - Choose a V_i^{unseen} randomly ;
 - Choose a V_j^{train} randomly, compute appearance prior by $I'_j = fwd(I_j^{train}, D_j^{mvs}, P_j^{train}, V_i^{unseen})$;
 - Render image \hat{I}_i for V_i^{unseen} based on \mathcal{G} by splatting ;
 - $\mathcal{L} \leftarrow L_{fwd}(\hat{I}_i, I'_j)$;
- end**
- $\nabla_{\mathcal{G}} \mathcal{L} \leftarrow \frac{\partial \mathcal{L}}{\partial \mathcal{G}}$;
- if** $IsRefinementIteration(it)$ **then**
 - | PruneAndDensify($\mathcal{G}, \nabla_{\mathcal{G}} \mathcal{L}$) ;
- end**
- Update parameters: $\mathcal{G} \leftarrow Adam(\mathcal{G}, \nabla_{\mathcal{G}} \mathcal{L})$;

end

Geometric Filter We utilize a geometric filter [49] to identify regions exhibiting high view consistency, followed by reprojecting the estimated depth D^{mvs} to highlight these areas. Specifically, we project a reference view pixel p_1 based on its depth d_1 to pixel p_i in a source view V_i . Then we project p_i back to the reference view by p_i 's depth d_i in source view's D_i^{mvs} to get the reprojected coordinate p_{reproj} and reprojected depth d_{reproj} . We consider depth estimation d_1 of p_1 exhibits high-consistency with source view V_i , when it satisfies $|p_{reproj} - p_1| < 1$ and $|d_{reproj} - d_1|/d_1 < 0.01$ [49]. For the masks $\{M_i\}$, we require the predicted depth to be highly consistent with at least two known viewpoints.

For the fusion of depth maps, We use the method mentioned in MVSNet [49] to merge depths into point clouds for the initialization of 3DGS. We down-sample the initial point cloud with a downsampling rate of 0.1 for memory conservation.

B.5 Computational Time Costs

We test the computational time costs of our method on a single V100 GPU. The MVS depth estimation and filtering processes take approximately 0.57 seconds and 0.19 seconds per image for LLFF (res. 1/8) and DTU (res. 1/4), respectively. The time costs for depth maps fusion (3 views) are 2.9 seconds and 3.2 seconds per scene for LLFF (res. 1/8) and DTU (res. 1/4) datasets, respectively. Comparatively, the original COLMAP takes around 293 seconds and 83 seconds per scene for dense point cloud reconstruction on the LLFF and DTU datasets.