CoherentGS: Sparse Novel View Synthesis with Coherent 3D Gaussians

Avinash Paliwal[1,2*], Wei Ye[2], Jinhui Xiong[2], Dmytro Kotovenko[3], Rakesh Ranjan[2], Vikas Chandra[2], and Nima Khademi Kalantari[1]

[1] Texas A&M University
[2] Meta Reality Labs
[3] LMU Munich

https://people.engr.tamu.edu/nimak/Papers/CoherentGS

Abstract. The field of 3D reconstruction from images has rapidly evolved in the past few years, first with the introduction of Neural Radiance Field (NeRF) and more recently with 3D Gaussian Splatting (3DGS). The latter provides a significant edge over NeRF in terms of the training and inference speed, as well as the reconstruction quality. Although 3DGS works well for dense input images, the unstructured point-cloud like representation quickly overfits to the more challenging setup of extremely sparse input images (e.g., 3 images), creating a representation that appears as a jumble of needles from novel views. To address this issue, we propose regularized optimization and depth-based initialization. Our key idea is to introduce a structured Gaussian representation that can be controlled in 2D image space. We then constraint the Gaussians, in particular their position, and prevent them from moving independently during optimization. Specifically, we introduce single and multiview constraints through an implicit convolutional decoder and a total variation loss, respectively. With the coherency introduced to the Gaussians, we further constrain the optimization through a flow-based loss function. To support our regularized optimization, we propose an approach to initialize the Gaussians using monocular depth estimates at each input view. We demonstrate significant improvements compared to the state-of-the-art sparse-view NeRF-based approaches on a variety of scenes.

Keywords: Sparse View Synthesis · 3D Gaussian Splatting · Implicit Decoder

1 Introduction

In recent years, the field of 3D reconstruction from posed images has been in the spotlight with the integration of learned scene representations, such as Multi-Plane Images (MPI) [61] and Neural Radiance Field (NeRF) [32]. More recently,

*The work was primarily done during an internship at Meta Reality Labs.

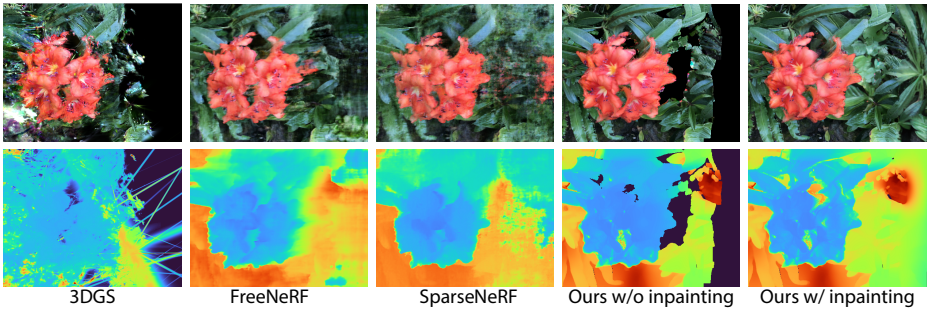

Fig. 1: For sparse input views, the quality of 3DGS deteriorates. Notable artifacts are observed in the results of the NeRF-based methods by Yang et al. [55] (FreeNeRF) and Wang et al. [18] (SparseNeRF). Our approach ("Ours w/o inpainting") yields high-quality synthesized views. Note that our constraints do not allow the Gaussians to move freely in the 3D space. As a result, our approach does not reconstruct the areas that are occluded in all the input images. This is an advantage of our technique over other methods that fill in these areas with blurry and repetitive structure, as we can identify and inpaint these regions and produce realistic hallucinated details. As a proof of concept, we inpaint these regions using a diffusion model and project them to 3D using monocular depth. As shown on the right, the hallucinated details and their corresponding depth are reasonable.

an explicit representation known as 3D Gaussian Splatting (3DGS) [23] has been introduced that significantly improves the training speed while providing real-time inference and better 3D reconstruction quality. However, it struggles to generate a good representation given a sparse set of training images. In such cases, the representation severely overfits to the training views and appears as a collection of semi-random anisotropic blobs from novel views, as shown in Fig. 1.

While utilizing 3DGS for sparse input view synthesis is under-explored, several NeRF-based methods have been proposed to tackle this task. Since reconstruction from sparse images is an ill-posed problem, these approaches often employ various regularizations to constrain the optimization [34,35,41,48,55]. However, as shown in Fig. 1, these state-of-the-art NeRF-based approaches produce sub-optimal results as their regularizations do not provide sufficient constraints for a reasonable 3D reconstruction. Additionally, most of these approaches rely on the coherency of the implicit representation learned by a neural network and are not directly applicable to 3DGS with an explicit unstructured representation.

Our key idea is to augment the explicit unstructured representation with coherency, by constraining the movement of the Gaussian blobs during optimization. Because of the unstructured nature of the Gaussians, constraining them in the 3D space is difficult. To overcome this issue, we propose to assign a single Gaussian to every pixel of each input image and enforce single and multiview constraints in 2D image space. Specifically, we force the Gaussians of each image with similar depth to move coherently using an implicit decoder [5, 29, 36]. To enforce constraints over Gaussians across different views, we ensure the rendered depth using all the Gaussians are smooth through a total variation loss. With

the introduced coherency, we additionally propose a flow-based loss to ensure the position of the Gaussians of the corresponding pixels in two images are similar.

Furthermore, to help with the optimization, we propose to initialize the position of Gaussians using an existing monocular depth prediction model. While these models provide high-quality depth estimates, single image depth is relative and lacks consistency across views. Our depth-based initialization properly positions the Gaussians in the world space, while our regularized optimization encourages the updates, particularly to the positions, to be coherent and smooth. This allows us to reconstruct high-quality texture and geometry, as shown in Fig. 1. As an added benefit of preventing the Gaussians to freely move in the 3D space, we can easily identify and inpaint the occluded regions to produce high-quality hallucinated texture and geometry (see Fig. 1 - right).

In summary, we make the following key contributions:

- We present an approach to perform 3D reconstruction using 3DGS from extremely sparse set of inputs.
- We propose a structured Gaussian representation and introduce coherency using various regularizations.
- We introduce a depth-based initialization of 3D Gaussians that complements the regularized optimization.

2 Related Work

In this section, we mainly focus on the closely related work by discussing the radiance field approaches and sparse novel view synthesis methods that utilize this concept.

2.1 Radiance Fields

The introduction of Neural Radiance Field (NeRF) [32], an optimization based approach that leverages implicit neural networks, has revolutionized scene reconstruction and novel view synthesis. A large number of techniques have presented various ways to improve the rendering quality [1–4, 19, 43, 47, 49], generalization [10,11,22,45,51,59], performance on dynamic scenes [5,14,15,27,36,38,46], and the need for precise camera poses [6,12,28,52,53,57]. Moreover, the success of this approach has also led to its use as the representation of choice in 3D generative modeling [7,8,17,20,37].

However, NeRF models based on MultiLayer Perceptron (MLP) utilize the neural network to implicitly encode the scene density and view-dependent color. To render a pixel color, the MLP is queried several times along the sampled ray which leads to slow optimization (hours) and rendering (seconds per frame). While several approaches have been proposed to improve the optimization and inference speed [9,16,33,39,40,44], they often achieve the speed up by sacrificing the rendering quality. Recently, Kerbl et al. [23] achieved a breakthrough with the 3D Gaussian Splatting (3DGS) approach which enables fast optimization with high-quality real-time rendering. We introduce a novel approach that enables application of 3DGS to the sparse input setting.

2.2 Sparse Novel View Synthesis

An interesting and more practical variation of the view synthesis problem is to utilize a sparse set of input images. NeRF and 3DGS require tens of images to render high quality novel views. With only a few images, these methods quickly overfit to the input images and produce unsatisfactory novel view results. Several NeRF-based approaches [13, 21, 34, 35, 41, 48, 55, 59] tackle this problem by introducing various regularizations. Specifically, RegNeRF [35] proposes a geometry and color regularization from unobserved viewpoints. DS-NeRF [13] relies on sparse 3D points from COLMAP for depth supervision. ViP-NeRF [34] computes a visibility prior to impose a multi-view constraint during optimization. FreeNeRF [55] uses a coarse-to-fine refinement scheme by gradually increasing the positional encoding frequencies. SparseNeRF [48] introduces a local depth ranking and spatial continuity regularization based on monocular depth.

While these approaches produce impressive results, their regularization in extremely sparse settings (e.g., 3 images) is not able to provide sufficient constraints. Therefore, as shown in Fig. 1, their results exhibit significant blurring in these scenarios. Additionally, the regularizations in most of these techniques are sparse and rely on the coherency of the MLP to propagate the constraints to the other regions. Unfortunately, these regularizations are not directly applicable to 3DGS which is a discrete and unstructured representation. We propose to introduce coherency in the 3DGS representation by utilizing the combination of an implicit decoder and total variation loss.

Concurrent to our technique, Zhu et al. [62] (FSGS) and Xiong et al. [54] (SparseGS) propose to utilize 3DGS for sparse view synthesis. However, unlike our method, they do not present a way to enforce coherency between nearby Gaussians, producing occasional floaters. Additionally, these methods fill in the regions that are occluded in all the input images with elongated Gaussians producing blurry results. In contrast, our approach allows us to identify and inpaint the occluded regions and produce high-quality hallucinated details (see Fig. 1 - right).

3 Background

In this section, we provide a brief introduction to the 3DGS [23] approach. This technique represents a 3D scene using a dense set of anisotropic 3D Gaussians, enabling fast and differentiable rendering via α-blending. Each Gaussian is parameterized by a set of attributes such as position $\mathbf{x}$, color $\mathbf{c}$, covariance matrix Σ [58], and opacity value α. Similar to a typical differentiable point-based approach [25], the image formation model for the volumetric rendering can be written as:

$$R_{\Sigma,\alpha,\mathbf{x},\mathbf{c}}(\mathbf{p}) = \sum_{i \in \mathcal{N}(\mathbf{p})} \mathbf{c}_i \gamma_i \prod_{j=1}^{i-1} (1 - \gamma_j) \text{ , where } \gamma_k = f(\Sigma_k, \alpha_k, \mathbf{x}_k, \mathbf{p}). \quad (1)$$

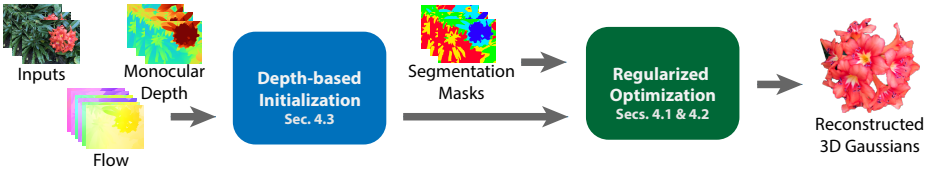

Fig. 2: Overview of the optimization pipeline. For every input image, we obtain monocular depth (Depth Anything [56]) and dense flow correspondences between all image pairs (FlowFormer++ [42]). These inputs are utilized to initialize a good set of 3D Gaussians for the subsequent optimization stage. The initialized 3D Gaussians, along with depth-based segmentation masks, are then used to perform a regularized 3D Gaussian optimization to obtain high-quality reconstruction.

Here, $\mathcal{N}(\mathbf{p})$ is the number of ordered points overlapping the pixel of interest $\mathbf{p}$. Moreover, the function $f(\Sigma, \alpha, \mathbf{x}, \mathbf{p})$ computes the effective opacity by evaluating the Gaussian at $\mathbf{p}$ and multiplying it with the global opacity α. Note that in practice, 3DGS decomposes Σ into scaling and rotation components, and uses spherical harmonics to represent the view-dependent color. However, we will continue using Σ and $\mathbf{c}$ in the following sections for simplicity of notation.

During optimization, optimal Gaussian parameters and colors are obtained by minimizing the following objective:

$$\Sigma^*, \alpha^*\mathbf{x}^*, \mathbf{c}^* = \arg\min_{\Sigma,\alpha,\mathbf{x},\mathbf{c}} \sum_{\mathbf{p}\in\mathcal{P}} \mathcal{L}(R_{\Sigma,\alpha,\mathbf{x},\mathbf{c}}(\mathbf{p}), R(\mathbf{p})), \tag{2}$$

where $\mathcal{P}$ contains all pixels of every input image, $R_{\Sigma,\alpha,\mathbf{x},\mathbf{c}}(\mathbf{p})$ and $R(\mathbf{p})$ correspond to the rendered and reference pixel colors, and $\mathcal{L}$ is the combination of $\mathcal{L}_1$ and SSIM losses in 3DGS. Our main objective is to utilize 3DGS in the extremely sparse setting and avoid overfitting to the input images.

4 Algorithm

Given a sparse set of N images (e.g., 3 or 4 images), our goal is to reconstruct a 3D Gaussian representation of the static scenes. Our key idea is to introduce coherency to the 3D Gaussians during optimization. In other words, when the position of a Gaussian is updated, the neighboring Gaussians should also be similarly affected during optimization. With this coherency, we can then use sparse regularization to further constrain the optimization and avoid overfitting to the input images. The overview of our method is shown in Fig. 2. We discuss our approach in detail in the following sections.

4.1 Coherent 3D Gaussian Optimization

Introducing coherency to unstructured Gaussian particles in 3D is challenging. Our main idea is to transform the representation to a more structured form in 2D image domain. To do this, we propose to assign a single Gaussian to each

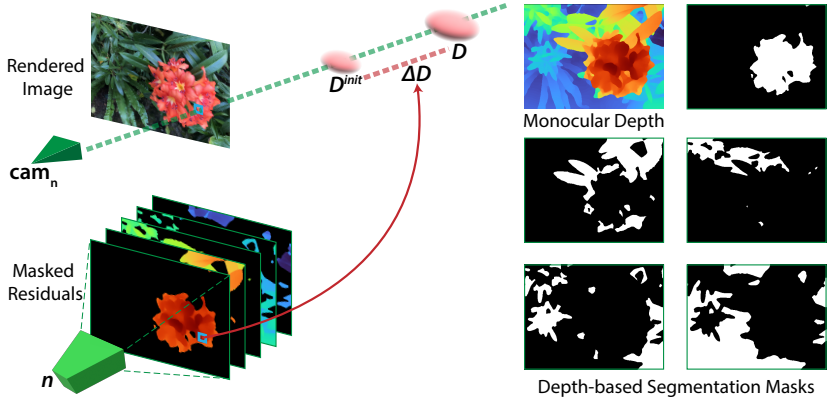

Fig. 3: During regularized optimization, the implicit decoder predicts the residual depth ΔD that moves the Gaussians from their initial position towards the true scene depth D. The input coordinate n to the decoder corresponds to the input view with camera cam_n. To preserve sharp discontinuities, we apply binary segmentation masks to the decoder output obtained by thresholding the monocular depth.

pixel in every input image. We further restrict the movement of the Gaussians at each pixel to a ray connecting the center of the camera to that pixel. Under this representation, the position of each Gaussian can be controlled using a scalar depth value. Specifically, given an initial depth estimate at each pixel (see Sec. 4.3), we update the position of each Gaussian through residual depth as follows:

$$\mathbf{x} = g(D_n^{\text{init}}[\mathbf{p}] + \Delta D_n[\mathbf{p}], \mathbf{p}), \tag{3}$$

where the function $g(d, p)$ projects pixel p into 3D according to depth d. Assuming that the local surface geometry is reasonably captured by the initial depth map at each view D_n^{init}, the residual depth should only vary smoothly to adjust the inaccuracies. Based on this observation, we propose single-view and multiview constraints as discussed below.

Single-view Constraint We enforce per-view smoothness by utilizing an implicit convolutional decoder. Specifically, as shown in Fig. 3, this decoder takes the view index n as the input and estimates the residual depth for the entire image, i.e., $\Delta D_n = f_\phi(n)$, where ϕ refers to the parameters of the decoder. Instead of updating the residual depth at individual pixels, we perform the optimization by modifying the decoder parameters ϕ. This ensures that the residual depth is smooth and object surfaces are coherently deformed during optimization.

However, smooth deformation means that the decoder struggles to handle the sharp depth discontinuities between the objects. We address this by obtaining a C-channel binary segmentation mask through dividing each image into C (5 in our implementation) separate regions, each with similar depths. Specifically, we use the approach by Yang et al. [56] to estimate the monocular depth and follow the strategy by Wang et al. [50] to divide the image into C regions based on the input depth (see Fig. 3 - right). The decoder in this case, also produces a

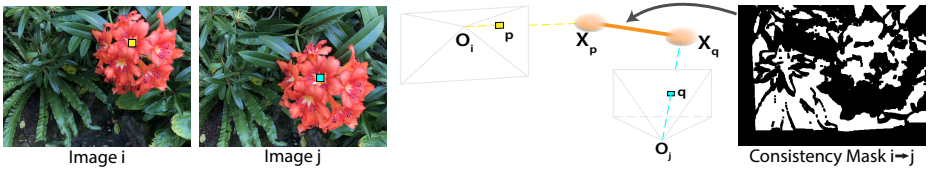

Fig. 4: Our flow based regularization forces the Gaussians of corresponding pixels in a pair of images (e.g., yellow and cyan squares) to have similar positions, by minimizing their distance. The binary mask is utilized to mask out unreliable correspondences.

C-channel residual depth. The final residual is obtained as $\Delta D_n = \text{chsum}(S \odot f_\phi(n))$, where chsum refers to the channelwise summation operation.

We also apply a similar constraint to the global opacity α of the Gaussians to ensure coherency. In this case, we start from an initial opacity and the decoder estimates a C-channel residual opacity which is combined with the initial value to obtain the final opacity at each pixel.

Multiview Constraint While the single-view constraint ensures smooth deformation of Gaussians corresponding to each image, it does not enforce the 3D surfaces, formed by the Gaussians from all the images, to be smooth. To encourage the reconstructed geometry to be smooth, we propose to utilize a total variation (TV) regularization $\mathcal{L}_{\text{TV}}$ on the rendered depth. Specifically, we replace the color $\mathbf{c}$ with depth d in Eq. 1 to obtain the rendered depth $R_{\Sigma,\alpha,\mathbf{x},d}$ in each view. We then apply a TV loss on the rendered disparity (inverse of depth) in two ways as follows:

$$\mathcal{L}_{\text{TV}} = \left\| \nabla \left(\frac{1}{1 + R_{\Sigma,\alpha,\mathbf{x},d}} \right) \right\|_1, \quad \mathcal{L}_{\text{MTV}} = \left\| \nabla \left(\mathbf{S} \odot \left(\frac{1}{1 + R_{\Sigma,\alpha,\mathbf{x},d}} \right) \right) \right\|_1. \tag{4}$$

Here, $\mathcal{L}_{\text{TV}}$ enforces the depth to be overall smooth, while $\mathcal{L}_{\text{MTV}}$ ensures that the depth in each segmented region is smooth. We propose to start by minimizing $\mathcal{L}_{\text{TV}}$ to obtain a globally smooth and connected geometry (no sharp discontinuities) and gradually increase the contribution of $\mathcal{L}_{\text{MTV}}$ to improve the structure details. We do this through the following loss:

$$\mathcal{L}_{\text{multi}} = (1 - \lambda_s)\mathcal{L}_{\text{TV}} + \lambda_s \mathcal{L}_{\text{MTV}}, \tag{5}$$

where λ_s is initialized to 0 and is gradually increased to reach 1 by the end of the optimization. With the coherency introduced to the 3D Gaussian optimization, we can now apply additional sparse regularization to improve the results as discussed below.

4.2 Additional Regularization

Inspired by consistent depth estimation techniques [26,31,60], we propose a flow-based regularization term to further constrain the problem, as shown in Fig. 4. Specifically, the key idea is that the corresponding points in two input images

come from the same 3D point. Therefore, we force the position of the Gaussians of the corresponding pixels in two images to be similar. This can be formally written as:

$$\mathcal{L}_{\text{flow}} = \sum_{(i,j)} \sum_{\mathbf{p}} \left\| M_{i \to j} \odot \left(g(D_i[\mathbf{p}], \mathbf{p}) - g(D_j[\mathbf{q}], \mathbf{q}) \right) \right\|_1 \tag{6}$$

where $D_i[\mathbf{p}] = D_i^{\text{init}}[\mathbf{p}] + \Delta D_i[\mathbf{p}]$, and $g(d, p)$ is a function that projects pixel p into the 3D space according to its depth d. Moreover, $\mathbf{p}$ and $\mathbf{q}$ are the corresponding pixels in cameras i and j, respectively, and are calculated using an existing optical flow method (Shi et al. [42] in our implementation). Furthermore, $M_{i \to j}$ is a binary mask, indicating the reliable correspondences, obtained using the forward-backward consistency check [31].

In summary, our proposed coherent 3D Gaussian optimization with the additional regularization is performed by minimizing the following objective:

$$\Sigma^*, \phi^*, \mathbf{c}^* = \arg\min_{\Sigma, \phi, \mathbf{c}} \sum_{\mathbf{p} \in \mathcal{P}} \mathcal{L}(R_{\Sigma, \alpha, \mathbf{x}, \mathbf{c}}(\mathbf{p}), R(\mathbf{p})) + \beta_m \mathcal{L}_{\text{multi}} + \beta_f \mathcal{L}_{\text{flow}}, \tag{7}$$

where β_m and β_f control the contribution of the multiview and flows terms and we set them to 5 and 0.1, respectively. Note that the global opacity α and position $\mathbf{x}$ of the Gaussians are optimized indirectly by updating the parameters of the implicit decoder ϕ.

While 3DGS optimizes the objective by only sampling the center of each pixel, we found this strategy to be problematic for sparse input setting. In this case, the Gaussians will be deformed to match the color at the center of each pixel, leaving the remaining areas uncovered. As a result, the surfaces, when viewed from a novel view, will appear semi-transparent. To address this issue, we perform the optimization at multiple samples within each pixel. The multisampling ensures that the Gaussians properly cover each pixel, resulting in significantly improved images, as shown in the supplementary video and Table 3.

4.3 3D Gaussian Initialization

To facilitate our regularized optimization pipeline, we need a suitable initialization. In particular, our optimization requires an initial depth estimate D^{init} that captures the local geometry of objects reasonably well. We do so using a monocular depth estimation method (Yang et al. [56] in our implementation). While these approaches produce high-quality depth maps, the estimated depth is relative and often not consistent across different views. Therefore, by performing the initialization using these approaches, the Gaussians corresponding to the same surfaces across different views exhibit significant misalignments, as shown in Fig. 5 (left), hindering the optimization process.

To mitigate this issue, we use a flow-based loss, similar to the one described in Sec. 4.2, to optimize the monocular depth. Directly optimizing the depth, however, could be problematic as the loss is only enforced in the areas where the flow is accurate, leaving the depth corresponding to remaining areas unaltered.

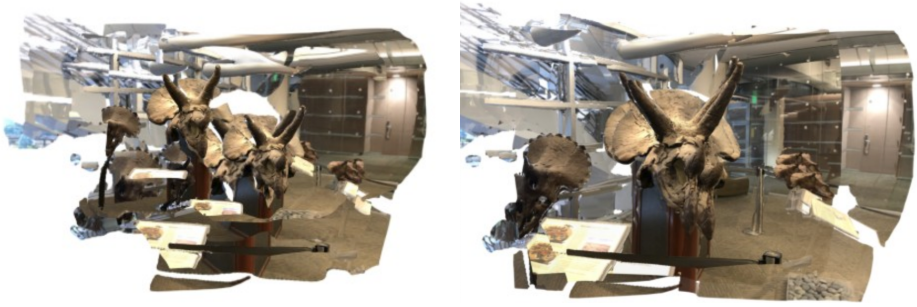

Fig. 5: We initialize a set of 3D Gaussians from each view using monocular depth to support our regularized optimization. However, since the monocular depth is relative, the initialized representation is not multi-view consistent (left). Therefore, before Gaussian initialization, we coarsely align the representations from different images using flow correspondences (right). This ensures that the optimization begins from a sensible starting point, which proves to be essential for the training of 3D Gaussian representation under the challenging ill-conditioned setting.

Therefore, we propose to only optimize the scale and offset of the depth at each image by minimizing the following objective:

$$\mathbf{s}^*, \mathbf{o}^* = \arg\min_{\mathbf{s},\mathbf{o}} \sum_{(i,j)} \sum_{\mathbf{p}} \left\| M_{i\to j} \odot \left(g(s_i \cdot D_i^{\mathrm{m}}[\mathbf{p}] + o_i, \mathbf{p}) - g(s_j \cdot D_j^{\mathrm{m}}[\mathbf{q}] + o_j, \mathbf{q}) \right) \right\|_1, \tag{8}$$

where D_i^{m} is the monocular depth at camera i. Once the optimization is complete, we obtain the optimal scale and offset for each depth map. Our initial depth can then be obtained by applying the scale and offset to the monocular depth, i.e., $D^{\mathrm{init}} = s \cdot D^{\mathrm{m}} + o$. The result of this alignment can be observed in Fig. 5 (right).

Following 3DGS, we represent the covariance matrix in terms of rotation and scale matrices. We initialize the rotation matrix to identity. For the scale matrix, we use isometric scale and treat the Gaussians as spheres. We then compute the scale according to the initial depth such that each Gaussian covers its corresponding pixel properly, as shown in Fig. 6. This is done as follows:

$$r = f \cdot D^{\mathrm{init}}/H, \tag{9}$$

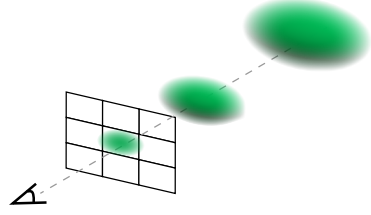

Fig. 6: Gaussian Scaling

where r is the radius of the sphere and is used to create the isometric scale matrix, and f represents the vertical focal length of the input image with height H. For the color, represented in terms of the SH coefficients, we set the DC value to the pixel color and other coefficients to zero. Finally, we initialize the global opacity to a constant.

Table 1: Numerical comparisons on the LLFF [32] dataset with 2 to 4 views.

Methods	PSNR			SSIM			LPIPS		
	2	3	4	2	3	4	2	3	4
3DGS	12.83	14.99	17.31	0.311	0.483	0.584	0.470	0.362	0.297
RegNeRF	16.55	19.41	21.49	0.468	0.627	0.713	0.417	0.306	0.257
FlipNeRF	16.57	19.74	21.55	0.485	0.668	0.721	0.407	0.282	0.260
FreeNeRF	17.07	19.97	21.80	0.513	0.652	0.713	0.376	0.280	0.259
SparseNeRF	17.74	20.33	21.90	0.513	0.657	0.720	0.386	0.302	0.260
Ours	18.32	20.33	21.58	0.644	0.725	0.762	0.220	0.180	0.167

5 Experiments

We implement our technique in PyTorch and use Adam [24] for optimization. We perform the coarse alignment during initialization for 1,000 iterations and the regularized optimization for 13,000 iterations. During the first 8,000 iterations of the regularized optimization, we keep the rotation matrix as identity and assign the scale according to Eq. 9. For the remaining 5,000 iterations, we freely optimize both the rotation and scale using the objective function.

In the following sections, we provide quantitative and qualitative comparisons against state-of-the-art approaches and evaluate the effect of various components of our method.

5.1 Comparisons

We compare our approach against the state-of-the-art NeRF-based approaches for sparse view synthesis. We specifically compare with approaches by Yang et al. [55] (FreeNeRF), Seo et al. [41] (FlipNeRF), Wang et al. [48] (SparseNeRF) and Niemeyer et al. [35] (RegNeRF). Moreover, the vanilla 3DGS serves as an additional baseline. For all the approaches, we utilize the implementations provided by the authors. We showcase both the quantitative and qualitative performances on two datasets, LLFF [32] and NVS-RGBD [48]. We follow previous methods to divide the images in both datasets into training and test views.

LLFF Dataset We begin by showing the quantitative comparisons in terms of PSNR, SSIM, and LPIPS in Table 1. Since our approach constrains the movement of the Gaussians to a ray connecting the camera center and the corresponding pixel, the regions that are occluded in all the views will not be reconstructed. To have a fair comparison against the other methods, we identify the occluded areas and mask them out in numerical evaluation. We use the same mask for all the approaches. To identify these regions, we render the opacity and consider any region with opacity below $1e-3$ as occluded. As seen in Table 1, in most cases, our method produces better results across all the metrics. In particular, the perceptual quality of our results, measure by LPIPS, is significantly better than the other techniques in all the input settings.

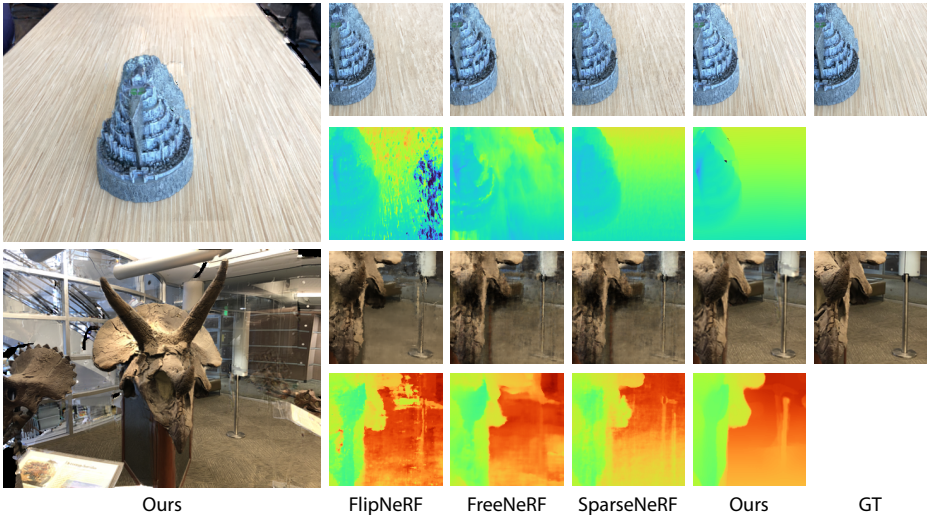

Fig. 7: LLFF 3 input. We show comparisons against other sparse-view NeRF-based approaches, SparseNeRF [48], FlipNeRF [41] and FreeNeRF [55]. Our approach produces high-quality novel views while reconstructing significantly better geometry.

Furthermore, we visually compare our approach against a subset of other methods on two scenes from the LLFF dataset with 3 input images. As shown in Fig. 7, our method significantly enhances the reconstruction of both texture and geometry compared to NeRF-based techniques. For the FORTRESS scene, both FlipNeRF and FreeNeRF generate noticeable artifacts and fail to accurately capture the underlying shape. SparseNeRF appears to generate an adequate depth, but upon closer examination, noisy artifacts in both texture and depth become apparent. Our method, on the other hand, is capable of creating high-quality texture with a clean and smooth depth. The second scene, HORNS, is complex with multiple objects with intricate details at various depths. All the NeRF-based approaches introduce significant artifacts in the rendered results. In contrast, our method is able to generate a coherent and high-quality texture and geometry. Note that since our approach is based on 3DGS, our inference speed is significantly higher than the alternatives. On an Nvidia Tesla V100 GPU, our approach has an average inference speed of 278 fps for LLFF data with 3 inputs, while the other approaches perform rendering at 0.08 fps.

As discussed previously, unlike other methods that fill in the occluded regions with blurry and repetitive textures, our method does not reconstruct these areas. This gives us a unique advantage as we can identify and inpaint these areas to hallucinate high-quality details. As a proof of concept, we use a simple strategy to inpaint these areas with a diffusion model and project them to 3D using monocular depth. As shown in Fig. 1 (right) the hallucinated details are of high-quality and consistent (see supplementary video).

Table 2: Numerical comparisons on the NVS-RGBD [48] dataset with 2 and 3 views.

Methods	ZED 2						iPhone					
	PSNR		SSIM		LPIPS		PSNR		SSIM		LPIPS	
	2	3	2	3	2	3	2	3	2	3	2	3
3DGS	13.51	14.43	0.313	0.315	0.545	0.545	12.66	14.22	0.330	0.382	0.563	0.531
RegNeRF	18.90	25.56	0.580	0.803	0.352	0.169	16.04	22.19	0.520	0.708	0.420	0.271
FlipNeRF	17.28	25.56	0.565	0.817	0.402	0.221	17.68	21.92	0.581	0.714	0.386	0.293
FreeNeRF	20.23	25.60	0.689	0.817	0.265	0.166	18.33	22.73	0.598	0.723	0.372	0.281
SparseNeRF	22.07	26.56	0.694	0.835	0.285	0.154	17.87	22.50	0.557	0.725	0.409	0.275
Ours	23.05	24.93	0.774	0.840	0.164	0.135	18.07	21.99	0.615	0.725	0.292	0.228

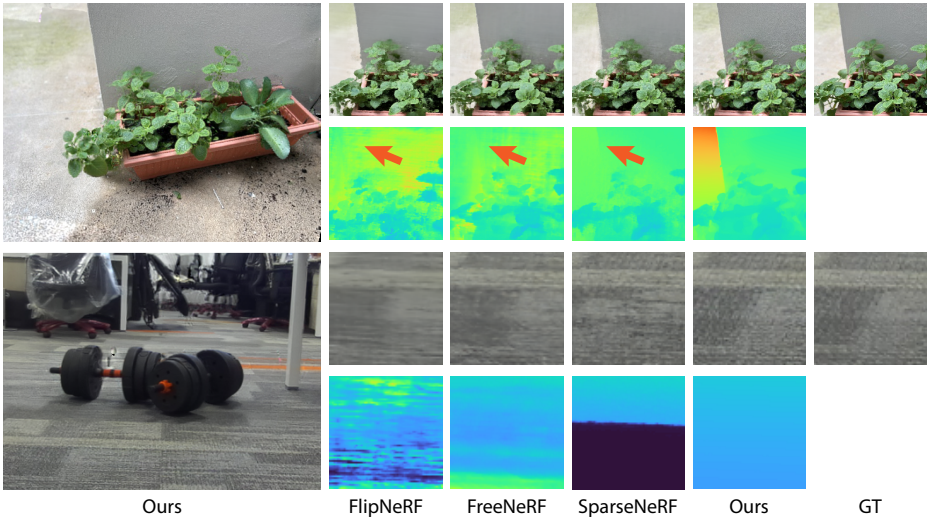

Fig. 8: NVS-RGBD 3 input (iPhone on the top and ZED 2 on the bottom). We show comparisons against other sparse-view NeRF-based approaches. Our approach produces better texture details on the gray wall (top) and carpet (bottom) while reconstructing significantly better geometry.

NVS-RGBD Dataset Next, we numerically compare our method against the other approaches on NVS-RGBD dataset (ZED 2 and iPhone) with 2 and 3 input images. Again our approach produces overall better results than the other methods, particularly in terms of SSIM and LPIPS. The advantage is even larger with 2 input images.

Moreover, we provide visual results on two scenes with 3 inputs from the NVS-RGBD dataset in Fig. 8. For the top scene, NeRF-based approaches are not able to reconstruct the texture details on the gray wall. Additionally, none of the other methods are able to reconstruct the geometry of the floor properly, as indicated by the arrows. Our approach, on the other hand, clearly distinguishes the wall and floor and produces results with detailed texture. Similarly, other

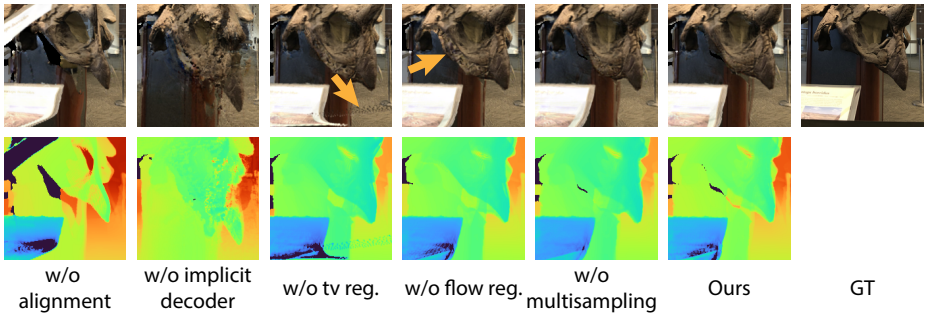

Fig. 9: We visually compare the effect of different components on the reconstruction quality. As seen, all the components are necessary to achieve high-quality results. The effect of multisampling might be difficult to see here and we encourage the readers to see our supplementary video.

techniques fail to properly reconstruct the detailed texture of the carpet for the bottom scene. However, our method produces results that are close to ground truth with a smooth geometry.

5.2 Ablations

Here, we evaluate the effect of various components of our method on the reconstruction quality, both visually (Fig. 9) and numerically (Table. 3). The numerical results are obtained on the LLFF dataset with 3 input views. Without the flow-based coarse alignment (Eq. 8) in the initialization stage, the optimization is unable to align the representation from different images for the complex scenes. During the regularized optimization process, the implicit decoder plays a critical role and ensures the reconstructed geometry is smooth. Furthermore, TV regularization suppresses the isolated point clouds that are not visible from the training views, but appear in a novel view, as indicated by the arrow. Moreover, without the flow-based regularization our approach has difficulty properly reconstructing the texture details in the unconstrained areas, as indicated by the arrow. Finally, multisampling improves the pixel coverage and avoids producing surfaces that appear as semi-transparent from the novel views. While this is difficult to see in Fig. 9, we encourage the readers to see the supplementary video where this effect is clearly visible.

6 Conclusion

In this paper, we present a novel approach to regularize the 3DGS optimization for the sparse input setting. Specifically, we propose to assign a single Gaussian to every pixel of the input images to be able to constrain the Gaussians in 2D image space. We introduce coherency to the 3D Gaussian optimization pipeline using single and multi-view constraints through an implicit decoder and a total

Table 3: Numerical comparisons to highlight the contribution of each component during the optimization on the LLFF dataset with 3 views.

Method	PSNR	SSIM	LPIPS
w/o alignment	19.06	0.679	0.217
w/o implicit decoder	16.68	0.477	0.331
w/o tv reg.	20.20	0.724	0.186
w/o flow reg.	20.32	0.723	0.185
w/o multisampling	19.99	0.718	0.194
Ours	**20.33**	**0.725**	**0.180**

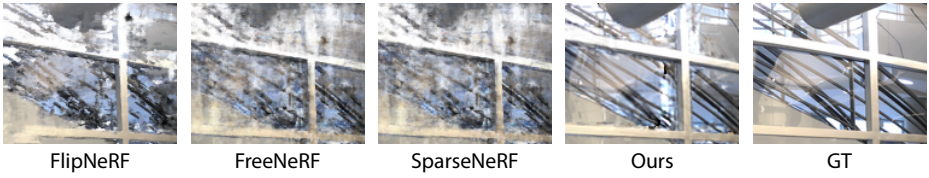

Fig. 10: Our approach assigns a single Gaussian to each pixel and as such reconstructing both the hand rails and the glass pane is difficult. Nonetheless, we are still better than existing NeRF-based approaches.

variation loss, respectively. We use monocular depth and flow correspondences to initialize a set of per-pixel 3D Gaussians to support the regularized optimization. This enables us to learn high quality texture and smooth geometry in the extreme sparse input setting. We demonstrate the superiority of our approach on various scenes from multiple datasets.

Limitations Since we assign a single Gaussian to each pixel, our approach has difficulty handling scenes with transparent objects. An example of such a case is shown in Fig. 10 where our technique is not able to properly reconstruct both the reflections on the glass and the hand rails behind it. Nevertheless, our results are still significantly better than the competing methods. Additionally, our approach relies on the monocular depth and may not be able to produce reasonable results if the depth is highly inaccurate.

References

1. Attal, B., Huang, J.B., Zollhöfer, M., Kopf, J., Kim, C.: Learning neural light fields with ray-space embedding networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 3
2. 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 (ICCV). pp. 5855–5864 (October 2021) 3
3. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. CVPR (2022) 3

4. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Zip-nerf: Anti-aliased grid-based neural radiance fields. ICCV (2023) 3
5. Bemana, M., Myszkowski, K., Seidel, H.P., Ritschel, T.: X-fields: Implicit neural view-, light- and time-image interpolation. ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2020) **39**(6) (2020). https://doi.org/10.1145/3414685.3417827 2, 3, 19
6. Bian, W., Wang, Z., Li, K., Bian, J., Prisacariu, V.A.: Nope-nerf: Optimising neural radiance field with no pose prior. In: CVPR (2023) 3
7. Cao, Y., Cao, Y.P., Han, K., Shan, Y., Wong, K.Y.K.: Dreamavatar: Text-and-shape guided 3d human avatar generation via diffusion models. arXiv preprint arXiv:2304.00916 (2023) 3
8. Chan, E., Monteiro, M., Kellnhofer, P., Wu, J., Wetzstein, G.: pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In: Proc. CVPR (2021) 3
9. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: European Conference on Computer Vision (ECCV) (2022) 3
10. 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) 3
11. Chibane, J., Bansal, A., Lazova, V., Pons-Moll, G.: Stereo radiance fields (srf): Learning view synthesis from sparse views of novel scenes. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (jun 2021) 3
12. Chng, S.F., Ramasinghe, S., Sherrah, J., Lucey, S.: Gaussian activated neural radiance fields for high fidelity reconstruction and pose estimation. In: The European Conference on Computer Vision: ECCV (2022) 3
13. 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 (CVPR) (June 2022) 4
14. Du, Y., Zhang, Y., Yu, H.X., Tenenbaum, J.B., Wu, J.: Neural radiance flow for 4d view synthesis and video processing. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021) 3
15. Gao, C., Saraf, A., Kopf, J., Huang, J.B.: Dynamic view synthesis from dynamic monocular video. In: Proceedings of the IEEE International Conference on Computer Vision (2021) 3
16. Garbin, S.J., Kowalski, M., Johnson, M., Shotton, J., Valentin, J.: Fastnerf: High-fidelity neural rendering at 200fps. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 14346–14355 (October 2021) 3
17. Gu, J., Liu, L., Wang, P., Theobalt, C.: Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=iUuzzTMUw9K 3
18. Guangcong, Chen, Z., Loy, C.C., Liu, Z.: Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. IEEE/CVF International Conference on Computer Vision (ICCV) (2023) 2
19. Guo, Y.C., Kang, D., Bao, L., He, Y., Zhang, S.H.: Nerfren: Neural radiance fields with reflections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18409–18418 (June 2022) 3
20. Höllein, L., Cao, A., Owens, A., Johnson, J., Nießner, M.: Text2room: Extracting textured 3d meshes from 2d text-to-image models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 7909–7920 (October 2023) 3

21. 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 (ICCV). pp. 5885–5894 (October 2021) 4
22. Johari, M., Lepoittevin, Y., Fleuret, F.: Geonerf: Generalizing nerf with geometry priors. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 3
23. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (July 2023), https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/ 2, 3, 4
24. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015) 10
25. Kopanas, G., Leimkühler, T., Rainer, G., Jambon, C., Drettakis, G.: Neural point catacaustics for novel-view synthesis of reflections. ACM Transactions on Graphics (TOG) **41**(6), 1–15 (2022) 4
26. Kopf, J., Rong, X., Huang, J.B.: Robust consistent video depth estimation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021) 7
27. Li, Z., Niklaus, S., Snavely, N., Wang, O.: Neural scene flow fields for space-time view synthesis of dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021) 3
28. Lin, C.H., Ma, W.C., Torralba, A., Lucey, S.: Barf: Bundle-adjusting neural radiance fields. In: IEEE International Conference on Computer Vision (ICCV) (2021) 3
29. Liu, R., Lehman, J., Molino, P., Petroski Such, F., Frank, E., Sergeev, A., Yosinski, J.: An intriguing failing of convolutional neural networks and the coordconv solution. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), https://proceedings.neurips.cc/paper_files/paper/2018/file/60106888f8977b71e1f15db7bc9a88d1-Paper.pdf 2
30. Liu, R., Lehman, J., Molino, P., Petroski Such, F., Frank, E., Sergeev, A., Yosinski, J.: An intriguing failing of convolutional neural networks and the coordconv solution. Advances in neural information processing systems **31** (2018) 19
31. Luo, X., Huang, J., Szeliski, R., Matzen, K., Kopf, J.: Consistent video depth estimation. ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH) **39**(4) (2020) 7, 8
32. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020) 1, 3, 10, 19
33. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. **41**(4), 102:1–102:15 (Jul 2022). https://doi.org/10.1145/3528223.3530127, https://doi.org/10.1145/3528223.3530127 3
34. N, N.S., Soundararajan, R.: Vip-nerf: Visibility prior for sparse input neural radiance fields. ACM SIGGRAPH 2023 Conference Proceedings (2023), https://api.semanticscholar.org/CorpusID:258426778 2, 4
35. Niemeyer, M., Barron, J.T., Mildenhall, B., Sajjadi, M.S.M., Geiger, A., Radwan, N.: Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In: Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (2022) 2, 4, 10
36. Paliwal, A., Tsarov, A., Kalantari, N.K.: Implicit view-time interpolation of stereo videos using multi-plane disparities and non-uniform coordinates. In: Proceedings

of the IEEE Conference on Computer Vision and Pattern Recognition (2023) 2, 3, 19
37. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. arXiv (2022) 3
38. 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 (2020) 3
39. Reiser, C., Peng, S., Liao, Y., Geiger, A.: Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 14335–14345 (October 2021) 3
40. Sara Fridovich-Keil and Alex Yu, Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: CVPR (2022) 3
41. Seo, S., Chang, Y., Kwak, N.: Flipnerf: Flipped reflection rays for few-shot novel view synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 22883–22893 (2023) 2, 4, 10, 11, 19, 20, 21
42. Shi, X., Huang, Z., Li, D., Zhang, M., Cheung, K.C., See, S., Qin, H., Dai, J., Li, H.: Flowformer++: Masked cost volume autoencoding for pretraining optical flow estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1599–1610 (2023) 5, 8
43. Suhail, M., Esteves, C., Sigal, L., Makadia, A.: Light field neural rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8269–8279 (June 2022) 3
44. Sun, C., Sun, M., Chen, H.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: CVPR (2022) 3
45. T, M.V., Wang, P., Chen, X., Chen, T., Venugopalan, S., Wang, Z.: Is attention all that neRF needs? In: The Eleventh International Conference on Learning Representations (2023), https://openreview.net/forum?id=xE-LtsE-xx 3
46. Tretschk, E., Tewari, A., Golyanik, V., Zollhöfer, M., Lassner, C., Theobalt, C.: Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In: IEEE International Conference on Computer Vision (ICCV). IEEE (2021) 3
47. Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J.T., Srinivasan, P.P.: Ref-nerf: Structured view-dependent appearance for neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5491–5500 (June 2022) 3
48. Wang, G., Chen, Z., Loy, C.C., Liu, Z.: Sparsenerf: Distilling depth ranking for few-shot novel view synthesis. In: IEEE/CVF International Conference on Computer Vision (ICCV) (2023) 2, 4, 10, 11, 12, 19, 20, 21
49. Wang, P., Liu, Y., Chen, Z., Liu, L., Liu, Z., Komura, T., Theobalt, C., Wang, W.: F2-nerf: Fast neural radiance field training with free camera trajectories. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4150–4159 (June 2023) 3
50. Wang, Q., Li, Z., Salesin, D., Snavely, N., Curless, B., Kontkanen, J.: 3d moments from near-duplicate photos. In: CVPR (2022) 6
51. Wang, Q., Wang, Z., Genova, K., Srinivasan, P., Zhou, H., Barron, J.T., Martin-Brualla, R., Snavely, N., Funkhouser, T.: Ibrnet: Learning multi-view image-based rendering. In: CVPR (2021) 3
52. Wang, Z., Wu, S., Xie, W., Chen, M., Prisacariu, V.A.: NeRF−−: Neural radiance fields without known camera parameters. arXiv preprint arXiv:2102.07064 (2021) 3

53. Xia, Y., Tang, H., Timofte, R., Gool, L.V.: Sinerf: Sinusoidal neural radiance fields for joint pose estimation and scene reconstruction. In: 33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, November 21-24, 2022. BMVA Press (2022), https://bmvc2022.mpi-inf.mpg.de/0131.pdf 3
54. Xiong, H., Muttukuru, S., Upadhyay, R., Chari, P., Kadambi, A.: Sparsegs: Real-time 360° sparse view synthesis using gaussian splatting. Arxiv (2023) 4
55. 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) 2, 4, 10, 11, 19, 20, 21
56. Yang, L., Kang, B., Huang, Z., Xu, X., Feng, J., Zhao, H.: Depth anything: Unleashing the power of large-scale unlabeled data. In: CVPR (2024) 5, 6, 8
57. Yen-Chen, L., Florence, P., Barron, J.T., Rodriguez, A., Isola, P., Lin, T.Y.: iNeRF: Inverting neural radiance fields for pose estimation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2021) 3
58. Yifan, W., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. ACM Transactions on Graphics (TOG) **38**(6), 1–14 (2019) 4
59. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: pixelNeRF: Neural radiance fields from one or few images. In: CVPR (2021) 3, 4
60. Zeng, L., Kalantari, N.K.: Test-time optimization for video depth estimation using pseudo reference depth. Computer Graphics Forum (2023). https://doi.org/https://doi.org/10.1111/cgf.14729, https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14729 7
61. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: Learning view synthesis using multiplane images. arXiv preprint arXiv:1805.09817 (2018) 1
62. Zhu, Z., Fan, Z., Jiang, Y., Wang, Z.: Fsgs: Real-time few-shot view synthesis using gaussian splatting (2023) 4

Supplementary Material

7 Implicit Decoder Architecture

We use the convolutional decoder network as proposed by Bemana et al. [5,36]. The input coordinate is concatenated to the coordconv [30] layer in the first layer of the decoder. This is followed by a series of convolutional and upsampling layers (bi-linear interpolation) until the desired output resolution is reached. The number of parameters is controlled using a scalar capacity factor that is multiplied to a preset of each layer, e.g., [2c, 4c, 8c] where c is the capacity factor. We use capacity factor [10, 15, 18] for the depth decoder corresponding to 2, 3 and 4 inputs, respectively. Similarly, we use [6, 10, 12] for the opacity decoder.

8 Additional Qualitative Results

We provide additional qualitative results for 2 and 4 input configurations on the LLFF [32] and NVS-RGBD [48] datasets.

LLFF Dataset As shown in Fig. 11, our method produces significantly better texture and geometry compared to other approaches on the LLFF dataset with 2 inputs. For the Orchids scene, all the NeRF-based approaches fail to handle the complex jumble of leaves behind the flower, producing glaring artifacts. Our approach is able to capture the intricate geometry and details. The other scene, Trex, contains a lot of thin details as highlighted by the insets. NeRF-based approaches fail to capture the thin details and produce blurry texture on the trex bones. Our approach is able to reconstruct details while providing a smooth geometry. We show the 4 input results in Fig. 13. SparseNeRF [48] and FlipNeRF [41] produce noisy texture for the Fern scene, while FreeNeRF [55] produces over-blurred results. Our approach produces texture much closer to ground truth in comparison without noise or blurriness. NeRF-based approaches struggle to handle regions that contain relatively sparse supervision (e.g., regions visible in 1 or 2 views out of 4). This is highlighted in the Flower scene. As shown, NeRF-based approaches produce ghosted artifacts while our method is able to generate a coherent geometry and thus high quality details.

NVS-RGBD Dataset NVS-RGBD is a sparse input dataset with 2 and 3 views. We show the 2 input results in Fig. 12. On the Plant scene, the input views have a large angular difference in pose. NeRF-based approaches overfit to the training views and produce geometry and texture with significant artifacts. Our method is able to reconstruct a coherent geometry and high-quality texture in comparison. For the Basketball scene, SparseNeRF and FlipNeRF generate significant artifacts on the shoes while FreeNeRF is unable to capture texture details. Our approach produces details closer to ground truth without any noticeable artifacts. The differences are more prominent in the video comparisons provided in supplementary video.

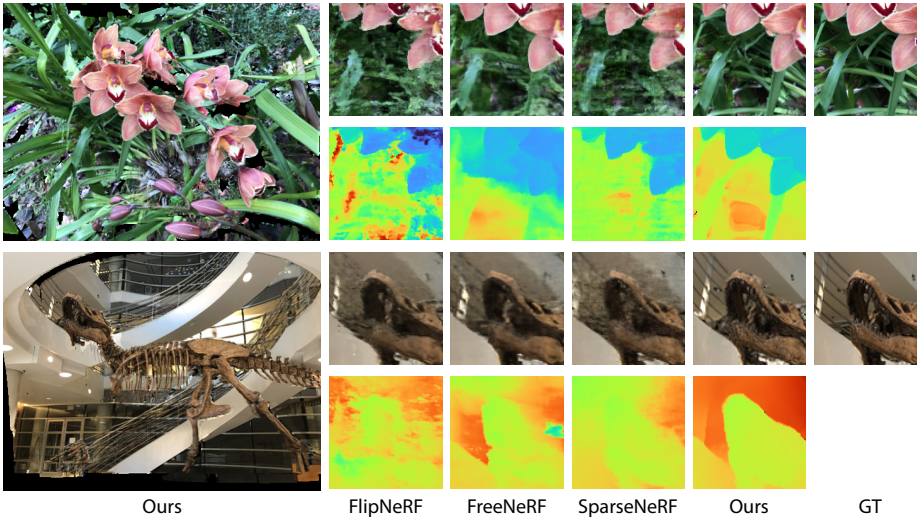

Fig. 11: LLFF 2 input. We show comparisons against other sparse-view NeRF-based approaches, SparseNeRF [48], FlipNeRF [41] and FreeNeRF [55].

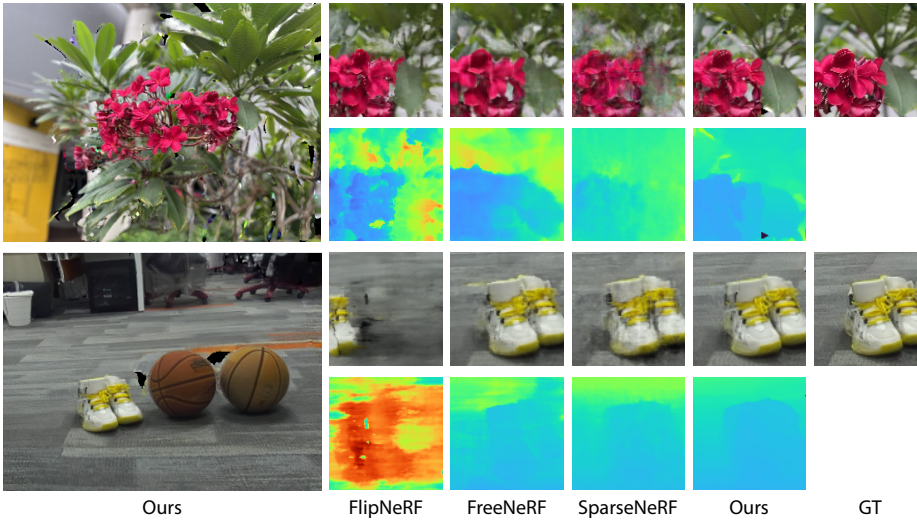

Fig. 12: NVS-RGBD 2 input (iPhone on the top and ZED 2 on the bottom). We show comparisons against other sparse-view NeRF-based approaches.

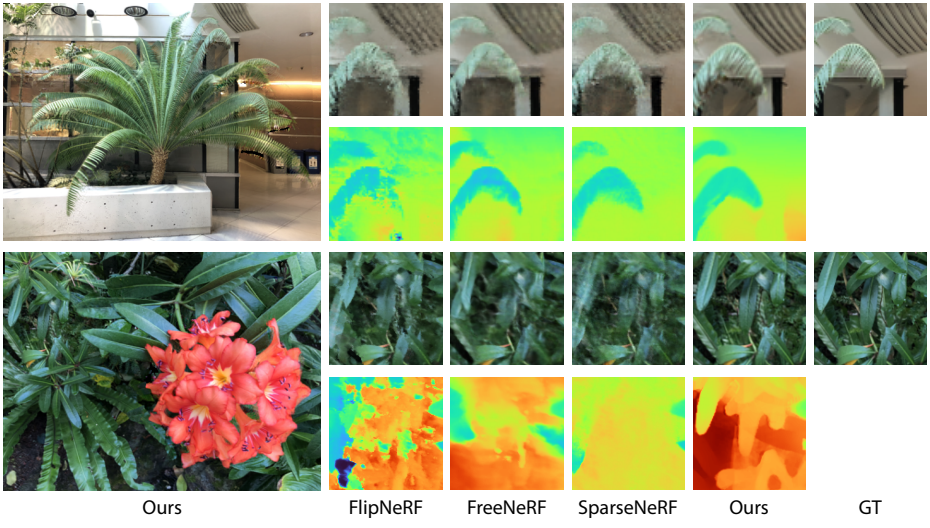

Fig. 13: LLFF 4 input. We show comparisons against other sparse-view NeRF-based approaches, SparseNeRF [48], FlipNeRF [41] and FreeNeRF [55].