This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

SfM-Free 3D Gaussian Splatting via Hierarchical Training

Bo Ji Angela Yao
National University of Singapore
{jibo,ayao}@comp.nus.edu.sg

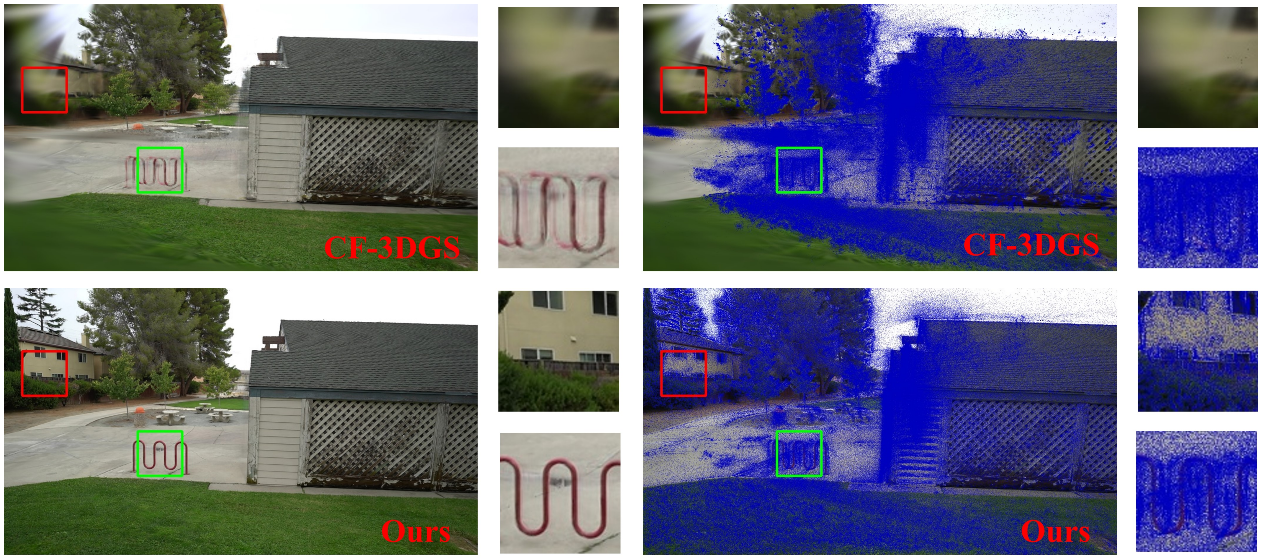


Figure 1. **Novel view synthesis results (left) alongside the projected centers of 3D Gaussians (right).** Each blue dot represents a projected 3D Gaussian center. Our proposal offers two key advantages: 1) Our 3D Gaussians are well-distributed across the scene, whereas CF-3DGS [11] has a notable absence of 3D Gaussians on the image's left side (e.g., in the red region); 2) Our learned 3D Gaussians are of high quality. While CF-3DGS places numerous 3D Gaussians in the green region, the rendering quality there is notably inferior to ours.

Abstract

Standard 3D Gaussian Splatting (3DGS) relies on known or pre-computed camera poses and a sparse point cloud, obtained from structure-from-motion (SfM) preprocessing, to initialize and grow 3D Gaussians. We propose a novel SfM-Free 3DGS (HT-3DGS) method for video input, eliminating the need for known camera poses and SfM preprocessing. Our approach introduces a hierarchical training strategy that trains and merges multiple 3D Gaussian representations – each optimized for specific scene regions – into a single, unified 3DGS model representing the entire scene. To compensate for large camera motions, we leverage video frame interpolation models. Additionally, we incorporate multi-source supervision to reduce overfitting and enhance representation. Experimental results reveal that our approach significantly surpasses state-of-the-art SfM-free novel view synthesis methods. On the Tanks and Temples dataset, we improve PSNR by an average of 2.25dB, with a maximum gain of 3.72dB in the best scene. On the CO3D-V2 dataset, we achieve an average PSNR boost of 1.74dB, with a top gain of 3.90dB. The code is available at https://github.com/jibo27/3DGS_Hierarchical_Training.

1. Introduction

3D Gaussian Splatting (3DGS) [15] represents a 3D scene from multi-view images based on camera intrinsic and extrinsic parameters along with an initial point cloud. Obtaining camera poses and the initial point cloud requires preprocessing, which is often performed using a structure-from-motion (SfM) algorithm [29]. However, SfM can be time-consuming and may struggle with repetitive patterns, textureless regions, or feature extraction errors. Additionally, SfM lacks differentiability, which can limit its applicability in future research [4]. As such, a class of new methods for novel view synthesis is trying to eliminate the need for SfM preprocessing [9, 11, 14, 20].

Removing SfM preprocessing introduces two obvious questions for 3DGS. First, how can the camera poses of the input images be estimated? Second, how can 3D gaussians be initialized and grown within the scene? Inspired by CF-3DGS [11], we address the challenge of constructing an SfM-free 3DGS from video sequences. Assuming video input with small camera movement, we address the issue of camera pose estimation by predicting the relative poses between temporally adjacent frames. By sequentially stacking these relative poses, we obtain the overall camera poses.

To improve camera pose estimation, a key innovation in our work is leveraging a video frame interpolation (VFI) model to generate additional frames. We using an off-the-shelf deep model [18] to double the input video length by interpolating between frames. Although these interpolated frames are not rendered from an underlying 3D model and may lack perfect geometric consistency, they provide sufficient quality (see Fig. 3) to bridge relative poses between frames, which is particularly beneficial for sequences with larger camera movements. They also provide additional supervision, covering viewpoints not present in the original training frames. Incorporating these interpolated frames into 3DGS training yields a 0.35 dB performance boost on the Tanks and Temples dataset [17].

To address the second question of initializing and growing the 3D Gaussians, a straightforward way would be to use a point cloud derived from the depth map of the first frame; however, this often leads to sparse Gaussian coverage for regions not visible in the first frame. The standard adaptive density control [15] – which adjusts 3D Gaussians by splitting, cloning, and pruning – struggles in these sparsely covered regions. In these areas, the Gaussians may have very small gradients, making it challenging to activate densification processes [5, 39].

To this end, we propose a novel hierarchical training strategy that merges multiple base 3DGS models, each optimized for specific parts of the scene, into a unified model representing the entire scene. Intuitively, the adaptive density control encounters difficulties in regions with sparse 3D Gaussians; however, with our strategy, these regions are populated with 3D Gaussians merged from other 3DGS models. Interestingly, this merging strategy can be viewed as a densification process: we discard unimportant 3D Gaussians and densify the representation by merging essential Gaussians from different base 3DGS models. Fig 1 illustrates the improved 3D Gaussian coverage achieved by our approach compared to a naive strategy without hierarchical training. This strategy boosts PSNR by 1.19–1.58 dB on the Tanks and Temples dataset[17].

Furthermore, we enhance the representation quality through multi-source supervision, leveraging both base 3DGS models and interpolated frames from VFI.

Our approach achieves a significant PSNR improvement of 2.25 dB on the Tanks and Temples [17] and 1.74 dB on the CO3D-V2 [28] over state-of-the-art SfM-free novel view synthesis methods. Even without known camera intrinsics, our method surpasses the state-of-the-art methods by 0.89 dB in PSNR. Our contributions are as follows:

- We improve pose estimation by leveraging video frame interpolation to smooth camera motion.
- We introduce a hierarchical training strategy to address initialization and density control challenges without SfM preprocessing. Interestingly, this approach can be interpreted as a densification step.
- We employ multi-source supervision, reusing base 3DGS models and VFI-interpolated frames to reduce overfitting.
- Together, these innovations yield a 3DGS approach that requires no SfM preprocessing, significantly outperforming existing SfM-free novel view synthesis methods.

2. Related Works

Novel view synthesis is the task of predicting realistic images from an unobserved viewpoint. NeRFs [24] are an implicit 3D representation that encode scenes within an MLP. They are remarkable at rendering images from novel views, but despite the various proposed improvements [1, 2, 10, 12], they are still relatively slow and may require up to several minutes to render a scene. In contrast, 3D Gaussian splats [15], as an explicit representation, are able to achieve high-quality renderings at much faster speeds. Subsequent works build upon 3D Gaussian splatting, dealing with revising the density [5, 39], compressing the representation [19, 25, 26] and anti-aliasing [35, 37]. Others extend 3D Gaussian splatting to dynamic [8, 30, 33] and large, city-scale scenes [16, 22].

SfM-free novel view synthesis for both NeRFs and 3D Gaussian splatting is a class of works that try to do away with known or estimated camera pose from SfM. Examples include i-NeRF [36], which estimates camera poses by aligning keypoints using a pre-trained NeRF. Follow-ups like NeRFmm [32], SiNeRF [34], BARF [21] and GARF [7] learn both the NeRF model and camera pose embeddings simultaneously [32], addressing the gradient inconsistency [7, 21], leveraging pre-trained networks for monocular depth estimation or optical flow, incorporating prior geometric knowledge or correspondence information [4, 6, 23].

For 3D Gaussian splatting, CF-3DGS [11] and GGRt [20], InstantSplat [9], COGS [14] was developed to support SfM-free optimization. CF-3DGS [11] performs an affine transformation on the positions of the 3D Gaussians to predict relative poses, progressively expanding the representations from the first frame to the last frame. However, its performance is limited by the accuracy of estimated camera poses. It also suffers when there is insufficient initialization of 3D Gaussians, and has challenges in density control. GGRt [20] jointly learns two modules for iterative pose optimization and a generalizable 3DGS. On the other hand, InstantSplat [9] and COGS [14] are designed primarily for scenarios with sparse image views. In this paper, we focus on the video input with small camera movement, similar to CF-3DGS [11]. We address two main challenges associated with applying 3DGS in SfM-free tasks: improving camera pose estimation and enhancing the initialization and learning of 3D Gaussians.

3. Approach

3.1. Overview

Consider a video sequence $\mathcal{I} = \{I_i \mid i = 1, \ldots, N\}$ captured with small camera movements. We aim to reconstruct a 3D Gaussian splatting representation (3DGS) $\mathcal{S}$ from $\mathcal{I}$ and camera intrinsics K. We first estimate a series of camera poses $\mathcal{P} = \{P_i \mid i = 1, \ldots, N\}$ (Section 3.2). Then, we partition the video into overlapping segments $\{C_j\}$. For each segment C_j, a base 3DGS model $\mathcal{S}_{C_j}$ is trained. These models are then iteratively merged from adjacent segments to form a unified representation (Section 3.3). After each merge, we retrain the merged 3DGS model using original training frames, pseudo-view frames from the base models, and interpolated frames from VFI on the combined segments (Section 3.4). This merging and retraining process continues until we obtain the final 3DGS model $\mathcal{S}$, representing the entire sequence $\mathcal{I}$. Fig. 2 provides an overview of the pipeline.

3.2. Camera pose estimation

We estimate the camera poses $\mathcal{P}$ by stacking relative camera poses between temporally adjacent pairs of frames. The camera pose of the first frame is set to have no rotation or translation, i.e., $P_1 = [\mathbb{I}|\mathbf{0}]$, serving as the reference frame. The estimated poses for all subsequent frames are with respect to this first frame. For each frame pair (I_i, I_{i+1}), we estimate the relative pose $T_{i \to i+1}$ across all $N-1$ pairs. The camera pose for the i-th frame is the matrix multiplication of the previous relative camera poses:

$$P_i := T_{1 \to i} = T_{i-1 \to i} \odot \cdots \odot T_{2 \to 3} \odot T_{1 \to 2}. \qquad (1)$$

While stacking relative poses can accumulate error, directly estimating each frame's pose with respect to the first frame is more challenging due to the larger camera displacement.

Relative pose estimation. As identified in [11], the relative poses between two frames can be approximated by estimating an affine transformation, denoted as A, which is applied to 3D Gaussians from the first frame. After the transformation, the rendered image with respect to the second frame should align with the second frame. Specifically, the 2D projection $\mu_{2\text{D}}$ of a 3D Gaussian with position μ under the pose P is given by $\mu_{2\text{D}} = K(P\mu)/(P\mu)_z$. This can be approximated by applying an affine transformation A to μ, followed by a projection using the identity camera pose $[\mathbb{I}|0]$, yielding $\mu_{2\text{D}} = K(\mathbb{I}\mu')/(\mathbb{I}\mu')_z$, where $\mu' = A\mu$. As a result, the relative pose can be estimated from A.

In practice, to estimate the relative pose from I_i to I_{i+1}, we first construct a single-image 3DGS model $\mathcal{S}_i$ optimized exclusively on I_i. We then apply the affine transformation A to each Gaussian in $\mathcal{S}_i$ and render the image $\hat{I}_{i+1}$ using the camera pose $[\mathbb{I}|\mathbf{0}]$. The reconstructed $\hat{I}_{i+1}$ is expected to match I_{i+1}. We optimize A by minimizing the photometric loss between the rendered image $\hat{I}_{i+1}$ and the target image I_{i+1}. During optimization, the attributes of 3D Gaussians in $\mathcal{S}_i$ are fixed, and only A is optimized. The optimized transformation matrix A corresponds to the estimated relative pose $T_{i \to i+1}$.

Relative pose estimation with video frame interpolation. When the camera movement between adjacent frames is small, the relative pose estimation described above performs well because of sufficient frame overlap. However, with larger camera motions, performance decreases, and the fixed single-image 3DGS model from the previous frame may fail to render a high-quality image $\hat{I}_{i+1}$. This is because larger camera motions introduce more unseen content that the previous frame may not cover, leading to optimization objectives impacted by artifacts and resulting in poorly estimated poses. For example, the rendered $\hat{I}_{i+1}$ exhibits artifacts in such regions, as shown in Fig. 3d.

A key insight of our work is compensating for large camera motions with a well-trained video frame interpolation (VFI) model [18]. Let $I_{i+0.5}$ represent an interpolated frame between I_i and I_{i+1}, where the decimal 0.5 indicates the interpolated result. We estimate the relative poses between I_i and $I_{i+0.5}$, and between $I_{i+0.5}$ and I_{i+1}, separately. The overall relative pose is then given by $T_{i \to i+1} = T_{i \to i+0.5} \odot T_{i+0.5 \to i+1}$. By reducing the relative camera motion in each step, fewer artifacts are introduced (see Figs. 3e and 3f).

3.3. Hierarchical Training

After estimating the camera poses $\mathcal{P}$, the next step is to initialize and grow 3D Gaussians from the input video. A straightforward approach is to initialize the 3D Gaussians using point clouds lifted from the depth map [3, 27] of the first frame, then gradually grow the 3D Gaussians by processing frames sequentially. This initialization, based on a single frame, is incomplete in covering the scene. Moreover, the standard adaptive density control [15], which relies on accumulated gradients to split and clone Gaussians or prunes them based on low opacity, struggles

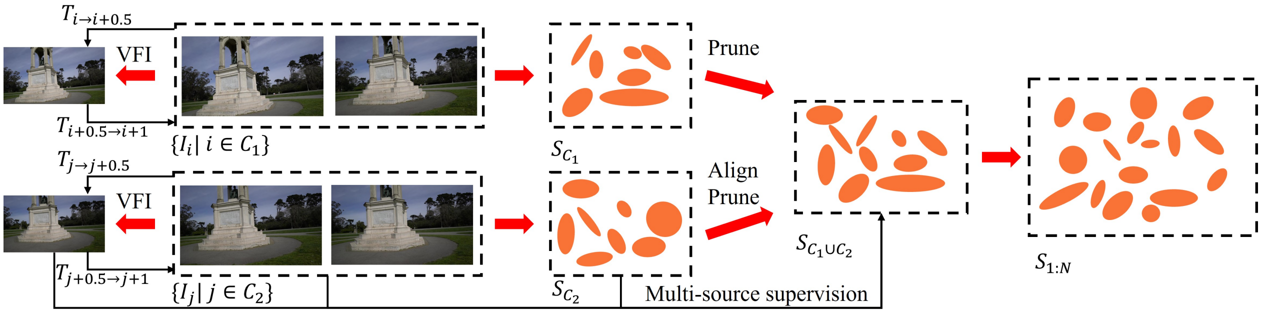
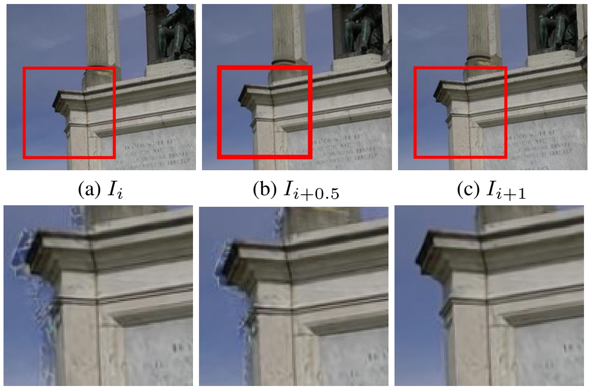


Figure 2. **Overview of our proposal.** We partition the video into multiple segments, train a base 3DGS model on each segment individually, and then iteratively merge these base models into a single, unified 3DGS model representing the entire scene.



(a) I_i (b) $I_{i+0.5}$ (c) I_{i+1}

(d) $\hat{I}_{i+1}$ w/o VFI (e) $\hat{I}_{i+0.5}$ w/ VFI (f) $\hat{I}_{i+1}$ w/ VFI

Figure 3. **Effect of VFI on relative pose estimation between I_i (3a) and I_{i+1} (3c).** Fig 3b shows the interpolated frame. In Fig 3d, artifacts are noticeable in regions affected by camera movement, which VFI helps reduce. Fig 3e and 3f show fewer artifacts in the rendered interpolated and original frames.

in regions with sparse Gaussians distributions. In such areas, Gaussians may have very small gradients, and thus fail to activate the densification process. Consequently, as shown in Fig. 1, areas in the scene not covered in the first frame exhibit a noticeable lack of Gaussians. To overcome this limitation, we propose the hierarchical training strategy.

Video partitioning. We first partition the N frames into overlapping segments, with the j-th segment denoted as C_j. Ideally, each segment features similar scene content so that the 3DGS model is trained without encountering many unseen regions. To achieve this, we reuse the estimated relative camera poses. Segments C_j and C_{j+1} are separated by locating adjacent image pairs with the largest camera movement. Empirically, we observe that evenly partitioning the video into segments yields similar results.

Training of base 3DGS models. After partitioning the video into segments, we train the 3DGS model $\mathcal{S}_{C_j}$ for each segment C_j. The 3D Gaussians are initialized using a point cloud predicted from the first frame of C_j by depth estimators [3, 27]. The Gaussians are then optimized from the first to the last frame in each segment, growing as needed to capture the scene details.

Merging of base 3DGS models. To represent the entire scene, we merge these base 3DGS models. Consider the merging of two base 3DGS models $\mathcal{S}_{C_j}$ and $\mathcal{S}_{C_{j+1}}$. Since each model is optimized with its 3D Gaussian positions aligned to the first frame of its respective segment, it is necessary to align the two models before merging. The alignment is performed by transforming the 3D Gaussians in $\mathcal{S}_{C_{j+1}}$ to match those in $\mathcal{S}_{C_j}$, based on the relative pose $T_{C_{j+1,1}\to C_{j,1}}$, where $C_{j,1}$ and $C_{j+1,1}$ denote the first frames of their respective segment.

For merging two 3DGS models, one potential idea is to identify correspondences between 3D Gaussians in the two models and interpolate matched pairs. However, this idea presents several challenges in our task. First, establishing accurate correspondences between Gaussians is non-trivial. For instance, one should not interpolate the Gaussians in the non-overlapping regions in the scene. Yet distinguishing between overlapping and non-overlapping regions can be difficult. Secondly, even with correspondences identified, variations in properties like opacity, scale, covariance, and color among matched Gaussians could complicate the interpolation process [16].

Therefore, instead of interpolation, we take a simple yet effective merging strategy: first, prune unimportant 3D Gaussians in each model, then concatenate the remaining Gaussians. Specifically, we first assign an importance score to each 3D Gaussian. Our importance score is inspired by a 3DGS compression strategy [26], which defines the importance of a given parameter p for a 3D Gaussian as:

$$f(p) = \frac{1}{\sum_{i=1}^{N} H_i W_i} \sum_{i=1}^{N} \left| \frac{\partial \left(\sum_{x,y} \hat{I}_i(x,y) \right)}{\partial p} \right| . \quad (2)$$

Above, $\sum_{x,y} \hat{I}_i(x,y)$ represents the sum of RGB values in the rendered image $\hat{I}_i$, and H_i and W_i are the height and width of image $\hat{I}_i$. Parameter p is a general notation representing variables such as color c, opacity α, or covariance. The importance score evaluates the sensitivity of the rendering quality with respect to changes to parameter p of a 3D Gaussian. If a minor change in the parameter significantly affects the rendered image, that parameter is considered important. We compute the importance score for each 3D Gaussian in base 3DGS models using Eq 2 and keep the top γ percent from base 3DGS models. Finally, we take the union of the selected 3D Gaussians to form the merged representation.

It is feasible to take the union since 3DGS is an explicit representation. While this approach may introduce redundant Gaussians, we empirically observe that adaptive density control [15] is more effective at pruning redundant Gaussians than at generating new or high-quality Gaussians in sparse regions. If we anchor the model to $\mathcal{S}_{C_j}$, our merging strategy can be viewed as a densification step: it removes less important Gaussians and densifies the representation by adding importance Gaussians from $\mathcal{S}_{C_{j+1}}$.

Hierarchical training. After defining the partition and merging strategy, we describe the hierarchical training pipeline that merges multiple base 3DGS models, each optimized for individual segments, into a unified model. First, we define a hierarchical level L and partition the input into 2^L overlapping segments, resulting in 2^L base models. We iteratively merge adjacent pairs, reducing the number of models by half in each step, until only one unified model remains. For example, with $L = 2$, we create four base models: $\mathcal{S}_{C_1}$, $\mathcal{S}_{C_2}$, $\mathcal{S}_{C_3}$, and $\mathcal{S}_{C_4}$. The first merge yields $\mathcal{S}_{C_1 \cup C_2}$ and $\mathcal{S}_{C_3 \cup C_4}$, and the final merge gives $\mathcal{S}_{1:N} := \mathcal{S}_{C_1 \cup C_2 \cup C_3 \cup C_4}$. Hierarchical training requires all input frames in advance. We also explore an online approach in which we sequentially merge $\mathcal{S}_{C_1 \cup C_2}$ with $\mathcal{S}_{C_3}$, then with $\mathcal{S}_{C_4}$, yielding $\mathcal{S}_{1:N}$. We refer to this variant as *progressive training*. Both strategies improve the baseline by at least 1.32dB (Table 5), with slightly better results for hierarchical training.

3.4. Multi-source supervision

After merging $\mathcal{S}_{C_j}$ and $\mathcal{S}_{C_{j+1}}$ and pruning unimportant 3D Gaussians, the newly merged representation needs to be retrained. Simply retraining on the set of images from $C_j \cup C_{j+1}$ leads to overfitting on those specific images. To address this, we propose to augment training with two additional sets of images: (1) from the base 3DGS models $\mathcal{S}_{C_j}$ and $\mathcal{S}_{C_{j+1}}$ before merging, and (2) from the interpolated frames (see Section 3.2).

Supervision from base 3DGS models. As the base 3DGS models are better-optimized for their respective segments, novel pseudo-views rendered from these models can serve as a source of training images for the merged 3DGS. To generate novel views, we first sample a virtual camera pose $P_{i+\tau}$ between two poses, P_i and P_{i+1}, using the formula:

$$P_{i+\tau} = P_i \exp\left(\tau \log\left(P_i^{-1} \cdot P_{i+1}\right)\right), \tag{3}$$

where $\tau \in (0, 1)$ and $P_i, P_{i+1} \in \mathrm{SE}(3)$ represent camera poses in the $\mathrm{SE}(3)$ space. This smooth interpolation enables the creation of pseudo-views, which are then rendered as additional supervision for the merged 3DGS model.

Supervision from video frame interpolation. The interpolated images from VFI are of sufficiently high quality at viewpoints not covered by the training frames, making them suitable for supervision. For instance, Fig. 3b shows an example of an interpolated frame. To supervise the merged 3DGS with the interpolated frames, we need to estimate their corresponding camera pose. For an interpolated frame $I_{i+0.5}$ between frames I_i and I_{i+1}, the camera pose $P_{i+0.5}$ is computed as $P_{i+0.5} = T_{i\rightarrow i+0.5} \odot \cdots \odot T_{1.5\rightarrow 2} \odot T_{1\rightarrow 1.5}$. Each relative pose was previously calculated in Section 3.2, so no additional computational overhead is required.

Loss function. We optimize the 3DGS model using the photometric loss between the rendered image and the training image or the pseudo image [15]:

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}, \tag{4}$$

where $\mathcal{L}_1$ is the L1 loss, and $\mathcal{L}_{\text{D-SSIM}}$ is the D-SSIM term.

4. Experiment

4.1. Experimental setting

Datasets. We conduct experiments on the Tanks and Temples dataset [17] and the CO3D-V2 dataset [28]. For Tanks and Temples, following [4, 11], we sequentially divide the frames into groups of 8, using one of the frames as the test frame and the remaining frames for training. For the *Family* scene on Tanks and Temples, we alternate images, using every other image as a test image. The CO3D-V2 dataset is more challenging due to its larger camera motions. We adopt the same sampling strategy, using every eighth frame for testing and the rest for training.

Metrics. We use PSNR, SSIM [31], and LPIPS [38] to evaluate the effectiveness of novel view synthesis. For camera pose estimation, we report the Absolute Trajectory Error (ATE) and Relative Pose Error (RPE), similar to [4, 11]. ATE measures the difference of the camera positions. RPE measures the relative pose errors, containing relative rotation error (RPE$_r$) and relative translation error (RPE$_t$). Additionally, we report the memory size required to store the optimized parameters.

Implementation details. The hierarchical training level is set to $L = 2$ as it reaches the saturation point (see Table 4).

Scenes	BARF [21]			SC-NeRF [13]			Nope-NeRF [4]			CF-3DGS [11]			Ours		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Church	23.17	0.62	0.52	21.96	0.60	0.53	25.17	0.73	0.39	30.23	0.93	0.11	31.34	0.94	0.08
Barn	25.28	0.64	0.48	23.26	0.62	0.51	26.35	0.69	0.44	31.23	0.90	0.10	34.95	0.97	0.05
Museum	23.58	0.61	0.55	24.94	0.69	0.45	26.77	0.76	0.35	29.91	0.91	0.11	31.59	0.95	0.08
Family	23.04	0.61	0.56	22.60	0.63	0.51	26.01	0.74	0.41	31.27	0.94	0.07	34.71	0.97	0.05
Horse	24.09	0.72	0.41	25.23	0.76	0.37	27.64	0.84	0.26	33.94	0.96	0.05	35.82	0.98	0.03
Ballroom	20.66	0.50	0.60	22.64	0.61	0.48	25.33	0.72	0.38	32.47	0.96	0.07	34.12	0.97	0.04
Francis	25.85	0.69	0.57	26.46	0.73	0.49	29.48	0.80	0.38	32.72	0.91	0.14	34.09	0.93	0.13
Ignatius	21.78	0.47	0.60	23.00	0.55	0.53	23.96	0.61	0.47	28.43	0.90	0.09	31.64	0.95	0.06
Mean	23.42	0.61	0.54	23.76	0.65	0.48	26.34	0.74	0.39	31.28	0.93	0.09	33.53	0.96	0.07

Table 1. **Novel view synthesis results on Tanks and Temples [17].** We achieve the best performance among all competitors.

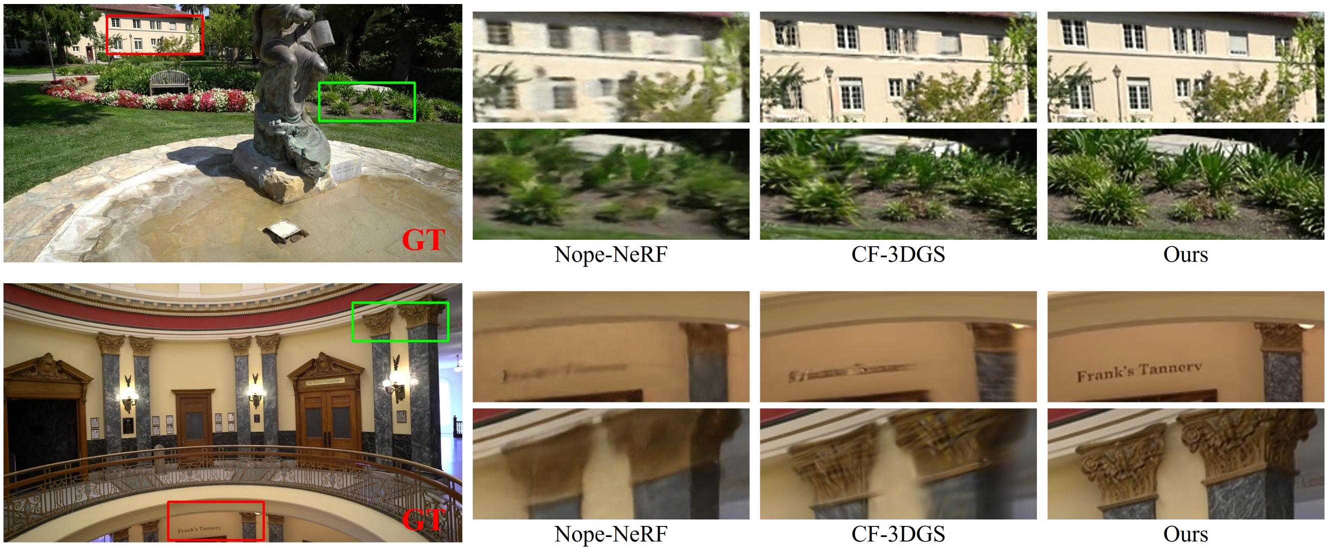


Figure 4. **Qualitative novel view synthesis results on Tanks and Temples [17].** Our proposal achieves superior rendering quality.

Each segment's base 3DGS model is trained from start to end frame with 300 iterations per frame. Multi-source supervision also employs 300 iterations per frame. During the merging, we select the top 50% of 3D Gaussians from each model based on importance score. Multi-source supervision involves two steps: first, using pseudo-view images from base models and original frames; second, using interpolated frames and original frames, with a 50% probability of selecting pseudo-view or interpolated frame for each step. Gaussians are grown and pruned every 100 and 2000 iterations on the Tanks and Temples and CO3D-V2 datasets, respectively. All experiments are conducted on a single RTX A5000 GPU.

4.2. Evaluation on the Tanks and Temples Dataset

Quantitative comparison. We perform the comparison with the state-of-the-art novel view synthesis methods without SfM preprocessing, including BARF [21], SC-NeRF [13], Nope-NeRF [4] and CF-3DGS [11] on Tanks and Temples [17]. As shown in Table 1, our method achieves superior performance compared to all of them. Specifically, compared to CF-3DGS, we improve the average PSNR by 2.25 dB, SSIM by 0.03, and reduce LPIPS by 0.02. The most significant improvement is observed in the *Barn* scene, with a PSNR increase of 3.72 dB, SSIM improvement of 0.07, and LPIPS reduction of 0.05.

Qualitative comparison. Fig. 4 shows that we achieve finer detail and higher fidelity, especially in highly detailed regions where CF-3DGS struggles to grow 3D Gaussians, highlighting the advantages of our training strategy.

4.3. Evaluation on the CO3D-V2 Dataset

Quantitative comparison. Due to the challenges posed by this dataset, we limit our comparison to the most advanced methods, namely Nope-NeRF [4] and CF-3DGS [11]. Table 2 shows that our method outperforms these approaches, with an average PSNR boost of 1.74 dB. The most significant improvement is seen in the scene *34_1403_4393* (teddybear), where we increase the PSNR by over 3.90 dB, improve SSIM by 0.07, and reduce LPIPS by 0.06.

Qualitative comparison. Fig. 5 demonstrates that even with challenging input videos, our method maintains high performance compared to CF-3DGS, which exhibits blur and unrealistic red artifacts in the *34_1403_4393* (teddy-

Method	34_1403_4393			106_12648_23157			110_13051_23361			245_26182_52130			415_57112_110099		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Nope-NeRF [4]	28.62	0.80	0.35	20.41	0.46	0.58	26.86	0.73	0.47	25.05	0.80	0.49	24.78	0.64	0.55
CF-3DGS [11]	27.75	0.86	0.20	22.14	0.64	0.34	29.69	0.89	0.29	27.24	0.85	0.30	26.21	0.73	0.32
Ours	32.52	0.93	0.14	23.43	0.73	0.28	29.95	0.87	0.19	28.59	0.87	0.27	27.23	0.78	0.30

Table 2. **Novel view synthesis results on CO3D-V2 [28].** We achieve superior performance over all competitors, with the largest improvement on *34_1403_4393*, where our method increases PSNR by 3.90 dB compared to Nope-NeRF.

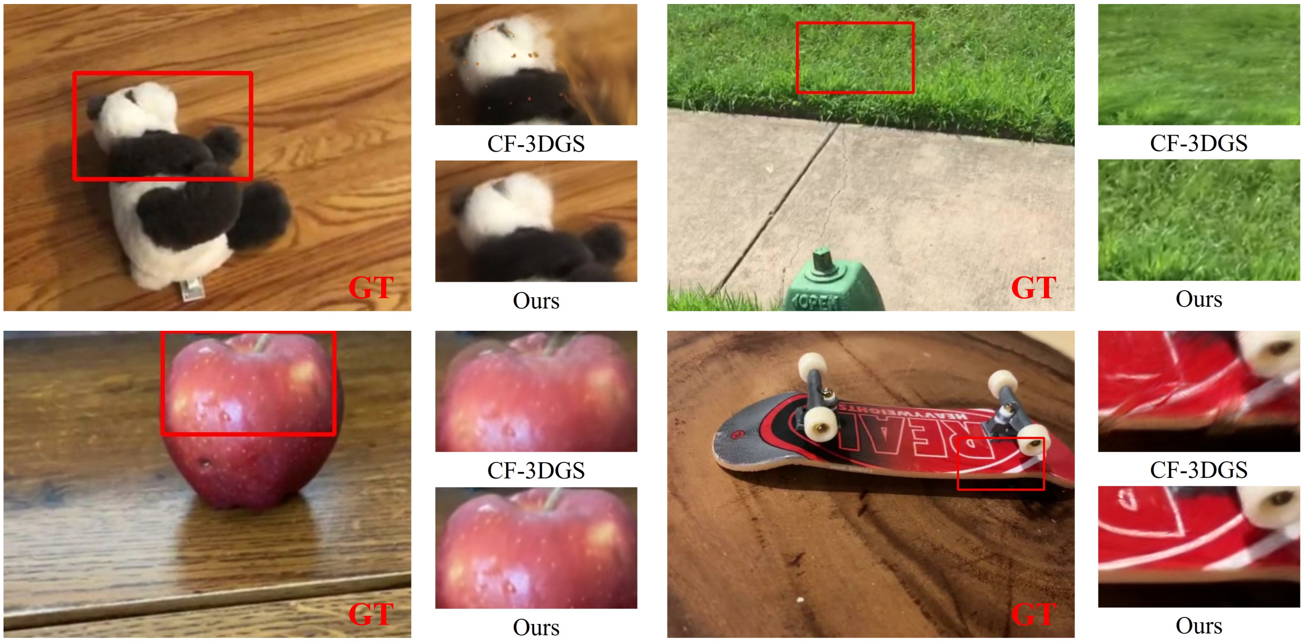


Figure 5. **Qualitative novel view synthesis results on CO3D-V2 [28].** Our proposal achieves superior rendering quality.

bear) scene due to suboptimal 3D Gaussian learning.

Camera pose estimation. We conduct camera pose estimation comparisons only on CO3D-V2, as it provides ground-truth camera poses. As shown in Table 3, our method achieves comparable or superior performance to all competitors, reducing RPE$_t$ and RPE$_r$ by up to 0.464 and 0.078, respectively. Our approach's improvements on ATE are less consistent. We hypothesize that this is due to errors in the interpolated images generated by the VFI model, to which camera position estimation is particularly sensitive.

4.4. Ablation study

We conduct an ablation study on the Tanks and Temples dataset [17]. Table 5 reports the average PSNR, SSIM, and LPIPS across all scenes. The baseline generally follows CF-3DGS [11], with adjustments to certain hyperparameters to align with our strategy.

Progressive vs. hierarchical training. Table 5 shows that hierarchical training (HT) outperforms progressive training (PT) by 0.2 dB (Variant 6 vs. 7), with both improving PSNR over the baseline (Variant 3) by more than 1.32 dB. This supports our claim that merging 3D Gaussians from different base models enhances results. HT performs better by balancing training across early and late frames, whereas PT, which allocates more iterations to early frames, tends to overfit them. However, PT is suitable for online tasks, unlike HT. Moreover, to test if the performance gains result from additional training iterations, we retrain the baseline model on the entire input (referred to as global training, GT). GT shows minimal improvement, indicating that retraining alone does not improve the performance.

Hierarchical training level. We evaluate the effectiveness of hierarchical training at various levels L, with results in Table 4. At $L = 0$, the 3DGS model is trained by treating the entire input as a single segment. Our strategy notably boosts PSNR by 1.19–1.58 dB and SSIM by 0.02, while reducing LPIPS by 0.01–0.02. Performance increases with higher levels, saturating at $L=2$, where the video is divided into four segments, sufficient for 3D Gaussian learning on the about 150-frame Tanks and Temples dataset. Our proposal also lowers memory storage by 0.22–0.26 GB. While it may seem counterintuitive, hierarchical training reduces memory storage for two reasons: (1) pruning unimportant Gaussians before merging, maintaining a stable count; (2) optimized segment-wise Gaussians are more representative, unlike the baseline model, which redundantly clones and

Method	34_1403_4393			106_12648_23157			110_13051_23361			245_26182_52130			415_57112_110099		
	RPE$_t$ ↓	RPE$_r$ ↓	ATE ↓	RPE$_t$ ↓	RPE$_r$ ↓	ATE ↓	RPE$_t$ ↓	RPE$_r$ ↓	ATE ↓	RPE$_t$ ↓	RPE$_r$ ↓	ATE ↓	RPE$_t$ ↓	RPE$_r$ ↓	ATE ↓
Nope-NeRF [4]	0.591	1.313	0.053	0.387	1.312	0.049	0.400	1.966	0.046	0.587	1.867	0.038	0.326	1.919	0.054
CF-3DGS [11]	0.505	0.211	0.009	0.094	0.360	0.008	0.140	0.401	0.021	0.239	0.472	0.017	0.110	0.424	0.014
Ours	0.041	0.170	0.009	0.045	0.282	0.014	0.093	0.331	0.020	0.064	0.438	0.017	0.049	0.351	0.024

Table 3. **Camera pose estimation results on CO3D-V2 [28].** We achieve significant improvements in RPE$_t$ and RPE$_r$, with a slight decrease in performance on ATE. We hypothesize that ATE is more sensitive to errors in the interpolated frames from VFI.

Scenes	$L = 0$				$L = 1$				$L = 2$				$L = 3$			
	PSNR ↑	SSIM ↑	LPIPS ↓	Mem ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Mem ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Mem ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Mem ↓
Church	30.44	0.93	0.09	1.06	31.40	0.94	0.08	0.84	31.67	0.95	0.08	0.85	31.61	0.94	0.08	0.83
Barn	30.09	0.88	0.11	1.44	32.06	0.92	0.08	1.43	32.27	0.92	0.08	1.41	32.20	0.92	0.08	1.38
Museum	30.24	0.91	0.10	1.35	30.95	0.94	0.08	1.17	31.75	0.94	0.07	1.10	31.97	0.95	0.07	1.09
Family	33.12	0.96	0.05	1.44	33.71	0.96	0.05	1.18	34.20	0.97	0.05	1.21	34.15	0.97	0.05	1.18
Horse	34.08	0.96	0.05	1.07	35.44	0.98	0.04	0.96	35.44	0.98	0.04	0.90	35.54	0.98	0.03	0.92
Ballroom	32.82	0.96	0.05	1.39	33.17	0.97	0.05	1.14	33.41	0.97	0.04	1.14	33.67	0.97	0.04	1.13
Francis	32.84	0.92	0.14	0.81	33.64	0.92	0.13	0.68	33.66	0.92	0.13	0.75	33.62	0.92	0.13	0.64
Ignatius	28.37	0.91	0.09	2.09	31.15	0.94	0.06	1.50	31.78	0.94	0.06	1.43	31.90	0.95	0.06	1.40
Mean	31.50	0.93	0.09	1.33	32.69	0.95	0.08	1.11	33.02	0.95	0.07	1.10	33.08	0.95	0.07	1.07

Table 4. **Ablation study of the hierarchical training level.** Memory storage is measured in gigabytes.

Id	Variant	PSNR ↑	SSIM ↑	LPIPS ↓
(1)	CF-3DGS	31.28	0.93	0.09
(2)	CF-3DGS + VFI	31.45	0.93	0.09
(3)	Baseline	31.50	0.93	0.09
(4)	+ Global Training (GT)	31.52	0.93	0.08
(5)	+ GT + VFI	31.95	0.93	0.08
(6)	+ Progressive Training	32.82	0.95	0.07
(7)	+ Hierarchical Training (HT)	33.02	0.95	0.07
(8)	+ HT + VFI	33.37	0.95	0.07
(9)	+ HT + VFI + Base (Ours)	33.53	0.96	0.07
(10)	Ours w/o intrinsic	32.17	0.94	0.09
(11)	NeRF + SfM	26.61	0.75	0.38
(12)	3DGS + SfM	30.20	0.92	0.10

Table 5. **Ablation study on Tanks and Temples [17].** Incorporating all components yields the best performance.

splits Gaussians in regions with sparse Gaussians.

Effectiveness of video frame interpolation (VFI). We evaluate the impact of VFI in Table 5. VFI smooths camera motion and provides additional supervision, resulting in an average PSNR gain of 0.17dB (Variant 1 vs. 2), 0.43dB (Variant 4 vs.5) 0.35 dB (Variant 7 vs. 8). VFI improves various 3DGS models, including CF-3DGS, beyond our hierarchical training. Moreover, hierarchical training contributes a PSNR gain exceeding 1.52 dB, showing that the core improvement stems from hierarchical training rather than augmented data from VFI.

Effectiveness of supervision from base 3DGS models. Table 5 shows that the supervision from base 3DGS models enhances performance, with an average PSNR increase of 0.16 dB and an SSIM increase of 0.01 (Variant 8 vs. 9). Pseudo-views generated by base 3DGS models mitigate overfitting and provide additional supervision for views not covered by the training images.

Unknown camera intrinsics. We experiment with heuristics instead of known camera intrinsics by setting the FoV to 70°. As shown in Table 5, PSNR dropped from 33.53dB to 32.17dB (Variant 9 vs. 10). Inaccurate intrinsics hinder pose estimation and may introduce scale ambiguity. Nonetheless, our method, even without known intrinsics, outperforms CF-3DGS by 0.89dB (Variant 1 vs. 10).

Comparison to 3DGS and NeRF with SfM poses. Variants 11 and 12 demonstrate that incorporating SfM results in lower performance for both NeRF and 3DGS compared to our method, with reductions of 6.92 dB and 3.33 dB, respectively. This performance gap arises from COLMAP's challenges in accurately estimating poses in low-texture environments, such as those found in Tanks & Temples.

5. Conclusion

We propose a hierarchical training strategy for 3D Gaussian splatting without known camera poses or SfM preprocessing, merging segment-specific base 3DGS models for enhanced representation. We further incorporate video frame interpolation to smooth camera motion and mitigate overfitting by reusing interpolated images and base models. This approach outperforms state-of-the-art SfM-free novel view synthesis methods, enabling broader generalization across datasets without SfM preprocessing.

Limitations. Our approach requires longer training and can face challenges with large camera motion or low-quality inputs. While training time increases, rendering is faster due to fewer 3D Gaussians. In practice, training time can be reduced by lowering iterations or removing VFI, which is less necessary with small camera motion or abundant input frames. Large motion or poor inputs may cause alignment errors in 3DGS model merging.

References

[1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 2

[3] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth. *arXiv preprint arXiv:2302.12288*, 2023. 3, 4

[4] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023. 1, 2, 5, 6, 7, 8

[5] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kontschieder. Revising densification in gaussian splatting. *arXiv preprint arXiv:2404.06109*, 2024. 2

[6] Zezhou Cheng, Carlos Esteves, Varun Jampani, Abhishek Kar, Subhransu Maji, and Ameesh Makadia. Lu-nerf: Scene and pose estimation by synchronizing local unposed nerfs. *arXiv preprint arXiv:2306.05410*, 2023. 2

[7] Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Garf: Gaussian activated radiance fields for high fidelity reconstruction and pose estimation. *arXiv e-prints*, 2022. 2

[8] Yuanxing Duan, Fangyin Wei, Qiyu Dai, Yuhang He, Wenzheng Chen, and Baoquan Chen. 4d gaussian splatting: Towards efficient novel view synthesis for dynamic scenes. *arXiv preprint arXiv:2402.03307*, 2024. 2

[9] Zhiwen Fan, Wenyan Cong, Kairun Wen, Kevin Wang, Jian Zhang, Xinghao Ding, Danfei Xu, Boris Ivanovic, Marco Pavone, Georgios Pavlakos, Zhangyang Wang, and Yue Wang. Instantsplat: Unbounded sparse-view pose-free gaussian splatting in 40 seconds, 2024. 2, 3

[10] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5501–5510, 2022. 2

[11] Yang Fu, Sifei Liu, Amey Kulkarni, Jan Kautz, Alexei A. Efros, and Xiaolong Wang. Colmap-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20796–20805, 2024. 1, 2, 3, 5, 6, 7, 8

[12] Wenbo Hu, Yuling Wang, Lin Ma, Bangbang Yang, Lin Gao, Xiao Liu, and Yuewen Ma. Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19774–19783, 2023. 2

[13] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *ICCV*, 2021. 6

[14] Kaiwen Jiang, Yang Fu, Mukund Varma T, Yash Belhe, Xiaolong Wang, Hao Su, and Ravi Ramamoorthi. A construct-optimize approach to sparse view synthesis without camera pose. *SIGGRAPH*, 2024. 2, 3

[15] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (ToG)*, 42(4):1–14, 2023. 1, 2, 3, 5

[16] Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. A hierarchical 3d gaussian representation for real-time rendering of very large datasets. *ACM Transactions on Graphics (TOG)*, 43(4):1–15, 2024. 2, 4

[17] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics*, 2017. 2, 5, 6, 7, 8

[18] Lingtong Kong, Boyuan Jiang, Donghao Luo, Wenqing Chu, Xiaoming Huang, Ying Tai, Chengjie Wang, and Jie Yang. Ifrnet: Intermediate feature refine network for efficient frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1969–1978, 2022. 2, 3

[19] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21719–21728, 2024. 2

[20] Hao Li, Yuanyuan Gao, Chenming Wu, Dingwen Zhang, Yalun Dai, Chen Zhao, Haocheng Feng, Errui Ding, Jingdong Wang, and Junwei Han. Ggrt: Towards pose-free generalizable 3d gaussian splatting in real-time. *CoRR*, 2024. 2, 3

[21] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 2, 6

[22] Yang Liu, He Guan, Chuanchen Luo, Lue Fan, Junran Peng, and Zhaoxiang Zhang. Citygaussian: Real-time high-quality large-scale scene rendering with gaussians. *arXiv preprint arXiv:2404.01133*, 2024. 2

[23] Andreas Meuleman, Yu-Lun Liu, Chen Gao, Jia-Bin Huang, Changil Kim, Min H Kim, and Johannes Kopf. Progressively optimized local radiance fields for robust view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16539–16548, 2023. 2

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

[25] Wieland Morgenstern, Florian Barthel, Anna Hilsmann, and Peter Eisert. Compact 3d scene representation via self-organizing gaussian grids. *arXiv preprint arXiv:2312.13299*, 2023. 2

[26] Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. Compressed 3d gaussian splatting for accelerated

novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10349–10358, 2024. 2, 4

[27] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, 2021. 3, 4

[28] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10901–10911, 2021. 2, 5, 7, 8

[29] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1

[30] Richard Shaw, Michal Nazarczuk, Jifei Song, Arthur Moreau, Sibi Catley-Chandar, Helisa Dhamo, and Eduardo Pérez-Pellitero. Swings: Sliding windows for dynamic 3d gaussian splatting. 2

[31] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 2004. 5

[32] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF−−: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 2

[33] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, 2024. 2

[34] Yitong Xia, Hao Tang, Radu Timofte, and Luc Van Gool. Sinerf: Sinusoidal neural radiance fields for joint pose estimation and scene reconstruction. 2022. 2

[35] Zhiwen Yan, Weng Fei Low, Yu Chen, and Gim Hee Lee. Multi-scale 3d gaussian splatting for anti-aliased rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20923–20931, 2024. 2

[36] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *IROS*, 2021. 2

[37] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19447–19456, 2024. 2

[38] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[39] Zheng Zhang, Wenbo Hu, Yixing Lao, Tong He, and Hengshuang Zhao. Pixel-gs: Density control with pixel-aware gradient for 3d gaussian splatting. *arXiv preprint arXiv:2403.15530*, 2024. 2