Tile-wise vs. Image-wise: Random-Tile Loss and Training Paradigm for Gaussian Splatting

Xiaoyu Zhang[1*] Weihong Pan[2*] Xiaojun Xiang[1] Hongjia Zhai[2] Liyang Zhou[1]
Hanqing Jiang[1†] Guofeng Zhang[2†]
[1]SenseTime Research [2]State Key Lab of CAD&CG, Zhejiang University

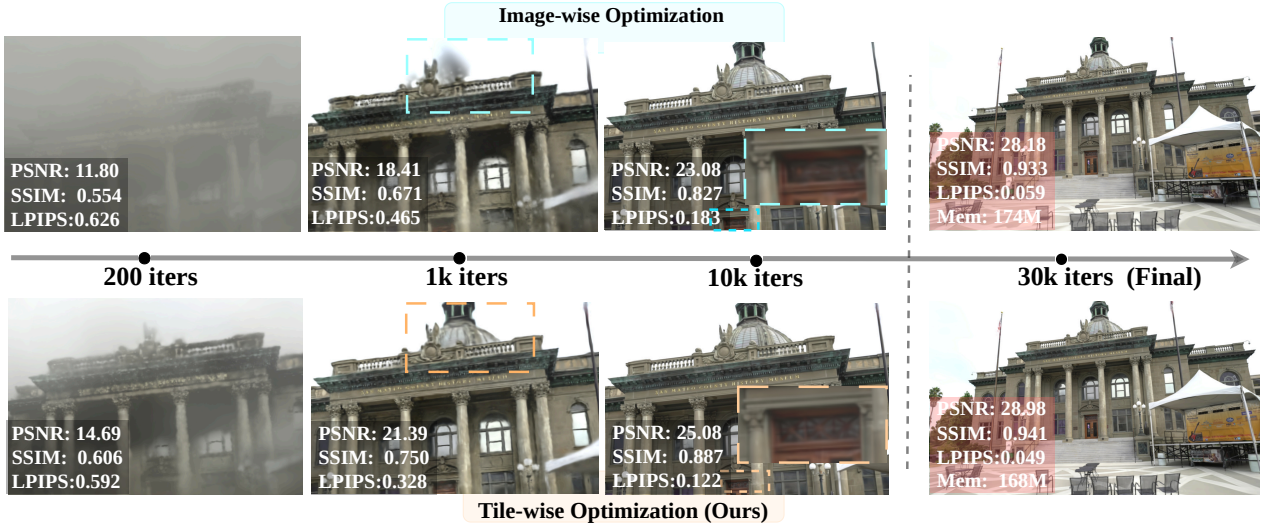

Figure 1. **Random-Tile Loss** integrates multi-view appearance and structural constraints during each optimization step. Under the same batch size, it consistently improves convergence efficiency and rendering quality while preserving compactness and real-time capability. The results on test views across different training iterations suggest a more stable and efficient convergence compared to the image-wise loss in Gaussian Splatting. Please refer to the **Supplementary Video** for more detailed comparisons on convergence performance.

Abstract

3D Gaussian Splatting (3DGS) has drawn significant attention for its advantages in rendering speed and quality. Most existing methods still rely on the image-wise loss and training paradigm because of its intuitive nature in the Splatting algorithm. However, image-wise loss lacks multi-view constraints, which are generally essential for optimizing 3D appearance and geometry. To address this, we propose RT-Loss along with a tile-based training paradigm, which uses randomly sampled tiles to integrate multi-view appearance and structural constraints in 3DGS. Additionally, we introduce an tile-based adaptive densification control strategy tailored for our training paradigm. Extensive experiments show that our approach consistently improves performance metrics while maintaining efficiency across various benchmark datasets.

1. Introduction

Neural Radiance Fields (NeRF) [33] has brought revolutionary advances to photorealistic novel view synthesis (NVS) through exceptional rendering quality, attracting significant attention from the computer vision, graphics, and robotics communities [44]. Recently, 3D Gaussian Splatting (3DGS) [19] has significantly enhanced rendering efficiency by replacing traditional ray marching with the splatting algorithm [22]. Numerous studies have focused on optimizing the Gaussian feature or heuristic split algorithm in 3DGS to improve the appearance quality and geometric accuracy [6, 24, 31, 51] .

However, while the splatting algorithm significantly improves rendering speed, its potential for incorporating Tile-based multi-view constraints remains underexplored. Specifically, 3DGS initializes with point clouds derived from Structure from Motion (SfM) [1, 12, 36] and optimizes a set of anisotropic 3D Gaussians to represent the scene. Tile-based splatting plays a crucial role in achieving real-time rendering by directly splatting visible Gaussians onto

*Authors contributed equally.
†Corresponding authors.

the camera view, rather than querying every ray point. As a result, defining the loss function on each training image $\mathcal{I}$ is a intuitive and straightforward choice:

$$\mathcal{L}(\Theta) = (1-\lambda)L_{\text{MAE}}(\Theta, \mathcal{I}) + \lambda L_{\text{D-SSIM}}(\Theta, \mathcal{I}), \quad (1)$$

where L_{MAE} and L_{D-SSIM} are computed image-wise on the entire training image. Although this approach benefits the calculation of structural loss SSIM across the dense image, it also means that gradients from multiple viewpoints cannot be integrated in a single optimization step. In contrast, NeRF typically optimizes a pixel-wise loss:

$$\mathcal{L}(\Theta) = \frac{1}{\|\mathcal{R}\|} \sum_{\boldsymbol{r} \in \mathcal{R}} L_{\text{MSE}}(\Theta, \boldsymbol{r}), \quad (2)$$

where each individual ray $r = (\boldsymbol{x}, \boldsymbol{d}, \boldsymbol{c})$ in the mini-batch $\mathcal{R}$ is randomly sampled from the entire training dataset. Point-wise sampling from random viewpoints provides more comprehensive gradients. However, it fails to utilize the collective supervision of distant pixels, thus inevitably overlooking structural information in the overall scene [42].

To address this issue, we propose a Tile-based training paradigm that integrates both multi-view constraints and structural information within the splatting process. We sample a set of tiles $\mathcal{T}$ from multiple training views, each individual tile t is the minimal unit in splatting . The loss is formulated as follows:

$$\mathcal{L}_{RT}(\Theta) = \frac{1}{\|\mathcal{T}\|} \sum_{\boldsymbol{t} \in \mathcal{T}} L_{\text{MAE}}(\Theta, \boldsymbol{t}) + \lambda L_{\text{RT-SSIM}}(\Theta, \mathcal{T}), \quad (3)$$

which is computed over a random-tile batch $\mathcal{T}$ that contains muti-view appearance and structure. Our Tile-based training paradigm maintains the training efficiency of the 3DGS rendering process, with detailed definitions and calculations provided in Sec. 3.2. Overall, our contributions are as follows:

- We propose a Tile-based training paradigm and loss formulation, incorporating both multi-view appearance constraints and structural similarity constraints with random tile sampling. To the best of our knowledge, we are the first to introduce a random-tile approach as a replacement for image-wise training paradigms in 3DGS.
- Extensive experiments indicate that this Tile-based training paradigm yields notable improvements in training performance while maintaining model compactness.

2. Related Work

Novel View Synthesis. Neural Radiance Field (NeRF) [33] has significantly advanced novel view synthesis by employing a multilayer perceptron (MLP) to model scenes with volumetric rendering, enabling the generation of realistic images from different perspectives. NeRF introduced a transformative approach to scene representation, sparking extensive research focused on enhancing rendering quality [2, 3, 27, 37, 50, 54, 55], speeding up computation [5, 14, 17, 34, 35], enabling 3D content generation [38], handling dynamic scenes [9, 15, 25], and improving generalization to novel scenes [45]. Due to the ray marching rendering technique, most NeRF-based methods are trained by utilizing batches of randomly sampled rays, with mean squared error (MSE) as the primary loss function. To integrate structural similarity, S3IM [42] enhances NeRF by incorporating patches of random rays in the loss calculation with a structural similarity index (SSIM), improving performance in tasks such as sparse view synthesis.

Gaussian Splatting Training Paradigm. 3D Gaussian Splatting (3DGS) has gained significant attention in novel view synthesis tasks due to its impressive performance. Using a rasterization-based rendering approach [20], 3DGS avoids the computational overhead of ray marching. With its high rendering speed and quality, 3DGS has found applications across various fields [48], including dynamic scene reconstruction [7, 10, 28, 29, 32, 41, 47], sparse view synthesis [43, 52], reflective object modeling [53], and geometric surface recovery [24, 30, 40]. Additional optimizations target specific issues, such as the geometric improvements seen in Scaffold-GS [31], which combines neural field models with Gaussian Splatting to achieve state-of-the-art surface accuracy. [18] introduces adaptive strategies to reduce dependency on SfM initialization. While most approaches adopt an image-wise training paradigm, some recent research explores multi-view constraints to enhance Gaussian Splatting. For example, [46] leverages multi-view stereo (MVS) to enhance initialization geometry, using multi-view consistency to optimize Gaussian geometry. [8, 10, 26] utilize batch accumulation to alleviate data pressure from the temporal dimension by rendering multiple images per iteration and accumulating gradients. However, this straightforward image-wise accumulation not only impacts performance but also fails to fully leverage the benefits of multi-view constraints with the same batch size.

3. Methods

3.1. Preliminaries

Gaussian Splatting. 3DGS [19] represents scenes using anisotropic 3D Gaussians, which retains the differential properties of volumetric representations while enabling efficient rendering through Tile-based rasterization. Each Gaussian is assigned a color c, represented by spherical harmonics, and an opacity α, which modulates $G(x)$ during

blending. Its spatial distribution is then defined by:

$$G(x) = \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right), \quad (4)$$

where x denotes a spatial location in the 3D scene and μ, initialized from Structure-from-Motion (SfM) points, specifies the Gaussian center. The covariance matrix Σ is constructed using a scaling matrix S and rotation matrix R as $\Sigma = RSS^TR^T$ to ensure positive semi-definiteness.

Tile-based Rendering. Tile-based rendering is the key to achieving high-resolution and real-time performance in 3DGS. Unlike conventional volumetric methods, it projects 3D Gaussians into 2D Gaussians $G'(x)$ on the image plane rather than relying on resource-intensive ray marching. The image is divided into tiles of size 16×16 to process all sorted 2D Gaussians in parallel. Each pixel within a tile applies α-blending:

$$C(x') = \sum_{i\in\mathcal{N}} c_i\sigma_i \prod_{j=1}^{i-1}(1-\sigma_j), \quad \sigma_i = \alpha_i G'_i(x'). \quad (5)$$

Here, x represents the queried pixel position, and $\mathcal{N}$ denotes the number of sorted 2D Gaussians linked to that tile. By leveraging the Tile-based differentiable rasterizer, all 3D Gaussians projected onto the training view can be directly optimized in an end-to-end manner.

MAE, MSE, and SSIM. These three loss functions are widely used for image quality assessment. The Mean Absolute Error (MAE) and Mean Squared Error (MSE) are two methods for calculating RGB loss:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i-\hat{y}_i)^2, \quad (6)$$

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy}+C_2)}{(\mu_x^2+\mu_y^2+C_1)(\sigma_x^2+\sigma_y^2+C_2)}, \quad (7)$$

where μ_x and μ_y are the means of the images x and y, σ_x^2 and σ_y^2 are the variances, σ_{xy} is the covariance between the images, and C_1, C_2 are constants to stabilize the division when the denominator is small. MAE measures the average absolute error, treating all deviations equally. MSE squares each error before averaging, making it more sensitive to large errors and outliers. The Structural Similarity (SSIM) focuses on perceptual quality by comparing structural patterns, luminance, and contrast between images.

3.2. Tile-based Training via RT-Loss

In this section, we introduce RT-Loss and our Tile-based training paradigm. Our goal is to incorporate random viewpoint observations within each training batch, while preserving structural information to enforce more comprehensive constraints. In contrast to the individual rays used in NeRFs, Gaussian Splatting utilizes a Tile-based splatting process that projects Gaussians onto the entire view screen, rather than performing active ray marching. Hence, the smallest unit in the loss calculation must align with the splatting rendering and is set as a 16×16 tile $\boldsymbol{t}$. We load the datasets as tiles instead of images, randomly sample N tiles as a training batch $\mathcal{T}$ and obtain the rendered tiles set $\mathcal{T}(\hat{\mathcal{C}})$ along with their corresponding ground truth $\mathcal{T}(\mathcal{C})$, where $\hat{\mathcal{C}} = \{\hat{\boldsymbol{C}}(\boldsymbol{t}) \mid \boldsymbol{t} \in \mathcal{T}\}$ and $\mathcal{C} = \{\boldsymbol{C}(\boldsymbol{t}) \mid \boldsymbol{t} \in \mathcal{T}\}$. We use MAE as the color loss, which can be written as:

$$L_{\text{RT}-\text{MAE}} = \frac{1}{\|\mathcal{T}\|}\sum_{\boldsymbol{t}\in\mathcal{T}}\|\hat{\boldsymbol{C}}(\boldsymbol{t}) - \boldsymbol{C}(\boldsymbol{t})\|_1. \quad (8)$$

The tiles contain viewpoint's structural information. If we solely rely on rgb loss L_{MAE} , the positional relationship contained within the tiles would be completely lost [42]. Inspired by S3IM's adaptation of SSIM on random rays, we propose RT-SSIM for 3DGS to retain the structural information embedded in random tiles. The concept is straightforward: while keeping the total batch size unchanged, we compute and average the SSIM across multiple independent and non-overlapping random tiles from different views, with kernel size and stride set to $K \times K$ (e.g., 9×9 for overall structrue or $3 \times 3$for fine detail) with a stride of s. RT-SSIM can be written as:

$$\text{RT-SSIM}(\hat{\mathcal{T}},\mathcal{T}) = \frac{1}{N}\sum_{\boldsymbol{t}\in\mathcal{T},v\in\boldsymbol{V}}\text{SSIM}\left(\boldsymbol{t}^{(v)}(\hat{\mathcal{C}}),\boldsymbol{t}^{(v)}(\mathcal{C})\right), \quad (9)$$

where V controls the number of views in all N tiles. In particular, when $V = N$, it implies all tiles are selected from different views. As RT-SSIM positively correlated with image similarity, we define the structure loss $L_{\text{RT-SSIM}}$ as

$$L_{\text{RT}-\text{SSIM}}(\Theta,\mathcal{T}) = 1 - \text{RT-SSIM}(\hat{\mathcal{T}},\mathcal{T}) \quad (10)$$

We note that tiles are particularly suitable for parallel computation, thus the additional computational cost of Tile-based training is minimal. The final hybrid loss $\mathcal{L}_{\text{RT}}$ can be written as:

$$\mathcal{L}_{\text{RT}}(\Theta,\mathcal{T}) = (1-\lambda_s)L_{\text{RT}-\text{MAE}} + \lambda_s L_{\text{RT}-\text{SSIM}} \quad (11)$$

where λ_s balances structure and color losses.

We present the pseudocode in Algorithm 1. In the standard 3DGS training paradigm, tiles are rendered in parallel using the same view parameters. In contrast, we sample tiles from multiple different views to integrate multi-view constraints. In our random-tile training paradigm, We introduce two additional hyperparameters, λ_s and V, where λ_s controls the loss weight and V determines the number of viewpoints. When $V = N$, each tile is sampled from a

Algorithm 1: Tile-based Training via RT-loss

Let $\mathcal{A}$ be an SGD training algorithm;
while *the termination criteria has not been satisfied* **do**
  Sample a training batch $\mathcal{T}$ consisting of N tiles from V views in dataset $\mathcal{D}$;
  for $i = 1$ **to** V **do**
    Randomly select tiles $\mathcal{T}_i = [t_1, t_2 ... t_{V/N}]$, where $t_j \in \mathcal{T}_i \subseteq \mathcal{T}$;
    VisibilityCheck();
  end
  for *each tile* $\boldsymbol{t}$ *in* $\mathcal{T}$ **do**
    Compute the forward splatting result after $\hat{\mathcal{C}} = \{\hat{\boldsymbol{C}}(\boldsymbol{t}) \mid \boldsymbol{t} \in \mathcal{T}\}$;
    Obtain the ground-truth $\mathcal{C} = \{\boldsymbol{C}(\boldsymbol{t}) \mid \boldsymbol{t} \in \mathcal{T}\}$;
  end
  for $i = 1$ **to** V **do**
    let rendered pixels $\hat{\mathcal{C}}$ form the pixel patch $\boldsymbol{t}^{(v)}(\hat{\mathcal{C}})$ in view V_i;
    Compute SSIM $\left(\boldsymbol{t}^{(v)}(\hat{\mathcal{C}}), \boldsymbol{t}^{(v)}(\mathcal{C})\right)$ with the given kernel $K \times K$;
  end
  Obtain the RT-SSIM loss $L_{\text{RT}-\text{SSIM}}(\Theta, \mathcal{T}) = 1 - \frac{1}{N}\sum \text{SSIM}\left(\boldsymbol{t}^{(v)}(\hat{\mathcal{C}}), \boldsymbol{t}^{(v)}(\mathcal{C})\right)$;
  Obtain the conventional L_{MAE} loss as color loss $L_{\text{RT-MAE}}(\Theta) = \frac{1}{\|\mathcal{T}\|}\sum_{\boldsymbol{t}\in\mathcal{T}} \|\hat{\boldsymbol{C}}(\boldsymbol{t}) - \boldsymbol{C}(\boldsymbol{t})\|_1$;
  $\mathcal{L}(\Theta, \mathcal{T}) = (1 - \lambda_s) L_{\text{RT}-\text{MAE}}(\Theta, \mathcal{T}) + \lambda_s L_{\text{RT}-\text{SSIM}}(\Theta, \mathcal{T})$;
  Compute the gradient $\nabla\mathcal{L}(\Theta)$;
  Update model parameters Θ by $\mathcal{A}$
end

distinct view, and their visibility is computed in parallel during preprocessing to accelerate the rendering process. For a fair comparison, we set the total tiles count N to match that in image-wise training, ensuring a consistent batch size in gradient descent algorithm $\mathcal{A}$. Our random-tile strategy uniformly samples tiles across each view, balancing observations for each perspective. Our ablation study in Sec. 4.4 shows that this strategy achieves slightly better results than purely random selection.

3.3. Tile-based Adaptive Density Control

In 3D Gaussian Splatting, a Gaussian is split or cloned depending on whether the average cumulative 2D gradient on the NDC coordinates $\sum \|\mathbf{g}_i\|$ across interval iterations exceeds a specified threshold τ_{densify}, as shown in Fig. 2. During our Tile-based training, Gaussians in one iteration will

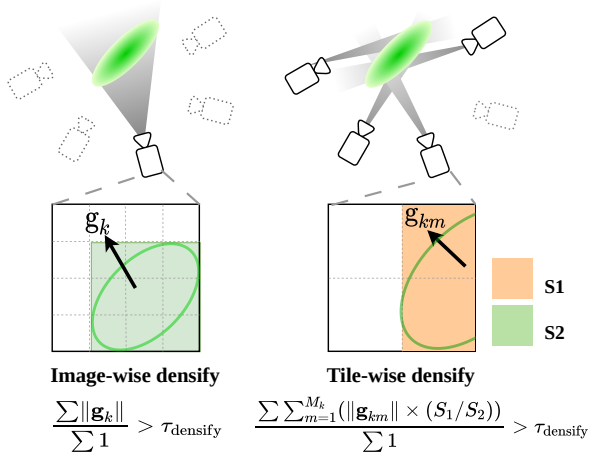

Figure 2. **Image-wise vs Tile-wise Adaptive Density Control.** In Tile-based training, NDC gradients for densification are computed as a weighted sum across all observations, balancing multi-view contributions, where $S1$ represents the area of the Gaussian projection onto the tile, and $S2$ is the total projected area.

have multiple NDC gradients from different training views. Thus, there are two methods for computing density gradients: Tile-Count Densification and Iteration-Count Densification.

$$\frac{\sum_{k=1}^{\text{Iter}}\sum_{m=1}^{M_k}\|\mathbf{g}_{km}\|}{\sum_{k=1}^{\text{Iter}} M_k} > \tau_{\text{densify}} \tag{12}$$

$$\frac{\sum_{k=1}^{\text{Iter}}\sum_{m=1}^{M_k}(\|\mathbf{g}_{km}\| \times r_m)}{\sum_{k=1}^{\text{Iter}} 1} > \tau_{\text{densify}}, \tag{13}$$

where *Iter* represents the number of iterations in which a Gaussian participates in α-blending within each densification interval. In Tile-count Densification (Eq. (12)), the NDC gradient is averaged over the number of observed tiles M_k, while in Iteration-Count Densification (Eq. (13)), it is averaged over the number of observed iterations. Additionally, in Iteration-Count Densification, the NDC gradient is weighted by r_m, the ratio of the projected area shown in Fig. 2. To maintain consistency with the split parameters used in the original 3DGS and to emphasize the effect of RT-Loss, we default to using Iteration-Count for adaptive density control. Further details on NDC gradients and densification algorithm can be found in the Supplementary Material.

4. Experiments

4.1. Experimental Setup

Dataset and Metrics. We conduct a comprehensive evaluation on 26 publicly available scenes widely used

Table 1. **Quantitative comparison to previous methods on benchmark datasets.** We report results on three commonly used datasets.The best , second best , and third best results are denoted by red, orange, and yellow, respectively.

Dataset	Mip-NeRF360			Tanks&Temples			Deep Blending		
Method | Metrics	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Plenoxels [13]	23.62	0.670	0.443	22.13	0.762	0.334	23.06	0.795	0.510
Instant-NGP [34]	26.43	0.725	0.339	22.94	0.752	0.304	23.62	0.797	0.423
Mip-NeRF360 [3]	29.23	0.844	0.207	23.12	0.771	0.250	29.40	0.901	0.245
3D-GS [19]	28.69	0.870	0.182	24.36	0.831	0.213	29.41	0.903	0.243
Scaffold-GS [31]	28.84	0.848	0.220	24.81	0.840	0.205	30.21	0.906	0.254
Ours	29.66	0.892	0.171	25.13	0.852	0.192	30.15	0.911	0.224
Ours (+ Scaffold-GS)	29.86	0.885	0.206	25.32	0.857	0.189	30.49	0.913	0.231

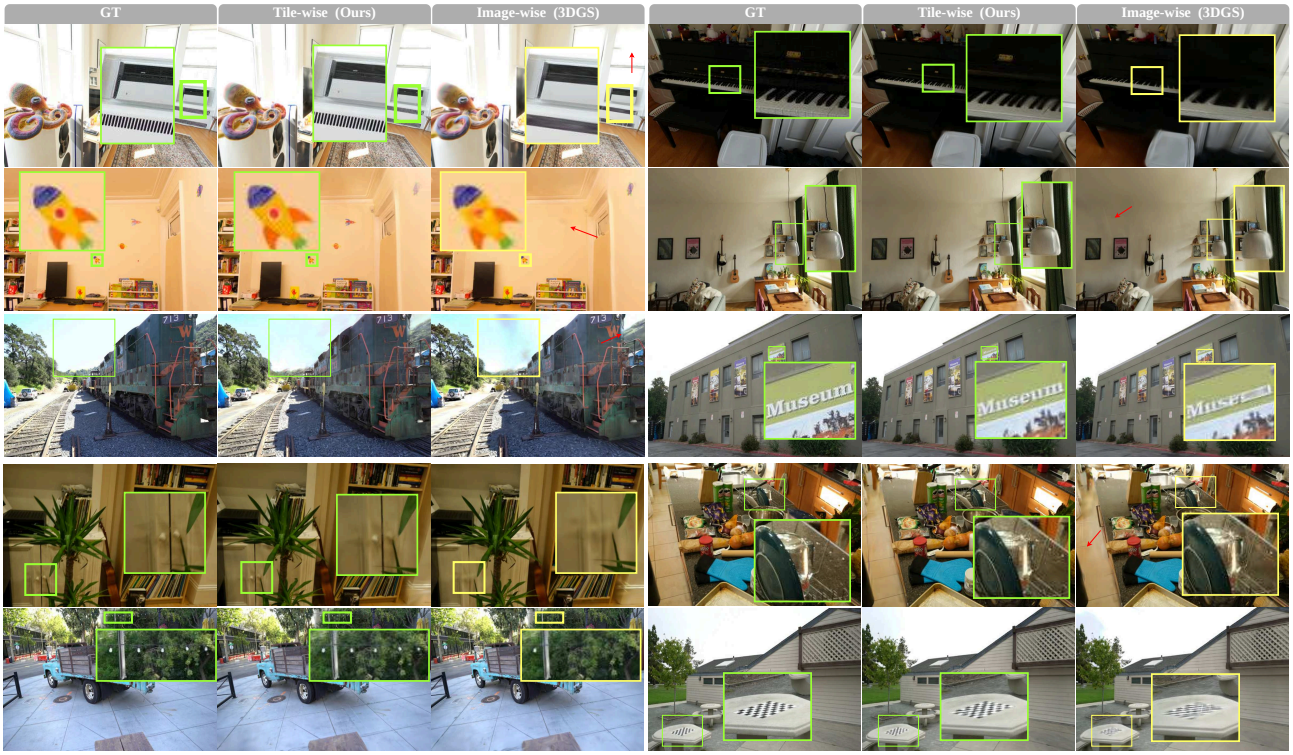

Figure 3. **Qualitative comparison across benchmark datasets [3, 11, 21].** We highlight the differences in the image using patches for clearer visibility. Our approach consistently outperforms 3D-GS on these scenes, demonstrating clear advantages in challenging scenarios, such as fine-scale details (MIP360-ROOM(a), TANDT-BARN), thin geometry (SMERF-NYC(a), SMERF-NYC(b)), texture-less regions (TANDT-TRAIN), light effects (MIP360-COUNTER, TANDT-TRUCK), and insufficient observations (TANDT-BARN, SMERF-NYC(b)).

in the field of 3D reconstruction across multiple diverse tasks. Specifically, we evaluate our method on seven scenes from Mip-NeRF360 [3], another seven from Tanks&Temples [21], and two additional scenes from Deep-Blending [16]. To further demonstrate the limitations of image-wise loss in capturing global structure, our evaluation incorporated all four scenes from [4], with 5–10× more training views. We additionally evaluated dynamic benchmarks, specifically using the Neural 3D Video Dataset containing six scenes, to demonstrate the effectiveness of RT-Loss in 4D reconstruction. In addition to the commonly used metrics (PSNR, SSIM [39], and LPIPS [49]), we further provide the storage size (MB) and rendering speed (FPS) to assess model compactness and efficiency, with all experiments performed on an NVIDIA RTX 4090 GPU. The main paper reports the average metrics across all scenes in each dataset, while detailed per-scene quantitative results are presented in the supplementary materials.

Baseline and Implementation. 3DGS[19], Scaffold-GS[31] and 4D-GS [10] are selected as our baselines due to their state-of-the-art performance. We also present the results of Mip-NeRF360 [3], iNGP [34] and Plenoxels [13] as in [19]. To demonstrate the generality of the tile-based ap-

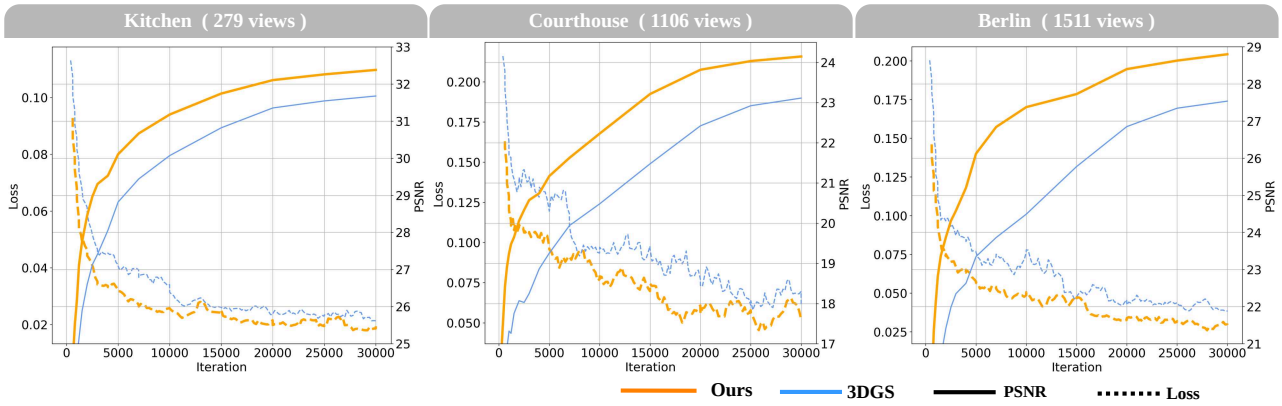

Figure 4. **Convergence performance.** We evaluated the convergence of PSNR for both the training and testing views during the training process. With the same batch size, the convergence of image-wise loss in 3DGS is slower than RT-Loss, with the gap widening on larger datasets, *e.g.*, COURTHOUSE and BERLIN

proach, we implemented the tile-wise method in Scaffold-GS for comparison. We trained for $30k$ iterations and rendered the same number of pixels per iteration, ensuring the total pixels from random tiles matched a complete image. To independently demonstrate the effectiveness of the tile-based training paradigm and RT-loss, we modified only the sampling and loss calculations for random tile training while keeping the default 3DGS parameters unchanged, including the densification threshold $\tau_{\text{densify}} = 0.0002$ and all learning rates. For RT-loss, we sample random tiles from $V = 5$ views and apply the same structure loss weight, $\lambda_s = 0.2$, as used in the original 3DGS.

4.2. Results Analysis

Benchmark Evaluation. The quantitative results on Mip-NeRF360 and Tanks&Temples are presented in Tab. 1. Results show that we outperform existing methods on standard datasets. Fig. 3 illustrates that tile-wise training achieves enhanced visual quality, with improved geometry accuracy and richer texture details. Moreover, our method retains the efficiency of Gaussian methods, as shown in Tab. 2. We compared the rendering speed and storage size with 3DGS. We achieve improved performance while using a comparable or fewer number of Gaussians, demonstrating high-quality reconstruction without compromising speed. By incorporating our tile-based approach, Scaffold-GS also achieved notable improvements, suggesting that the image-wise training paradigm, which lacks multi-view constraints, limits the reconstruction quality of various GS methods.

Convergence Performance. We evaluated our convergence performance across several representative scenarios. As shown in Fig. 4, compared to the original image-wise loss in 3DGS, the RT-Loss based on multi-view observations provides more effective constraints, resulting in significantly faster convergence. The difference in convergence speed is visually evident in Fig. 1. This advantage in convergence is even more pronounced in scenes with a large number of training views, *e.g.*, COURTHOUSE and BERLIN.

Scene with More Training Views. We compared our method with the original 3DGS in scenarios with more

Table 2. **Rendering FPS and storage comparison.** The rendering speed of both methods is measured on the same machine.

Scenes	Mip-NeRF360		ROOM			KITCHEN		
	Mem	FPS	PSNR	Mem	FPS	PSNR	Mem	FPS
3DGS	671	78	31.79	**317**	123	31.63	372	118
Ours	**625**	**93**	**32.81**	318	**125**	**32.35**	**320**	**135**
Scenes	Tanks&Temples		COURTHOUSE			MEETINGROOM		
	Mem	FPS	PSNR	Mem	FPS	PSNR	Mem	FPS
3DGS	317	126	23.13	174	**146**	26.27	276	120
Ours	**302**	**130**	**24.14**	**168**	139	**26.98**	**260**	**133**

Table 3. **PSNR evaluation for scenes with increased training views.** The convergence capability of RT-Loss is particularly evident on the SMERF dataset[11], which includes approximately 10 times more training views than Mipnerf360. Additionally, we recorded the PSNR gap at 4k and 30k iterations.

Scene(views)	Alameda(1734)		Berlin(1511)		London(1874)	
test iteration	4k	30k	4k	30k	4k	30k
3DGS	17.62	22.27	22.63	27.53	22.32	25.75
Ours	**20.11**	**23.916**	**24.80**	**28.79**	**24.07**	**26.626**

Table 4. **Quantitative comparison on dynamic datasets.** We applied RT loss for dynamic scene reconstruction and compared it with **4D-GS**[10] on the challenging real-world dynamic dataset Neu3D[23].

Model	PSNR↑	D-SSIM↓	LPIPS↓	FPS↑	Storage (MB)↓
4D-GS[10]	31.15	0.974	0.049	30	90
Ours	**31.74**	**0.973**	**0.043**	**34**	**78**

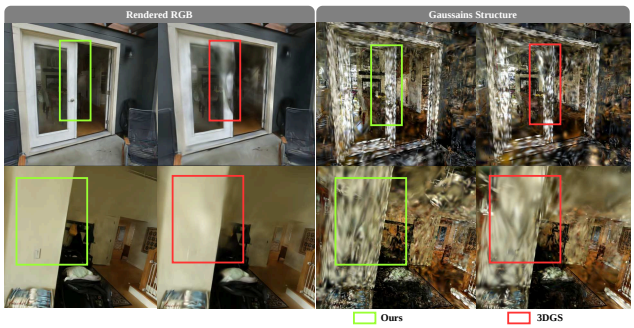

Figure 5. **Gaussians structure comparison.** RT-Loss demonstrates stronger structural reconstruction ability in scenes with more training views. In contrast, 3DGS not only loses details but also fails to recover the geometric structure of the scene.

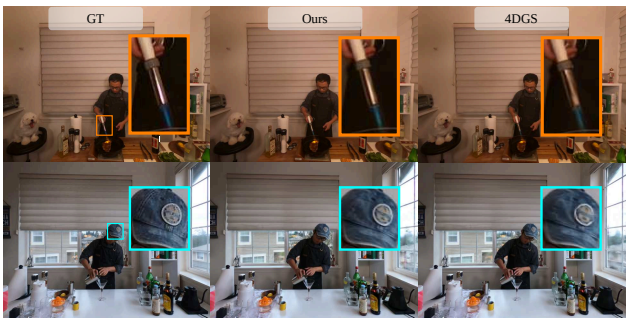

Figure 6. **Qualitative comparisons on the Neural 3D Video Dataset.** In 4D reconstruction, constraints from multiple viewpoints and time also help generate more accurate details.

training views. In Tab. 3, our method achieved a significant improvement in metrics. The gap is particularly noticeable in the early training stages (4k iterations), as image-wise training naturally requires more iterations to fully capture the scene. Fig. 5 further illustrates that our method produces a more refined geometric structure on large datasets. This supports our hypothesis about the current training paradigm: as the number of training views increases, image-wise training struggles to comprehensively capture scene information, resulting in less reliable optimization and densification. Furthermore, through experiments with extended training durations, we demonstrate that naively increasing the iteration count for image-wise training does not effectively address this limitation. This is because once incorrect geometry is formed during Gaussian optimization, it is difficult to correct by simply extending the training process. For further details, please refer to the supplementary material.

Dynamic Scenes. We evaluated our method on the real-world dynamic scene dataset, the Neural 3D Video Dataset, using 4D-GS [10] as the baseline. The expansion of the time dimension in the dynamic dataset resulted in a larger

Table 5. **Ablation study on RT-Loss components.** 'Img-' refers to image-wise loss, and 'RT-' refers to random-tile loss.

Scene(views)	KITCHEN(279)		CORTHOUSE(1106)	
	PSNR	Mem(Mb)	PSNR	Mem(Mb)
$L_{\text{Img-MAE}} + L_{\text{Img-SSIM}}$ (3DGS)	31.67	372	23.11	173
$L_{\text{Img-MAE}}$	31.09	131	21.35	**44**
$L_{\text{RT-MAE}}$	31.65	**127**	22.63	63
$L_{\text{RT-MAE}} + L_{\text{Img-SSIM}}$	32.02	352	23.50	186
$L_{\text{RT-MAE}} + L_{\text{RT-SSIM}}$ (Ours)	**32.35**	320	**24.15**	168

Table 6. **Effect of Adaptive Density Control.** 'Tile-Count' and 'Iteration-Count' are densification strategies described in Sec. 3.3, with 'Iteration-Count' being the default.

Scene(views)	KICHEN(279)		COURTHOUSE(1106)	
	PSNR	Mem(Mb)	PSNR	Mem(Mb)
3DGS w/o Densification	29.84	58	22.96	118
Ours w/o Densification	30.32	58	23.83	118
Ours w/ Tile-Count	31.95	175	23.56	125
Ours w/ Iteration-Count	32.35	320	24.15	168

Table 7. **Effect of Tile Sampling and Batch Training.**. 'V' refers to the number of randomly selected viewpoints, determining the spread of observations. 'B' refers to Batch train, which means using multiple images as a batch for training. 'Non-Uniform' and 'Uniform' indicate whether the tiles are distributed evenly.

Scene(views)	KICHEN(279)		COURTHOUSE(1106)	
	PSNR	Mem(Mb)	PSNR	Mem(Mb)
tile-wise $V = 2$	31.80	343	23.76	182
tile-wise $V = 4$	32.22	337	23.92	175
tile-wise $V = 6$	32.40	332	24.29	161
image-wise $B = 2$	31.67	352	23.82	214
image-wise $B = 4$	31.22	348	23.69	207
Non-Uniform Sample	32.30	310	24.02	192
Uniform Sample	32.38	320	24.14	168

volume of training data. Although most 4D Gaussian methods address this issue by using batch training and rendering multiple images together as a training batch, our results show that, with the same batch size, tile-based training provides more effective constraints across multiple viewpoints and time frames. The results show that RT-Loss is also effective for dynamic scene reconstruction. As depicted in Tab. 4, we notably improved the rendering performance on dynamic scenes without sacrificing efficiency. As shown in Fig. 6, our method reconstructs finer details, attributed to our randomized approach, which provides stronger constraints across both viewpoints and time frames compared to image-wise training. Details are provided in the supplementary material.

4.3. Ablation Studies

Effect of RT-Loss components. We conducted an ablation on the loss components of Eq. (11), which includes the base color loss $L_{\text{RT-MAE}}$ and the structural loss $L_{\text{RT-SSIM}}$. As shown in the 2^{nd} and 3^{rd} row in Tab. 5, when only MAE loss is used, the PSNR significantly decreases, with a lower

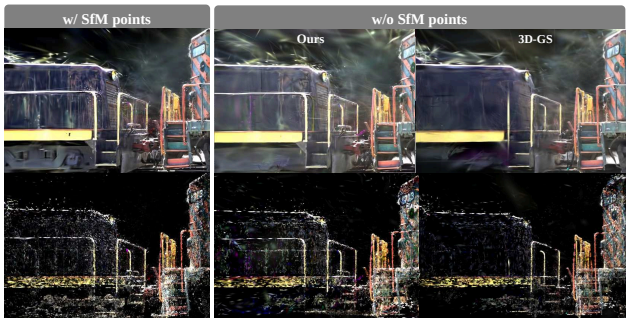

Figure 7. **Effect of initialization on reconstruction quality.** (Left) Gaussians optimized from SfM point initialization. (Right) Gaussians optimized from random points. Our method reconstructs better structures and finer details even with random initialization, demonstrating superior robustness to initialization issues.

Table 8. **Sensitivity to Initialization.** In the 'w/o SfM initialization' setting, Gaussians are initialized with random positions. RT-Loss demonstrates stronger robustness in addressing initialization issues.

Scene(views)	KICHEN(279)		COURTHOUSE(1106)	
	PSNR	Mem(Mb)	PSNR	Mem(Mb)
3DGS w/ SfM initialization	31.63	372	23.13	174
3DGS w/o SfM initialization	29.89	269	21.77	121
Ours w/ SfM initialization	32.35	320	24.14	168
Ours w/o SfM initialization	31.22	272	23.21	186

tendency for Gaussian densification. However, our $L_{\text{RT-MAE}}$ still outperforms the image-wise constraints, and this effect becomes more pronounced in the *CourtHouse* scene with more training views, where the PSNR difference reaches up to 1.3 dB. Notably, in the *Kitchen* scene, the $L_{\text{RT-MAE}}$ constraint achieves PSNR on par with 3DGS, despite using only one-third of the Gaussian points, as shown in the 1^{st} and 3^{rd} row in Tab. 5. Additionally, incorporating SSIM improves the PSNR as the number of Gaussians increases. Furthermore, to demonstrate the improvement of structure loss brought by RT-SSIM, we rendered the complete image and incorporated image-wise SSIM as a structural loss during random tile training. In contrast, the metrics using RT-SSIM consistently yield higher values, showing that the multi-view observations in RT-SSIM provide more accurate reconstruction constraints than image-wise SSIM.

Effect of Adaptive Density Control. We ablate our densification strategy described in Sec. 3.3 on Tab. 6.Without any densification, the optimization of GS relies solely on the loss function without any splitting or cloning. Our improvements demonstrate the effectiveness of multi-view constraints in enhancing performance. Tile-Count densification, which averages the NDC gradient across tiles, causes the Gaussians to resist densifying and lower PSNR. This is because random-tile training increases the frequency with which a Gaussian is observed, leading to a smaller average gradient.

Effect of Tile Sampling and Batch Training. We analyzed the impact of viewpoint count V and tile uniformity described in Sec. 3.2 on Tab. 7. Uniform sampling better distributes viewpoint constraints in the loss, yielding slight improvements and serving as our default. In data-rich scenes like COURTHOUSE, more viewpoints provide greater benefits, as larger scenes require sufficient and evenly distributed observations. This highlights the adaptability of our random-tile training paradigm to different scene requirements. We also tested the comparative effect of image-wise batch training. Under a similar total training volume (iterations = 15k), introducing multiple images as a batch to incorporate multi-view observations does not improve quality as effectively as the tile-wise approach. Moreover, our observations indicate that such dense inclusion of multiple images significantly increases computational overhead.

Sensitivity to Initialization. The initialization problem has long been considered a critical factor impacting 3DGS performance [18]. We conduct experiments under two initialization conditions. Fig. 7 shows that our method recovers improved structure and appearance even with random initialization. As shown in Tab. 8, our method demonstrates greater robustness to initialization variations. This suggests that part of the initialization issue may arise from uneven and insufficient training constraints, and RT-Loss has the potential to address this limitation.

4.4. Discussions and Limitations

We propose a practical and effective approach by incorporating tile-wise constraints into Gaussian optimization to replace image-wise constraints. We have demonstrated its clear effectiveness and universality through both the main experiments and ablation studies. However, in experiments with large texture-less regions, our method fails to improve the geometry. To further enhance the tile-wise optimization, finer strategies could be explored in practice, such as further allocating multi-view constraints to identical content, in order to provide more accurate and uniform multi-view constraints for improvement.

5. Conclusion

In this work, we introduce a Tile-based training paradigm that integrates multi-view appearance and structural constraints. Extensive experiments demonstrate its effectiveness across various scenarios. We expect tile-wise training to hold strong potential as a promising paradigm for future research and applications in Gaussian Splatting.

Acknowledgments

This work was partially supported by Key R&D Program of Zhejiang Province (No. 2023C01039).

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54 (10):105–112, 2011. 1

[2] 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, 2021. 2

[3] 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, 5, 3

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19697–19705, 2023. 5

[5] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16569–16578, 2023. 2

[6] Kai Cheng, Xiaoxiao Long, Kaizhi Yang, Yao Yao, Wei Yin, Yuexin Ma, Wenping Wang, and Xuejin Chen. Gaussianpro: 3d gaussian splatting with progressive propagation, 2024. 1

[7] Jaeyoung Chung, Suyoung Lee, Hyeongjin Nam, Jaerin Lee, and Kyoung Mu Lee. Luciddreamer: Domain-free generation of 3d gaussian splatting scenes. *arXiv preprint arXiv:2311.13384*, 2023. 2

[8] Xiaobiao Du, Yida Wang, and Xin Yu. Mvgs: Multi-view-regulated gaussian splatting for novel view synthesis, 2024. 2

[9] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B. Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing, 2021. 2

[10] 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, 5, 6, 7

[11] Daniel Duckworth, Peter Hedman, Christian Reiser, Peter Zhizhin, Jean-François Thibert, Mario Lučić, Richard Szeliski, and Jonathan T Barron. Smerf: Streamable memory efficient radiance fields for real-time large-scene exploration. *ACM Transactions on Graphics (TOG)*, 43(4): 1–13, 2024. 5, 6, 3

[12] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, et al. Building rome on a cloudless day. In *European Conference on Computer Vision*, pages 368–381, 2010. 1

[13] 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. 5, 3

[14] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14346–14355, 2021. 2

[15] Xiang Guo, Guanying Chen, Yuchao Dai, Xiaoqing Ye, Jiadai Sun, Xiao Tan, and Errui Ding. Neural deformable voxel grid for fast optimization of dynamic view synthesis, 2022. 2

[16] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018. 5, 3

[17] Tao Hu, Shu Liu, Yilun Chen, Tiancheng Shen, and Jiaya Jia. Efficientnerf: Efficient neural radiance fields, 2022. 2

[18] Jaewoo Jung, Jisang Han, Honggyu An, Jiwon Kang, Seonghoon Park, and Seungryong Kim. Relaxing accurate initialization constraint for 3d gaussian splatting. *arXiv preprint arXiv:2403.09413*, 2024. 2, 8

[19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 5, 3

[20] Leonid Keselman and Martial Hebert. Approximate differentiable rendering with algebraic surfaces, 2022. 2

[21] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4):1–13, 2017. 5, 3

[22] Christoph Lassner and Michael Zollhöfer. Pulsar: Efficient sphere-based neural rendering, 2020. 1

[23] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 6

[24] Yanyan Li, Chenyu Lyu, Yan Di, Guangyao Zhai, Gim Hee Lee, and Federico Tombari. Geogaussian: Geometry-aware gaussian splatting for scene rendering, 2024. 1, 2

[25] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023. 2

[26] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis, 2024. 2

[27] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021. 2

[28] Haotong Lin, Sida Peng, Zhen Xu, Tao Xie, Xingyi He, Hujun Bao, and Xiaowei Zhou. High-fidelity and real-time novel view synthesis for dynamic scenes. In *SIGGRAPH Asia Conference Proceedings*, 2023. 2

[29] Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. *arXiv preprint arXiv:2312.13763*, 2023. 2

[30] Xian Liu, Xiaohang Zhan, Jiaxiang Tang, Ying Shan, Gang Zeng, Dahua Lin, Xihui Liu, and Ziwei Liu. Humangaussian: Text-driven 3d human generation with gaussian splatting. *arXiv preprint arXiv:2311.17061*, 2023. 2

[31] Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-gs: Structured 3d gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024. 1, 2, 5, 3

[32] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023. 2

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

[34] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 2, 5, 3

[35] Christian Reiser, Rick Szeliski, Dor Verbin, Pratul Srinivasan, Ben Mildenhall, Andreas Geiger, Jon Barron, and Peter Hedman. Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023. 2

[36] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016. 1

[37] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022. 2

[38] Dan Wang, Xinrui Cui, Septimiu Salcudean, and Z. Jane Wang. Generalizable neural radiance fields for novel view synthesis with transformer, 2022. 2

[39] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 5

[40] Christopher Wewer, Kevin Raj, Eddy Ilg, Bernt Schiele, and Jan Eric Lenssen. latentsplat: Autoencoding variational gaussians for fast generalizable 3d reconstruction. *arXiv preprint arXiv:2403.16292*, 2024. 2

[41] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Wang Xinggang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023. 2

[42] Zeke Xie, Xindi Yang, Yujie Yang, Qi Sun, Yixiang Jiang, Haoran Wang, Yunfeng Cai, and Mingming Sun. S3im: Stochastic structural similarity and its unreasonable effectiveness for neural fields, 2023. 2, 3

[43] Haolin Xiong, Sairisheek Muttukuru, Rishi Upadhyay, Pradyumna Chari, and Achuta Kadambi. Sparsegs: Real-time 360sparse view synthesis using gaussian splatting, 2024. 2

[44] Linning Xu, Vasu Agrawal, William Laney, Tony Garcia, Aayush Bansal, Changil Kim, Samuel Rota Bulò, Lorenzo Porzi, Peter Kontschieder, Aljaž Božič, Dahua Lin, Michael Zollhöfer, and Christian Richardt. Vr-nerf: High-fidelity virtualized walkable spaces. In *SIGGRAPH Asia 2023 Conference Papers*, page 1–12. ACM, 2023. 1

[45] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields, 2023. 2

[46] Wangze Xu, Huachen Gao, Shihe Shen, Rui Peng, Jianbo Jiao, and Ronggang Wang. Mvpgs: Excavating multi-view priors for gaussian splatting from sparse input views. *arXiv preprint arXiv:2409.14316*, 2024. 2

[47] Zeyu Yang, Hongye Yang, Zijie Pan, Xiatian Zhu, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv 2310.10642*, 2023. 2

[48] Taoran Yi, Jiemin Fang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *arXiv preprint arXiv:2310.08529*, 2023. 2

[49] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 5

[50] Xiaoyu Zhang, Weihong Pan, Chong Bao, Xiyu Zhang, Xiaojun Xiang, Hanqing Jiang, and Hujun Bao. Lookcloser: Frequency-aware radiance field for tiny-detail scene. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 16122–16132, 2025. 2

[51] 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. 1

[52] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. 2

[53] Zuo-Liang Zhu, Beibei Wang, and Jian Yang. Gs-ror: 3d gaussian splatting for reflective object relighting via sdf priors, 2024. 2

[54] C Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM TOG*, 2004. 2

[55] Yiming Zuo and Jia Deng. View synthesis with sculpted neural points. *arXiv preprint arXiv:2205.05869*, 2022. 2