Holistic Large-Scale Scene Reconstruction via Mixed Gaussian Splatting

Chuandong Liu[1] **Huijiao Wang**[3] **Lei Yu**[2,3,*] **Gui-Song Xia**[2,4,5,*]

[1]School of Computer Science, Wuhan University
[2]School of Artificial Intelligence, Wuhan University
[3]School of Electronic Information, Wuhan University
[4]State Key Lab. of LIESMARS, Wuhan University
[5]Institute for Math & AI, Wuhan University
[*]Corresponding Authors

Project Page: mixgs.github.io

Abstract

Recent advances in 3D Gaussian Splatting have shown remarkable potential for novel view synthesis. However, most existing large-scale scene reconstruction methods rely on the divide-and-conquer paradigm, which often leads to the loss of global scene information and requires complex parameter tuning due to scene partitioning and local optimization. To address these limitations, we propose MixGS, a novel holistic optimization framework for large-scale 3D scene reconstruction. MixGS models the entire scene holistically by integrating camera pose and Gaussian attributes into a view-aware representation, which is decoded into fine-detailed Gaussians. Furthermore, a novel mixing operation combines decoded and original Gaussians to jointly preserve global coherence and local fidelity. Extensive experiments on large-scale scenes demonstrate that MixGS achieves state-of-the-art rendering quality and competitive speed, while significantly reducing computational requirements, enabling large-scale scene reconstruction training on a single 24GB VRAM GPU.

1 Introduction

Accurate 3D scene representation is fundamental for a wide range of applications, including aerial surveying [4], autonomous driving [26, 66], environmental monitoring [33, 65], and AR/VR [18, 20], all of which demand high-fidelity visual quality and real-time rendering. In recent years, implicit representations such as neural radiance fields (NeRF) [40, 63, 53] have achieved impressive results in novel view synthesis, but often suffer from limited detail reconstruction and slow rendering speed. More recently, explicit representations like 3D Gaussian Splatting (3DGS) [22] have demonstrated superior performance in both visual quality and computational efficiency, enabling real-time rendering and showing strong adaptability to dynamic scenes [62, 67] and content generation [75, 28].

Nevertheless, directly applying naive 3DGS to large-scale scene reconstruction faces significant challenges, such as out-of-memory issues and degraded novel view synthesis quality [30]. To address these limitations, most existing 3DGS-based methods [34, 10] adopt a divide-and-conquer strategy, inspired by its success in previous NeRF-based approaches [52, 53]. As illustrated in Fig. 1 (left), the large scene is partitioned into multiple independent blocks, enabling parallel training and optimization across multiple GPUs. After optimization, these blocks are merged to produce the final reconstruction for novel view rendering. Various improvements, such as progressive partitioning [30], recursive scene splitting [10], and adaptive data selection [34], have been proposed to enhance

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

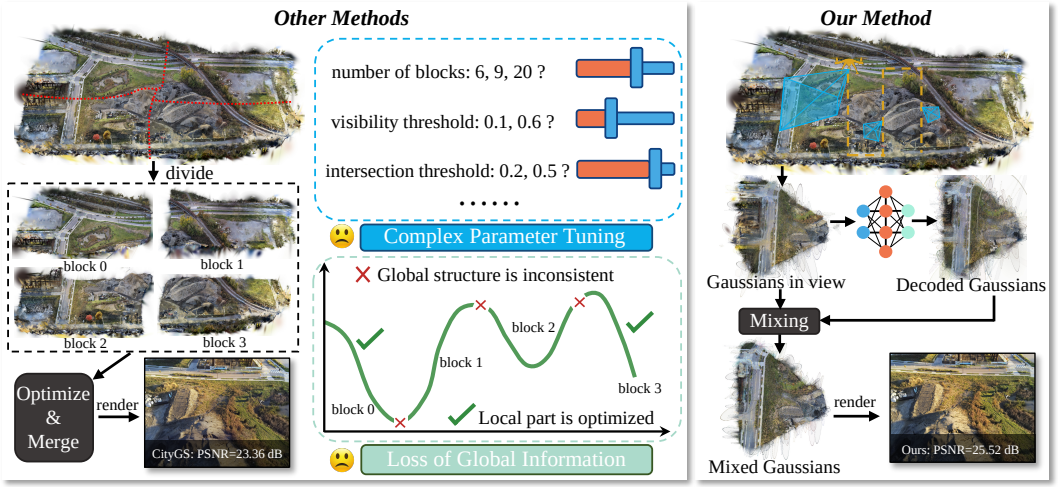

Figure 1. **Left:** Conventional divide-and-conquer approaches partition the scene into multiple independent blocks, each optimized and rendered separately before merging. This strategy introduces two major limitations: (1) complex parameter tuning, such as selecting the number of blocks and adjusting thresholds (e.g., visibility and intersection [30, 34]), which often requires extensive manual intervention and scene-specific reconfiguration; (2) loss of global information, leading to inconsistencies in global structure, illumination, and geometric continuity across block boundaries, as local optimizations do not guarantee global coherence. **Right:** In contrast, our MixGS framework treats the entire scene and the Gaussian Representation network as a holistic optimization problem. By extracting and encoding visible Gaussians, decoding new Gaussians via implicit feature representations, and mixing them with the originals, MixGS achieves both global consistency and fine-grained detail reconstruction.

the representation capability of Gaussian primitives within each block. Despite these advances, divide-and-conquer methods still suffer from two major issues.

On the one hand, scene partitioning heavily relies on prior knowledge and manual parameter tuning. The choice of division intervals along the x, y, and z axes determines the number and size of blocks, which significantly affects the convergence behavior and reconstruction quality. Besides, assigning appropriate training images to each block also requires additional threshold-based adjustments, such as visibility [30] and intersection [34] thresholds, and these configurations are often scene-specific, necessitating re-tuning for different environments.

On the other hand, independently optimizing each block can lead to locally optimal but globally sub-optimal solutions, resulting in the loss of global scene consistency. For instance, lighting conditions that should be uniform across the scene may become inconsistent, and continuous structures like buildings or grasslands can suffer from geometric discontinuities and texture misalignments at block boundaries due to fragmented optimization.

To address the aforementioned limitations of the divide-and-conquer paradigm, *i.e.,* complex parameter tuning and loss of global information, we propose MixGS, a novel 3DGS-based pipeline for large-scale scene reconstruction. Unlike traditional approaches that independently optimize and densify each scene block, as shown in Fig. 1 (right), MixGS treats the entire large-scale scene and the Gaussian Representation network as a holistic optimization framework. Specifically, Gaussian primitives within the visible frustum are first extracted based on camera poses and further filtered using opacity information. The positions of these selected Gaussians are encoded via 3D hash encoding to obtain spatial feature representations, while other Gaussian attributes and camera poses are incorporated as auxiliary inputs to enrich the feature space. These features are then decoded to generate refined Gaussians, with a designed offset pool. The decoded Gaussians are mixed with the original filtered Gaussians to form a set of mixed Gaussians for rendering, where the original Gaussians primarily capture coarse scene structure and the decoded Gaussians compensate for missing fine-grained details. This holistic design not only alleviates the high GPU memory requirements by transforming explicit Gaussian densification into implicit high-dimensional feature encoding, enabling training on a single GPU, but also achieves rendering speeds comparable to other methods.

To comprehensively evaluate our approach, we conduct extensive experiments on two prominent benchmarks [31, 53] for large-scale 3D scene reconstruction, encompassing four challenging scenes.

Our results demonstrate that MixGS significantly improves reconstruction performance in these scenarios. To sum up, our key contributions are as follow:

- We introduce MixGS, a novel holistic optimization method designed to overcome the limitations of existing approaches that rely on divide-and-conquer strategies for large-scale scene reconstruction.
- We develop a mixed Gaussians rasterization pipeline that simultaneously captures global and local scene information through view-aware Representation modeling for implicit feature learning.
- Extensive experimentation demonstrate that MixGS achieves state-of-the-art results while maintaining comparable rendering speed on established benchmarks for large-scale scene reconstruction.

2 Related Work

Neural Radiance Fields. Neural radiance fields [40] (NeRF) revolutionized 3D scene representation by learning a continuous volumetric function that maps spatial coordinates to color and density values through ray sampling. This breakthrough has inspired numerous follow-up workss [6, 16, 69] that address its computational and representational limitations. Recent advances have focused on improving training efficiency and rendering quality through various scene encoding strategies, including sparse voxel representations [32], multi-resolution hash tables [42], and orthogonal axis-plane decompositions [43]. These approaches have significantly enhanced NeRF's practical applicability while maintaining its high-quality rendering capabilities. Beyond per-scene reconstruction, significant progress has been made in generalizable NeRF approaches [7, 21, 36, 51, 57], which aim to transfer learned priors across different scenes. These methods often incorporate additional supervision signals, such as depth information [12, 47, 60], to improve training stability and reconstruction quality. To address the inherent aliasing artifacts in NeRF, researchers have proposed various solutions, including cone tracing approximations with scale-aware positional encodings [1, 2] and hexagonal sampling strategies [42, 3]. For large-scale scene reconstruction, recent works have explored the alignment of multiple NeRF blocks through both traditional optimization techniques [17] and geometry-aware transformers [9] pretrained on 3D datasets.

3D Gaussian Splatting. 3D Gaussian Splatting (3DGS) [22] represents a paradigm shift from NeRF's volumetric rendering approach, offering a more efficient explicit representation through differentiable rasterization of 3D Gaussians. While 3DGS achieves real-time rendering performance, it faces challenges in identifying and optimizing Gaussian primitives, particularly in textureless regions. Recent works have addressed this limitation through various innovative approaches. Scaffold-GS [38] introduces a sparse voxel grid representation to encode Gaussian characteristics, effectively reducing unnecessary densification while maintaining scene fidelity. Further advancements include implicit scene encoding through graph neural networks (SAGS [55]) and hierarchical level-of-detail structures (Octree-GS [45]). The aliasing issue, common to both NeRF and 3DGS due to fixed-window Gaussian kernels during rasterization, has been addressed in Mip-Splatting [70] and subsequent works [29, 50, 64, 5]. Additional research directions include 2D Gaussian [19] learning for improved surface fitting [35], dynamic scene modeling through deformation learning [62, 67, 56], and model compression through quantization techniques [14, 58].

Large-Scale Scene Reconstruction. Large-scale scene reconstruction has evolved significantly from early Structure-from-Motion (SfM) approaches [49, 48], which relied on traditional keypoint extraction methods like SIFT [37]. The availability of camera poses has facilitated scene partitioning, making the divide-and-conquer strategy a mainstream approach for large-scale reconstruction. This paradigm has been successfully adopted in NeRF-based methods [25, 44, 54, 63, 73], with notable contributions including Block-NeRF [52], which incorporates appearance embeddings [39] and learnable pose refinement for handling environmental variations. Switch-NeRF [74] and Mega-NeRF [53] further enhance this approach through switch transformer-based ray assignment and network sparsity optimization, respectively.

The recent adoption of 3D Gaussian Splatting [22] has brought new possibilities to large-scale reconstruction, offering an optimal balance between visual fidelity [24, 13, 71, 61] and rendering speed [8, 15, 46]. Following the successful divide-and-conquer strategy from NeRF, methods like CityGaussian [34] and VastGaussian [30] have developed specialized blocking strategies for 3DGS. These approaches ensure optimal training image distribution within blocks, enabling efficient techniques such as Level of Detail (LOD) [23, 35, 11]. DOGS [10] introduces a recursive splitting approach for balanced block sizes, coupled with distributed training for accelerated optimization.

While these methods have demonstrated impressive progress in large-scale scene reconstruction, they still face two fundamental challenges: the need for complex parameter tuning and the potential loss of global scene consistency, which our work aims to address.

3 Approach

3.1 Problem Definition and Formulation

3D Gaussian Splatting [22] explicitly represents a static scene as a set of anisotropic 3D Gaussians $\mathcal{G} = \{G_i\}_{i=1}^{N}$, offering both high visual fidelity and efficient training. Each Gaussian kernel is parameterized by its center $\mu_i \in \mathbb{R}^3$ and covariance matrix $\Sigma_i \in \mathbb{R}^{3\times3}$, where i indexes the i-th Gaussian. The 3D Gaussian function at a point $\mathbf{x} \in \mathbb{R}^3$ is defined as:

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

where the covariance matrix Σ_i is further factorized as $\Sigma_i = R_i S_i S_i^\top R_i^\top$, with $R_i \in \mathbb{R}^{3\times3}$ as the rotation matrix and $S_i \in \mathbb{R}^{3\times3}$ as the scale matrix. To render an image from the 3D Gaussian representation, the Gaussians are transformed from world to camera coordinates via a view transformation matrix $W \in \mathbb{R}^{3\times3}$ and projected onto the 2D image plane using a Jacobian matrix $J \in \mathbb{R}^{3\times3}$ [68, 76]. The covariance matrix in the camera plane is computed as $\mathbf{\Sigma}' = \mathbf{JW\Sigma W}^\top \mathbf{J}^\top$. Given a pixel $\mathbf{p}$ on the image plane, the final color is rendered by compositing the splatted Gaussians from front to back, following the conventional volume rendering equation:

$$\mathbf{C}(\mathbf{p}) = \sum_i \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{2}$$

where $\mathbf{c}_i$ is the color of the i-th Gaussian (computed from spherical harmonic coefficients), and α_i is the opacity associated with G_i. Each Gaussian G_i is thus parameterized as $(\mu_i, \alpha_i, r_i, s_i, c_i)$, where r_i is the quaternion corresponding to R_i, s_i is the scale vector (the diagonal of S_i), and c_i is the color.

Based on the above concept, for small-scale scene reconstruction, Gaussians can be directly optimized via the simplified objective function:

$$\min_{\mathcal{G}} \sum_{i=1}^{M} \mathcal{L}\left(\xi(\mathcal{G}, \tau_i), I_i^{\text{gt}}\right), \tag{3}$$

where $\mathcal{G}$ denotes the set of Gaussians that need to be optimized, I^{gt} is the set of ground truth images, τ_i represents camera poses, M is the total number of training images, ξ is the rendering process, and $\mathcal{L}$ is the loss function. As defined in [22], $\mathcal{L} = \mathcal{L}_1 + \lambda\mathcal{L}_{\text{SSIM}}$. However, directly training large-scale scenes with naive 3DGS is limited by memory constraints.

Divide-and-conquer schema: To address the limitations of naive 3DGS for large-scale scene, most existing methods adopt the divide-and-conquer strategy [53, 34]. Specifically, the set of Gaussians $\mathcal{G}$ is partitioned into n blocks, *i.e.,* $\mathcal{G} \rightarrow \{\mathcal{G}_{b1}, \ldots, \mathcal{G}_{bn}\}$, and each block is assigned a corresponding subset of training images: $I^{\text{gt}} \rightarrow \{I_{b1}^{\text{gt}}, \ldots, I_{bn}^{\text{gt}}\}$. Next, each block is optimized independently:

$$\mathcal{G}_{bj}^* = \min_{\mathcal{G}_{bj}} \sum_{i=1}^{M_{bj}} \mathcal{L}\left(\xi(\mathcal{G}_{bj}, \tau_i), I_{i,bj}^{\text{gt}}\right), \quad \forall bj \in \{b1, \ldots, bn\}, \tag{4}$$

where M_{bj} denotes the number of training images for block bj. Finally, all optimized blocks are merged to produce the final set of Gaussians for rendering. While divide-and-conquer methods have achieved success through sophisticated partitioning strategies, they still face two major limitations: complex parameter tuning and loss of global scene information.

Our MixGS schema: In contrast, we propose a holistic optimization framework that jointly models the entire scene, aiming to design an objective function capable of optimizing large-scale scenes in the same manner as small-scale ones. Specifically, we pretrain coarse Gaussians $\mathcal{G}_c$ from the original large-scale scene and employ a view-aware representations to model the feature attributes under the current viewpoint. Further, the representations with learnable parameter Φ can be decoded to

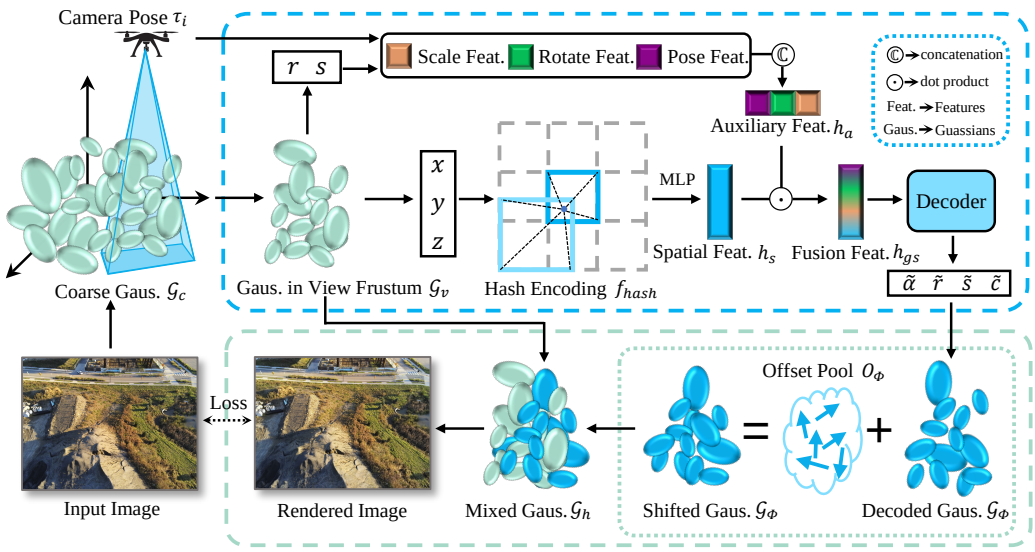

Figure 2. Overview of the proposed MixGS method pipeline. We first train the original Gaussians to capture the coarse information of the scene. Then, based on the camera poses, we extract the Gaussians within the view frustum, thereby enabling implicit feature extraction that integrates Gaussian attributes with camera poses. These features are decoded using a tiny multi-head MLP to generate the decoded Gaussians. Next, the positions of the Gaussian primitives are adjusted via an offset pool and combined with the original Gaussians to form the mixed Gaussians. Finally, the mixed Gaussians are splatted through the differentiable rasterization operation [22] to render images for supervision.

Gaussian primitives $\mathcal{G}_\Phi$ enriched with scene details, which are then mixed with the coarse Gaussians for final rendering. To sum up, our approach can be formulated as the following objective:

$$\mathcal{G}_c^*, \Phi^* = \min_{\mathcal{G}_c, \Phi} \sum_{i=1}^{M} \mathcal{L}\left(\xi(\mathcal{G}_c \cup \mathcal{G}_\Phi, \tau_i), I_i^{\text{gt}}\right). \tag{5}$$

It is easy to see that we are able to optimize large-scale scenes holistically in the same manner as small-scale scenes. Besides, we achieve high-quality rendering that captures both global structure and local details, thereby eliminating the need for complex parameter tuning and mitigating the loss of global information, while enabling training on a single 24GB consumer-grade GPU. The following sections detail the design and holistic optimization of Gaussians $\mathcal{G}_c$ and representations obtained from learnable parameters Φ.

3.2 View-Aware Representation Modeling

Coarse 3D Gaussians Training. To facilitate subsequent feature extraction for different attributes of Gaussians, it is necessary to first capture the coarse geometry and texture information of the large-scale scene. Following the standard 3DGS pipeline [22], we optimize all Gaussians using images and initial point clouds from COLMAP [48]. The resulting set of trained Gaussian primitives is denoted as $\mathcal{G}_c = \{G_i \mid i = 1, \ldots, N_c\}$, where N_c represents the total number of Gaussians. This coarse representation provides a strong geometric prior for further attribute learning.

Multi-Resolution Hash Encoding. Given coarse Gaussians $\mathcal{G}_c$, we extract the subset $\mathcal{G}_v = \{G_i\}_{i=1}^{N_v}$ within the current view frustum, *i.e.*, $\mathcal{G}_v \in \mathcal{G}_c$, determined by the camera pose τ_i (see in Fig. 2). The mean positions $\{\mu_i = (x_i, y_i, z_i)\}_{i=1}^{N_v}$ of these Gaussians are used for spatial encoding. We employ multi-resolution hash encoding [42] to capture geometric structure at multiple scales:

$$N_l = \left\lfloor N_{\min} \cdot b^l \right\rfloor, \quad b = \exp\left(\frac{\ln N_{\max} - \ln N_{\min}}{L - 1}\right), \tag{6}$$

where $N_{\min}$ and $N_{\max}$ represent the coarsest and finest resolutions, respectively, L denotes the largest number of levels, and l is the index of the current level. The selected resolution N_l determines the grid voxel resolution at each level. For a given input $\mathbf{k}$, the corresponding voxel position on the grid

can then be computed via rounding down and up $\lfloor \mathbf{k}_l \rfloor = \lfloor \mathbf{k} \cdot N_l \rfloor$, $\lceil \mathbf{k}_l \rceil = \lceil \mathbf{k} \cdot N_l \rceil$. To retrieve voxel values, positions are hashed into the corresponding hash table using the following hash function:

$$h_l(\mathbf{k}_l) = \left(\oplus_{i=1}^{d} k_i \pi_i\right) \mod T, \tag{7}$$

where $\oplus$ denotes the bit-wise XOR operation, $\{\pi_i\}$ are distinct large prime numbers assigned to each dimension, d represents the input dimensionality, and T indicates the hash table size. The encoded features are then obtained through trilinear interpolation over the corresponding grid voxel values.

Generally, we denote the above hash encoder as f_{hash}, the final spatial features h_s are obtained via:

$$h_s = f_{\theta_1} \circ f_{hash}(x_i, y_i, z_i), \tag{8}$$

where f_{θ_1} is a tiny MLP with parameters θ_1. We adopt multi-resolution hash encoding for its hierarchical structure, which enables progressive learning from coarse to fine details for performance gains, and for its efficient implementation [41], which ensures fast training and inference.

Auxiliary Gaussian Enhancement. To further enhance representation capacity, we independently encode the rotation quaternions r, scale vectors s of $\mathcal{G}_v$, and the camera pose τ_i, capturing local geometric variations and viewpoint-dependent effects. These auxiliary features are concatenated to form h_a and transformed into attention scores, which modulate the spatial features:

$$h_{gs} = (2 \cdot \sigma(f_{\theta_2}(h_a)) - 1) \odot h_s, \tag{9}$$

where h_{gs} is the fused feature, f_{θ_2} is a lightweight MLP, $\odot$ denotes element-wise multiplication, and σ is the Sigmoid function.

3.3 Mixed Gaussians Rasterization

In this section, the primary objective is to decode the high-dimensional feature h_{gs}, which models the view-aware Gaussian attributes, thereby generating corresponding Gaussian attributes $\{\mu, \alpha, r, s, c\}$ as described in Sec. 3.1. Note that we call the generated Gaussians as decoded Gaussians $\mathcal{G}_\Phi$. In detail, we use a multi-head MLP f_{θ_3} to decode the features h_{gs} and predict the Gaussian opacity $\tilde{\alpha} \in \mathbb{R}$, color $\tilde{c} \in \mathbb{R}^3$, and covariance matrix Σ, which is reparameterization by a rotation quaternion $\tilde{r} \in \mathbb{R}^4$ and a scaling matrix $\tilde{s} \in \mathbb{R}^3$. This process can be formulated as:

$$\{(\tilde{\alpha}_i, \tilde{r}_i, \tilde{s}_i, \tilde{c}_i)\}_{i=1}^{N_\Phi} = f_{\theta_3}(h_{gs}). \tag{10}$$

Note that the number of decoded Gaussians are the same with Gaussians in the view frustum, *i.e.,* $N_\Phi = N_v$. Before being used to formulate Gaussian representations, the rotation quaternion should be normalized and the opacity should be activated by Sigmoid function, to satisfy their range and feature. Moreover, we design a offset pool $O_c = \{o_1, o_2, \ldots, o_{N_c}\}$ to save the shifted value for each decoded Gaussian primitives. Further, the position of decoded Gaussians are calculated as:

$$\{\tilde{\mu}_1, \ldots, \tilde{\mu}_{N_\Phi}\} = \{\mu_1, \ldots, \mu_{N_v}\} + \{o_1, \ldots, o_{N_\Phi}\}, \tag{11}$$

where $O_\Phi \in O_c$ is the offset pool in view, and μ_{N_v} is the position attribute of Gaussians $\mathcal{G}_v$. Compared to directly predicting absolute positions, when the coarse Gaussian position is already close to the target structure, fine-tuning through local offset prediction can more effectively capture small-scale geometric variations. By predicting residual deviations rather than complete positional information, the model converges more easily and achieves greater precision in detail refinement.

With the aforementioned decoded Gaussians $\mathcal{G}_\Phi$, a novel view can be rendered using the rasterization splatting pipeline [22]. However, the obtained view easily focus on view-aware local details, thereby lacking stable global structure. Hence, we propose to combine Gaussians $\mathcal{G}_\Phi$ and $\mathcal{G}_v$, generating mixed Gaussians $\mathcal{G}_h$ for further rendering. This process can be formulated as:

$$\mathcal{G}_h = \mathcal{G}_v \cup \mathcal{G}_\Phi = \{G_1, \ldots, G_{N_v}, G_{N_v+1}, \ldots, G_{N_\Phi+N_v}\}. \tag{12}$$

Therefore, we can obtain final rendered view via the mixed Gaussians. In this way, Gaussian densification is implicitly achieved, which facilitates the reconstruction of fine details and complex lighting in large-scale scenes. On the other hand, it significantly reduces the memory requirements during training, enabling efficient training even on a consumer GPU with 24GB of VRAM.

3.4 Training with Multi-Stage Optimization

To ensure effective optimization during training, we adopt a three-stage optimization strategy. This approach progressively refines the model from both global and local perspectives, enabling a better balance between global structure and local details in large-scale scene reconstruction.

(1) Coarse Stage. Coarse Gaussians $\mathcal{G}_c$ are trained to rapidly capture the global geometric structure, providing a stable prior and preventing premature overfitting to local details.

(2) Detail Stage. With coarse Gaussian parameters fixed, we optimize other parameters Φ, such as the hash encoder, MLP with learnable parameters, and offset pool, *i.e.,* $\Phi = (f_{\theta_1}, f_{\theta_2}, f_{\theta_3}, f_{hash}, O_c)$, allowing the model to fit local details and high-frequency geometry based on the global structure.

(3) Joint Stage. After capturing both global and local information, we jointly fine-tune all parameters ($\mathcal{G}_c$ and Φ) using the mixed Gaussians. This stage integrates global and local cues, eliminates boundary artifacts, and leverages priors from earlier stages, further enhancing overall quality.

4 Experiments

4.1 Experimental Settings

Datasets and Evaluation Metrics. Following state-of-the-art methods [10, 34], we conduct experiments on large-scale scenes across two real-world urban scene datasets: UrbanScene3D [31] dataset with Residence and Sci-Art, and Mill19 [53] dataset with Building and Rubble. Both of datasets consist of thousands of high-resolution images captured by drones. For fair comparison [10, 53, 30], we adopt the same data splits as previous works to construct the training and testing sets. Specifically, the Building, Rubble, Residence, and Sci-Art scenes contain 1920, 1657, 2582, and 3620 training images, respectively, and 20, 21, 21, and 21 testing images. For the evaluation of novel view synthesis, we report PSNR, SSIM [59], and LPIPS [72] metrics. Additionally, we measure Frame Per Second (FPS) to assess model effectiveness.

Compared Methods. We compare MixGS against seven state-of-the-art methods, categorized into NeRF-based methods: Switch-NeRF [74], Mega-NeRF [53], GP-NeRF [63] and 3DGS-based methods: VastGaussian [30], CityGaussian [34], Hierarchy-GS [23], DOGS [10]. To ensure a fair comparison, we downsample all images by 4 times. Note that DOGS [10] differs slightly from other methods, which downsamples images by a factor of 6 times. Therefore, we also conduct a experiments with 6 times downsampling for direct comparison, as reported in the Appendix.

Implementation Details. All experiments are conducted on an NVIDIA RTX 3090 GPU with 24GB VRAM using PyTorch. Following previous methods [34, 10], we use the official camera poses provided by Mega-NeRF [53] and initialize 3D Gaussians with COLMAP [48]. The three training stages are run for 30,000, 40,000, and 260,000 iterations, respectively. To fit the original 3DGS [22] for large-scale scene reconstruction, we train for 60,000 iterations, applying densification every 200 steps until 30,000 iterations. VastGaussian [30] applies color correction to rendered images before metric evaluation, which can significantly improve results. For fair comparison, we report the results of VastGaussian using a reproduced version [10] that excludes both color correction and decoupled appearance encoding.

4.2 Main Results

Quantitative results. Table 1 reports the mean PSNR, SSIM, and LPIPS metrics across the four large-scale scenes. MixGS achieves the highest SSIM across all scenes, highlighting its superior perceptual quality. In terms of PSNR and LPIPS, our method consistently delivers either the best or highly competitive results. Notably, on the challenging Residence dataset, MixGS achieves a PSNR of 23.39, enabling 3DGS-based approaches to outperform NeRF-based methods. It is worth mentioning that superior PSNR values are obtained by NeRF-based methods on the Sci-Art dataset, which exhibits inherent blur due to capture conditions. This can be attributed to the tendency of NeRF-based reconstructions to produce smoother outputs that align with the dataset's characteristics. In contrast, 3DGS-based methods, including our MixGS, maintain better structural and perceptual fidelity across all scenes, as reflected in the SSIM and LPIPS scores.

Table 1. Quantitative results of our method compared to previous work on Mill19 dataset [53] and UrbanScene3D dataset [31]. We report metrics for PSNR↑, SSIM↑, and LPIPS↓ on test views. The best, second best, and third best results are highlighted in red, orange, and yellow respectively. † represents without applying the decoupled appearance encoding.

Scenes	Building			Rubble			Residence			Sci-Art		
Metrics	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Mega-NeRF [53]	20.92	0.547	0.454	24.06	0.553	0.508	22.08	0.628	0.401	25.60	0.770	0.312
Switch-NeRF [74]	21.54	0.579	0.397	24.31	0.562	0.478	22.57	0.654	0.352	26.51	0.795	0.271
GP-NeRF [63]	21.03	0.566	0.486	24.06	0.565	0.496	22.31	0.661	0.448	25.37	0.783	0.373
3DGS [22]	20.46	0.720	0.305	25.47	0.777	0.277	21.44	0.791	0.236	21.05	0.830	0.242
VastGaussian† [30]	21.80	0.728	0.225	25.20	0.742	0.264	21.01	0.699	0.261	22.64	0.761	0.261
Hierarchy-GS [23]	21.52	0.723	0.297	24.64	0.755	0.284	–	–	–	–	–	–
DOGS [10]	22.73	0.759	0.204	25.78	0.765	0.257	21.94	0.740	0.244	24.42	0.804	0.219
CityGaussian [34]	21.67	0.764	0.262	24.90	0.785	0.256	21.90	0.805	0.217	21.34	0.833	0.232
MixGS (Ours)	23.03	0.771	0.261	26.66	0.792	0.267	23.39	0.815	0.219	24.20	0.856	0.220

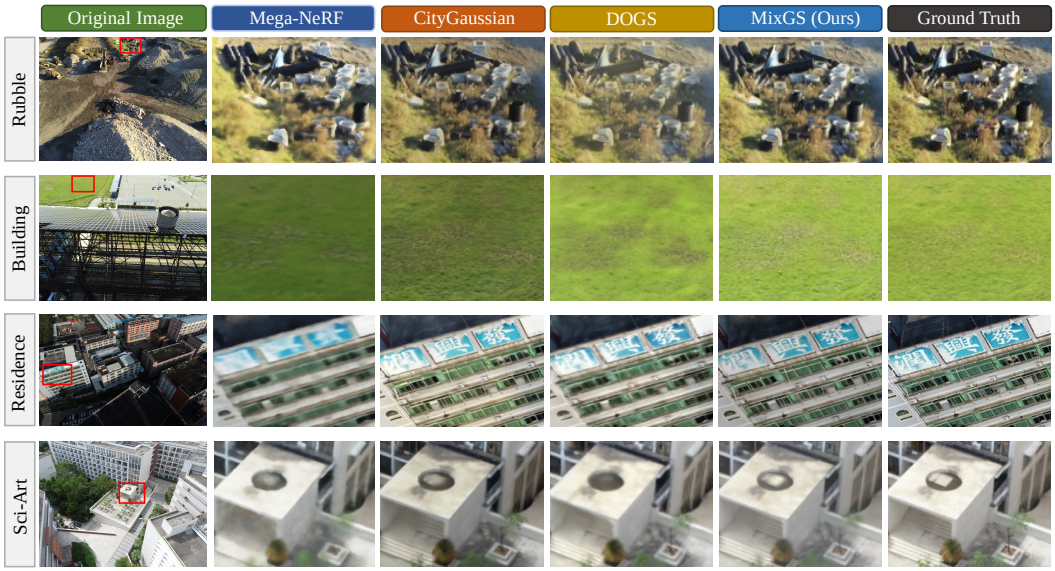

Figure 3. Qualitative comparison of rendering on the Mill19 [53] and UrbanScene3D [31] datasets. We demonstrate the zoomed-in detail of the selected areas in red box.

Qualitative results. Fig. 3 presents the qualitative comparisons of the novel view synthesis results. For MegaNeRF[1], CityGaussian[2], and DOGS[3], all visualizations are generated using official pretrained models or results. It can be observed that MixGS outperforms existing methods by producing sharper textures, more accurate illumination, and improved geometric consistency. Specifically, MegaNeRF lacks fine details and exhibit blurry and erroneous structures in image rendering. Additionally, MixGS demonstrates significantly better visual quality under challenging lighting conditions, such as the example on the 2^{nd} row in Fig. 3. Additional visualization results are provided in the Appendix.

Rendering speed. To further assess the efficiency of our method, we compare the rendering speed (FPS) with other approaches. As shown in Table. 2, traditional NeRF-based methods, such as Mega-NeRF and Switch-NeRF, exhibit significantly slower rendering speed (<0.1 FPS). In contrast, 3DGS-based methods achieve much higher rendering speeds. Benefit from the acceleration provided by the CUDA/C++ implementation and lightweight network design, MixGS achieve real-time speed ($\geq$ 30 FPS) on all large-scale scenes, even when running on a single RTX 3090 GPU, and outperforms or matches methods with more powerful A100 GPUs. To ensure accurate and fair measurements of rendering times per frame, we explicitly synchronize all CUDA streams prior to timing. For more analysis about training time and consumption across methods, please refer to the Appendix.

[1]https://github.com/cmusatyalab/mega-nerf
[2]https://github.com/Linketic/CityGaussian
[3]https://github.com/AIBluefisher/DOGS

Table 2. Rendering speed comparison on the UrbanScene3D [31] and Mill19 [53] datasets. We report the FPS and used GPU type.

Methods	GPU Type	Building	Rubble	Residence	Sci-Art
Mega-NeRF [53]	A100 40G	<0.1	<0.1	<0.1	<0.1
Switch-NeRF [74]	A100 40G	<0.1	<0.1	<0.1	<0.1
GP-NeRF [63]	A100 40G	0.42	0.40	0.31	0.34
3DGS [22]	A100 40G	45.0	47.8	62.1	72.2
CityGaussian (w/ LoD) [34]	A100 40G	37.4	52.6	41.6	64.6
CityGaussian (w/o LoD) [34]	A100 40G	24.3	43.9	32.7	56.1
MixGS (Ours)	3090 24G	33.5	42.5	36.6	53.9

Table 3. Quantitative ablation results on the *Rubble* dataset.

Model	PSNR↑	SSIM↑	LPIPS↓
w/o HE	24.70	0.728	0.324
w/o AE	26.24	0.784	0.277
w/o CT	25.86	0.763	0.307
w/o IT	25.99	0.759	0.311
w/o OP	26.35	0.779	0.275
w/o GV	23.65	0.704	0.360
full model	26.66	0.792	0.267

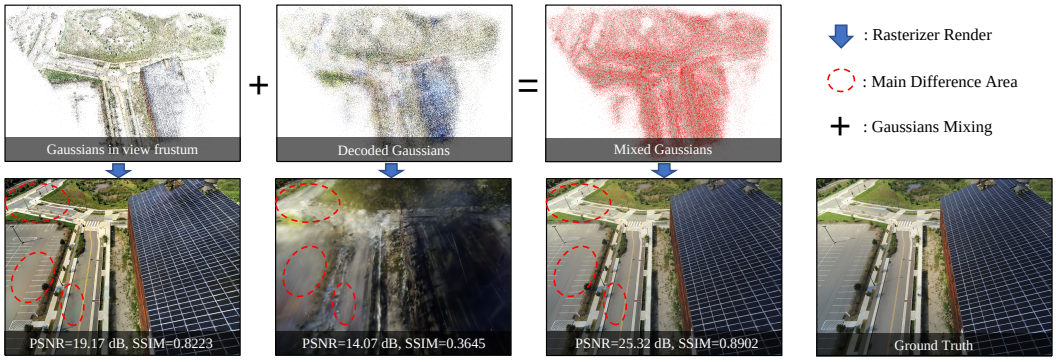

Figure 4. Visualizations of the Gaussians in the view frustum, decoded Gaussians, and mixed Gaussians, along with their corresponding rendered images. It can be observed that rendering with mixed Gaussians leads to better reconstruction of lighting consistency and fine-grained local details.

4.3 Ablation Study

Key components. Table 3 presents the results of ablation experiments designed to assess the individual contributions of key components in the proposed MixGS framework. We remove or modify each module and evaluate its impact on reconstruction performance, using the Rubble scene as a example.

First, we examine the effect of multi-resolution hash encoding (*w/o* HE). Removing this module results in a substantial performance drop, with PSNR decreasing by approximately 1.96dB compared to the full model. This clearly demonstrates the critical role of hash encoding in representing the 3D positional attributes of Gaussians. Second, we assess the impact of auxiliary Gaussian enhancement (*w/o* AE). Excluding this component leads to a noticeable decline in performance, primarily due to the absence of camera pose and other Gaussian attribute information. This result highlights the necessity of incorporating auxiliary features to effectively capture the complex variations present in large-scale scenes. We further validate the effectiveness of the multi-stage training strategy. The 3^{rd} and 4^{th} rows of Table 3 correspond to the ablation of the first-stage coarse Gaussian training (*w/o* CT) and the second-stage implicit feature training (*w/o* IT), respectively. Omitting either stage prevents the optimization process from converging to an optimal solution, indicating that both global structure initialization and local detail refinement are essential for high-quality reconstruction. Next, we investigate the role of the offset pool (*w/o* OP), where positional offsets are directly regressed by the decoder rather than learned through a dedicated pool. As shown in the 5^{th} row, this leads to a slight decrease in reconstruction performance, as direct regression introduces greater uncertainty and makes it more challenging to capture fine geometric details.

Additionally, we conduct an ablation on mixed Gaussian rendering (*w/o* GV), where only the decoded Gaussians are used for rendering, without combining them with the original Gaussians in the view frustum. This configuration results in a significant degradation of reconstruction quality, as relying solely on implicit features makes it difficult to maintain the global structure of large-scale scenes. To further illustrate the benefits of Gaussian mixing, Fig. 4 visualizes the Gaussians in the view frustum, the decoded Gaussians, and the mixed Gaussians, along with their corresponding rendered images. It is evident that rendering with mixed Gaussians leads to superior reconstruction of lighting consistency and fine-grained local details.

Process in highly textured regions. To ensure that richly textured areas are faithfully reconstructed, MixGS employs two complementary mechanisms.

Figure 5. Qualitative results near scene boundary on the Building scene.

Adaptive Densification: On the one hand, in the coarse training stage, Gaussians are duplicated and split in regions with high-frequency textures and detailed structure (as in 3DGS [22]). As a result, areas with complex details are initially covered by a denser set of coarse Gaussians. This seeding provides a strong prior for the decoded Gaussians to refine fine details.

Offset-Driven Refinement: On the other hand, benefiting from the offset pool, the decoded Gaussians will move toward under-reconstructed and high-frequency zones during training, instead of locating near the corresponding coarse Gaussians. We show the center location of the coarse and decoded Gaussians in Fig. 4. It can be observed that the decoded Gaussians shift toward more complex areas (e.g., bush and rooftop tiles), further increasing the Gaussian density exactly where it is needed most.

Moreover, MixGS can generate multiple decoded Gaussians per coarse Gaussian by simply modifying the decoder's output dimension. We evaluated different ratios K (i.e., $N_\Phi = N_v$) on the Rubble scene in Table 4. It can be observed that higher ratios improve quality (PSNR reaches 27.02 dB at 3:1) but increase memory usage. We use the 1:1 setting for optimal balance between performance and memory efficiency on a single 24GB GPU.

Table 4. Quantitative results of different ratio K on the Rubble scene.

Ratio K	PSNR↑	SSIM↑	LPIPS↓	Peak VRAM↓
1	26.66	0.792	0.267	20.45 GB
2	26.76	0.796	0.263	22.91 GB
3	27.02	0.801	0.255	25.41 GB

Importance of holistic optimization. We include additional visualization results in Fig. 5 to illustrate the differences between holistic optimization and the divide-and-conquer strategy. For clarity, the block boundaries are explicitly visualized. Although some post-processing operations [10, 34] with increased overlap are applied to mitigate boundary effects, it can still be observed that the divide-and-conquer approach leads to inconsistent illumination and discontinuities along the block boundaries (red box), as well as obvious black artifacts along the block borders (green box). In contrast, thanks to the advantages of holistic optimization, MixGS effectively avoids such issues.

5 Conclusion

In this paper, we propose MixGS, a novel framework for large-scale novel view synthesis. In contrast to mainstream approaches that adopt the divide-and-conquer paradigm, MixGS treats the entire scene as a unified whole and performs holistic optimization. Our method introduces a view-aware representation modeling strategy that integrates camera pose and Gaussian attributes to extract informative local features. By leveraging a multi-head decoder and an offset pool, MixGS generates decoded Gaussians that effectively capture fine-grained scene details. The final image rendering is achieved by mixing the Gaussians within the view frustum with the decoded Gaussians, resulting in high-quality scene reconstruction. Extensive experiments on multiple challenging datasets demonstrate that MixGS achieves state-of-the-art performance in view synthesis quality, while maintaining efficiency on a single GPU. Importantly, MixGS provides a fundamentally new perspective for large-scale scene reconstruction, opening up promising directions for future research.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (No.62325111, U22B2011, 62271354) and the Fundamental Research Funds for the Central Universities (204205kf0063).

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 *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 5835–5844. IEEE, 2021. 3

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5460–5469. IEEE, 2022. 3

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 19640–19648. IEEE, 2023. 3

[4] Ilker Bozcan and Erdal Kayacan. Au-air: A multi-modal unmanned aerial vehicle dataset for low altitude traffic surveillance. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8504–8510. IEEE, 2020. 1

[5] Brian Chao, Hung-Yu Tseng, Lorenzo Porzi, Chen Gao, Tuotuo Li, Qinbo Li, Ayush Saraf, Jia-Bin Huang, Johannes Kopf, Gordon Wetzstein, and Changil Kim. Textured gaussians for enhanced 3d scene appearance modeling. *CoRR*, abs/2411.18625, 2024. 3

[6] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *Computer Vision - ECCV 2022 - 17th European Conference*, volume 13692, pages 333–350, 2022. 3

[7] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 14104–14113. IEEE, 2021. 3

[8] Junyi Chen, Weicai Ye, Yifan Wang, Danpeng Chen, Di Huang, Wanli Ouyang, Guofeng Zhang, Yu Qiao, and Tong He. Gigags: Scaling up planar-based 3d gaussians for large scene surface reconstruction. *arXiv preprint arXiv:2409.06685*, 2024. 3

[9] Yu Chen and Gim Hee Lee. Dreg-nerf: Deep registration for neural radiance fields. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pages 22646–22656. IEEE, 2023. 3

[10] Yu Chen and Gim Hee Lee. Dogs: Distributed-oriented gaussian splatting for large-scale 3d reconstruction via gaussian consensus. In *Advances in Neural Information Processing Systems*, volume 37, pages 34487–34512, 2024. 1, 3, 7, 8, 10, 15, 16

[11] Jiadi Cui, Junming Cao, Fuqiang Zhao, Zhipeng He, Yifan Chen, Yuhui Zhong, Lan Xu, Yujiao Shi, Yingliang Zhang, and Jingyi Yu. Letsgo: Large-scale garage modeling and rendering via lidar-assisted gaussian primitives. *ACM Transactions on Graphics (TOG)*, 43(6):1–18, 2024. 3

[12] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 12872–12881. IEEE, 2022. 3

[13] Jixuan Fan, Wanhua Li, Yifei Han, and Yansong Tang. Momentum-gs: Momentum gaussian self-distillation for high-quality large scene reconstruction. *arXiv preprint arXiv:2412.04887*, 2024. 3

[14] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ FPS. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. 3

[15] Guofeng Feng, Siyan Chen, Rong Fu, Zimu Liao, Yi Wang, Tao Liu, Zhilin Pei, Hengjie Li, Xingcheng Zhang, and Bo Dai. Flashgs: Efficient 3d gaussian splatting for large-scale and high-resolution rendering. *arXiv preprint arXiv:2408.07967*, 2024. 3

[16] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5491–5500, 2022. 3

[17] Lily Goli, Daniel Rebain, Sara Sabour, Animesh Garg, and Andrea Tagliasacchi. nerf2nerf: Pairwise registration of neural radiance fields. In *IEEE International Conference on Robotics and Automation, ICRA 2023, London, UK, May 29 - June 2, 2023*, pages 9354–9361. IEEE, 2023. 3

[18] Jiaming Gu, Minchao Jiang, Hongsheng Li, Xiaoyuan Lu, Guangming Zhu, Syed Afaq Ali Shah, Liang Zhang, and Mohammed Bennamoun. Ue4-nerf: Neural radiance field for real-time rendering of large-scale scene. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. 1

[19] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024. 3

[20] Ying Jiang, Chang Yu, Tianyi Xie, Xuan Li, Yutao Feng, Huamin Wang, Minchen Li, Henry Y. K. Lau, Feng Gao, Yin Yang, and Chenfanfu Jiang. VR-GS: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In Andres Burbano, Denis Zorin, and Wojciech Jarosz, editors, *ACM SIGGRAPH 2024 Conference Papers, SIGGRAPH 2024, Denver, CO, USA, 27 July 2024- 1 August 2024*, page 78. ACM, 2024. 1

[21] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 18344–18347. IEEE, 2022. 3

[22] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139:1–139:14, 2023. 1, 3, 4, 5, 6, 7, 8, 9, 10, 15, 16

[23] 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. 3, 7, 8, 16

[24] Bingling Li, Shengyi Chen, Luchao Wang, Kaimin Liao, Sijie Yan, and Yuanjun Xiong. Retinags: Scalable training for dense scene rendering with billion-scale 3d gaussians. *arXiv preprint arXiv:2406.11836*, 2024. 3

[25] Ruilong Li, Sanja Fidler, Angjoo Kanazawa, and Francis Williams. Nerf-xl: Scaling nerfs with multiple gpus. In *European Conference on Computer Vision*, pages 92–107. Springer, 2024. 3

[26] Wei Li, CW Pan, Rong Zhang, JP Ren, YX Ma, Jin Fang, FL Yan, QC Geng, XY Huang, HJ Gong, et al. Aads: Augmented autonomous driving simulation using data-driven algorithms. *Science robotics*, 4(28):eaaw0863, 2019. 1

[27] Yixuan Li, Lihan Jiang, Linning Xu, Yuanbo Xiangli, Zhenzhi Wang, Dahua Lin, and Bo Dai. Matrixcity: A large-scale city dataset for city-scale neural rendering and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3205–3215, 2023. 15

[28] Zhiqi Li, Yiming Chen, and Peidong Liu. Dreammesh4d: Video-to-4d generation with sparse-controlled gaussian-mesh hybrid representation. *Advances in Neural Information Processing Systems*, 37:21377–21400, 2024. 1

[29] Zhihao Liang, Qi Zhang, Wenbo Hu, Lei Zhu, Ying Feng, and Kui Jia. Analytic-splatting: Anti-aliased 3d gaussian splatting via analytic integration. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part XVII*, volume 15075 of *Lecture Notes in Computer Science*, pages 281–297. Springer, 2024. 3

[30] Jiaqi Lin, Zhihao Li, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Jiayue Liu, Yangdi Lu, Xiaofei Wu, Songcen Xu, Youliang Yan, et al. Vastgaussian: Vast 3d gaussians for large scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5166–5175, 2024. 1, 2, 3, 7, 8, 15, 16

[31] Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, reconstructing, and simulating: The urbanscene3d dataset. In Shai Avidan, Gabriel J. Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part VIII*, volume 13668 of *Lecture Notes in Computer Science*, pages 93–109. Springer, 2022. 2, 7, 8, 9, 15, 16, 17, 18

[32] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *Advances in Neural Information Processing Systems 33*, 2020. 3

[33] Shuhong Liu, Xiang Chen, Hongming Chen, Quanfeng Xu, and Mingrui Li. Deraings: Gaussian splatting for enhanced scene reconstruction in rainy environments. In Toby Walsh, Julie Shah, and Zico Kolter, editors, *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, pages 5558–5566. AAAI Press, 2025. 1

[34] Yang Liu, Chuanchen Luo, Lue Fan, Naiyan Wang, Junran Peng, and Zhaoxiang Zhang. Citygaussian: Real-time high-quality large-scale scene rendering with gaussians. In *European Conference on Computer Vision*, pages 265–282. Springer, 2024. 1, 2, 3, 4, 7, 8, 9, 10, 15, 16

[35] Yang Liu, Chuanchen Luo, Zhongkai Mao, Junran Peng, and Zhaoxiang Zhang. Citygaussianv2: Efficient and geometrically accurate reconstruction for large-scale scenes. *arXiv preprint arXiv:2411.00771*, 2024. 3

[36] Yuan Liu, Sida Peng, Lingjie Liu, Qianqian Wang, Peng Wang, Christian Theobalt, Xiaowei Zhou, and Wenping Wang. Neural rays for occlusion-aware image-based rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 7814–7823. IEEE, 2022. 3

[37] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, 2004. 3

[38] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 20654–20664. IEEE, 2024. 3

[39] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7210–7219, 2021. 3

[40] 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. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow*, volume 12346, pages 405–421, 2020. 1, 3

[41] Thomas Müller. tiny-cuda-nn, 4 2021. 6

[42] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 3, 5

[43] Seungtae Nam, Daniel Rho, Jong Hwan Ko, and Eunbyung Park. Mip-grid: Anti-aliased grid representations for neural radiance fields. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. 3

[44] Konstantinos Rematas, Andrew Liu, Pratul P Srinivasan, Jonathan T Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12932–12942, 2022. 3

[45] Kerui Ren, Lihan Jiang, Tao Lu, Mulin Yu, Linning Xu, Zhangkai Ni, and Bo Dai. Octree-gs: Towards consistent real-time rendering with lod-structured 3d gaussians. *CoRR*, abs/2403.17898, 2024. 3

[46] Xuanchi Ren, Yifan Lu, Jay Zhangjie Wu, Huan Ling, Mike Chen, Sanja Fidler, Francis Williams, Jiahui Huang, et al. Scube: Instant large-scale scene reconstruction using voxsplats. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 3

[47] Barbara Roessle, Jonathan T. Barron, Ben Mildenhall, Pratul P. Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 12882–12891. IEEE, 2022. 3

[48] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 4104–4113. IEEE Computer Society, 2016. 3, 5, 7

[49] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. *ACM Trans. Graph.*, 25(3):835–846, 2006. 3

[50] Xiaowei Song, Jv Zheng, Shiran Yuan, Huan-ang Gao, Jingwei Zhao, Xiang He, Weihao Gu, and Hao Zhao. SA-GS: scale-adaptive gaussian splatting for training-free anti-aliasing. *CoRR*, abs/2403.19615, 2024. 3

[51] Mohammed Suhail, Carlos Esteves, Leonid Sigal, and Ameesh Makadia. Light field neural rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 8259–8269. IEEE, 2022. 3

[52] 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. 1, 3

[53] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12922–12931, 2022. 1, 2, 3, 4, 7, 8, 9, 15, 16, 17, 18

[54] Haithem Turki, Jason Y Zhang, Francesco Ferroni, and Deva Ramanan. Suds: Scalable urban dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12375–12385, 2023. 3

[55] Evangelos Ververas, Rolandos Alexandros Potamias, Jifei Song, Jiankang Deng, and Stefanos Zafeiriou. SAGS: structure-aware 3d gaussian splatting. In Ales Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part XIX*, volume 15077 of *Lecture Notes in Computer Science*, pages 221–238. Springer, 2024. 3

[56] Linhan Wang, Kai Cheng, Shuo Lei, Shengkun Wang, Wei Yin, Chenyang Lei, Xiaoxiao Long, and Chang-Tien Lu. Dc-gaussian: Improving 3d gaussian splatting for reflective dash cam videos. In *Neural Information Processing Systems*, 2024. 3

[57] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P. Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas A. Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 4690–4699. Computer Vision Foundation / IEEE, 2021. 3

[58] Yufei Wang, Zhihao Li, Lanqing Guo, Wenhan Yang, Alex C. Kot, and Bihan Wen. Contextgs : Compact 3d gaussian splatting with anchor level context model. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. 3

[59] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004. 7

[60] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 5590–5599. IEEE, 2021. 3

[61] Yongchang Wu, Zipeng Qi, Zhenwei Shi, and Zhengxia Zou. Blockgaussian: Efficient large-scale scene novelview synthesis via adaptive block-based gaussian splatting. *arXiv preprint arXiv:2504.09048*, 2025. 3

[62] Jiawei Xu, Zexin Fan, Jian Yang, and Jin Xie. Grid4d: 4d decomposed hash encoding for high-fidelity dynamic gaussian splatting. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. 1, 3

[63] Linning Xu, Yuanbo Xiangli, Sida Peng, Xingang Pan, Nanxuan Zhao, Christian Theobalt, Bo Dai, and Dahua Lin. Grid-guided neural radiance fields for large urban scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8296–8306, 2023. 1, 3, 7, 8, 9

[64] Zhiwen Yan, Weng Fei Low, Yu Chen, and Gim Hee Lee. Multi-scale 3d gaussian splatting for anti-aliased rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 20923–20931. IEEE, 2024. 3

[65] Daniel Yang, John J. Leonard, and Yogesh A. Girdhar. Seasplat: Representing underwater scenes with 3d gaussian splatting and a physically grounded image formation model. *CoRR*, abs/2409.17345, 2024. 1

[66] Zhenpei Yang, Yuning Chai, Dragomir Anguelov, Yin Zhou, Pei Sun, Dumitru Erhan, Sean Rafferty, and Henrik Kretzschmar. Surfelgan: Synthesizing realistic sensor data for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11118–11127, 2020. 1

[67] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 20331–20341. IEEE, 2024. 1, 3

[68] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions On Graphics (TOG)*, 38(6):1–14, 2019. 4

[69] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields. In *IEEE/CVF International Conference on Computer Vision*, pages 5732–5741, 2021. 3

[70] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 19447–19456. IEEE, 2024. 3

[71] Chubin Zhang, Hongliang Song, Yi Wei, Chen Yu, Jiwen Lu, and Yansong Tang. Geolrm: Geometry-aware large reconstruction model for high-quality 3d gaussian generation. In *Advances in Neural Information Processing Systems*, volume 37, pages 55761–55784, 2024. 3

[72] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 586–595. Computer Vision Foundation / IEEE Computer Society, 2018. 7

[73] Yuqi Zhang, Guanying Chen, and Shuguang Cui. Efficient large-scale scene representation with a hybrid of high-resolution grid and plane features. *Pattern Recognition*, 158:111001, 2025. 3

[74] MI Zhenxing and Dan Xu. Switch-nerf: Learning scene decomposition with mixture of experts for large-scale neural radiance fields. In *The Eleventh International Conference on Learning Representations*, 2022. 3, 7, 8, 9, 15, 16

[75] Junsheng Zhou, Weiqi Zhang, and Yu-Shen Liu. Diffgs: Functional gaussian splatting diffusion. *Advances in Neural Information Processing Systems*, 37:37535–37560, 2024. 1

[76] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 371–378, 2001. 4

A Analysis of $6\times$ Downsampling

Table 5. Quantitative results of our method compared to previous work on Mill19 dataset [53] and UrbanScene3D dataset [31]. We report metrics for PSNR↑, SSIM↑, and LPIPS↓ on test views. The *best* and second best are highlighted.

Scenes	Building			Rubble			Residence			Sci-Art		
Metrics	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
DOGS [10] (6×)	22.73	0.759	0.204	25.78	0.765	0.257	21.94	0.740	0.244	**24.42**	0.804	0.219
MixGS (Ours) (4×)	23.03	0.771	0.261	26.66	0.792	0.267	23.39	0.815	0.219	24.20	0.856	0.220
MixGS (Ours) (6×)	**23.82**	**0.8142**	**0.197**	**27.95**	**0846**	**0.189**	**23.96**	**0.859**	**0.159**	24.36	**0.878**	**0.1648**

Consistent with most previous works [53, 34, 30], we primarily downsample all images by a factor of 4, while DOGS [10] employs a higher downsampling rate of 6 or more. We observe that increasing the downsampling factor to 6 leads to significantly improved results compared to a factor of 4, as the reconstruction task becomes less challenging due to the reduction of high-frequency information. As shown in Table 5, our method achieves substantially better performance when images are downsampled by a factor of 6.

B Evaluation on Additional Datasets

we have evaluated MixGS on the MatrixCity and Campus scenes. The MatrixCity dataset [27] comprises 5,620 training views and 741 test views, while the Campus subset of UrbanScene3D includes 5,850 training images and 21 test images. Both datasets are recognized for their challenging scale and high-frequency details. As shown in Table 6, on both MatrixCity and Campus, MixGS achieves the highest rendering quality across PSNR, SSIM, and LPIPS, demonstrating its robustness across diverse large-scale scenes.

Why LPIPS Is Relatively Higher? From Table 1, it can be observed that MixGS has better PSNR metrics, while the LPIPS metrics are not that significant. Our MixGS choice to prioritizes holistic consistency (critical for large scenes), instead of trivial LPIPS optimization. Hence, this does not reflect poor detail recovery. Methods like DOGS and CityGaussian use divide-and-conquer, which can create "pixel-perfect but lighting inconsistent" local regions (lower LPIPS for small patches) but fail to align textures/lighting across block boundaries (e.g., grassland suddenly shifts in the 4 rows and 2 columns in Fig. 6). Our holistic framework avoids this by enforcing global consistency, which may introduce minor local color adjustments to align large structures, and these adjustments are what LPIPS occasionally penalizes.

Table 6. Additional quantitative results of novel view synthesis on the MatrixCity [27] dataset and Campus [53] scene. † indicates results obtained without applying decoupled appearance encoding.

Method	Matrixcity			Campus		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
MegaNeRF [53]	–	–	–	23.42	0.537	0.636
SwitchNeRF [74]	–	–	–	23.62	0.541	0.616
3DGS [22]	23.67	0.735	0.384	18.95	0.630	0.447
VastGaussian† [30]	28.33	0.835	0.220	23.82	0.695	0.329
CityGaussian [34]	27.32	0.813	0.211	–	–	–
DOGS [10]	28.58	0.847	0.219	24.01	0.681	0.377
Ours	**28.63**	**0.857**	**0.210**	**24.28**	**0.715**	**0.315**

C Computational Overhead

To analysis the trade off between different methods, we compare the computational overhead on both the Mill19 and UrbanScene3D datasets. In Table 7, we report the optimization time, GPU memory consumption during evaluation, and the type of GPU used. For Mega-NeRF, Switch-NeRF, VastGaussian, and DOGS, we directly adopt the corresponding data reported in the original DOGS paper, where training was conducted on five RTX 6000 GPUs with 48GB memory each. For 3DGS and our MixGS, we conduct experiments on a single RTX 3090 GPU paired with an Intel(R) Xeon(R) Gold 6154 CPU @ 3.00GHz, and report the corresponding metrics. The software environment

Table 7. Quantitative results of novel view synthesis on the Mill19 and UrbanScene3D datasets. We report the training time (hh:mm) and the allocated memory (GB). † indicates results obtained without applying decoupled appearance encoding.

Methods	Used GPU	Building		Rubble		Residence		Sci-Art	
		Opt. Time	Mem.	Opt. Time	Mem.	Opt. Time	Mem.	Opt. Time	Mem.
Mega-NeRF [53]	5× RTX6000 48GB	19:49	5.84	30:48	5.88	27:20	5.99	27:39	5.97
Switch-NeRF [74]	5× RTX6000 48GB	24:46	5.84	38:30	5.87	35:11	5.94	34:34	5.92
3DGS [22]	1× RTX3090 24GB	02:39	4.62	02:24	2.18	02:49	3.23	02:03	1.61
VastGaussian† [30]	5× RTX6000 48GB	03:26	3.07	02:30	2.74	03:12	3.67	02:33	3.54
DOGS [10]	5× RTX6000 48GB	03:51	3.39	02:25	2.54	04:33	6.11	04:23	3.53
MixGS (Ours)	1× RTX3090 24GB	14:39	3.25	12:07	2.76	17:09	2.83	14:48	2.24

includes PyTorch version 2.0.1 and CUDA version 11.7. Traditional NeRF-based methods, Mega-NeRF and Switch-NeRF, exhibit significantly longer optimization times and require substantial computational resources. In contrast, Gaussian-based approaches demonstrate considerably lower optimization times and more efficient memory usage. Compared with other methods, MixGS delivers more stable memory consumption across scenes, with memory usage tightly bounded.

Limitations and future work: Currently, due to the single-GPU setup, our method requires more training time compared to multi-GPU approaches. As a future direction, we plan to adopt the open-source distributed training and accelerate training framework[4] to reduce the overall time consumption.

D More Qualitative Results

We provide additional visual comparisons for the Rubble [53], Building [53], Residence [31], and Sci-Art [31] scenes. Our method consistently reconstructs finer details across all these scenes. Notably, MixGS demonstrates superior capability in reconstructing luminance, as exemplified by the Rubble and Building results shown in Fig. 6. For better visualization, we use red arrow to indicate the differences. This highlights the effectiveness of our holistic optimization framework, which prevents local optimum and loss of global information.

Furthermore, we present depth rendering results in Fig. 7. Depth maps are computed using the default depth rendering technique [23] for 3DGS, without employing any depth regularization or additional supervision. For CityGaussian [34], we utilize the official checkpoints. As shown in Fig. 7, our method effectively captures fine-grained geometric details while avoiding floating artifacts. This demonstrates its strong ability to maintain multi-view consistency and preserve appearance information across a large set of images, ultimately enabling large-scale scene reconstruction with highly accurate geometric representation. These results further highlight the potential of our approach for future large-scale mesh reconstruction.

[4]https://github.com/nerfstudio-project/gsplat

Figure 6. Qualitative results of ours and other methods in image rendering on Mill-19 [53] and Urban-scene3D [31] datasets.

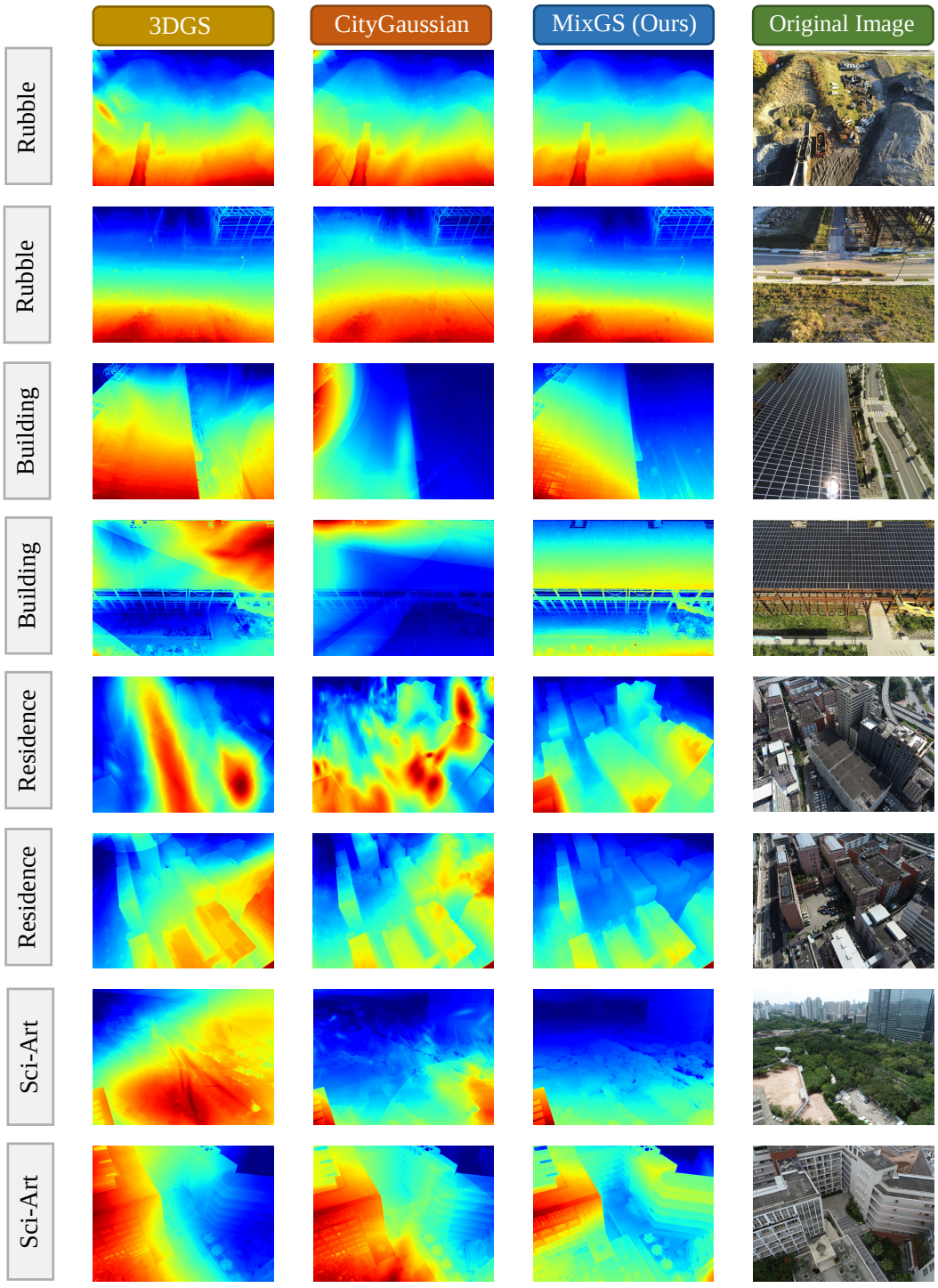

Figure 7. Qualitative results of ours and other methods in depth rendering on Mill-19 [53] and Urbanscene3D [31] datasets.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: We clearly state the claims made, including the contributions made in the paper and important assumptions and limitations.

2. **Limitations**

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

   Answer: [Yes]

   Justification: We create a separate "Limitations" subsection in our paper.

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

   Justification: We clearly state all assumptions and theorems.

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: We provide the full details in the paper and will release the code.

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

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

   Answer: [Yes]

   Justification: The code and data will be included in the project page.

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: We provide the full details in the paper and will release the code.

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: We provide all the detail of training and testing.

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: We provide sufficient information of the computer resources in Implementation Detail.

9. **Code of ethics**

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

Answer: [Yes]

Justification: We read the NeurIPS Code of Ethics.

10. **Broader impacts**

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

    Answer: [NA]

    Justification: To our best knowledge, there is no societal impact of the work performed.

11. **Safeguards**

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

    Answer: [NA]

    Justification: The paper poses no such risks.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: we cite the original paper that produced the code package or dataset.

13. **New assets**

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

    Answer: [Yes]

    Justification: We will includes details about training and license in our code, and the license is Apache License 2.0.

14. **Crowdsourcing and research with human subjects**

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

    Answer: [NA]

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

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

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

    Answer: [NA]

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

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The paper does not involve LLMs in our core method development.