OccluGaussian: Occlusion-Aware Gaussian Splatting for Large Scene Reconstruction and Rendering

Shiyong Liu[1*] Xiao Tang[1*] Zhihao Li[1*] Yingfan He[2]
Chongjie Ye[2] Jianzhuang Liu[3] Binxiao Huang[4] Shunbo Zhou[5] Xiaofei Wu[1†]
[1]Huawei Noah's Ark Lab [2] The Chinese University of Hong Kong (Shenzhen)
[3]Shenzhen Institutes of Advanced Technology [4]The University of Hong Kong
[5]Huawei Embodied Intelligence Lab

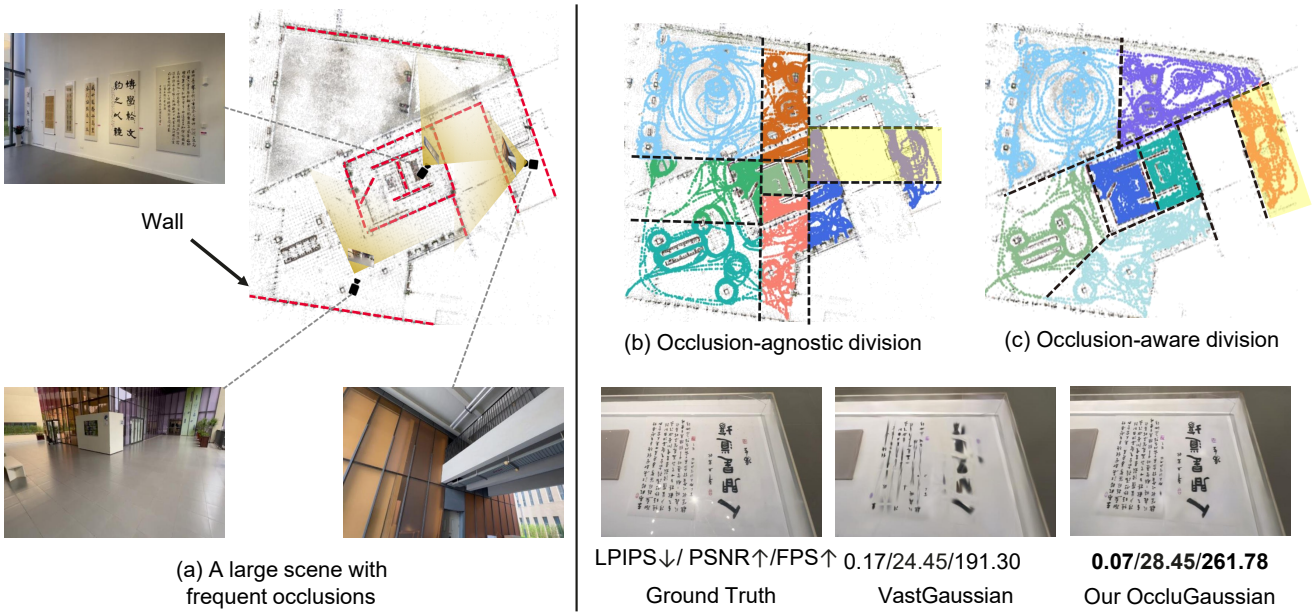

Figure 1. (a) In large scene reconstruction from ground-level captures, there are frequent occlusions such as walls and buildings. (b) Previous scene division methods are occlusion-agnostic, so they produce regions with severe internal occlusions, leading to poor reconstruction results. (c) We propose an occlusion-aware division method to generate regions that better align with the scene layout (see the two regions highlighted in yellow in (b) and (c)), thus improving the reconstruction quality significantly. Besides, we introduce a region-based rendering technique to accelerate the rendering speed of 3D Gaussian splatting in a large scene with massive primitives.

Abstract

In large-scale scene reconstruction using 3D Gaussian splatting, it is common to partition the scene into multiple smaller regions and reconstruct them individually. However, existing division methods are occlusion-agnostic, meaning that each region may contain areas with severe occlusions. As a result, the cameras within those regions are less correlated, leading to a low average contribution to the overall reconstruction. In this paper, we propose an occlusion-aware scene division strategy that clusters training cameras based on their positions and co-visibilities to acquire multiple regions. Cameras in such regions exhibit stronger correlations and a higher average contribution, facilitating high-quality scene reconstruction. We further propose a region-based rendering technique to accelerate large scene rendering, which culls Gaussians invisible to the region where the viewpoint is located. Such a technique significantly speeds up the rendering without compromising quality. Extensive experiments on multiple large scenes show that our method achieves superior reconstruction results with faster rendering speed compared to existing state-of-the-art approaches. Project page: https://occlugaussian.github.io.

* Equal contribution † Corresponding author

1. Introduction

Large scene reconstruction needs to process a huge amount of geometry and appearance information in 2D images to recover 3D structures for novel view synthesis. It is crucial in various applications, such as autonomous navigation [2, 9, 18, 47, 48], cultural heritage preservation [45], and immersive virtual/augmented reality [9, 13, 35], *etc*.

Recent works in large scene reconstruction [22, 28, 32, 47, 49, 51] mostly take radiance fields as the basic 3D representation, *e.g*., Neural Radiance Fields (NeRF) [33] and 3D Gaussian Splatting (3DGS) [21]. NeRF-based approaches struggle to scale for large scenes with rich details, as their implicit representations demand substantial resources for both training and rendering. 3DGS, a primitive-based rasterization technique, also faces scalability issues due to its memory-intensive representation, which quickly exceeds the capacity of high-end GPUs. To address these issues, a divide-and-conquer strategy is commonly applied, which partitions the scene into several smaller and more manageable regions. These regions are then individually reconstructed and finally merged to a complete model. Existing scene division strategies are mainly based on camera positions or point clouds [22, 28, 32, 47, 51]. While these approaches are suitable for occlusion-free scenes like aerial imagery or open spaces, they overlook scene layouts and occlusions, which are commonly present in indoor environments, as shown in Fig. 1. Comparing the two regions yellow highlighted in Fig. 1(b) and Fig. 1(c), the former have fewer training cameras with visible parts in common. The cameras for one area occupy a certain amount of training time and resources, but contribute little to the reconstruction of the other. In other words, putting them together involves too many training cameras whose average contribution to the region's reconstruction is thus significantly reduced.

In this paper, we introduce Occlusion-aware 3D Gaussian splatting (OccluGaussian) for large scene reconstruction and rendering. OccluGaussian incorporates an occlusion-aware scene division strategy that considers scene layout and occlusions. It enhances training efficiency and resource allocation by focusing on cameras that substantially contribute to the reconstruction of specific areas, rather than diluting efforts across less relevant regions, thus finally improving the reconstruction quality. Specifically, we first create an attributed view graph where nodes represent cameras with position features and edges denote their visibility correlations, based on the co-visibilities of the capturing cameras. Next, we apply a graph clustering algorithm to this graph, yielding division results that align with the scene layout, as shown in Fig. 1(c). Training cameras in such regions have stronger correlations, enhancing their average contribution and leading to improved reconstruction results. Finally, we reconstruct these regions individually and merge them together to produce the complete model.

The reconstructed complete model often contains a huge amount of Gaussians, leading to slow rendering speeds. Existing methods [22, 32] often apply the level-of-details (LoD) technique to dynamically prune unnecessary small and distant Gaussians to boost the rendering efficiency. However, these methods are still occlusion-agnostic and process all 3D Gaussians within the view frustum of the rendering camera. The occluded Gaussians that are invisible from the camera can be culled in advance to accelerate rendering without any quality drop. To this end, we further propose a region-based rendering technique to accelerate the rendering of the large scene. Specifically, for each region generated from our scene division strategy, we record the visible 3D Gaussians of all training cameras located in it. During rendering, we process only the recorded 3D Gaussians of the region in which the rendering camera is located. Since our scene division strategy is occlusion-aware, we can effectively prune the occluded Gaussians that are invisible from the camera. This technique achieves much faster rendering without transition artifacts and noticeable loss in quality.

Our contributions are summarized as follows:

- We propose an occlusion-aware scene division strategy that considers the scene layout and camera co-visibilities. The resulting regions barely contain occlusions, and the corresponding training cameras have a higher average contribution, leading to improved reconstruction results.
- We present a region-based rendering technique that accelerates 3D Gaussian splatting in large scenes. It eliminates much of the time-consuming processing of invisible 3D Gaussians, boosting rendering speeds without noticeable quality degradation.
- We conduct extensive experiments on several large-scene datasets, and demonstrate that OccluGaussian achieves superior rendering quality and faster rendering speed compared to previous state-of-the-art methods.

2. Related Work

2.1. Large Scene Reconstruction

3D reconstruction of large scenes from captured images has been a popular research topic for decades. Traditional methods use structure-from-motion (SfM) algorithms to generate sparse point clouds of the scene or further extract dense point clouds and meshes via multiview stereo methods [1, 14, 15, 17, 25, 40, 44, 46, 56]. Recently, Neural Radiance Fields (NeRF) [33, 47, 51, 55] and 3D Gaussian Splatting (3DGS) [6, 21, 28, 29, 31, 32, 50, 54] have been widely applied to large-scale scene reconstruction, as they outperform point clouds and meshes for novel view synthesis. A divide-and-conquer strategy is commonly applied for both NeRF and 3DGS methods. Such as, BlockNeRF [47], as a pioneering work, explicitly splits city-scale scenes into

multiple blocks, then optimizes individual NeRF model for each, and seamlessly combining them by aligning their appearances. Mega-NeRF [49] also uses a grid-based division, and assigns each pixel's ray to different grids that the ray intersects as it passes through the scene when optimizing NeRF models. NeRF-XL [24] distributes the divided blocks across multiple GPUs, allowing to reconstruct arbitrarily large scenes with more parameters and faster speed. As for 3DGS-based methods, VastGaussian [28] uniformly divides the scene and introduces a progressive partitioning strategy to ensure sufficient supervision for each block. CityGaussian [32] splits a large scene in the contracted space to balance the optimization workload for each block. Distributed training systems with multiple GPUs [6, 54] are also explored to achieve faster optimization speed or hold more Gaussian primitives to improve reconstruction quality. However, these division approaches are mainly based on camera positions, and overlook the visibility correlations between cameras, making them occlusion-agnostic. While such methods work well for aerial captures in popular large-scene datasets, they are less suitable for ground-level captures where occlusions occur frequently.

2.2. Camera Clustering

In camera pose estimation of large-scale scenes, partitioning methods are commonly utilized to cluster cameras and segment scenes for parallel processing, which helps to reduce the computational complexity [20, 26, 37, 43, 52]. Some of them [5, 7, 37, 52, 56] use graph cut algorithms to aggregate cameras acoording to their co-visibilities, e.g., Out-of-Core-BA [37] and COLMAP [44] use the Metis graph partitioner [20]. Others [26, 43] use clustering algorithms based on the features of images and 3D key points. However, these methods often adopt only image or matching features, which are not sufficient enough to avoid occlusions within the segmented regions, as shown in Fig. 6. In this paper, we propose a scene division technique that exploits both camera position and co-visibility information, which is occlusion-aware and capable of aligning with the scene layout to conduct partitioning.

2.3. 3DGS Rendering Acceleration

As large scenes contain a substantial number of Gaussian primitives, it is crucial to speed up 3DGS rendering for real-time performance. Recently, several methods [12, 16, 19, 23, 34, 36, 38] have been proposed to compress 3DGS by reducing the number of Gaussian primitives, which also helps accelerate rendering in large scenes. These methods prune 3D Gaussians that contribute minimally to the overall rendering, such as those with low opacities, to minimize the impact on visual quality. Another popular approach for rendering acceleration customizes the rendering for large scenes by using different level-of-details (LoD) at varying distances. For example, Octree-GS [42] incorporates an octree structure to hierarchically organize 3D Gaussians into multiple levels. CityGaussian [32] uses the compression algorithm from LightGaussian [12] to generate different detail levels in a block-wise manner. Hierarchical-3DGS [22] introduces a novel hierarchy for 3DGS, enabling efficient level-of-detail selection and interpolation. From a different perspective compared to existing methods, OccluGaussian proposes a region-base rendering strategy, which culls occluded Gaussians that are invisible to the viewpoint in advance. This significantly reduces redundant computations and further boosts rendering speeds.

3. Method

We propose an occlusion-aware 3DGS framework (OccluGaussian) that achieves superior reconstruction quality and significantly faster rendering speed for large-scale scenes. The overview of OccluGaussian is given in Fig. 2. We first present a novel occlusion-aware scene division strategy based on attributed graph clustering in Sec. 3.1. Next, Sec. 3.2 details the optimization of each individual region and the merging process to obtain a complete model. Finally, we introduce a region-based rendering technique to accelerate rendering in large scenes in Sec. 3.3.

3.1. Occlusion-Aware Scene Division

Previous division methods [22, 28, 32, 47, 49, 51] typically ignore the scene layout and divide the scene uniformly based on camera positions or point clouds. A more effective division strategy should consider the occlusions in the scene, so that the cameras in each region have strong correlations and a higher average contribution to the reconstruction. Here, we introduce an occlusion-aware scene division strategy, as illustrated in the top-left part of Fig. 2.

Attributed view graph. First, we create a view graph of n posed training cameras, which is an undirected attributed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, X)$, where $\mathcal{V}$, $\mathcal{E}$ and X denote the nodes, the edges with weights, and the feature matrix, respectively. Each node in $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ corresponds to a camera. To derive the edges $\mathcal{E}$, we directly utilize the structure-from-motion results obtained during the camera pose estimation. An edge is established between two cameras if they share some visual contents in common, and we set its weight as the number of matched feature points between these two cameras. The set of edges $\mathcal{E}$ is finally represented as an adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{n \times n}$. The feature matrix $X = [\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}]^\top \in \mathbb{R}^{n \times d}$ contains the 3D coordinates of each camera with positional encoding [33]. Intuitively, occluded or distant cameras typically share minimal overlapped views, which can be distinguished in our occlusion-aware view graph.

View graph clustering. Then, we split the view graph $\mathcal{G}$ into multiple parts using graph clustering. We employ a

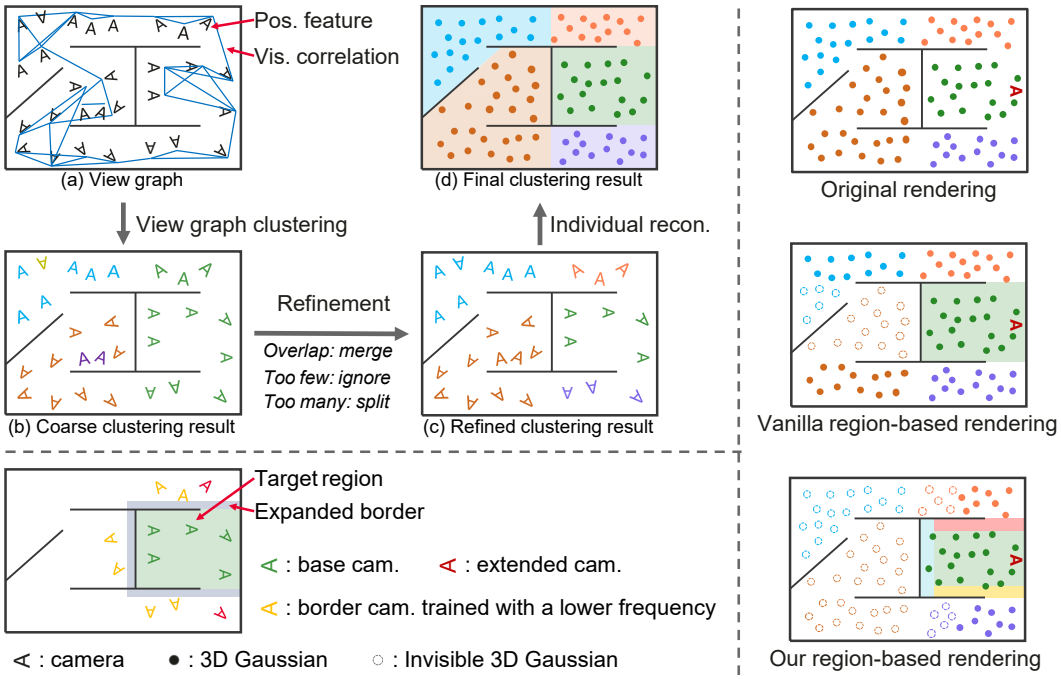

Figure 2. **Overview of OccluGaussian. Top left:** To reconstruct a large scene, we divide it into multiple regions by adopting an occlusion-aware scene division strategy. (a) We first create an attributed view graph from the posed cameras, where nodes represent cameras with positional features, and edges represent visibility correlations between them. (b) A graph clustering algorithm is applied to the view graph to cluster the cameras into multiple regions, and (c) we further refine them to obtain more balanced sizes. (d) The region boundaries are calculated based on the clustered cameras. Each region is individually reconstructed and finally merged into a complete model. **Bottom left:** Each region is reconstructed using three sets of training cameras: base cameras located inside the region, extended cameras providing adequate visual content of the region, and border cameras used to constrain Gaussian primitives near the boundaries. **Right:** We introduce a region-based rendering technique, which culls 3D Gaussians that are occluded from the region where the rendering viewpoint is located. Furthermore, we subdivide the scene into smaller sub-regions with fewer essential 3D Gaussians. This approach effectively reduces redundant computations and further boosts our rendering speed.

attributed graph clustering algorithm [53], which performs graph convolution to generate smooth feature representations, followed by spectral clustering on the resulting features to group the nodes. The clustering process is outlined as follows:

Given the adjacency matrix A and the degree matrix $D = \text{diag}(d_1, \cdots, d_n)$, $d_i = \sum_{j=1}^{n} a_{ij}$, the symmetrically normalized graph Laplacian is defined as:

$$L_s = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}. \tag{1}$$

Graph convolution is defined as the multiplication of a graph signal f with a graph filter G:

$$\bar{\boldsymbol{f}} = G \cdot \boldsymbol{f}, \quad G = (I - \frac{1}{2} L_s)^r, \tag{2}$$

where $\bar{\boldsymbol{f}}$ is the filtered graph signal, and r is a positive integer. Each column of the feature matrix X can be considered as a graph signal. By performing graph convolution on the feature matrix X, we obtain the filtered feature matrix $\bar{X} = GX$. Then, we calculate the similarity matrix $\bar{W}$:

$$\bar{W} = \frac{1}{2}(|H| + |H^T|), \quad H = \bar{X}\bar{X}^T. \tag{3}$$

Since $\bar{W}$ indicates the closeness between nodes in $\mathcal{V}$, we apply spectral clustering algorithm [39] on $\bar{W}$ to cluster $\mathcal{V}$, which correspond to the training cameras. Cameras that share a large overlapping view or are spatially close are clustered into the same region, as shown in Fig. 2(b).

Determining the clustering number. Most clustering algorithms need a predefined number of clusters. To address this, we propose an adaptive approach to determine the optimal number of clusters. The process starts by selecting an initial cluster number K, enabling graph clustering to generate K preliminary clusters. We then refine them by splitting clusters that contain too many cameras by further applying graph clustering, and by ignoring any cluster that either has too few cameras or whose convex hull is entirely covered by the convex hull of another cluster. These refinement steps are applied recursively until all clusters achieve a balanced number of cameras, thereby determining the optimal number of clusters for different scenes, (see Fig. 2(c)).

Boundary calculation. Finally, we calculate explicit boundaries of each region based on its clustered cameras. These boundaries are crucial for border expansion (Sec. 3.2)

and region-based rendering (Sec. 3.3). We use a linear classification method [8] to derive decision functions as boundary lines. Thanks to the occlusion-aware camera clustering, these boundary lines closely align with the scene layout.

3.2. Individual Region Reconstruction

After the occlusion-aware scene division, we reconstruct each region individually and finally merge them to form the complete model. In this subsection, we detail the strategy for selecting training cameras for each region and the algorithm used to seamlessly merge these regions.

Training camera selection. As pointed out in [28, 32], it is crucial to select appropriate training cameras for 3D reconstruction. Too few training cameras cannot provide sufficient supervision, while too many irrelevant cameras will waste computational resources and reduce the average contribution of every camera. We select three sets of training cameras as shown in the bottom-left part of Fig. 2: 1) The base set, whose cameras are located within the region. 2) The extended set, whose cameras are outside the region but capture adequate visible content of it. We follow [32] to select the extended cameras based on their visibility contribution to the region. 3) The border set, whose cameras face the region but are occluded. These cameras help constrain Gaussian primitives near the boundaries for the final seamless merging. Without the border set, these Gaussian primitives can become very large or elongated, resulting in floaters or artifacts (see the qualitative results in the supplementary material). Compared to existing methods, our occlusion-aware scene division strategy requires fewer extended cameras for adequate supervision to reconstruct the target region, increasing the average contribution per camera and thereby improving reconstruction quality.

Seamless region merging. We optimize 3D Gaussians for each region individually using its selected training cameras. And then, we remove all Gaussian primitives from each region that lie outside the region to create sharp borders. Finally, these regions are merged to form a complete model.

3.3. Region-Based Rendering

Large scenes contain a substantial number of 3D Gaussians, making rendering computationally expensive and slow. Considering that many Gaussian primitives in the scene are occluded, invisible to the viewpoint, culling them in advance can significantly reduce the processing load, thereby accelerating rendering speeds without sacrificing visual quality. Therefore, leveraging our occlusion-aware scene division, we propose a region-based rendering strategy that culls invisible Gaussians for efficient rendering in large scenes, as shown in the right part of Fig. 2.

Region-based visibility calculation. Consider a large scene reconstructed by N_g 3D Gaussians and divided into k regions $\{R_i\}_{i=1}^{k}$. For each region R_j, we define a visibility mask $M_j = \{m_i^j \in \{0,1\}\}_{i=1}^{N_g}$ for all N_g 3D Gaussians, indicating their visibility from any training viewpoint in R_j. To derive M_j, we perform 3DGS rasterization for each camera that is clustered into R_j, and calculate each 3D Gaussian i's maximal accumulated weight as its contribution to the rendered image. We mark $m_i^j = 1$ if 3D Gaussian i's contribution to any pixel exceeds 0.01. Note that we generate both the original view and the backside view from the camera position to comprehensively collect visible 3D Gaussians for each region. After iterating over all cameras in R_j, Gaussians with $m_i^j = 0$ are treated as *occluded* and can be culled in advance when rendering.

Region subdivision. Viewpoints near region borders often observe too many 3D Gaussians from neighboring regions, leading to redundant computations when rendering the interior of a region. To address this issue, we further subdivide each region into smaller sub-regions when deriving visibility masks: one interior sub-region, and several border sub-regions. First, we identify the boundary lines between the region and its neighboring regions. Then, we shrink the boundary lines inward by $0.1d_{max}$, where d_{max} is the maximal distance between two cameras within the region, to obtain the corresponding border sub-regions. The remaining area in the region, excluding all the border sub-regions, forms the interior sub-region. Our region subdivision strategy leads to more compact region-based visibility masks, and thus further boosts the rendering speed.

Rendering with region-based culling. To render at an arbitrary viewpoint in the complete model, we first identify the region R_j where the viewpoint belongs to, then use the corresponding visibility mask M_j to cull the occluded 3D Gaussians, and finally perform 3DGS rasterization for the remaining 3D Gaussians to generate the rendering image. By knowing the occlusions between regions, our culling strategy retains only the minimal set of necessary 3D Gaussians for rendering within each region. This approach leads to significant rendering speedup without noticeable quality drop, as shown in the supplementary material.

4. Experiments

4.1. Experimental Setup

Dataset. Most existing large-scene datasets [27, 30] are primarily captured from aerial perspectives and thus with rare occlusions. Therefore, we construct a new dataset called OccluScene3D, containing three large-scale scenes in a campus with complex layouts and significant occlusions: GALLERY, CANTEEN and CLASSBUILDING. More details of OccluScene3D are in the supplementary material. We also assess our method on the Zip-NeRF dataset [4], including four large-scale scenes such as apartments and houses. To show the generality of our method, we further conduct experiments on occlusion-free scenes in the Mill-19 [49]

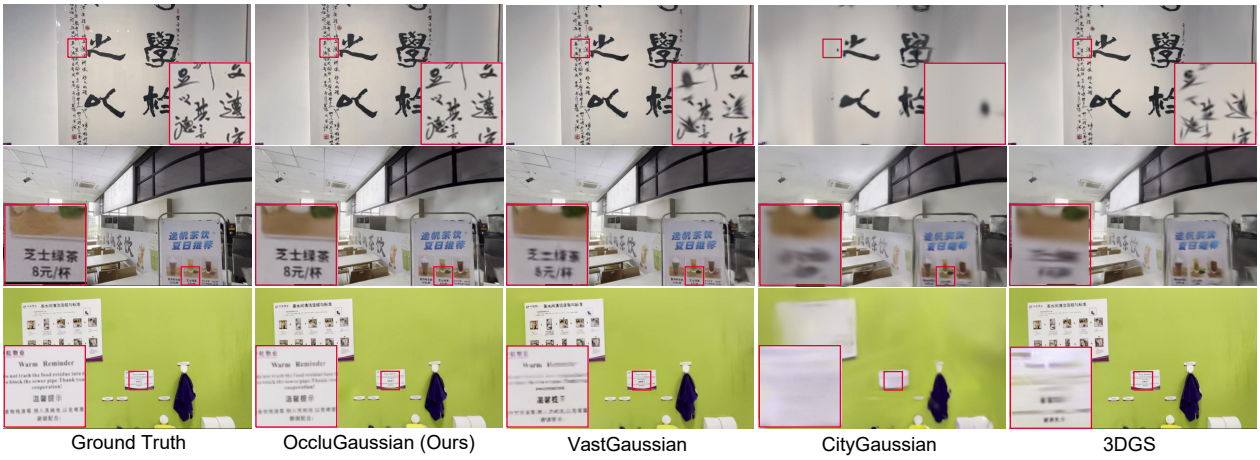

Figure 3. Qualitative comparison with SOTA methods on three large scenes in the OccluScene3D dataset.

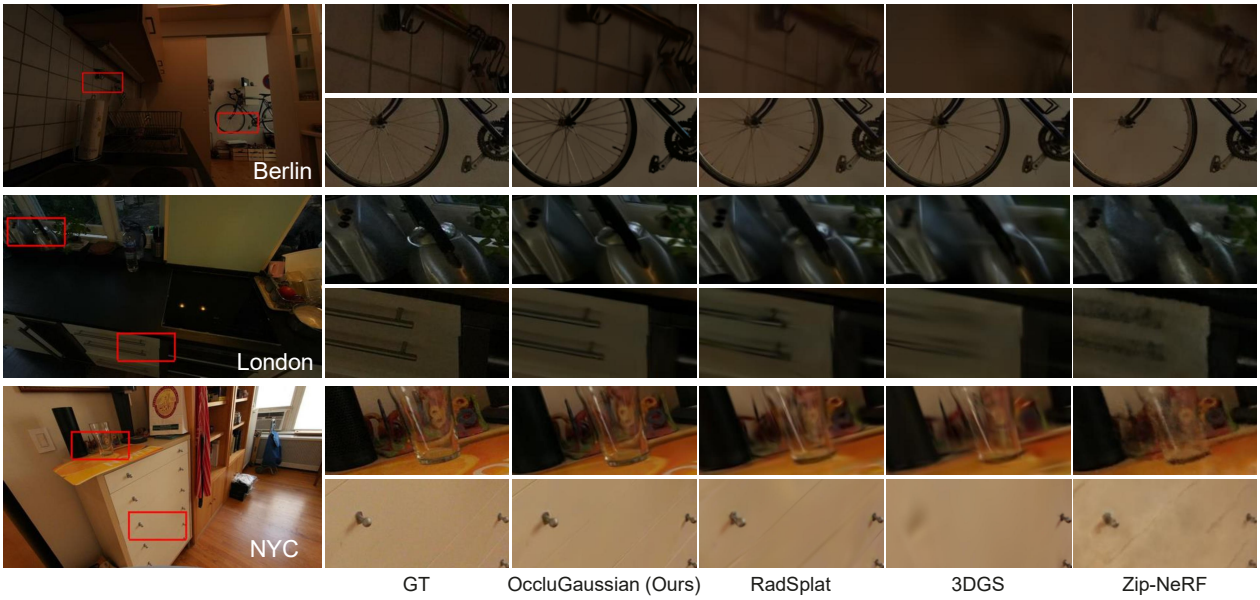

Figure 4. Qualitative comparison with SOTA methods on three large scenes in the Zip-NeRF dataset.

and UrbanScene3D [30] datasets.

Implementation details. The models are optimized for 90,000 iterations with densification from iteration 1,500 to 45,000 at intervals of 100. We adopt the appearance modeling in VastGaussian [28] to fit appearance variations across captured images. For clustering the view graph, we set a fixed initial cluster number $K = 10$. We iteratively refine the number of clusters until the camera count in each cluster falls within $[M_c - \sigma_c M_c, M_c + \sigma_c M_c]$, where $\sigma_c = 0.5$ and M_c is the average camera count across all clusters.

Baselines. On the OccluScene3D dataset, we compare with scalable 3DGS-based methods [21, 22, 28, 32]. Since VastGaussian [28] is not open-sourced, we re-implement its scene division strategy and keep all other hyperparameters the same as ours during 3DGS optimization. For the Zip-NeRF dataset, we compare with methods that have evaluated on it [4, 10, 38, 41]. Following the evaluation protocol in RadSplat [38], we optimize two models: one for visualization, and another for quantitative analysis, excluding appearance modeling. We do not use the monocular depth prior in Hierarchical-GS [22] for fairness.

Metrics. We evaluate the rendering quality using three commonly-used quantitative metrics: SSIM, PSNR and LPIPS. We also report the average rendering speed at 1080p resolution in frames per second (FPS).

4.2. Result Analysis

Reconstruction quality and rendering speeds. We present quantitative results for occluded scene reconstruction in Tab. 1 and Tab. 2. We are unable to evaluate

Scene	GALLERY				CANTEEN				CLASSBUILDING			
Metrics	PSNR	SSIM	LPIPS	FPS	PSNR	SSIM	LPIPS	FPS	PSNR	SSIM	LPIPS	FPS
VastGaussian* [28]	<u>25.09</u>	**0.903**	<u>0.095</u>	215.22	<u>24.60</u>	<u>0.890</u>	<u>0.105</u>	211.02	<u>24.05</u>	<u>0.884</u>	<u>0.111</u>	269.97
CityGaussian [32]	21.98	0.808	0.294	119.86	20.41	0.794	0.275	54.02	20.48	0.840	0.244	65.57
Hierarchical-GS [22]	22.23	0.800	0.182	216.00	22.71	0.825	0.178	199.33	23.87	0.881	0.128	198.58
3DGS [21]	21.36	0.843	0.213	**344.92**	21.86	0.847	0.183	**525.93**	19.41	0.871	0.186	**395.13**
OccluGaussian	**25.81**	**0.903**	**0.094**	<u>288.94</u>	**25.25**	**0.900**	**0.100**	<u>311.59</u>	**25.33**	**0.921**	**0.083**	<u>339.64</u>

Table 1. Quantitative comparison on the OccluScene3D dataset. We report SSIM ↑, PSNR ↑, LPIPS ↓ and FPS ↑ on the test views. The **best** and <u>second best</u> results are highlighted. * denotes that it is our re-implementation of VastGaussian.

Metrics	PSNR	SSIM	LPIPS
MERF [41]	23.49	0.747	0.445
SMERF [10]	27.28	0.829	0.340
Zip-NeRF [4]	27.37	0.836	0.305
3DGS [21]	25.50	0.809	0.369
RadSplat [38]	26.17	0.839	0.364
OccluGaussian	**28.63**	**0.880**	**0.281**

Table 2. Quantitative comparison on the Zip-NeRF dataset.

Scene	MILL-19			URBANSCENE3D		
Metrics	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Mega-NeRF [49]	23.09	0.572	0.393	24.35	0.659	0.369
Switch-NeRF [55]	23.50	0.590	0.355	24.84	0.678	0.333
Grid-NeRF [51]	24.37	0.807	0.142	24.94	0.787	0.158
Modified 3DGS [28]	24.90	0.785	0.163	24.18	0.793	0.199
CityGaussian [32]	24.29	0.771	0.166	23.87	0.821	0.161
Hierarchical-GS [22]	22.95	0.739	0.291	-	-	-
DOGS [6]	24.26	0.762	0.231	23.40	0.742	0.280
VastGaussian [28]	25.21	0.814	0.131	25.69	0.851	0.132
OccluGaussian	**25.97**	**0.854**	**0.103**	**25.75**	**0.858**	**0.125**

Table 3. Quantitative comparison on aerial capture datasets, Mill-19 and UrbanScene3D. We fail to test Hierarchical-GS [22] on the URBANSCENE3D dataset due to an out-of-memory issue.

DOGS [6] on the OccluScene3D and Zip-NeRF datasets due to insufficient GPU resources required from its distributed optimization strategy. OccluGaussian almost significantly outperforms the compared methods in all metrics, especially in PSNR. Regarding FPS, our method is faster than all methods except the original 3DGS [21], due to its less detailed reconstruction and fewer Gaussian primitives. Visual comparisons in Fig. 3 and Fig. 4 illustrate OccluGaussian's superior detail in rendering. Additionally, Fig. 5 shows OccluGaussian's scene division aligns better with the scene layout. More visualizations are provided in the supplementary material.

In Tab. 3, we validate our method on occlusion-free scenes using the Mill-19 [49] and UrbanScene3D [30] datasets. Our method maintains a clear advantage, highlighting the generality of our approach. These results show the effectiveness of our scene division strategy, which enhances the correlation among training cameras within each region and improves reconstruction quality.

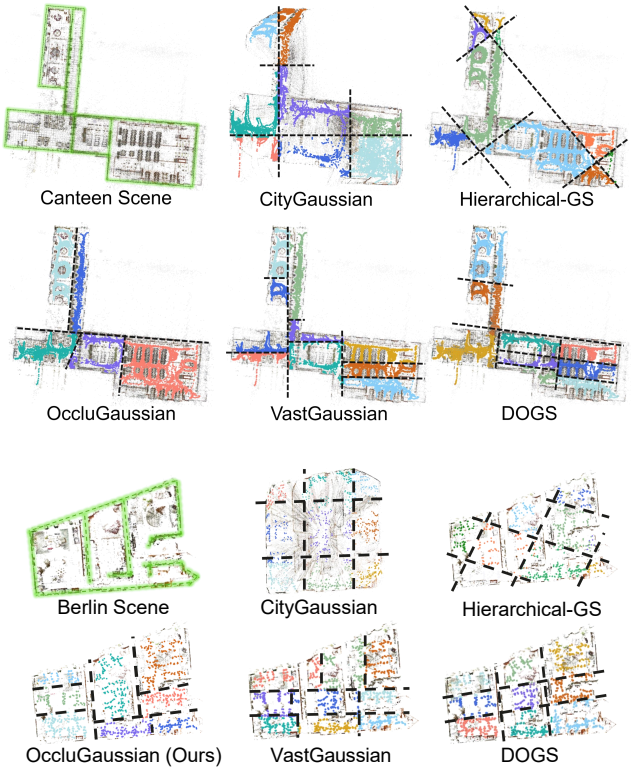

Figure 5. Different division results on the CANTEEN scene (upper) and BERLIN scene (lower). Green lines denote the physical walls, while black lines denote the boundaries of the divided regions. Notably, CityGaussian's division is projected onto a contracted space.

4.3. Comparison of Camera Clustering Methods

We compare various camera clustering methods on the GALLERY scene in the OccluScene3D dataset. As shown in Fig. 6, grid-based methods partition scenes into grids without considering clustering features and occlusions. The K-means algorithm, which relies solely on camera position features, is also oblivious to occlusions and performs worse when incorporating image features extracted by NetVLAD [3], failing to extract region boundaries. DBSCAN [11] does not require an initial cluster number, but

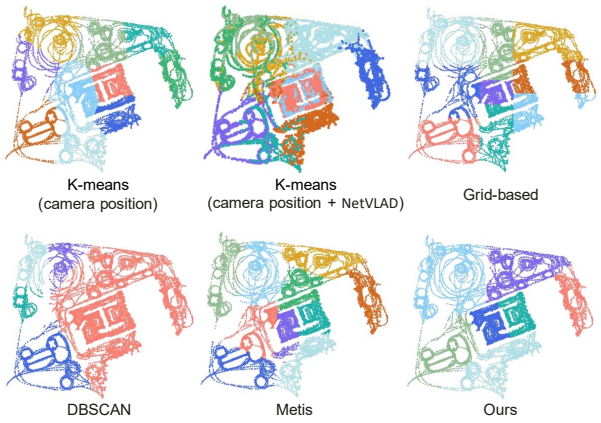

Figure 6. Scene division results by different methods are presented with points for cameras, color-coded by their respective regions.

Method	PSNR↑	SSIM↑	LPIPS↓
Grid-based	24.58	0.892	0.104
K-means (CamP)	24.94	0.903	0.094
K-means (CamPN)	-	-	-
Metis [20]	25.02	0.903	0.097
DBSCAN [11]	25.00	0.898	0.111
Ours	**25.46**	**0.908**	**0.094**

Table 4. Comparison of clustering methods. 'CamP' denotes camera position, 'CamPN' denotes camera position + NetVLAD. K-means (CamPN) fails due to its poor clustering results that prevent effective distinctions by the classification model.

	PSNR	SSIM	LPIPS	FPS
w/o RBR & RSD	25.81	0.903	0.099	189.52
w/o RSD	25.81	0.903	0.099	271.79
Full	**25.81**	**0.903**	**0.099**	**288.94**

Table 5. Ablation study of the region-based rendering on the GALLERY scene. 'RBR' denotes vanilla region-based rendering; 'RSD' denotes region subdivision in our region-based rendering.

produces unevenly sized clusters and also lacks occlusion awareness. Metis [20] provides slightly better results by conducting graph clustering, but severe occlusions persist in the partitioned regions. We further perform 3DGS optimization based on these clustering results, as shown in Tab. 4, which demonstrates that our clustering method outperforms others in all metrics.

4.4. Ablation Study

Region-based rendering. We ablate our region-based rendering on the GALLERY scene, as shown in Tab. 5. Our region-based culling strategy significantly enhances rendering speeds without noticeable loss in visual quality. Additionally, the proposed region subdivision technique further accelerates rendering. It is important to note that our culling strategy automatically removes Gaussian primitives with minimal contribution, reducing the overall number of Gaussian primitives in the final reconstructed model.

	Canteen			Berlin		
	PSNR	LPIPS	#Blocks	PSNR	LPIPS	#Blocks
VastGaussian [28]	24.60	0.105	9	29.48	0.085	9
CityGaussian [32]	24.16	0.114	9	28.68	0.091	9
Hierarchical-GS [22]	23.68	0.131	9	27.26	0.105	9
DOGS [6]	24.66	0.108	9	28.15	0.095	9
OccluGaussian	**25.25**	**0.100**	5	**30.37**	**0.076**	8

Table 6. Different division strategies used in OccluGaussian.

Initial K	Final K	PSNR	SSIM	LPIPS
7	6	30.90	0.898	0.126
10	7	31.33	0.902	0.121
15	8	31.35	0.901	0.121

Table 7. Different initial clustering numbers K on the NYC scene of Zip-NeRF dataset.

Division strategy. To demonstrate the effectiveness of our occlusion-aware scene division, we evaluate it alongside VastGaussian, CityGaussian, Hierarchical-GS, and DOGS by replacing our scene division with theirs while keeping the same hyperparameters as ours for 3DGS optimization. We test on the CANTEEN scene of OccluScene3D dataset and the BERLIN scene of Zip-NeRF. Results in Tab. 6 shows that our occlusion-aware scene division still demonstrates a significant advantage. Additional visualizations can be found in the supplementary material.

Initial clustering numbers. To show the robustness of our method, we vary the manually-set initial clustering number K on the NYC scene of Zip-NeRF dataset and find similar performance trends, as shown in Tab. 7. However, for huge scenes with dozens or hundreds of separated regions, setting an appropriate initial clustering number K remains crucial, which we leave for future exploration.

5. Conclusion

This paper presents OccluGaussian, a novel approach for high-quality reconstruction and real-time rendering of large-scale scenes. We propose an occlusion-aware scene division strategy that enhances reconstruction quality by optimizing the contribution of training cameras within each region. Furthermore, we propose a region-based rendering strategy that discards occluded 3D Gaussians, significantly accelerating rendering for large-scale scenes.

Limitation. While OccluGaussian has made significant progress, limitations remain. For instance, our camera clustering starts with a fixed initial number, which works well in our experiments but may be insufficient for extremely large scenes. Future work will focus on dynamically determine the initial clustering number based on the number of training cameras to enhance robustness and scalability.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *CACM*, 2011. 2

[2] Nurnajmin Qasrina Ann, MS Hendriyawan Achmad, Luhur Bayuaji, M Razali Daud, and Dwi Pebrianti. Study on 3D scene reconstruction in robot navigation using stereo vision. In *I2CACIS*, 2016. 2

[3] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: Cnn architecture for weakly supervised place recognition. In *CVPR*, 2016. 7

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-NeRF: Anti-aliased grid-based neural radiance fields. In *ICCV*, 2023. 5, 6, 7

[5] Brojeshwar Bhowmick, Suvam Patra, Avishek Chatterjee, Venu Madhav Govindu, and Subhashis Banerjee. Divide and conquer: Efficient large-scale structure from motion using graph partitioning. In *ACCV*, 2015. 3

[6] Yu Chen and Gim Hee Lee. DOGS: Distributed-oriented gaussian splatting for large-scale 3d reconstruction via gaussian consensus. *NeurIPS*, 37:34487–34512, 2025. 2, 3, 7, 8

[7] Yu Chen, Shuhan Shen, Yisong Chen, and Guoping Wang. Graph-based parallel large scale structure from motion. *PR*, 2020. 3

[8] Corinna Cortes. Support-vector networks. *Machine Learning*, 1995. 5

[9] Anurag Dalal, Daniel Hagen, Kjell G Robbersmyr, and Kristian Muri Knausgård. Gaussian splatting: 3d reconstruction and novel view synthesis: A review. *IEEE Access*, 2024. 2

[10] 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. *TOG*, 2024. 6, 7

[11] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1996. 7, 8

[12] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, Zhangyang Wang, et al. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *NeurIPS*, 2024. 3

[13] Ben Fei, Jingyi Xu, Rui Zhang, Qingyuan Zhou, Weidong Yang, and Ying He. 3D gaussian splatting as new era: A survey. *TVCG*, 2024. 2

[14] Christian Früh and Avideh Zakhor. An automated method for large-scale, ground-based city model acquisition. *IJCV*, 2004. 2

[15] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *CVPR*, 2010. 2

[16] Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In *ECCV*. Springer, 2024. 3

[17] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M Seitz. Multi-view stereo for community photo collections. In *ICCV*, 2007. 2

[18] Lei He, Leheng Li, Wenchao Sun, Zeyu Han, Yichen Liu, Sifa Zheng, Jianqiang Wang, and Keqiang Li. Neural radiance field in autonomous driving: A survey. *arXiv preprint arXiv:2404.13816*, 2024. 2

[19] Yuheng Jiang, Zhehao Shen, Penghao Wang, Zhuo Su, Yu Hong, Yingliang Zhang, Jingyi Yu, and Lan Xu. Hifi4G: High-fidelity human performance rendering via compact gaussian splatting. In *CVPR*, 2024. 3

[20] George Karypis and Vipin Kumar. METIS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices. 1997. 3, 8

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *TOG*, 2023. 2, 6, 7

[22] 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. *TOG*, 2024. 2, 3, 6, 7, 8

[23] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *CVPR*, 2024. 3

[24] Ruilong Li, Sanja Fidler, Angjoo Kanazawa, and Francis Williams. NeRF-XL: Scaling nerfs with multiple gpus. In *ECCV*. Springer, 2024. 3

[25] Xiaowei Li, Changchang Wu, Christopher Zach, Svetlana Lazebnik, and Jan-Michael Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *ECCV*, 2008. 2

[26] Yunpeng Li, Noah Snavely, and Daniel P Huttenlocher. Location recognition using prioritized feature matching. In *ECCV*, 2010. 3

[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 *ICCV*, 2023. 5

[28] 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 *CVPR*, 2024. 2, 3, 5, 6, 7, 8

[29] Jiaqi Lin, Zhihao Li, Binxiao Huang, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Xiaofei Wu, Fenglong Song, and Wenming Yang. Decoupling appearance variations with 3d consistent features in gaussian splatting. In *AAAI*, 2025. 2

[30] Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, reconstructing, and simulating: the urbanscene3d dataset. In *ECCV*, 2022. 5, 6, 7

[31] Jiayue Liu, Xiao Tang, Freeman Cheng, Roy Yang, Zhihao Li, Jianzhuang Liu, Yi Huang, Jiaqi Lin, Shiyong Liu, Xiaofei Wu, et al. Mirrorgaussian: Reflecting 3d gaussians for reconstructing mirror reflections. In *ECCV*, 2024. 2

[32] 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 *ECCV*. Springer, 2024. 2, 3, 5, 6, 7, 8

[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. *CACM*, 2021. 2, 3

[34] Wieland Morgenstern, Florian Barthel, Anna Hilsmann, and Peter Eisert. Compact 3D scene representation via self-organizing gaussian grids. In *ECCV*. Springer, 2024. 3

[35] Michela Mortara, Chiara Eva Catalano, Francesco Bellotti, Giusy Fiucci, Minica Houry-Panchetti, and Panagiotis Petridis. Learning cultural heritage by serious games. *Journal of Cultural Heritage*, 2014. 2

[36] KL Navaneet, Kossar Pourahmadi Meibodi, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Compact3D: Compressing Gaussian splat radiance field models with vector quantization. *arXiv preprint arXiv:2311.18159*, 2023. 3

[37] Kai Ni, Drew Steedly, and Frank Dellaert. Out-of-core bundle adjustment for large-scale 3D reconstruction. In *ICCV*, 2007. 3

[38] Michael Niemeyer, Fabian Manhardt, Marie-Julie Rakotosaona, Michael Oechsle, Daniel Duckworth, Rama Gosula, Keisuke Tateno, John Bates, Dominik Kaeser, and Federico Tombari. Radsplat: Radiance field-informed Gaussian splatting for robust real-time rendering with 900+ FPS. *3DV*, 2025. 3, 6, 7

[39] Pietro Perona and William Freeman. A factorization approach to grouping. In *ECCV*, 1998. 4

[40] Marc Pollefeys, David Nistér, J-M Frahm, Amir Akbarzadeh, Philippos Mordohai, Brian Clipp, Chris Engels, David Gallup, S-J Kim, Paul Merrell, et al. Detailed real-time urban 3D reconstruction from video. *IJCV*, 2008. 2

[41] 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. *TOG*, 2023. 6, 7

[42] 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. *PAMI*, 2025. 3

[43] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In *ECCV*, 2012. 3

[44] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2, 3

[45] Roberto Scopigno, Paolo Cignoni, Nico Pietroni, Marco Callieri, and Matteo Dellepiane. Digital fabrication techniques for cultural heritage: a survey. In *Computer graphics forum*, 2017. 2

[46] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3D. In *ACM siggraph*. 2006. 2

[47] 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 *CVPR*, 2022. 2, 3

[48] Sebastian Thrun et al. Robotic mapping: A survey. 2002. 2

[49] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-NeRF: Scalable construction of large-scale NeRFs for virtual fly-throughs. In *CVPR*, 2022. 2, 3, 5, 7

[50] Zipeng Wang and Dan Xu. PyGS: Large-scale scene representation with pyramidal 3D Gaussian splatting. *arXiv preprint arXiv:2405.16829*, 2024. 2

[51] 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 *CVPR*, 2023. 2, 3, 7

[52] Runze Zhang, Siyu Zhu, Tian Fang, and Long Quan. Distributed very large scale bundle adjustment by global camera consensus. In *ICCV*, 2017. 3

[53] Xiaotong Zhang, Han Liu, Qimai Li, Xiao-Ming Wu, and Xianchao Zhang. Adaptive graph convolution methods for attributed graph clustering. *KDE*, 2023. 4

[54] Hexu Zhao, Haoyang Weng, Daohan Lu, Ang Li, Jinyang Li, Aurojit Panda, and Saining Xie. On scaling up 3d gaussian splatting training. In *ECCV*. Springer, 2024. 2, 3

[55] MI Zhenxing and Dan Xu. Switch-NeRF: Learning scene decomposition with mixture of experts for large-scale neural radiance fields. In *ICLR*, 2022. 2, 7

[56] Siyu Zhu, Runze Zhang, Lei Zhou, Tianwei Shen, Tian Fang, Ping Tan, and Long Quan. Very large-scale global sfm by distributed motion averaging. In *CVPR*, 2018. 2, 3