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

# Prune Wisely, Reconstruct Sharply: Compact 3D Gaussian Splatting via Adaptive Pruning and Difference-of-Gaussian Primitives

Haoran Wang, Guoxi Huang, Fan Zhang, David Bull, and Nantheera Anantrasirichai
School of Computer Science, University of Bristol, Bristol, UK
{yp22378,guoxi.huang,fan.zhang,dave.bull,n.anantrasirichai}@bristol.ac.uk

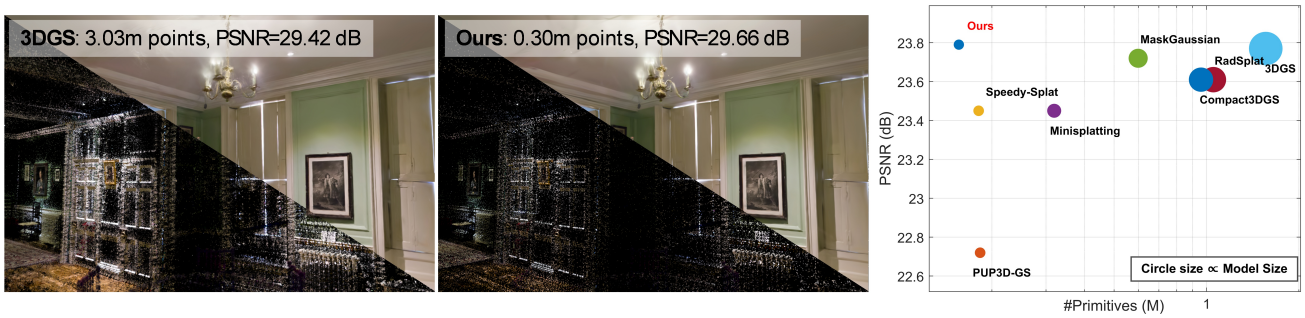



Figure 1. Performance improvement of the proposed pruning strategy and 3D Difference-of-Gaussian (DoG) primitives. Comparison on the Drjohnson scene, where the original 3DGS method is shown on the left and our result in the middle panel. The right panel shows PSNR versus the number of primitives on the Tanks and Temples dataset [24] across various 3DGS-based methods.

## Abstract

*Recent significant advances in 3D scene representation have been driven by 3D Gaussian Splatting (3DGS), which has enabled real-time rendering with photorealistic quality. 3DGS often requires a large number of primitives to achieve high fidelity, leading to redundant representations and high resource consumption, thereby limiting its scalability for complex or large-scale scenes. Consequently, effective pruning strategies and more expressive primitives that can reduce redundancy while preserving visual quality are crucial for practical deployment. We propose an efficient, integrated reconstruction-aware pruning strategy that adaptively determines pruning timing and refining intervals based on reconstruction quality, thus reducing model size while enhancing rendering quality. Moreover, we introduce a 3D Difference-of-Gaussians primitive that jointly models both positive and negative densities in a single primitive, improving the expressiveness of Gaussians under compact configurations. Our method significantly improves model compactness, achieving up to a 90% reduction in Gaussian count while delivering visual quality comparable to or better than that of state-of-the-art methods. Code available at: https://github.com/WangHaoran16/Prune-Wisely-Reconstruct-Sharply.*

## 1. Introduction

Novel View Synthesis (NVS) aims to generate photorealistic images from an unseen viewpoint based on a limited set of input observations. This research topic was initially dominated by NeRF-based methods [7, 12, 28, 30, 38], which implicitly represent a scene as a continuous radiance field and estimate volume density and colors using neural networks. While these methods have attracted significant attention and supported a wide range of applications [2], they commonly suffer from computational overhead and render very slowly, making them unsuitable for real-time applications.

3D Gaussian Splatting (3DGS) [23] has recently emerged as an alternative efficient and high-quality representation for 3D scene reconstruction and novel view synthesis. By explicitly modeling a scene as a set of spatially distributed Gaussian primitives, 3DGS achieves real-time photorealistic rendering while maintaining differentiability for optimization. Due to these advantages, 3DGS has rapidly become the dominant approach in 3D reconstruction. Nonetheless, achieving high-quality reconstruction typically requires a large number of Gaussian primitives, many of which are redundant or only contribute marginally to the final render [10]. This redundancy results in unnecessary memory consumption and computational burden, significantly limiting scalability for large or complex scenes.

To improve the efficiency of 3DGS, several pruning approaches have been proposed [8, 9, 14]. Although these have achieved promising results, most of them prune at fixed training iterations and use uniform refinement intervals, hence disregarding the dynamic nature of the reconstruction process. Such inflexible schedules often lead to unstable optimization. Early pruning may remove necessary primitives, while late pruning usually provides little efficiency gain. To address these issues, we propose a reconstruction-aware pruning framework that adaptively determines when to prune based on reconstruction quality. Instead of pruning in predetermined iterations, our method analyzes reconstruction quality and automatically identifies the optimal pruning time, adjusting the pruning ratio as the procedure progresses. This adaptive mechanism enables the model to maintain stability during training and progressively achieve compact representations. To retain reconstruction quality after compression, we also propose a Spatio-spectral Pruning Score that measures the importance of a Gaussian primitive in both spatial and spectral domains.

A second key factor that limits the performance of compact models lies in the nature of 3D Gaussian primitives. It is difficult for a small number of smooth Gaussian kernels to capture fine details accurately. Consequently, we introduce a 3D Difference-of-Gaussians (3D-DoG) primitive, a novel variant of the standard Gaussian that models both positive and negative spatial responses. Its positive-density lobe still contributes to rendering in the same way as normal 3D Gaussians, while its negative-density component can implement color subtraction in $\alpha$-blending. Compared to the original primitive, the 3D-DoG is more responsive to fine geometric details and edges, thereby preserving sharper structures under compact configurations. Figure 1 shows significant improvements with the proposed framework.

Our main contributions are summarized as follows:

1) Reconstruction-aware Pruning Scheduler (RPS). We propose a novel dynamic pruning strategy to address the inefficiency and instability of existing pruning schedules, where fixed or uniform pruning may remove important primitives too early or occur too late to yield meaningful efficiency gains.
2) Spatio-spectral Pruning Score (SPS). We design a new importance ranking mechanism that incorporates spectral information into the importance ranking of Gaussian primitives; it enables stable pruning, where iteratively removed low-score points are relatively unimportant across both spatial and spectral domains.
3) 3D Difference-of-Gaussians (3D-DoG) primitive. We introduce a new Gaussian variant with both positive and negative components, functioning as a primitive with intrinsic contrast to capture fine structural details.
4) Scalable efficiency. Our approach achieves comparable or superior rendering quality with 90% fewer primitives on the Mip-NeRF 360, Tanks & Temples, and Deep Blending datasets, offering a more scalable and efficient 3DGS framework for complex scenes.

## 2. Related Work

### 2.1. 3D Gaussian Splatting

3D Gaussian Splatting (3DGS) [23] is an explicit representation method that uses a cloud of differentiable 3D Gaussians to model a scene. Its popularity is due to its ability to render in real-time with high-fidelity reconstruction. In recent years, several works have extended its capabilities in quality, robustness, and generality. To address aliasing artifacts arising from unconstrained 3D frequencies, Mip-Splatting [42] and multi-scale splatting methods [41] introduced 3D smoothing and mipmap-based level-of-detail rendering. Other variants [5, 17, 18, 21, 22, 25] extend 3DGS to handle challenging conditions such as in-the-wild, hazy, and underwater scenarios by mitigating various degradation effects. Beyond static scenes, dynamic extensions such as 4DGS and Gaussian-Flow [19, 26, 40] incorporated temporal deformation networks to represent 4D scenes, allowing for moderate motion capture. Further, frequency regularization [43] and pixel-aware density control [45] help maintain detail consistency and temporal stability across scales and viewpoints. More recently, Liberated-GS [33] decoupled 3DGS from traditional Structure-from-Motion (SfM) [39] point clouds, achieving self-contained reconstruction without external initialization. EnliveningGS [34] explored the active locomotion of Gaussian primitives, extending 3DGS towards physically plausible and dynamic behaviors. In parallel, FlashGS [11] improved computational efficiency and scalability for large-scale and high-resolution rendering, making 3DGS more practical for real-world deployment.

### 2.2. Pruning Gaussian Splats

Although the aforementioned works have substantially advanced 3D Gaussian Splatting (3DGS), improving its computational efficiency remains a critical challenge. The naive density control strategy employed often introduces redundancy, thereby increasing computational overhead [3]. Pruning unnecessary points has thus emerged as the most common solution to this problem. Typically, Gaussians are ranked by importance, with the least important discarded. Several works [31, 44] approximate Gaussian importance using opacity values. Later studies proposed dedicated pruning scores: RadSplat [32] defines an efficient score based on the accumulated ray contributions of individual Gaussians; Mini-Splatting [9] aggregates blending weights to form a pruning score; PuP-3DGS [14] evaluates spatial sensitivity; and Speedy-Splat [13] further leverages

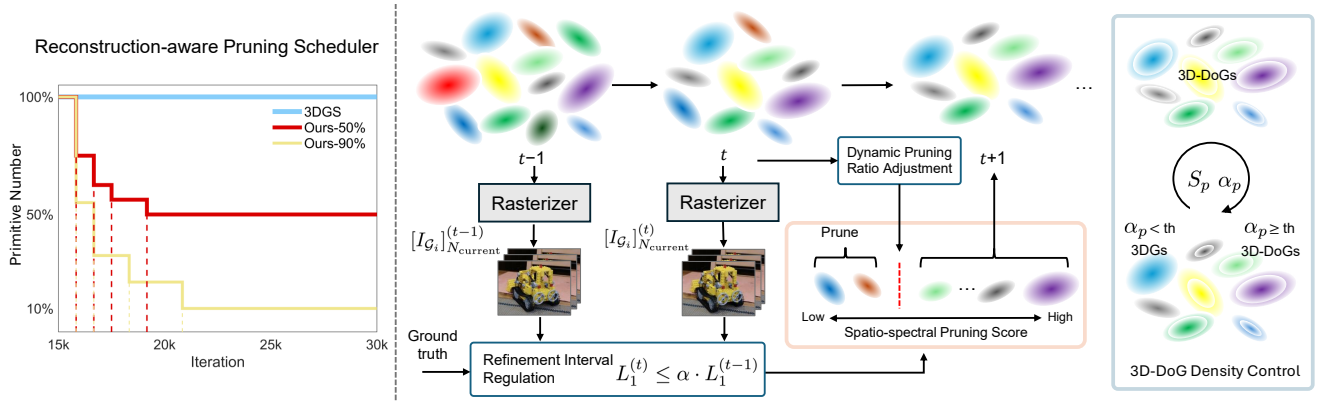



Figure 2. (Left) Gaussian primitive count comparison. Our method adaptively adjusts the refinement settings to meet different pruning targets, such as the 50% and 90% pruning ratios shown in the figure. (Right) Overview of the Reconstruction-aware Pruning Scheduler and 3D-DoG Density Control. We use L1 loss as a reconstruction quality indicator to dynamically determine pruning timing and ratio throughout optimization. In addition, we activate 3D-DoG after pruning and adaptively control its density.

per-Gaussian gradients for improved performance. Other works, such as MaskGaussian [27] and LP-3DGS [46], employ Gumbel-Softmax [20] to learn adaptive masks for Gaussian importance estimation.

Unlike previous heuristic pruning methods, our approach performs adaptive, reconstruction-aware pruning, leading to more efficient compression without sacrificing quality.

## 3. Method

The diagram of our method is illustrated in Figure 2. We propose a Reconstruction-aware Pruning Scheduler to assist the progressive prune-refine process. With this adaptive pruning strategy, we can efficiently compress the 3DGS model at reasonable ratios and appropriate time steps. Subsequently, our method introduces a divergent primitive model, 3D-DoG (3D Difference-of-Gaussians), to compensate for detail loss caused by pruning. Consequently, the obtained compact 3DGS model can retain rendering quality and geometric fidelity with enhanced efficiency.

### 3.1. Preliminaries

3DGS [23] models a scene as a collection of 3D Gaussian primitives. Each Gaussian primitive $\mathcal{G}_i$ is defined by a center 3D position $\mu_i$, a covariance matrix $\Sigma_i$, an opacity $\alpha_i$, and a view-dependent colour $c_i$ defined by using spherical harmonics (SH). To facilitate a differentiable optimization, the covariance matrix $\Sigma_i$ is factorized into a rotation matrix $R_i \in \mathbb{R}^{3\times3}$ and a scaling matrix $S_i \in \mathbb{R}^{3\times3}$ in the 3D Gaussian primitive.

$$\Sigma_i = R_i S_i S_i^T R_i^T. \tag{1}$$

To render images, 3D Gaussians need to be projected from 3D to 2D. Each 3D Gaussian undergoes a viewing transformation $W$ that projects it onto the image plane, and its 2D covariance matrix can be calculated as follows:

$$\Sigma_i' = JW\Sigma_i W^T J^T, \tag{2}$$

where $J$ denotes the Jacobian matrix [36] corresponding to an affine approximation of the projective transformation.

The original 3D Gaussian primitive can be considered as a smooth kernel; its density reaches a peak at its center and gradually declines to 0 as the rendered pixel moves away from its center. As a result, the effective blending weight $\alpha_i'$ of a 3D Gaussian $G_i$ in pixel $x$ can be computed as:

$$\alpha_i' = \alpha_i \exp\left(-\tfrac{1}{2}(x-\mu_i')^T(\Sigma_i')^{-1}(x-\mu_i')\right), \tag{3}$$

where $\mu_i'$ refers to the 2D Gaussian's mean after projecting 3D coordinate $\mu_i$ The pixel colour formation following the $\alpha$-blending scheme as follows:

$$C = \sum_i c_i\, \alpha_i' \prod_{j=1}^{i-1}(1-\alpha_j'), \tag{4}$$

$c_i$ corresponds to the colour decoded from the coefficients of the 3D Gaussian spherical harmonics $i$-th, which can vary depending on the viewing direction.

During 3DGS optimization, the 2D position gradients of 3D Gaussians serve as a key factor in controlling the model size growth [37]. However, this densification mechanism leads to over-densification, resulting in a high proportion of redundant Gaussian points [16]. Consequently, an additional pruning process is needed to achieve a trade-off between performance and efficiency. In addition, the smoothness of the 3D Gaussian primitive limits its ability to represent fine details in a compact model. Therefore, a more expressive primitive design is highly desirable.

### 3.2. Reconstruction-aware Pruning Scheduler

Previous work has typically performed pruning at fixed iterations and with a preset ratio [13]. However, this naive

**Algorithm 1** Refinement Interval Regulation strategy

**Require:** Current Model $\mathcal{M}_t$, reconstruction losses $L_1^{(t)}$, $L_1^{(t-1)}$, threshold $\beta$, and maximum interval $Iter_{\max}$.

**if** $L_1^{(t)} \leq \beta \cdot L_1^{(t-1)}$ **then**
  $\mathcal{M}_{t+1} = \text{prune}(\mathcal{M}_t)$
**else**
  $\text{refine}(\mathcal{M}_t)$ until criterion met or $Iter_{\max}$ reached
**end if**

strategy suffers from two significant limitations. First, the difficulty of reconstruction varies significantly across scenes, yet conventional iterative pruning methods adopt fixed pruning intervals regardless of this variation. In rapidly converging scenes, overly long intervals result in redundant points during rendering, wasting computational resources. Conversely, in more complex scenes, pruning too frequently prevents the point cloud from fully training, resulting in sub-optimal performance and loss of fine details. Second, pruning with fixed ratios neglects the changing redundancy level of the point cloud. As pruning progresses, redundancy naturally decreases; maintaining a large ratio then leads to over-pruning in later stages, thereby degrading reconstruction quality.

To address these issues, we propose a Reconstruction-aware Pruning Scheduler (RPS) that dynamically adjusts the pruning interval based on the reconstruction loss and progressively reduces the number of pruned primitives at each step. In this way, RPS enables more adaptive, efficient, and detail-preserving pruning across diverse scenes. An overview of the RPS process is shown in Figure 2, and its details are described below.

**Refinement Interval Regulation.** Previous refinement strategies [9] have pruned the 3D point cloud after each refinement step. However, the question of when to prune (i.e. how long the refining interval should be) remains underexplored. We address this by adaptively modulating the refinement interval throughout the pruning process. After each pruning step, we use the average L1 loss as a criterion to determine whether a further pruning operation should be executed. Let $L_1^{(t)}$ denote the L1 reconstruction loss computed on the whole training dataset after the $t$-th pruning step, and $L_1^{(t-1)}$ denote the loss recorded before the $t$-th pruning step. To integrate the prune–refine procedure into the training process rather than applying it as a post-processing step [8, 14]. We set a pruning criterion as follows:

$$L_1^{(t)} \leq \beta \cdot L_1^{(t-1)}, \tag{5}$$

where we set $\beta = 0.95$. If the above condition is satisfied, this indicates that the reconstruction quality has improved. In this case, we proceed to the next pruning step. Otherwise, the structure remains unchanged and refinement continues until the condition in (5) is met or the maximum refinement interval $Iter_{\max}$, which is set to 2000 iterations, is reached to avoid a bottleneck. The process is summarized in Algorithm 1. This procedure is carried out every 500 iterations until the pruning target is reached, at the cost of a slight degradation in efficiency.

**Dynamic Pruning Ratio Adjustment.** We observe that as the overall 3DGS model size decreases, pruning increasingly degrades the reconstruction quality, which is in agreement with the findings reported in [1]. A key reason for this is that the number of redundant GS primitives varies during the pruning process. When redundancy is high, many 3D points overlap spatially and contribute little to the final rendering; aggressive pruning is desirable. However, as the model size decreases, most remaining Gaussians are crucial for preserving fine geometric and photometric details. Building on this observation, we introduce a novel dynamic pruning scheduler in which the removal rate decreases over time. Aggressive pruning is allowed in the early stages, and only a relatively few points are removed in the later stages. Moreover, we integrate multistep pruning into the 3DGS model optimization rather than running them sequentially [8], thereby improving training efficiency and enabling the model to adaptively balance performance and sparsity. Inspired by [6], we control the pruning ratio of the round $t$ as follows:

$$N^{(t)} = N_{\text{current}} - \left(N_0 - N_{\text{target}}\right) \cdot \frac{1}{2^t}, \tag{6}$$

$$R^{(t)} = \frac{\left(N_{\text{current}} - N^{(t)}\right)}{\left(N_{\text{current}}\right)}, \tag{7}$$

where $N^{(t)}$ and $R^{(t)}$ denote the desired number of Gaussians and computed pruning ratio of pruning round $t$, $N_0$ is the number of Gaussians after complete densification, $N_{\text{target}}$ is the preset target number. We use the current number of primitives $N_{\text{current}}$ and $N^{(t)}$ to obtain the pruning ratio for the current round. By dynamically adjusting the pruning ratio, the model size can be significantly reduced in the early iterations, and the training process further accelerated.

**Spatio-spectral Pruning Score.** To further enhance the effectiveness of pruning, we introduce a new criterion for evaluating the importance of each Gaussian. Specifically, we develop a Spatio-spectral Pruning Score, which measures the contribution of each primitive from both spatial and spectral perspectives. Previous work [13] employs a pruning score that enables a fast estimation, expressed as follows.

$$\tilde{U}_i = \left(\nabla_{g_i} I_{\mathcal{G}}\right)^2, \tag{8}$$

where $\tilde{U}_i$ denotes the efficient pruning score of the $i$-th Gaussian, $\nabla_{g_i}$ denotes the gradient of the parameters of $i$-th Gaussian with respect to the rendered image $I_{\mathcal{G}}$.

However, this importance evaluation concentrates on the spatial domain but ignores the frequency domain. Incorporating the frequency domain ensures that Gaussians essential for preserving sharp structures are not mistakenly pruned. Hence, we propose a spectral pruning score:

$$\tilde{U}_i^f = \sum_{\omega \in \Omega} w(\omega) \left| \nabla_{g_i} \hat{I}_{\mathcal{G}}(\omega) \right|^2, \quad \hat{I}_{\mathcal{G}} = \mathrm{FFT}(I_{\mathcal{G}}), \tag{9}$$

where $|\cdot|$ is the complex modulus and $w(\omega) \geq 0$ is a frequency weight that emphasizes informative bands. We use a radial schedule:

$$w(\omega) = \left( \frac{\|\omega\|}{\omega_{\max}} \right)^{\gamma_f}, \ \gamma_f > 0, \quad w(\mathbf{0}) = 0, \tag{10}$$

By combining a spatially aware and spectrally aware importance score $\tilde{U}_i$ and $\tilde{U}_i^f$, we obtain our Spatio-spectral Pruning Score (SPS) $\tilde{U}_i^*$:

$$\tilde{U}_i^* = \lambda_s \cdot \frac{\left(\nabla_{g_i} I_{\mathcal{G}}\right)^2}{\|\tilde{U}\|_2} + \lambda_f \cdot \frac{\left(\nabla_{g_i} \mathrm{FFT}(I_{\mathcal{G}})\right)^2}{\|\tilde{U}^f\|_2}, \tag{11}$$

where $\lambda_s$ and $\lambda_f$ are weighting coefficients that balance the contributions of the spatial- and frequency-domain gradient terms, respectively. This formulation allows the model to consider both spatial variations jointly and frequency-domain characteristics for more stable pruning.

### 3.3. 3D Difference-of-Gaussians (3D-DoG)

**3D-DoG Kernel.** After pruning, we found that the compact model suffers a very significant loss in details. This issue is due to the intrinsic properties of 3DGS: splatting operates only in the positive-density domain. To represent boundary or texture areas, vanilla 3DGS can only stack dozens of small, overlapping Gaussians of different shapes to capture fine details [35]. However, after aggressive pruning, there are not enough primitives to reconstruct those areas. To address this, we introduce a novel 3D Difference-of-Gaussian (3D-DoG) primitive, which can be viewed as the difference between a primary-Gaussian and a pseudo-Gaussian as shown in Figure 3 (top).

We define the 3D-DoG primitive as follows:

$$DoG(x) = G(x) - G_p(x), \tag{12}$$

where $G_p(x)$ is a pseudo-Gaussian that shares the same center coordinates of the kernel and all other parameters with the primary-Gaussian $G(x)$ except for opacity and scales. Consequently, compared to the vanilla 3D Gaussians, 3D-DoG introduces four additional learnable parameters to define the pseudo-Gaussian, which are the opacity factor $f^\alpha$ and the scaling factors $[f_x^s, f_y^s, f_z^s]$. The scalings $S_p$ and opacity $\alpha_p$ are defined as follows:

$$S_p = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix} \cdot \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & f_z \end{bmatrix}^T, \tag{13}$$

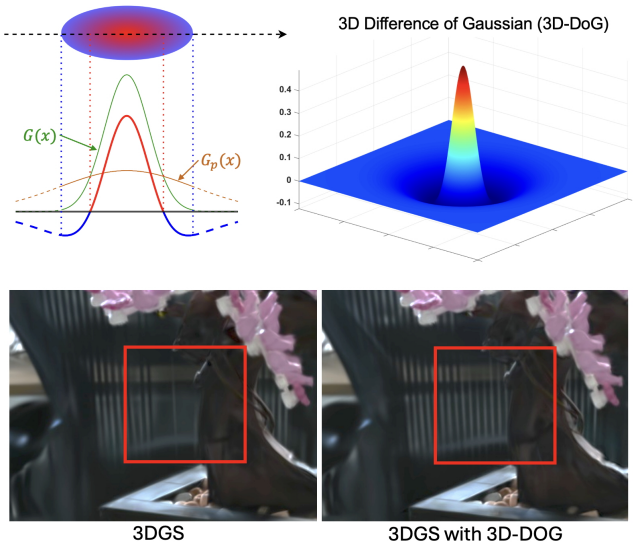



Figure 3. (Top) Illustration of the proposed 3D-DoG primitive in 1D and 3D, featuring a positive-density peak and a negative-density ring. (Bottom) 3DGS with 3D-DoG primitives achieves better detail representation.

$$\alpha_p = f^\alpha \cdot \alpha, \tag{14}$$

where $[s_x, s_y, s_z]$ and $\alpha$ are the scaling parameters and the opacity of the primary Gaussian, respectively. Combined with other existing parameters, all are set to values less than 1.0 to ensure that our 3D-DoG profile maintains a positive lobe that governs the radiance, consistent with the original 3D Gaussian formulation. The surrounding negative-density ring serves as a contrast component, subtracting the color of neighboring pixels it overlaps with [47]. In other words, the 3D-DoG inherently encodes contrast within its construction. Compared to 3D Gaussians, which capture overall global geometry, 3D-DoGs have a sharper response and can thus enhance features such as boundaries when the number of primitives is limited. This advantage is clearly demonstrated in Figure 3 (bottom).

**3D-DoG Density Control.** Since 3D-DoG introduces additional computational overhead, our observations indicate it takes longer to cover a dense point cloud. Thus, we only introduce 3D-DoG primitives in the compact model.

After the pruning phase, we employ 3D-DoG primitives to replace all primitives in the compact version model by activating $[f_x^s, f_y^s, f_z^s]$ and $\alpha_p$ to allow the adjustment of pseudo-Gaussians, and continue training with the same loss functions. By evaluating $\alpha_p$ across all primitives, we can identify those with the lowest $\alpha_p$ values, indicating that the pseudo-Gaussian representations of these 3D-DoGs have a marginal impact on representations of these primitives. When $\alpha_p$ = 0, 3D-DoGs become equivalent to a standard 3D Gaussians. Consequently, 3D-DoGs with $\alpha_p$ values below a predefined threshold are degenerated into 3D Gaus-

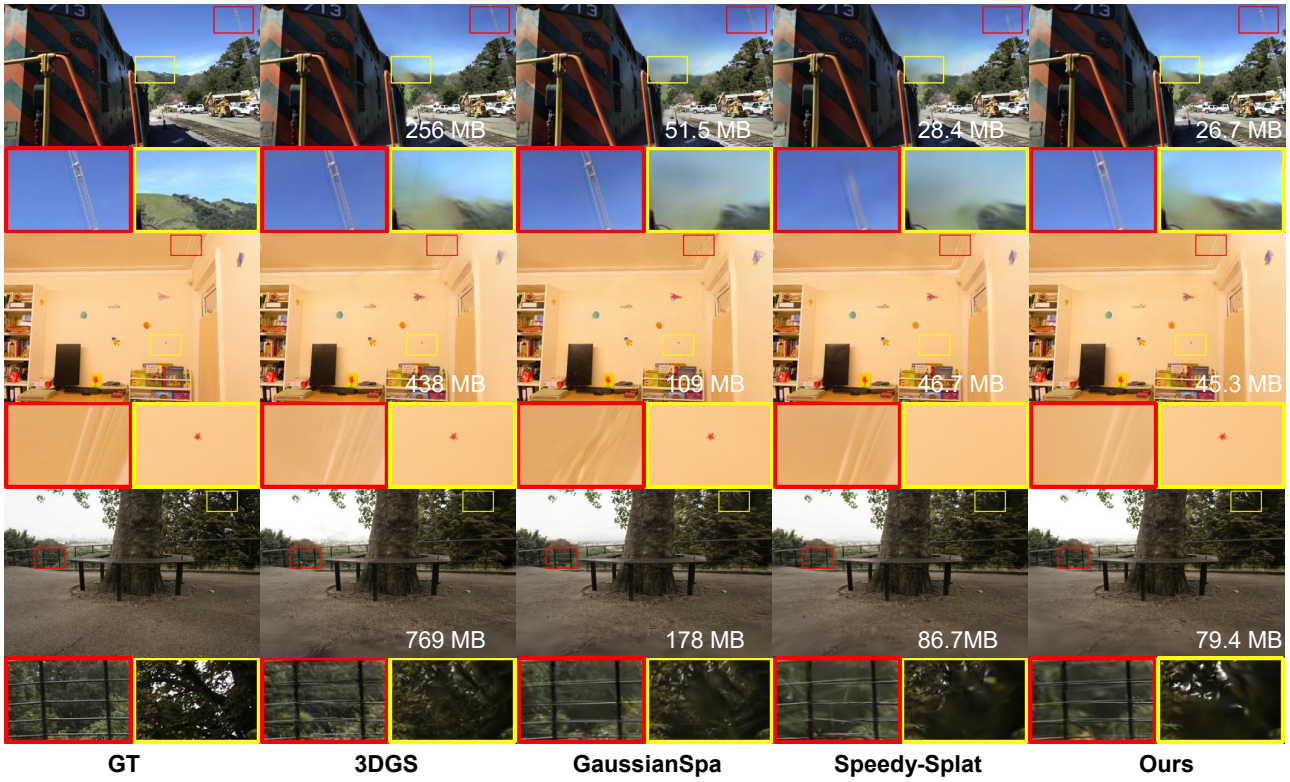



Figure 4. **Novel view rendering comparison with the baselines.** Top: **Train** from the Tanks & Temples. Middle: **Playroom** from the Deep Blending dataset. Bottom: **Treehill** from the Mip-NeRF 360 dataset. We have shown details below the images. Best viewed when zoomed in.

sian components at each training iteration. Hence, the computational overhead is reduced with minimal loss of fidelity.

This density control mechanism enables us to adaptively adjust the ratio of different types of primitives in the hybrid representation to accommodate various scene characteristics. By iteratively controlling the 3D-DoG density, our pipeline achieves a balance between efficiency and reconstruction quality. 3D-DoG and 3D Gaussian primitives are jointly optimized to play complementary roles, allowing the resulting mixture model to maintain stability in smooth regions while better recovering fine details.

# 4. Experiments

## 4.1. Experimental Setup

**Datasets.** To comprehensively evaluate our method, we use the Mip-NeRf 360 [4] dataset, which comprises four indoor and five outdoor scenes. Furthermore, we also tested Deep Blending [15] and Tanks & Temples [24], which provide two additional indoor and two additional outdoor scenes, respectively.

**Implementation.** The official code for the 3DGS implementation [23] is adopted as the backbone. In all experiments, we set a pruning target of 90%, retaining only 10% of primitives compared to the original model, since this setting is particularly challenging. Only the process after the first 15,000 iterations is modified, once the densification has finished. Our Reconstruction-aware Pruning Scheduler is integrated with the rest of the optimization to enable the progressive pruning-refine process until the model size reaches the preset pruning target. The duration of this step is flexibly adjusted as described in subsection 3.2, but will conclude after no more than 25,000 iterations. After the pruning process finishes, we introduce the 3D-DoG primitives into the model and optimize their overall proportion in the point cloud. Our experiments were carried out on a single RTX 3090 GPU. We adopt the accelerated rasterization CUDA module [29] to improve efficiency. The forward and backward branches have been modified to adapt to 3D-DoG splatting. More details are available in the supplementary material (SM).

**Baseline and Metrics.** Apart from the original 3DGS[23], we selected MaskGaussian [27], GaussianSpa [44], PuP-3DGS [29], and Speedy-Splat [13] as baselines; all of these methods aim to compress the original 3DGS model by pruning. MaskGaussian does not specify a pruning ratio, whereas GaussianSpa uses an 80% pruning ratio when built upon 3DGS. PuP-3DGS and Speedy-Splat both target a 90% pruning ratio. By testing these representative methods as baselines, we evaluate our approach not only on re-

Table 1. Quantitative evaluation of our method compared to previous work, computed over three datasets: Mip-NeRF 360, Deep Blending and Tanks and Temples. ↑ refers larger values are better while ↓ is opposite. The best, second best, and third best results are highlighted. † denotes accelerated diff-gaussian-rasterization module is adopted

| Method | Mip-NeRF 360 [4] | | | | | Deep Blending [15] | | | | | Tanks and Temples [24] | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Size ↓ | PSNR↑ | SSIM↑ | LPIPS↓ | Time↓ | Size ↓ | PSNR↑ | SSIM↑ | LPIPS↓ | Time↓ | Size ↓ | PSNR↑ | SSIM↑ | LPIPS↓ | Time↓ |
| 3DGS† [23] | 645.2 | 27.47 | 0.826 | 0.201 | 17m1s | 592.7 | 29.75 | 0.904 | 0.244 | 13m51s | 381.0 | 23.77 | 0.847 | 0.177 | 17m50s |
| MaskGaussian [27] | 280.7 | 27.43 | 0.811 | 0.227 | 24m11s | 172.4 | 29.69 | 0.907 | 0.244 | 15m10s | 140.0 | 23.72 | 0.847 | 0.181 | 13m45s |
| GaussianSpa [44] | 157.4 | 27.26 | 0.807 | 0.239 | 24m45s | 132.5 | 29.66 | 0.905 | 0.250 | 13m49s | 87.3 | 23.67 | 0.841 | 0.198 | 23m11s |
| PuP-3DGS [14] | 90.6 | 26.67 | 0.786 | 0.271 | - | 69.9 | 28.85 | 0.881 | 0.302 | - | 43.4 | 22.72 | 0.801 | 0.244 | - |
| Speedy-Splat [13] | 73.9 | 26.84 | 0.782 | 0.296 | 16m30s | 58.6 | 29.42 | 0.887 | 0.311 | 15m30s | 43.0 | 23.43 | 0.821 | 0.241 | 9m37s |
| Ours† | 65.3 | 27.16 | 0.789 | 0.285 | 13m48s | 59.9 | 29.87 | 0.904 | 0.254 | 10m19s | 38.4 | 23.79 | 0.823 | 0.229 | 8m9s |

construction quality but also on memory usage and training speed. We adopt common PSNR, SSIM, and perceptual-based LPIPS to measure the visual quality, and use model size and training time to evaluate computational efficiency.

### 4.2. Results and Discussion

**Quantitative comparison.** Table 1 shows the evaluation results of our method compared with all the benchmarks. Our method achieves state-of-the-art reconstruction performance among approaches that produce models of similar size, such as PUP-3DGS [14] and Speedy-Splat [13]. It slightly improves PSNR values in the Deep Blending [15] and Tanks & Temples [24] datasets. Our method achieves a favorable balance between quality, efficiency, and compactness across different datasets, maintaining competitive PSNR and SSIM scores while significantly reducing the size of the model. This highlights the effectiveness of our design in achieving compact 3D Gaussian representations with minimal visual quality degradation. Implemented with CUDA, our method achieves a frame rate of 330+ FPS.

**Qualitative comparison.** The subjective comparison between the baselines and our method is shown in Figure 4. Examples include the *Train* from Tanks and Temples [24], *Playroom* scenes from Deep Blending [15], and *Treehill* scenes from Mip-NeRF 360 [4]. These results demonstrate that our method can reconstruct fine details, showing its effectiveness and adaptability across various scenes. Temporal stability can be demonstrated via the $x$-$t$ plane at a fixed $y$ as shown in Fig. 6. Smooth, continuous bands indicate strong temporal consistency, while fragmented bands suggest flickering. The similar $x$-$t$ patterns indicate that our pruning method does not introduce temporal artifacts.

**Pruning analysis.** The pruning process and the PSNR increase curve are shown in Figure 5. This figure illustrates that our Reconstruction-aware Pruning Scheduler can be successfully integrated into the original training pipeline, with progressive pruning maintaining a consistent upward trend in PSNR.

Figure 5 also shows the trends in PSNR and primitive count during optimization. It demonstrates that our Reconstruction-aware Pruning Scheduler progressively compresses the overall number of primitives without hindering PSNR improvement. In some cases, such as the **Bicycle** scene (Figure 5 left), we inevitably observe a slight performance drop in the late stage of pruning due to the aggressive 90% pruning target set in our experiment. Note that the performance degradation in the 25k iteration is due to the activations of 3D-DoG attributes, which can be quickly recovered in subsequent iterations.

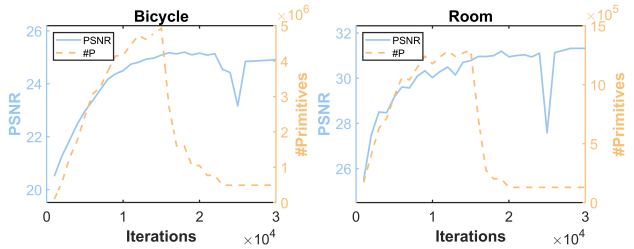



Figure 5. Variations in the primitive count and PSNR values of (Left) **Bicycle** and (Right) **Room** scenes using our method. The PSNR drop at the 25k iteration is due to the activation of 3D-DoG.

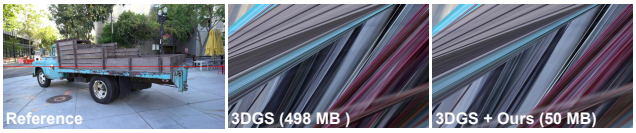



Figure 6. Temporal stability: $x$-$t$ plane at $y$=500 of Trunk video rendered by 3DGS (middle) and our pruned 3DGS (right).

**Collaboration Between 3D Gaussians and 3D-DoGs.** We isolate 3D Gaussians and 3D-DoGs in our mixture model for rendering, and the results are shown in Figure 7. These results demonstrate that the two primitives work collaboratively: 3D Gaussians capture the overall structure, while 3D-DoGs are predominantly placed around texture and edge regions.

Figure 8 compares the error maps of a compressed model without and with 3D-DoGs in the **Bonsai** scene along a red line that traverses both edge and textured regions. The error




Figure 7. The rendering results are obtained by separating 3D Gaussians and 3D-DoGs in our 90% primitives pruned 3DGS model. Compared to vanilla 3D Gaussian, the proposed 3D-DoG is more sensitive to capture details such as edges and textures. Examples are respectively from **Counter** and **Bicycle**.

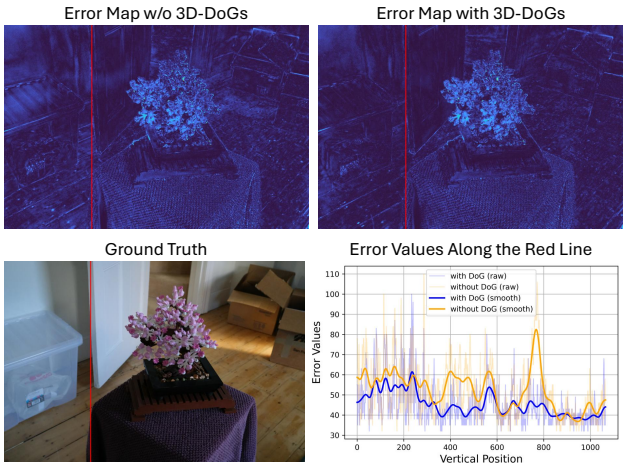



Figure 8. **Error Comparison.** (Top): Error heatmaps over the full image for the compact model without (left) and with (right) 3D-DoGs. (Bottom-left): Novel view of the **Bonsai** scene. (Bottom-right): Reconstruction error along the red line for both models.

values along this line are generally lower when 3D-DoGs are incorporated, indicating that 3D-DoGs effectively reduce reconstruction errors, particularly at structural boundaries and in texture-rich areas.

### 4.3. Ablation Study

We isolate our contributions using a set of modified frameworks, starting with the 3DGS backbone with 100% primitives. We first use the refinement interval Regulation (RIR) as **V1** to prune 90% of primitives. At each step, prune the same number of primitives based on opacity rank. In **V2**, the Dynamic Pruning Ratio Adjustment (DPRA) is integrated to tune the number of pruning primitives at each step. The Spatio-spectral Pruning Score (SPS) is adopted to replace opacity to measure per-primitive importance in the 3DGS model in **V3**. Finally, we introduce a novel 3D-DoG primitive into the architecture to formulate our method.

Table 2 provides a comprehensive comparison of both reconstruction quality and efficiency across different model variants. As the model scales, the variant with only 10% of the parameters consistently approaches the full 100% model in performance, validating the effectiveness and general robustness of our proposed architecture. In terms of efficiency, our techniques significantly accelerate 3DGS training and inference. Specifically, **V1** achieves approximately a $1.36\times$ reduction in training time and nearly $2.5\times$ higher FPS compared to the original 3DGS baseline. **V2** further improves training efficiency while maintaining a stable rendering speed. **V3** introduces SPS, slightly increasing train time without a noticeable loss in FPS. Finally, our complete model ("Ours"), which incorporates the 3D-DoG module, delivers the best reconstruction quality, with only a marginal drop in efficiency due to the additional computation introduced by 3D-DoG primitives but still accelerates the training by $1.23\times$ and inference by $2\times$ respectively.

Table 2. Comparison of different variants in terms of reconstruction quality and efficiency on the Mip-NeRF360 dataset using 3DGS as the backbone. ↑ indicates higher is better, while ↓ indicates lower is better.

| | Components | | | | Reconstruction Metrics | | | Efficiency Metrics | |
|---|---|---|---|---|---|---|---|---|---|
| Variant | RIR | DPRA | SPS | 3D-DoG | PSNR↑ | SSIM↑ | LPIPS↓ | Time↓ | FPS↑ |
| 3DGS | ✗ | ✗ | ✗ | ✗ | 27.47 | 0.826 | 0.201 | 17m1s | 143.5 |
| V1 | ✓ | ✗ | ✗ | ✗ | 26.03 | 0.742 | 0.331 | 12m17s | 362.4 |
| V2 | ✓ | ✓ | ✗ | ✗ | 26.17 | 0.751 | 0.324 | 11m34s | 363.2 |
| V3 | ✓ | ✓ | ✓ | ✗ | 26.99 | 0.771 | 0.299 | 13m28s | 361.9 |
| Ours | ✓ | ✓ | ✓ | ✓ | 27.16 | 0.789 | 0.285 | 13m48s | 289.0 |

## 5. Conclusion

We propose an adaptive framework for efficient 3DGS that jointly optimizes pruning and reconstruction quality. By introducing a reconstruction-aware pruning scheduler, our method dynamically balances compression and quality during training, enabling stable convergence and compact representations. In addition, the proposed 3D-DoG primitives enhance the expressive power of compact models, allowing them to retain fine structural details even under aggressive pruning. Through extensive experiments, our approach consistently achieves a favorable trade-off between rendering fidelity, model size, and computational cost. Our adaptive, reconstruction-driven compression can be a promising direction for scalable 3D scene representation. Future work includes extending the framework to dynamic, large-scale scenes and hardware-efficient rendering for real-time use.

## Acknowledgements

This work was supported by the EPSRC ECR International Collaboration Grants (EP/Y002490/1) and the UKRI MyWorld Strength in Places Programme (SIPF00006/1).

## References

[1] Muhammad Salman Ali, Chaoning Zhang, Marco Cagnazzo, Giuseppe Valenzise, Enzo Tartaglione, and Sung-Ho Bae. Compression in 3d gaussian splatting: A survey of methods, trends, and future directions. *arXiv preprint arXiv:2502.19457*, 2025. 4

[2] Nantheera Anantrasirichai, Fan Zhang, and David Bull. Artificial intelligence in creative industries: Advances prior to 2025. *arXiv preprint arXiv:2501.02725*, 2025. 1

[3] Milena T Bagdasarian, Paul Knoll, Y Li, Florian Barthel, Anna Hilsmann, Peter Eisert, and Wieland Morgenstern. 3dgs. zip: A survey on 3d gaussian splatting compression methods. In *Computer Graphics Forum*, page e70078. Wiley Online Library, 2025. 2

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5470–5479, 2022. 6, 7

[5] Wenbo Chen and Ligang Liu. Deblur-gs: 3d gaussian splatting from camera motion blurred images. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 7 (1):1–15, 2024. 2

[6] Youyu Chen, Junjun Jiang, Kui Jiang, Xiao Tang, Zhihao Li, Xianming Liu, and Yinyu Nie. Dashgaussian: Optimizing 3d gaussian splatting in 200 seconds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 11146–11155, 2025. 4

[7] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12882–12891, 2022. 1

[8] 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. *Advances in neural information processing systems*, 37: 140138–140158, 2024. 2, 4

[9] Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of gaussians. In *European Conference on Computer Vision*, pages 165–181. Springer, 2024. 2, 4

[10] Guangchi Fang and Bing Wang. Mini-splatting2: Building 360 scenes within minutes via aggressive gaussian densification. *arXiv preprint arXiv:2411.12788*, 2024. 1

[11] Guofeng Feng, Siyan Chen, Rong Fu, Zimu Liao, Yi Wang, Tao Liu, Boni Hu, Linning Xu, Zhilin Pei, Hengjie Li, et al. Flashgs: Efficient 3d gaussian splatting for large-scale and high-resolution rendering. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26652–26662, 2025. 2

[12] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5501–5510, 2022. 1

[13] Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedy-splat: Fast 3d gaussian splatting with sparse pixels and sparse primitives. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21537–21546, 2025. 2, 3, 4, 6, 7

[14] Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. Pup 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5949–5958, 2025. 2, 4, 7

[15] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6):1–15, 2018. 6, 7

[16] Binxiao Huang, Zhengwu Liu, and Ngai Wong. Decomposing densification in gaussian splatting for faster 3d scene reconstruction. *arXiv preprint arXiv:2507.20239*, 2025. 3

[17] Guoxi Huang, Haoran Wang, Zipeng Qi, Wenjun Lu, David Bull, and Nantheera Anantrasirichai. From restoration to reconstruction: Rethinking 3d gaussian splatting for underwater scenes. *arXiv preprint arXiv:2509.17789*, 2025. 2

[18] Guoxi Huang, Haoran Wang, Brett Seymour, Evan Kovacs, John Ellerbrock, Dave Blackham, and Nantheera Anantrasirichai. Visual enhancement and 3d representation for underwater scenes: a review. *arXiv e-prints*, pages arXiv–2505, 2025. 2

[19] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4220–4230, 2024. 2

[20] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 3

[21] Zhuodong Jiang, Haoran Wang, Guoxi Huang, Brett Seymour, and Nantheera Anantrasirichai. Rusplatting: Robust 3d gaussian splatting for sparse-view underwater scene reconstruction. *arXiv preprint arXiv:2505.15737*, 2025. 2

[22] Zhuodong Jiang, Haoran Wang, Guoxi Huang, Brett Seymour, and Nantheera Anantrasirichai. Swagsplatting: Semantic-guided water-scene augmented gaussian splatting. *arXiv preprint arXiv:2509.00800*, 2025. 2

[23] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 1, 2, 3, 6, 7

[24] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 1, 6, 7

[25] Byeonghyeon Lee, Howoong Lee, Xiangyu Sun, Usman Ali, and Eunbyung Park. Deblurring 3d gaussian splatting. *arXiv preprint arXiv:2401.00834*, 2024. 2

[26] Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21136–21145, 2024. 2

[27] Yifei Liu, Zhihang Zhong, Yifan Zhan, Sheng Xu, and Xiao Sun. Maskgaussian: Adaptive 3d gaussian representation from probabilistic masks. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 681–690, 2025. 3, 6, 7

[28] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V Sander. Deblur-nerf: Neural radiance fields from blurry images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12861–12870, 2022. 1

[29] Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Markus Steinberger, Francisco Vicente Carrasco, and Fernando De La Torre. Taming 3dgs: High-quality radiance fields with limited resources. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024. 6

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

[31] KL Navaneet, Kossar Pourahmadi Meibodi, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Compgs: Smaller and faster gaussian splatting with vector quantization. In *European Conference on Computer Vision*, pages 330–349. Springer, 2024. 2

[32] 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. In *2025 International Conference on 3D Vision (3DV)*, pages 134–144. IEEE, 2025. 2

[33] Weihong Pan, Xiaoyu Zhang, Hongjia Zhai, Xiaojun Xiang, Hanqing Jiang, and Guofeng Zhang. Liberated-gs: 3d gaussian splatting independent from sfm point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 26675–26685, 2025. 2

[34] Siyuan Shen, Tianjia Shao, Kun Zhou, Chenfanfu Jiang, and Yin Yang. Enliveninggs: Active locomotion of 3dgs. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 896–905, 2025. 2

[35] Yuang Shi, Simone Gasparini, Géraldine Morin, Chenggang Yang, and Wei Tsang Ooi. Sketch and patch: Efficient 3d gaussian representation for man-made scenes. In *Proceedings of the 17th International Workshop on IMmersive Mixed and Virtual Environment Systems*, pages 51–57, 2025. 5

[36] KJ Waldron, Shih-Liang Wang, and SJ Bolin. A study of the jacobian matrix of serial manipulators. 1985. 3

[37] Haoran Wang, Nantheera Anantrasirichai, Fan Zhang, and David Bull. Uw-gs: Distractor-aware 3d gaussian splatting for enhanced underwater scene reconstruction. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3280–3289. IEEE, 2025. 3

[38] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf--: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 1

[39] Matthew J Westoby, James Brasington, Niel F Glasser, Michael J Hambrey, and Jennifer M Reynolds. 'structure-from-motion'photogrammetry: A low-cost, effective tool for geoscience applications. *Geomorphology*, 179:300–314, 2012. 2

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

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

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

[43] Jiahui Zhang, Fangneng Zhan, Muyu Xu, Shijian Lu, and Eric Xing. Fregs: 3d gaussian splatting with progressive frequency regularization. In *the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21424–21433, 2024. 2

[44] Yangming Zhang, Wenqi Jia, Wei Niu, and Miao Yin. Gaussianspa: An" optimizing-sparsifying" simplification framework for compact and high-quality 3d gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26673–26682, 2025. 2, 6, 7

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

[46] Zhaoliang Zhang, Tianchen Song, Yongjae Lee, Li Yang, Cheng Peng, Rama Chellappa, and Deliang Fan. Lp-3dgs: Learning to prune 3d gaussian splatting. *Advances in Neural Information Processing Systems*, 37:122434–122457, 2024. 3

[47] Jialin Zhu, Jiangbei Yue, Feixiang He, and He Wang. 3d student splatting and scooping. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21045–21054, 2025. 5