Denoising-GS: Gaussian Splatting with Spatial-aware Denoising

Qingyuan Zhou[1], Xinyi Liu[1], Weidong Yang[1⋆], Ning Wang[2], Shuquan Ye[5], Ben Fei[3⋆], Ying He[4], and Wanli Ouyang[3,5]

[1] School of Computer Science and Artificial Intelligence, Fudan University, Shanghai, China
{zhouqy23,liuxiny24}@m.fudan.edu.cn wdyang@fudan.edu.cn
[2] School of Software, Dalian University of Technology, Dalian, China
nwang@dlut.edu.cn
[3] Department of Information Engineering, The Chinese University of Hong Kong, Hong Kong, China
feynben@gmail.com wlouyang@ie.cuhk.edu.hk
[4] College of Computing and Data Science, Nanyang Technological University, Singapore
yhe@ntu.edu.sg
[5] Shenzhen Loop Area Institute, Shenzhen, China
shuquanye@gmail.com

Abstract. Recent advances in 3D Gaussian Splatting (3DGS) have achieved significant success in high-fidelity Novel View Synthesis (NVS), yet the optimization process still introduces noisy Gaussian primitives due to the sparse and incomplete initialization from Structure-from-Motion (SfM) point clouds. Most existing methods focus only on adjusting the positions of primitives during optimization, while ignoring the underlying spatial structure. To this end, we introduce a new perspective by formulating the optimization of 3DGS as a primitive denoising process and propose **Denoising-GS**, a spatial-aware denoising framework for Gaussian primitives by taking both the positions and spatial structure into consideration. Specifically, we design an optimizer that preserves the spatial optimization flow of primitives, facilitating coherent and directed denoising rather than random perturbations. Building upon this, the **Spatial Gradient-based Denoising** strategy jointly considers the spatial supports of primitives to ensure gradient-consistent updates. Furthermore, the **Uncertainty-based Denoising** module estimates primitive-wise uncertainty to prune redundant or noisy primitives, while the **Spatial Coherence Refinement** strategy selectively splits primitives in sparse regions to maintain structural completeness. Experiments conducted on three benchmark datasets demonstrate that Denoising-GS consistently enhances NVS fidelity while maintaining representation compactness, achieving state-of-the-art performance across all benchmarks. Source code and models will be made publicly available.

⋆ Corresponding authors.

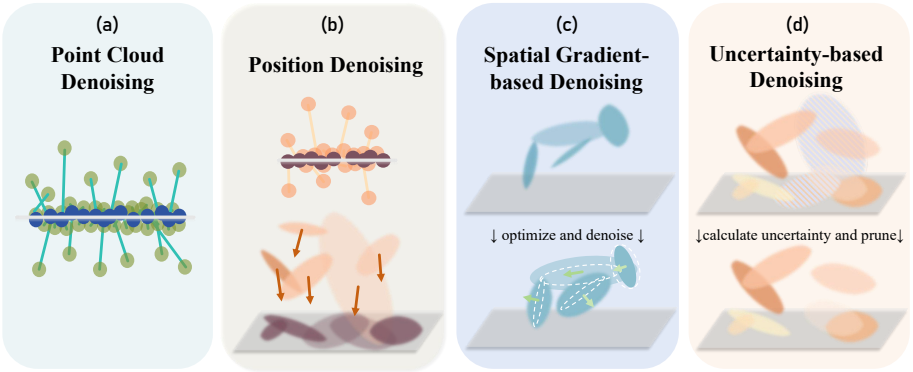

Fig. 1: Illustrative comparison of different denoising methods: (a) Point cloud denoising, which moves noisy points toward the GT surface; (b) Position denoising of Gaussian primitives, which adjusts the mean positions of noisy primitives toward the GT surface; (c) Spatial gradient-based denoising of Gaussian primitives, which optimizes both the mean positions and the spatial structure of primitives based on gradients; (d) Uncertainty-based denoising of Gaussian primitives, which estimates the uncertainty of each primitive and prunes those with high uncertainty.

1 Introduction

Recent progress in 3D scene representation [1,10,12] has greatly advanced novel view synthesis (NVS), enabling photorealistic rendering from multi-view images. Among these approaches, 3D Gaussian Splatting (3DGS) [21] has emerged as a leading explicit representation, modeling scenes with anisotropic Gaussian primitives that support real-time rendering and efficient optimization. Owing to its strong balance between rendering fidelity and computational efficiency, 3DGS has inspired numerous extensions across diverse applications, including specular modeling [29,36,37], surface reconstruction [4,18,50], SLAM [34,42], 3D generation [5,7,49], 3D copyright protections [19,28], and 3D human modeling [25,30].

Despite its success, 3DGS still faces a fundamental challenge due to its reliance on Structure-from-Motion (SfM) point clouds for initialization. Since these point clouds are often sparse and incomplete, the SGD-based optimization easily becomes trapped in local minima, producing noisy Gaussian primitives that compromise structural fidelity. Recent work, 3DGS-MCMC [22], addresses this issue by introducing opacity-guided relocation and Stochastic Gradient Langevin Dynamics (SGLD) [39] perturbations to help primitives escape local minima. However, these stochastic updates act only on the mean positions, leaving the covariance (spatial structure) unchanged, and the random perturbations in SGLD often lead to slow convergence, particularly near saddle points or in regions of high curvature [23].

While such stochastic approaches help escape local minima, they do not directly address the noisy nature of the primitives themselves. In practice, the inaccurate initialization and unstable updates in 3DGS generate primitives that

behave similarly to noisy points scattered around the true surface. This observation naturally connects 3DGS optimization with the well-studied problem of point cloud denoising.

In point cloud denoising [9, 51], noisy points are iteratively pulled toward the ground-truth (GT) surface (Fig. 1(a)), while GS-based surface reconstruction methods [4,8,18,50] likewise aim to position primitives near the underlying geometry (Fig. 1(b)). However, unlike point cloud denoising that involves positional refinement, the positions (means) and spatial structures (covariances) should be simultaneously denoised. This motivates us to reformulate 3DGS optimization as a Gaussian primitive denoising process, in which both the mean positions and covariance matrices are progressively refined to achieve a coherent, noise-reduced representation.

Building on this perspective, we propose **Denoising-GS**, a spatially aware denoising framework that jointly refines the mean and covariance of each primitive. To achieve stable and directed optimization beyond random perturbations, we introduce momentum-biased stochastic exploration into the optimization of 3DGS, which maintains the optimization flow of each primitive, leading to smoother convergence and more consistent updates.

On top of the momentum-biased stochastic exploration, we design the **Spatial Gradient-based Denoising** strategy that aligns the spatial updates of primitives with the underlying gradient field (Fig. 1(c)), ensuring geometrically consistent refinement. To further address redundancy and incomplete spatial coverage, we incorporate **Uncertainty-based Denoising**, which leverages uncertainty estimation to identify and prune noisy or redundant primitives (Fig. 1(d)). Finally, the **Spatial Coherence Refinement** strategy replenishes under-represented regions, preserving a compact yet complete scene representation for high-fidelity reconstruction. We evaluate Denoising-GS on three public datasets (Mip-NeRF 360 [2], Tanks and Temples [24], and Deep Blending [16]), and demonstrate state-of-the-art results among all baselines. Our contributions are fourfold:

- We reformulate the GS optimizing procedure as a Gaussian primitive denoising process for achieving accurate scene representation, and propose a spatial-aware 3DGS denoising framework.
- We introduce the momentum-biased stochastic exploration to enable primitives to stochastically explore global optimal positions while maintaining the optimization flow of the spatial process.
- We design the **Spatial Gradient-based Denoising** module that considers the spatial support of each primitive, ensuring its spatial updates remain consistent with the underlying gradient field.
- We leverage uncertainty estimation and design the **Uncertainty-based Denoising** module to identify and prune redundant or noisy primitives, and introduce the **Spatial Coherence Refinement** strategy for compensation of sparse regions.

2 Related Work

GS-based Novel View Synthesis. The advent of 3DGS has fundamentally redefined the landscape of NVS and has inspired numerous subsequent studies [10, 40] that aim to further enhance rendering fidelity, convergence speed, and computational efficiency. These methods address issues such as removing floating artifacts and achieving alias-free rendering [38, 43, 46], refining the densification and pruning strategies of Gaussian primitives [6,22,32,44,45,48]. Meanwhile, other approaches aim to balance rendering quality and efficiency through budget-aware optimization [14, 33], anchor-based Gaussian Splatting [31, 41, 47], improved Gaussian rasterization [3, 11, 35], and alternative primitive (kernel) representations8 [13, 26, 52].

Optimizing Gaussian Primitives in Spatial. Most existing GS-based methods follow an SGD-based optimizer to iteratively refine Gaussian primitives. In terms of geometric fidelity, GS-based surface reconstruction methods [4,8,18,50] flatten the primitives in spatial to produce scene representations that better align with GT mesh surfaces. To further accelerate optimization, 3DGS-LM [17] introduces a Levenberg-Marquardt optimizer, achieving comparable rendering quality with faster speed. Meanwhile, 3DGS-MCMC [22] adopts SGLD to introduce stochastic explorations on the mean positions during optimization, which helps prevent local optima and improve rendering robustness. In contrast to these approaches, our Denoising-GS preserves the spatial optimization trajectory of primitives in stochastic exploration, and models the spatial relationship between the mean and scaling, leveraging gradients to denoise primitives. It is also worth noting that Pixel-GS [48] and AbsGS [45] address a similar issue to our Spatial Gradient-based Denoising module, namely the influence of gradients. However, their main claims focus on utilizing gradients to facilitate the growth of large Gaussian primitives or to split oversized ones, without explicitly considering the spatial interaction and coordinated updates between the gradients of the mean and the scaling.

Uncertainty Estimation in GS. Recent works have explored uncertainty estimation within Gaussian Splatting representations, covering joint optimization of per-primitive uncertainty [27], next-best-view selection [20], and various downstream applications [15, 19]. FisherRF [20] and PUP 3D-GS [15] both use Fisher Information to estimate uncertainties in GS representations, similar to our Denoising-GS. However, FisherRF focuses on actively selecting informative training views based on uncertainty estimation, while PUP 3D-GS performs post-hoc uncertainty estimation to prune redundant Gaussians for downstream tasks that require sparse scene representations. In contrast, Denoising-GS integrates uncertainty estimation directly into the primitive denoising process, thereby achieving a tighter and more efficient scene representation.

3 Preliminaries

3DGS [21] is an explicit scene representation that utilizes a set of 3D anisotropic primitives to model a radiance field. Each of the primitives is represented by a

3D Gaussian distribution, initialized from SfM point clouds, as illustrated in Eq. (1),

$$G = e^{-\frac{1}{2}\mathcal{M}^\top \Sigma^{-1}\mathcal{M}}, \tag{1}$$

where $\mathcal{M} \in \mathbb{R}^3$ denotes the mean of the Gaussian distribution, Σ stands for the covariance matrix. Considering the inherent physical significance of the covariance matrix, which challenges to update conventionally in back-propagation, the covariance matrix is decomposed into rotation matrix $\mathbf{R}$ and scaling matrix $\mathbf{S}$ by Eq. (2),

$$\Sigma = \mathbf{RSS}^\top \mathbf{R}^\top, \tag{2}$$

where $\mathbf{R}$ represents quaternions, and $\mathbf{S}$ denotes to scaling factors.

3DGS-MCMC [22] reinterprets the training process of 3DGS as a Markov Chain Monte Carlo (MCMC) sampling procedure. Instead of relying on heuristic densification or pruning strategies, it views the set of Gaussian primitives as random samples drawn from an underlying probability distribution that characterizes the physical scene. Under this probabilistic perspective, the conventional gradient-based optimization of 3DGS is reformulated as a Stochastic Gradient Langevin Dynamics (SGLD) update, shown in Eq. (3),

$$\underbrace{\mathcal{M} \leftarrow \mathcal{M} - \lambda_{\text{lr}} \nabla \mathcal{M} L_{\text{total}}}_{\text{vanilla 3DGS update}} + \lambda_{\text{noise}} \lambda_{\text{lr}} \boldsymbol{\xi}, \tag{3}$$

which is derived by augmenting the Adam-based parameter update of vanilla 3DGS with an additional SGLD noise term to enable stochastic exploration in the parameter space. Here, L_{total} denotes total loss function, λ_{noise} controls the magnitude of the injected noise, and $\boldsymbol{\xi} \sim \mathcal{N}(0, \mathbf{I})$ denotes Gaussian noise.

Additionally, 3DGS-MCMC introduces a novel strategy for densifying and pruning Gaussian primitives. In practice, primitives with low opacity are relocated to the positions of primitives with high opacity, following the transformations defined in Eq. (4) and Eq. (5).

$$o^{\text{new}}_{1,\dots,N} = 1 - \sqrt[N]{1 - o^{\text{old}}_N}, \tag{4}$$

$$\Sigma^{\text{new}}_{1,\dots,N} = \left(o^{\text{old}}_N\right)^2 \left(\sum_{i=1}^{N} \sum_{k=0}^{i-1} \binom{i-1}{k} \frac{(-1)^k \left(o^{\text{new}}_N\right)^{k+1}}{\sqrt{k+1}} \right)^{-2} \Sigma^{\text{old}}_N, \tag{5}$$

where o is the opacity, Σ denotes the covariance matrix, and N indicates the number of new primitives cloning at the same position.

4 Method

4.1 Overview

Denoising-GS redefines the optimization of 3DGS as a denoising process of Gaussian primitives. Specifically, we first introduce the momentum-biased stochastic

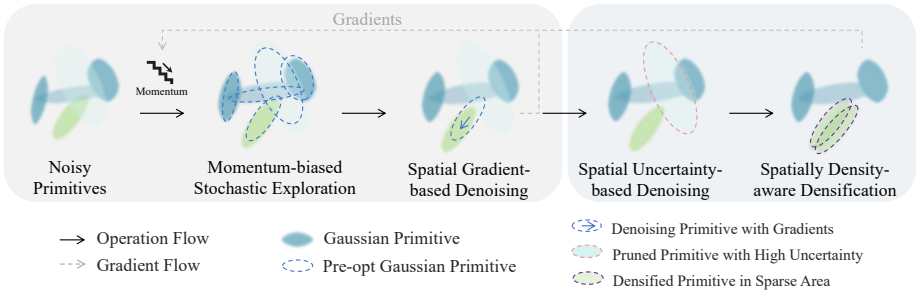

Fig. 2: Overview of the proposed Denoising-GS framework. The pipeline starts from noisy SfM-initialized Gaussian primitives and progressively refines them through a multi-stage denoising process, yielding a compact and accurate representation for high-fidelity NVS. The proposed momentum-biased exploration preserves the spatial optimization trajectory of primitives, ensuring that the denoising exploration is coherent and directed rather than purely stochastic. The spatial gradient-based denoising module jointly considers the spatial support of primitives, and performs denoising guided by the gradients. Furthermore, the uncertainty-based denoising module prunes redundant or noisy primitives, while the spatial coherence refinement strategy compensates for sparse regions to maintain completeness and fidelity.

exploration into 3DGS, which preserves the spatial optimization trajectory of primitives and enables more stable and direction-aware denoising (Sec. 4.2). To address the inefficiency issue where the translation of the mean is counteracted by concurrent scaling along the same axis, we design a spatial denoising strategy that jointly considers the gradients of means and scalings to align spatial variations with gradient directions (Sec. 4.3). Furthermore, we introduce an uncertainty estimation module to identify and prune redundant or noisy primitives (Sec. 4.4), and the spatial coherence refinement strategy to compensate for sparse regions (Sec. 4.5).

4.2 Momentum-biased Stochastic Exploration

3DGS-MCMC [22] introduces SGLD [39] perturbations to inject random noise into each Gaussian primitive during optimization, allowing exploration of more potential positions and alleviating the issue of local minima encountered in standard SGD. However, SGLD often results in unstable trajectories and slow convergence. To address these issues, we adopt the Momentum-biased Stochastic Exploration [23] strategy to 3DGS. In contrast to the vanilla SGLD, which updates parameters solely based on instantaneous gradients and stochastic noise, we introduce an exponential moving average of historical gradients (momentum) into the stochastic exploration, effectively stabilizing the stochastic updates while preserving the asymptotic correctness of the sampling process.

Formally, given the mean of a Gaussian primitive $\mathcal{M}_t \in \mathbb{R}^3$ at iteration t in optimization, the SGLD update is realized by introducing a drift component.

The update rule is defined as:

$$\mathcal{M}_{t+1} = \mathcal{M}_t - \lambda_t \nabla \mathcal{M}_t L_{\text{total}} + \sqrt{2\lambda_t \tau}\, \boldsymbol{\xi}_t, \tag{6}$$

where $\nabla \mathcal{M}_t$ is the gradient of $\mathcal{M}_t$, λ_t is the learning rate, τ is the temperature coefficient controlling the stochasticity, and $\boldsymbol{\xi}_t \sim \mathcal{N}(0, \mathbf{I})$ is Gaussian noise. 3DGS-MCMC simplifies this formula to Eq. (3).

To alleviate the aforementioned issues and endow the stochastic noise with a memory bank, our strategy incorporates a momentum term $\mathbf{m}_t$ which maintains an exponentially decayed history of past gradients to guide the exploration.

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \nabla \mathcal{M}_{t-1}, \tag{7}$$

where $\nabla \mathcal{M}_{t-1}$ is the gradient of the mean $\mathcal{M}$ of the Gaussian primitive, and β_1 is a smoothness controlling coefficient. Our parameter update is then formulated as:

$$\mathcal{M}_{t+1} = \mathcal{M}_t - \lambda_t \nabla \mathcal{M}_t L_{\text{total}} + \sqrt{2\lambda_t \tau}\, \boldsymbol{\xi}_t - \alpha \lambda_t \mathbf{m}_t, \tag{8}$$

where $\alpha \in (0, 1]$ is a bias factor that controls the relative contribution of the momentum term in the drift component.

In the implementation, the magnitude of the Gaussian noise disturbed to the means is opacity-aware and adapted to the spatial shape of each Gaussian primitive, whereas the momentum term serves purely as an offset determined by historical gradients, independent of both opacity and the covariance matrix. The detailed implementation is provided in the Appendix.

4.3 Spatial Gradient-based Denoising

Although SGLD and our momentum-biased stochastic exploration could eliminate the positional noise of Gaussian primitives, the spatial noise should also be taken into consideration. In other words, the mean position $\mathcal{M}$ and the covariance matrix $\boldsymbol{\Sigma}$ of each Gaussian primitive should be jointly denoised, as the latter defines the effective spatial support of the primitive.

During the iterative optimization of a Gaussian primitive, the gradients of the mean and the covariance (decomposed into scaling and rotation) are computed independently based on the total loss. However, in specific cases, the movement of the mean may align with the primitive's expansion or contraction, causing redundant updates and thus reducing the effectiveness of optimization. A simplified illustration is shown in Fig. 3, which represents an idealized case where rotation $R = I$. The dashed Gaussian primitive denotes the optimal region, and our objective is to iteratively optimize the yellow primitive to cover this target region. Primitives that lie outside the target region (colored in gray) are considered part of the noisy region. Fig. 3(a) shows the initial state, while Fig. 3(b) presents the state after the optimization of the momentum-biased stochastic exploration. During this process, the mean $\mathcal{M}$ shifts leftward under the guidance of gradient $\nabla \mathcal{M}$ and the exploration induced in Sec. 4.2 (ΔD). Meanwhile, both s_L and s_R expand synchronously with the gradient ∇s_j. This gradient-guided

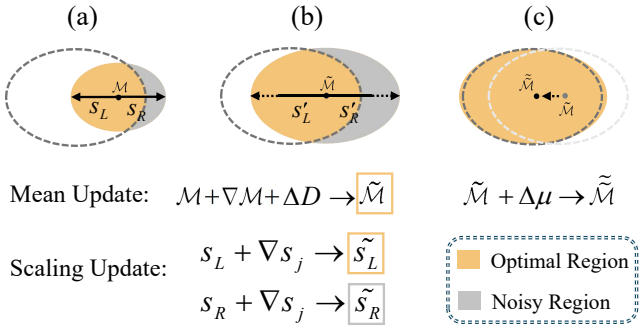

Fig. 3: A simplified illustration of the Spatial Gradient-based Denoising process. Note that the diagram represents an idealized case, and we assume that all Gaussian primitives move only along the horizontal direction, with rotation neglected. $\mathcal{M}$ is the mean position of Gaussian primitive, $\nabla\mathcal{M}$ is gradient of $\mathcal{M}$, ΔD is the momentum-biased stochastic exploration component, $s_L = s_R$ represent the half-lengths of the primitive's major axis j, and ∇s_j is the scaling gradient on j axis. $\tilde{\mathcal{M}}$ is the mean position after the momentum-biased stochastic exploration. $\tilde{\tilde{\mathcal{M}}}$ is the mean position of the denoised primitive, and $\Delta\mu$ is the denoising term.

update, however, also enlarges the gray region, which corresponds to the noisy region. To address this issue, the proposed Spatial Gradient-based Denoising module introduces a denoising term $\Delta\mu$ (Fig. 3(c)) that adaptively adjusts both the means and covariance matrices, ensuring that the Gaussian primitive is denoised toward its optimal position.

However, in a practical scenario, the Gaussian primitives no longer move strictly along the coordinate axes and may undergo rotations, resulting in directly computing the gradient of the covariance matrix $\boldsymbol{\Sigma}$ is computationally expensive. Following the standard parameterization in [21], we define the means as $\mathcal{M} = (\mathcal{M}_x, \mathcal{M}_y, \mathcal{M}_z)$, and represent the covariance matrix as $\boldsymbol{\Sigma}$, where the scaling is $\mathbf{S} = \mathrm{diag}(s_x, s_y, s_z)$ and the rotation is $\mathbf{R}$. At each iteration during optimization, the gradients of mean $\nabla\mathcal{M} = (\nabla\mathcal{M}_x, \nabla\mathcal{M}_y, \nabla\mathcal{M}_z)$ indicate the desired directional movement of the means. The scaling gradients $\nabla\mathbf{S} = (\nabla\mathbf{S}_x, \nabla\mathbf{S}_y, \nabla\mathbf{S}_z)$ correspond to the expansion or contraction along the principal axes. To ensure consistent updates, we consider both the mean and its gradients within the same local coordinate system of each Gaussian. Specifically, we first transform the mean gradients into the local coordinate system of the Gaussian via:

$$\nabla\mathcal{M}^G = \mathbf{R}^\top \nabla\mathcal{M}, \tag{9}$$

where $\nabla\mathcal{M}^G$ denotes the mean gradients in local Gaussian coordinate system, $\mathbf{R}$ is rotation matrix. Note that although the rotation is stored as a quaternion in the GS pipeline, the corresponding $\mathbf{R}$ is expressed as a 3×3 matrix. We then identify the principal axes of both the mean gradients $\nabla\mathcal{M}^G$ and scaling gradients $\nabla\mathbf{S}$ by locating their maximum components. If the two axes are aligned, a directional denoise term $\Delta\mu$ is added along the same axis on $\mathcal{M}$ to prevent

redundant movement:

$$\Delta\mu = |\nabla \mathbf{S}_j| \cdot \text{sign}(\nabla \mathcal{M}_j^G). \tag{10}$$

Here, $\nabla \mathbf{S}_j$ and $\nabla \mathcal{M}_j^G$ denote the scaling and mean gradient values along the j-th principal axis in the local Gaussian coordinate system, where j corresponds to one of the x, y, or z axes. The *sign* function preserves the direction of the mean gradient, ensuring that the denoising displacement occurs in the same directional sense as the original gradient update. Finally, the means of primitives are updated as:

$$\mathcal{M}_{t+1} = \mathcal{M}_t - \lambda_t \nabla \mathcal{M}_t L_{\text{total}} + \Delta D + \beta_2 \mathbf{R} \Delta\mu, \tag{11}$$

where $\Delta D = \sqrt{2\lambda_t \tau}\boldsymbol{\xi}_t - a\lambda_t \mathbf{m}_t$ comes from the momentum-biased stochastic exploration defined in Eq. (8), with consistent variable names. $\beta_2 \in (0, 1]$ is the controlling coefficient of the denoising term $\Delta\mu$.

4.4 Uncertainty-based Denoising

To achieve a more efficient and compact scene representation, we present a strategy that leverages uncertainty estimation to prune primitives and enable denoising. Inspired by [15, 20], we utilize Fisher Information to compute the primitive uncertainty based on means and scalings. Specifically, Fisher Information measures the sensitivity of model parameters to the observed data and can be defined as the variance of the score of the log-likelihood:

$$\mathbf{I}(\theta) = \mathbb{E}_{X \sim p(X|\theta)} \left[\nabla_\theta \log p(X \mid \theta) \, \nabla_\theta \log p(X \mid \theta)^\top \right], \tag{12}$$

where θ denotes the model parameters, X is the observed data, and $p(X \mid \theta)$ is the likelihood.

However, direct computation of this expectation is intractable in the Gaussian Splatting pipeline. In our Denoising-GS framework, we approximate the Fisher information for each Gaussian primitive using the gradients of the L2 rendering loss with respect to the primitive's parameters θ, including the $\mathcal{M}$ and scalings $\mathbf{S}$. For a single primitive, the Fisher information matrix is approximated via the outer product of the gradient:

$$F_i = \sum_{v \in \mathcal{V}} \nabla\theta_i \mathcal{L}_v \left(\nabla\theta_i \mathcal{L}_v \right)^\top, \tag{13}$$

where $\nabla\theta_i$ is the gradient of parameters θ_i, and $\mathcal{V}$ denotes the set of all camera views, and $\mathcal{L}_v = \|I_v - \hat{I}_v(\theta)\|_2^2$ is the L2 rendering loss of view v, where I_v is the GT image and $\hat{I}_v(\theta)$ is the rendered image. The uncertainty of the i-th primitive is calculated by:

$$u_i = \sum_{j=1}^{6} \sigma_j(F_i), \tag{14}$$

where u_i denotes the uncertainty of the i-th Gaussian primitive, F_i represents the approximate Fisher information matrix computed for the parameters of the i-th primitive, and $\sigma_j(F_i)$ is the j-th singular value of F_i obtained through singular value decomposition (SVD). The summation runs over six singular values, corresponding to the three dimensional $\mathcal{M}$ and three dimensional $\mathbf{S}$. The summation over all singular values corresponds to the total information magnitude of the primitive in the spatial domain. We interpret the uncertainty u as an inverse measure of the information contained in each primitive. Therefore, primitives with smaller u values are considered to have higher uncertainty and lower confidence, which typically correspond to noisy or redundant points. These primitives are identified and pruned during this denoising stage. The implementation is provided in the Appendix.

4.5 Spatial Coherence Refinement

To achieve a more expressive scene representation and improve rendering fidelity, we introduce a spatial coherence refinement strategy that exploits spatial density to adaptively guide the selective densification of Gaussian primitives in sparse regions, thereby enhancing scene details while maintaining representation compactness. Specifically, we compute the distance between the center of each primitive and its k nearest neighbors, defined as:

$$d_i = \frac{1}{k} \sum_{j=1}^{k} \|x_i - x_{i,j}\|_2^2, \tag{15}$$

where x_i denotes the position of the i-th primitive, and $x_{i,j}$ represents the position of one of the k-th nearest neighbors. Primitives with larger d are considered to lie in sparse regions and are densified according to Eq. (4) and Eq. (5). The detailed implementation is provided in the Appendix.

5 Experiments

5.1 Implementation

Our Denoising-GS framework is built using PyTorch 2.5.1 and CUDA 12.4. All experiments are conducted on an NVIDIA H100 GPU. For all experimental scenes, α is 0.05, β_1 is 0.9, β_2 is 0.5, and k is 3. In the Uncertainty-based Denoising stage, we remove the top 10% of the most uncertain primitives, while conducting Spatial Coherence Refinement a total of five times, where each iteration divides the 0.05% of the most spatially sparse primitives. These components will be ablated in Sec. 6.

Datasets and Metrics. Following the experimental setup of prior work [22], Denoising-GS is evaluated on eleven scenes from three datasets, including seven scenes from Mip-NeRF 360 [2], two outdoor scenes from Tanks and Temples [24], and two indoor scenes from Deep Blending [16]. We explore the use of denser

Table 1: Results of two baselines and eight GS-based NVS methods on the Mip-NeRF 360, Tank and Temples, and Deep Blending datasets. The red, orange and yellow colors represent the top three results.

#	Method	Venue	Mip-NeRF 360			Tanks and Temples			Deep Blending		
			SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓
(a)	Vanilla 3DGS [21]	ToG'23	0.880	29.258	0.168	0.845	23.649	0.178	0.900	29.475	0.247
	3DGS-MCMC [22]	NeurIPS'24	0.896	29.893	0.186	0.860	24.290	0.190	0.895	29.665	0.320
(b)	Variational-3DGS [27]	NeurIPS'24	0.871	28.744	0.185	0.839	23.324	0.184	0.890	28.732	0.265
	Taming 3DGS [33]	SIGGRAPH Asia'24	0.862	28.930	0.204	0.833	23.854	0.213	0.901	29.968	0.273
	Mip-Splatting [46]	CVPR'24	0.889	29.456	0.149	0.856	23.768	0.158	0.898	29.310	0.242
	GES [13]	CVPR'24	0.865	28.766	0.196	0.837	23.470	0.197	0.901	29.553	0.252
	AbsGS [45]	ACM MM'24	0.859	28.645	0.185	0.852	23.653	0.163	0.901	29.612	0.239
	Pixel-GS [48]	ECCV'24	0.886	29.454	0.159	0.853	23.730	0.151	0.892	28.858	0.251
	GS-LPM [44]	CVPR'25	0.887	29.378	0.160	0.845	23.880	0.186	0.900	29.421	0.246
	3D-HGS [26]	CVPR'25	0.881	29.772	0.162	0.857	24.395	0.165	0.901	29.554	0.248
(c)	**Denoising-GS**	**Ours**	0.897	29.925	0.142	0.868	24.561	0.149	0.906	29.872	0.239

and noisier input SfM point clouds on Mip-NeRF 360, which is optimized for 35K iterations. Tanks and Temples, and Deep Blending are optimized for 30K iterations. Structural Similarity Index Metric (SSIM), Peak Signal-to-Noise Ratio (PSNR), and Learned Perceptual Image Patch Similarity (LPIPS) are reported to evaluate the rendering (NVS) quality.

Baselines. We compare Denoising-GS with the ten most recent GS-based NVS methods, where vanilla 3DGS [21] and 3DGS-MCMC [22] serve as the baseline methods. These compared methods cover a wide range of recent developments, including GS with uncertainty estimation [27], faster convergence with limited resources [33], artifact suppression to improve rendering [46], alternative primitives (kernels) for scene representations [13,26], gradient-based rendering improvements [45, 48], and a reevaluation of the densification and pruning stages [44]. All baseline methods follow the original training protocol and are discussed further in the Appendix.

5.2 Results

Tab. 1 shows quantitative evaluations of novel view synthesis quality. Across all three datasets, Denoising-GS outperforms all compared methods. Under the standard initialization, the average SSIM/PSNR/LPIPS of Denoising-GS are 0.896/29.880/0.144 on the Mip-NeRF 360 Dataset.

Primitive Efficiency. In scene representation, there is a trade-off between rendering quality and the number of primitives used. As shown in Tab. 1, Denoising-GS outperforms methods [13, 21, 27, 33, 45] in terms of NVS quality on the Mip-NeRF 360 dataset, while methods [22, 26, 44, 46, 48] achieve similar quantitative results. To further examine the efficiency of scene representation, Tab. 2 compares the number of primitives used by the top five comparison methods [22, 26, 44, 46, 48] (based on PSNR results) and the vanilla 3DGS [21] on the Mip-NeRF 360 dataset. Specifically, Denoising-GS achieves the highest

Table 2: The comparison of Gaussian primitives required for scene representation among the top five NVS methods and the baselines on Mip-NeRF 360 dataset, with G denoting the number of required Gaussian primitives in millions (M). The red, orange and yellow colors represent the top three results.

Method	Bicycle		Bonsai		Counter		Garden		Kitchen		Room		Stump		AVG.	
	PSNR↑	$G\downarrow$	PSNR↑	$G\downarrow$	PSNR↑	$G\downarrow$	PSNR↑	$G\downarrow$	PSNR↑	$G\downarrow$	PSNR↑	$G\downarrow$	PSNR↑	$G\downarrow$	PSNR↑	$G\downarrow$
Vanilla 3DGS	25.64	5.83	32.29	1.25	29.08	1.16	27.73	5.05	31.35	1.73	31.75	1.46	26.97	4.55	29.258	3.004
3DGS-MCMC	26.15	5.90	32.88	1.30	29.51	1.20	28.16	5.20	32.27	1.80	32.48	1.50	27.80	4.75	29.893	3.093
Mip-Splatting	25.95	7.93	32.55	1.60	29.36	1.47	27.98	5.59	31.26	2.11	31.92	2.08	27.18	5.70	29.456	3.783
Pixel-GS	25.74	8.55	32.66	2.06	29.31	2.49	27.81	7.53	32.00	3.04	31.47	2.47	27.18	6.65	29.454	4.684
GS-LPM	25.90	5.75	32.35	1.31	29.21	1.21	27.77	5.01	31.72	1.80	31.49	1.56	27.21	5.03	29.378	3.096
3D-HGS	25.69	5.51	33.50	1.14	29.81	1.10	27.95	4.49	32.34	1.62	32.42	1.39	26.71	4.31	29.772	2.794
Denoising-GS	26.17	5.32	32.94	1.17	29.57	1.08	28.27	4.69	32.34	1.62	32.58	1.35	27.60	4.29	29.925	2.789

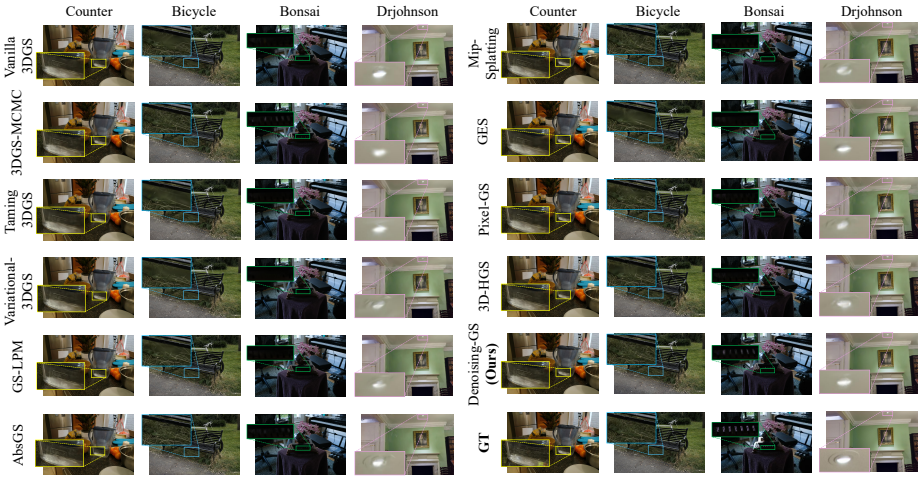

Fig. 4: Visual Comparisons on Novel View Synthesis of four scenes in Mip-NeRF 360 and Deep Blending datasets. More results are provided in the Appendix.

NVS performance while requiring the fewest primitives. 3D-HGS [26] employs nearly the same number of primitives as Denoising-GS but yields suboptimal NVS results. Additionally, Denoising-GS saves 10% or more primitives compared with [21, 22, 44], whereas [46, 48] use significantly more primitives than Denoising-GS.

Visualizing Comparisons. Fig. 4 shows comparisons across four scenes (*Counter*, *Bicycle*, *Bonsai*, and *Drjohnson*) from [2, 16]. More scenes will be discussed in the Appendix. Denoising-GS provides better visual quality than all ten other methods. Specifically, *Counter* is a well-lit indoor scene featuring semi-transparent objects and metallic specular surfaces. *Bicycle* is a large-scale outdoor scene featuring extensive areas of dense and finely detailed grass. *Bonsai* is a dimly-lit indoor environment where motion blur occurs due to camera shake during capture. *Drjohnson* is a complex indoor scene with multiple light

Table 3: Ablation results of NVS on Mip-NeRF 360, Tank and Temples, and Deep Blending datasets. The top results are indicated by bold (best), underline (second best), and red represent the coefficients and implementation reported in Sec. 5.

#	Coefficients			Prune	Densify	Mip-NeRF 360			Tanks and Temples			Deep Blending		
	α	β_1	β_2	Percent	Percent	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓	SSIM↑	PSNR↑	LPIPS↓
(a)	0.05	0.9	0.5	10%	0.05%	**0.897**	29.925	**0.142**	**0.868**	24.561	0.149	**0.906**	**29.872**	0.239
	0.05	0.9	0.1	10%	0.05%	**0.897**	29.930	**0.142**	0.867	**24.606**	0.150	**0.906**	29.849	0.239
	0.05	0.9	1	10%	0.05%	**0.897**	29.924	**0.142**	0.867	24.499	0.149	**0.906**	29.836	**0.238**
	0.1	0.9	0.5	10%	0.05%	**0.897**	29.901	0.143	0.867	24.458	**0.148**	0.904	29.400	0.240
	0.5	0.9	0.5	10%	0.05%	0.896	29.920	**0.142**	**0.868**	24.572	**0.148**	0.901	29.527	0.244
	0.05	0.5	0.5	10%	0.05%	**0.897**	29.891	**0.142**	0.867	24.486	0.149	**0.906**	29.827	0.239
	0.05	0.1	0.5	10%	0.05%	**0.897**	**29.942**	**0.142**	0.866	24.456	0.150	0.903	29.493	0.240
(b)	0.05	0.9	0.5	20%	0.05%	0.896	29.864	0.144	0.865	24.496	0.154	0.901	29.120	0.243
	0.05	0.9	0.5	10%	0.05%	**0.897**	**29.925**	**0.142**	**0.868**	**24.561**	**0.149**	**0.906**	**29.872**	**0.239**
	0.05	0.9	0.5	20%	0.1%	0.896	29.866	0.144	0.865	24.472	0.153	0.904	29.558	0.240
	0.05	0.9	0.5	10%	0.1%	**0.897**	29.900	0.143	**0.868**	24.521	**0.149**	0.905	29.700	**0.239**

sources, presenting substantial challenges for NVS under such intricate illumination conditions. The analysis below provides a detailed comparison of the four scenes. *Counter*: We evaluate the NVS capability for the semi-transparent brand text on the water bottle (yellow box). While [13, 22, 33] fails to render this region clearly, Denoising-GS recovers sharper details. *Bicycle*: The NVS of weeds beneath the seat (blue box) is examined. Because this area is invisible in most of the viewpoints, [13, 21, 26, 27, 33, 44, 48] produce noticeable floating artifacts. In contrast, with the Uncertainty-based Denoising and the Spatial Coherence Refinement, Denoising-GS produces clear rendering results while using fewer primitives (5.32M) than 3DGS-MCMC [21] (5.90M), AbsGS [45] (6.44M), and Mip-splatting [46] (7.93M). *Bonsai*: Rendering the structure beneath the flowerpot (green box) is particularly challenging because of severe illumination insufficiency. While almost all comparing methods hardly recover any structural details, Denoising-GS successfully reconstructs plausible geometry and texture in novel views. *Drjohnson*: We analyze the upper lighting area (pink box), where specular effects pose persistent challenges. Most comparison methods exhibit floating artifacts in this area, while Denoising-GS produces lighting reconstruction without artifacts. Moreover, our method maintains high rendering quality in other areas of the room outside the highlighted region.

6 Ablation Studies

Since the three benchmarks differ in characteristics such as image resolution and the quality of input SfM point clouds, our ablation aims to confirm the effectiveness of the proposed model across all scenes. Accordingly, we conduct ablations across all three datasets and select the hyperparameters that perform best on average, rather than optimizing separate settings for each dataset.

Controlling Coefficients. Tab. 3(a) presents ablation results for three coefficients in Sec. 4.2 and Sec. 4.3. The parameter α controls the strength of

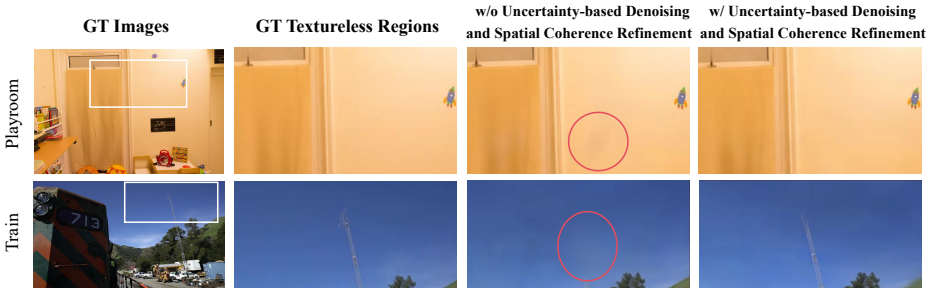

Fig. 5: Visual ablation of NVS in textureless regions, demonstrating the contributions of Uncertainty-based Denoising and Spatial Coherence Refinement.

correction applied to the momentum term relative to the exploration noise. A larger correction often slows down exploration, which in turn affects convergence. Therefore, we set α to a relatively small value, and experiments show that $\alpha = 0.05$ achieves the best and most balanced performance among the three datasets. The coefficient β_1 controls how smooth the momentum update is. Since the convergence behaviors differ between indoor and outdoor scenes, the different weights assigned to historical momentum and current gradients do not perform consistently across various datasets. We therefore select $\beta_1 = 0.9$, which achieves stable performance across all three datasets and matches the commonly used smoothing factor in momentum-biased optimization. The coefficient β_2 controls the strength of the denoising term. We set $\beta_2 = 0.5$ to achieve the best balance between denoising effectiveness and convergence speed in both indoor and outdoor scenes.

Uncertainty-based Denoising and Spatial Coherence Refinement. It is demonstrated [15] that removing too many primitives, while sometimes able to maintain visual quality, results in a significant decline in quantitative metrics. Conversely, pruning only a few primitives keeps rendering quality high but lowers the effectiveness of the denoising and pruning stages. To investigate this trade-off, we remove 10% and 20% of primitives, which corresponds to the removal of approximately 310K and 620K primitives, respectively, from the Mip-NeRF 360 dataset. At the same time, our Spatial Coherence Refinement strategy aims to improve scene representation while adding as few new points as possible. We examine densification rates of 0.05% and 0.1%, which correspond to roughly 5.6K and 13.9K additional points on Mip-NeRF 360 under the 10% pruning setting from the previous denoising stage. As shown in Tab. 3(b), pruning 10% of primitives combined with a 0.05% densification rate (an average reduction of 0.3M primitives per scene) achieves the best balance between compactness and rendering fidelity. Additionally, in Fig. 5, we present a visual ablation of the Uncertainty-based Denoising and Spatial Coherence Refinement strategies, demonstrating that our method provides a superior representation capability in textureless regions.

7 Conclusion

In this paper, we introduce a new perspective on GS optimization as a primitive denoising process and propose Denoising-GS, a spatial-aware denoising framework tailored for Gaussian primitives. We first introduce the momentum-biased stochastic exploration to allow primitives to remember the spatial process of optimization. On top of the momentum-biased optimizer, we design the Spatial Gradient-based Denoising strategy and the Uncertainty-based Denoising module for an accurate and compact scene representation. Finally, the Spatial Coherence Refinement strategy divides primitives in sparse areas to enhance the quality of the scene representation.

Limitations and Future Work. Rendering quality and the number of primitives involve a trade-off. In regions with sparse texture, such as large uniform areas of sky or low-texture walls, our method may exhibit a slight decline in rendering quality because of the denoising (pruning) of Gaussian primitives. For future work, we plan to extend its application to flattened primitives and investigate the potential for mesh extraction.

Acknowledgments

This research was supported by Shanghai's Key Technology Development Program, under the project 'Intelligent Integrated Arctic Ice-Snow-Atmosphere Observation Platform and Short- to Medium-Term High-Precision Sea Ice Forecasting' (Grant No. 25DZ3102600). The computational experiments were conducted using the CFFF platform at Fudan University.

References

1. Bao, Y., Ding, T., Huo, J., Liu, Y., Li, Y., Li, W., Gao, Y., Luo, J.: 3d Gaussian splatting: Survey, technologies, challenges, and opportunities. IEEE Transactions on Circuits and Systems for Video Technology **35**(7), 6832–6852 (2025)
2. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5470–5479 (2022)
3. Bulo, S.R., Bartolovic, N., Porzi, L., Kontschieder, P.: Hardware-rasterized ray-based Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 485–494 (2025)
4. Chen, D., Li, H., Ye, W., Wang, Y., Xie, W., Zhai, S., Wang, N., Liu, H., Bao, H., Zhang, G.: PGSR: Planar-based Gaussian splatting for efficient and high-fidelity surface reconstruction. IEEE Transactions on Visualization and Computer Graphics **31**(9), 6100–6111 (2024)
5. Chen, Z., Wang, F., Wang, Y., Liu, H.: Text-to-3d using Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 21401–21412 (2024)

6. Cheng, K., Long, X., Yang, K., Yao, Y., Yin, W., Ma, Y., Wang, W., Chen, X.: Gaussianpro: 3d Gaussian splatting with progressive propagation. In: International Conference on Machine Learning (ICML). pp. 8123–8140. PMLR (2024)
7. Chung, J., Lee, S., Nam, H., Lee, J., Lee, K.M.: LucidDreamer: Domain-free generation of 3d Gaussian splatting scenes. IEEE Transactions on Visualization and Computer Graphics **31**(12), 10640–10651 (2025)
8. Dai, P., Xu, J., Xie, W., Liu, X., Wang, H., Xu, W.: High-quality surface reconstruction using Gaussian surfels. In: ACM SIGGRAPH 2024 Conference Papers. pp. 1–11 (2024)
9. Edirimuni, D.d.S., Lu, X., Shao, Z., Li, G., Robles-Kelly, A., He, Y.: IterativePFN: True iterative point cloud filtering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 13530–13539 (2023)
10. Fei, B., Xu, J., Zhang, R., Zhou, Q., Yang, W., He, Y.: 3D Gaussian splatting as new era: A survey. IEEE Transactions on Visualization and Computer Graphics **31**(8), 4429–4449 (2025)
11. Feng, G., Chen, S., Fu, R., Liao, Z., Wang, Y., Liu, T., Hu, B., Xu, L., Pei, Z., Li, H., et al.: FlashGS: Efficient 3d Gaussian splatting for large-scale and high-resolution rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 26652–26662 (2025)
12. Gao, K., Gao, Y., He, H., Lu, D., Xu, L., Li, J.: NeRF: Neural radiance field in 3d vision, a comprehensive review. arXiv preprint arXiv:2210.00379 (2022)
13. Hamdi, A., Melas-Kyriazi, L., Mai, J., Qian, G., Liu, R., Vondrick, C., Ghanem, B., Vedaldi, A.: GES: Generalized exponential splatting for efficient radiance field rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 19812–19822 (2024)
14. Hanson, A., Tu, A., Lin, G., Singla, V., Zwicker, M., Goldstein, T.: Speedy-Splat: Fast 3d Gaussian splatting with sparse pixels and sparse primitives. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 21537–21546 (2025)
15. Hanson, A., Tu, A., Singla, V., Jayawardhana, M., Zwicker, M., Goldstein, T.: PUP 3D-GS: Principled uncertainty pruning for 3d Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 5949–5958 (2025)
16. Hedman, P., Philip, J., Price, T., Frahm, J.M., Drettakis, G., Brostow, G.: Deep blending for free-viewpoint image-based rendering. ACM Transactions on Graphics (ToG) **37**(6), 1–15 (2018)
17. Höllein, L., Božič, A., Zollhöfer, M., Nießner, M.: 3DGS-LM: Faster Gaussian-splatting optimization with Levenberg-Marquardt. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 26740–26750 (2025)
18. Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S.: 2d Gaussian splatting for geometrically accurate radiance fields. In: ACM SIGGRAPH 2024 Conference Papers. pp. 1–11 (2024)
19. Huang, X., Li, R., Cheung, Y.m., Cheung, K.C., See, S., Wan, R.: GaussianMarker: Uncertainty-aware copyright protection of 3d Gaussian splatting. Advances in Neural Information Processing Systems (NeurIPS) **37**, 33037–33060 (2024)
20. Jiang, W., Lei, B., Daniilidis, K.: FisherRF: Active view selection and mapping with radiance fields using Fisher information. In: European Conference on Computer Vision (ECCV). pp. 422–440. Springer (2024)

21. Kerbl, B., Kopanas, G., Leimkuehler, T., Drettakis, G.: 3d Gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics (ToG) **42**(4), 1–14 (2023)
22. Kheradmand, S., Rebain, D., Sharma, G., Sun, W., Tseng, Y.C., Isack, H., Kar, A., Tagliasacchi, A., Yi, K.M.: 3d Gaussian splatting as Markov chain Monte Carlo. Advances in Neural Information Processing Systems (NeurIPS) **37**, 80965–80986 (2024)
23. Kim, S., Song, Q., Liang, F.: Stochastic gradient Langevin dynamics algorithms with adaptive drifts. arXiv preprint arXiv:2009.09535 (2020)
24. Knapitsch, A., Park, J., Zhou, Q.Y., Koltun, V.: Tanks and Temples: Benchmarking large-scale scene reconstruction. ACM Transactions on Graphics (ToG) **36**(4), 1–13 (2017)
25. Kocabas, M., Chang, J.H.R., Gabriel, J., Tuzel, O., Ranjan, A.: HUGS: Human Gaussian splats. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 505–515 (2024)
26. Li, H., Liu, J., Sznaier, M., Camps, O.: 3D-HGS: 3d half-Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 10996–11005 (2025)
27. Li, R., Cheung, Y.m.: Variational multi-scale representation for estimating uncertainty in 3d Gaussian splatting. Advances in Neural Information Processing Systems (NeurIPS) **37**, 87934–87958 (2024)
28. Li, R., Zhang, X., Tong, C., Xu, Z., Zhang, J.: Gaussianseal: Rooting adaptive watermarks for 3d Gaussian generation model. Machine Intelligence Research pp. 1–18 (2026)
29. Liang, Z., Li, H., Jia, K., Guo, K., Zhang, Q.: GUS-IR: Gaussian splatting with unified shading for inverse rendering. IEEE Transactions on Pattern Analysis and Machine Intelligence **47**(10), 8364–8378 (2025)
30. Liu, X., Zhan, X., Tang, J., Shan, Y., Zeng, G., Lin, D., Liu, X., Liu, Z.: HumanGaussian: Text-driven 3d human generation with Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6646–6657 (2024)
31. Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., Dai, B.: Scaffold-GS: Structured 3d Gaussians for view-adaptive rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 20654–20664 (2024)
32. Lyu, Y., Cheng, K., Kang, X., Chen, X.: ResGS: Residual densification of 3d Gaussian for efficient detail recovery. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 28093–28102 (2025)
33. Mallick, S.S., Goel, R., Kerbl, B., Steinberger, M., Carrasco, F.V., De La Torre, F.: Taming 3DGS: High-quality radiance fields with limited resources. In: SIGGRAPH Asia 2024 Conference Papers. pp. 1–11 (2024)
34. Matsuki, H., Murai, R., Kelly, P.H., Davison, A.J.: Gaussian splatting SLAM. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 18039–18048 (2024)
35. Radl, L., Steiner, M., Parger, M., Weinrauch, A., Kerbl, B., Steinberger, M.: StopThePop: Sorted Gaussian splatting for view-consistent real-time rendering. ACM Transactions on Graphics (ToG) **43**(4), 1–17 (2024)
36. Shi, Y., Wu, Y., Wu, C., Liu, X., Zhao, C., Feng, H., Zhang, J., Zhou, B., Ding, E., Wang, J.: GIR: 3d Gaussian inverse rendering for relightable scene factorization. IEEE Transactions on Pattern Analysis and Machine Intelligence (2025)

37. Song, J., Ye, Z., Zhou, Q., Yang, W., Fei, B., Xu, J., He, Y., Ouyang, W.: Reflections Unlock: Geometry-aware reflection disentanglement in 3d Gaussian splatting for photorealistic scenes rendering. arXiv preprint arXiv:2507.06103 (2025)
38. Steiner, M., Köhler, T., Radl, L., Windisch, F., Schmalstieg, D., Steinberger, M.: AAA-Gaussians: Anti-aliased and artifact-free 3d Gaussian rendering. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 27650–27659 (2025)
39. Welling, M., Teh, Y.W.: Bayesian learning via stochastic gradient Langevin dynamics. In: Proceedings of the 28th International Conference on Machine Learning (ICML). pp. 681–688 (2011)
40. Wu, T., Yuan, Y.J., Zhang, L.X., Yang, J., Cao, Y.P., Yan, L.Q., Gao, L.: Recent advances in 3d Gaussian splatting. Computational Visual Media **10**(4), 613–642 (2024)
41. Xie, Y., Lin, Y., Li, L., Liu, L., Wei, X., Liu, Y., Lv, J.: Hash-GS: Anchor-based 3d Gaussian splatting with multi-resolution hash encoding for efficient scene reconstruction. In: 2025 IEEE International Conference on Robotics and Automation (ICRA). pp. 13964–13971. IEEE (2025)
42. Yan, C., Qu, D., Xu, D., Zhao, B., Wang, Z., Wang, D., Li, X.: GS-SLAM: Dense visual SLAM with 3d Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 19595–19604 (2024)
43. Yan, Z., Low, W.F., Chen, Y., Lee, G.H.: Multi-scale 3d Gaussian splatting for anti-aliased rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 20923–20931 (2024)
44. Yang, H., Zhang, C., Wang, W., Volino, M., Hilton, A., Zhang, L., Zhu, X.: Improving Gaussian splatting with localized points management. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Conference (CVPR). pp. 21696–21705 (2025)
45. Ye, Z., Li, W., Liu, S., Qiao, P., Dou, Y.: AbsGS: Recovering fine details in 3d Gaussian splatting. In: Proceedings of the 32nd ACM International Conference on Multimedia. pp. 1053–1061 (2024)
46. Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A.: Mip-Splatting: Alias-free 3d Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 19447–19456 (2024)
47. Zhang, J., Zhan, F., Shao, L., Lu, S.: SOGS: Second-order anchor for advanced 3d Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11167–11176 (2025)
48. Zhang, Z., Hu, W., Lao, Y., He, T., Zhao, H.: Pixel-GS: Density control with pixel-aware gradient for 3d Gaussian splatting. In: European Conference on Computer Vision (ECCV). pp. 326–342. Springer (2024)
49. Zhou, J., Zhang, W., Liu, Y.S.: DiffGS: Functional Gaussian splatting diffusion. Advances in Neural Information Processing Systems (NeurIPS) **37**, 37535–37560 (2024)
50. Zhou, Q., Gong, Y., Yang, W., Li, J., Luo, Y., Xu, B., Li, S., Fei, B., He, Y.: MGSR: 2d/3d mutual-boosted Gaussian splatting for high-fidelity surface reconstruction under various light conditions. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 27295–27304 (2025)
51. Zhou, Q., Yang, W., Fei, B., Xu, J., Zhang, R., Liu, K., Luo, Y., He, Y.: 3DMambaIPF: A state space model for iterative point cloud filtering via differentiable rendering. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 10843–10851 (2025)

52. Zhu, J., Yue, J., He, F., Wang, H.: 3d student splatting and scooping. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 21045–21054 (2025)