

# Opt3DGS: Optimizing 3D Gaussian Splatting with Adaptive Exploration and Curvature-Aware Exploitation

Ziyang Huang<sup>1</sup>, Jiagang Chen<sup>1</sup>, Jin Liu<sup>2</sup>, Shunping Ji<sup>1,✉</sup>

<sup>1</sup>Wuhan University <sup>2</sup>Hangzhou Dianzi University

{huangziyang2024, chenjiagang2015, jishunping}@whu.edu.cn, jinliu@hdu.edu.cn

## Abstract

3D Gaussian Splatting (3DGS) has emerged as a leading framework for novel view synthesis, yet its core optimization challenges remain underexplored. We identify two key issues in 3DGS optimization: entrapment in suboptimal local optima and insufficient convergence quality. To address these, we propose Opt3DGS, a robust framework that enhances 3DGS through a two-stage optimization process of adaptive exploration and curvature-guided exploitation. In the exploration phase, an Adaptive Weighted Stochastic Gradient Langevin Dynamics (SGLD) method enhances global search to escape local optima. In the exploitation phase, a Local Quasi-Newton Direction-guided Adam optimizer leverages curvature information for precise and efficient convergence. Extensive experiments on diverse benchmark datasets demonstrate that Opt3DGS achieves state-of-the-art rendering quality by refining the 3DGS optimization process without modifying its underlying representation.

## 1 Introduction

3D Gaussian Splatting (Kerbl et al. 2023) has recently emerged as a dominant method in novel view synthesis, significantly advancing scene modeling through its superior representational capabilities. Unlike implicit predecessors such as Neural Radiance Fields (NeRF) (Mildenhall et al. 2021), 3DGS leverages an explicit approach, modeling the radiance field of scenes using discrete Gaussian primitives. This explicit representation offers considerable advantages in computational efficiency and modeling flexibility, facilitating widespread adoption in diverse applications such as geometric reconstruction (Yu et al. 2024b; Chen et al. 2024; Guédon and Lepetit 2024), simultaneous localization and mapping (SLAM) (Matsuki et al. 2024), semantic scene understanding (Cen et al. 2025), and dynamic scene modeling (Yang et al. 2023).

Despite these advantages, effectively optimizing Gaussian primitives to reconstruct a radiance field remains a challenging non-convex optimization problem. Non-convex optimization inherently faces the risk of convergence to local optima, complicating the attainment of globally optimal scene representations. The original 3DGS method employs heuristic rules within its adaptive density control (ADC) step, us-

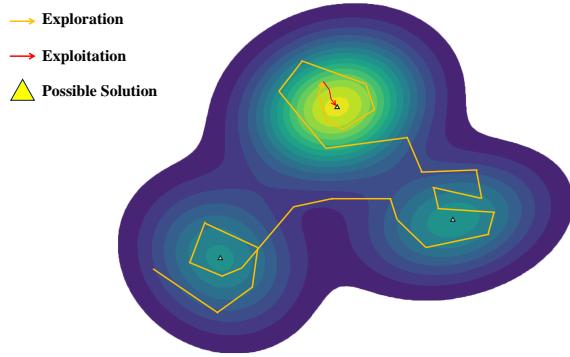


Figure 1: Exploration and Exploitation. The exploration stage promotes global search across modes using Adaptive Weighted SGLD, while the exploitation stage enables precise, curvature-aware convergence with Local Quasi-Newton direction-guided Adam optimizer.

ing fixed thresholds to guide Gaussian cloning, splitting, and pruning. Yet, these heuristics lack robustness and frequently yield suboptimal outcomes. Recent advancements, such as 3DGSMCMC (Kheradmand et al. 2024), attempt to address these limitations by modeling 3DGS optimization as a stochastic gradient Langevin dynamics (SGLD) process, incorporating opacity-based probabilistic sampling for Gaussian addition and removal. Although 3DGSMCMC unifies the training procedure, it does not completely resolve the local optima issue. Specifically, 3DGSMCMC introduces an inherent clustering effect, causing excessive Gaussian accumulation in already well-reconstructed regions. Based on Bayesian theorem, such clustering creates sampling bias, prematurely trapping the optimization in local modes and restricting exploration of the global solution space. Furthermore, standard first-order optimizers, like Adam (Kingma and Ba 2014), commonly used in existing 3DGS methods lack curvature information, hindering precise convergence during later training stages and limiting reconstruction quality. These limitations motivate the pursuit of optimization frameworks with enhanced exploration and precise convergence to tackle the complex non-convex landscape of 3DGS training.

In this paper, we propose Opt3DGS, a general and ef-

fective optimization framework for 3DGS. As shown in Figure 1, our approach divides the training process into two stages: Exploration and Exploitation. In the Exploration stage, inspired by the flat-histogram principle (Wang and Landau 2001), we introduce Adaptive Weighted SGLD (AW-SGLD). AW-SGLD flattens the posterior distribution to reduce energy barriers between modes, enabling the model to escape local traps and explore the solution space more thoroughly. This increases the likelihood of identifying the global optimal mode. In the Exploitation stage, once the model approaches a high-quality solution, we design a Local Quasi-Newton Direction-guided Adam optimizer. This optimizer leverages historical gradient information to estimate curvature-aware update directions, achieving more precise convergence than standard Adam while maintaining its robustness. Unlike full Quasi-Newton methods, it avoids the need for computationally expensive line searches. We evaluate Opt3DGS on public benchmarks, including MipNeRF360, Tanks & Temples, and DeepBlending, and compare it with state-of-the-art 3DGS methods. Experimental results demonstrate that Opt3DGS achieves superior rendering quality, validating the effectiveness of our optimization-centric approach.

Our contributions are summarized as follows:

- We analyze the local mode trapping issue in 3DGS from a Bayesian perspective and introduce AW-SGLD to enhance global exploration and improve convergence to the optimal mode.
- We overcome the limitation of first-order optimizers in the exploitation phase by developing a Local Quasi-Newton Direction-guided Adam optimizer, enabling precise convergence and enhanced rendering quality.
- By focusing solely on optimization without altering the Gaussian representation, Opt3DGS achieves state-of-the-art rendering performance.

## 2 Related Work

Neural rendering has recently made remarkable progress in novel view synthesis by learning scene representations directly from images. Among these methods, implicit approaches such as NeRF (Mildenhall et al. 2021) and explicit approaches like 3DGS (Kerbl et al. 2023) represent two dominant paradigms.

NeRF (Mildenhall et al. 2021) pioneered implicit 3D scene representation by learning a volumetric radiance field with an MLP and using volumetric rendering to synthesize novel views. Subsequent extensions improve different aspects of NeRF: NeRF++ (Zhang et al. 2020) models unbounded scenes, Mip-NeRF (Barron et al. 2021) and Mip-NeRF360 (Barron et al. 2022) address aliasing with conical frustum sampling, D-NeRF (Pumarola et al. 2021) incorporates temporal dynamics, and InstantNGP (Müller et al. 2022) accelerates training with multi-resolution hash encodings. Despite these advances, implicit methods remain computationally expensive due to the need for dense neural evaluations along rays.

3DGS (Kerbl et al. 2023) represents a scene using explicit 3D Gaussian primitives and renders them via parallel raster-

ization, thereby avoiding the expensive neural field evaluations required by NeRF-based volume rendering (Mildenhall et al. 2021). This design achieves real-time novel view synthesis with competitive visual quality. Building on this framework, subsequent works have sought to overcome its limitations from different perspectives. To improve rendering fidelity, methods such as Mip-Splatting (Yu et al. 2024a) and multi-scale splatting (Yan et al. 2024) address aliasing. To enhance expressiveness of representation, 2DGS (Huang et al. 2024) uses 2D surface-aligned Gaussians for higher geometric fidelity. 3DHGS (Li et al. 2025) refine the primitive model with half-Gaussians to better handle discontinuities. SSS (Zhu et al. 2025) introduce Student’s T-distribution to improve expressiveness. BBSplat (Svitov et al. 2024) using optimizable textured planar primitives to learn RGB textures and alpha maps achieving accurate representation. Other works focus on efficiency, including anchor-based model compression (Lu et al. 2024), faster training via resource allocation or Newtonian optimization (Chen et al. 2025; Lan et al. 2025; Pehlivan et al. 2025), sort-free rendering for lightweight devices (Hou et al. 2024). Densification strategies have also been actively explored: AbsGS (Ye et al. 2024) and RevisingGS (Rota Bulò et al. 2024) design more principled or error-driven adaptive density control, while FreGS (Zhang et al. 2024) mitigates over-reconstruction caused by densification through frequency-domain regularization. Methods based on Stochastic Gradient Markov Chain Monte Carlo (SGMCMC) unify the update, addition, and removal of Gaussian primitives within a single optimization framework. For instance, 3DGSM-CMC (Kheradmand et al. 2024) employ stochastic gradient Langevin dynamics and SSS (Zhu et al. 2025) adopts stochastic gradient Hamiltonian Monte Carlo for enhanced exploration.

Despite these efforts, stochastic methods still suffer from Gaussian over-clustering and lack mechanisms for precise, curvature-aware convergence, which motivates the optimization framework proposed in this work.

## 3 Background and Limitation

This section introduces the basics of 3DGS (Kerbl et al. 2023) and its extension 3DGSM-CMC (Kheradmand et al. 2024), and discusses the key limitations of these methods that motivate our proposed approach.

### 3.1 Preliminary

3DGS models a scene as a collection of explicit Gaussian primitives. Each primitive is parameterized by a 3D position  $\mu \in \mathbb{R}^3$ , an opacity scalar  $o \in \mathbb{R}$ , view-dependent spherical harmonic coefficients for appearance modeling, and a covariance matrix  $\Sigma = \mathbf{R}\mathbf{s}\mathbf{s}^\top\mathbf{R}^\top$  that determines the spatial extent and orientation of the Gaussian, where  $\mathbf{s} \in \mathbb{R}^3$  is a scale vector for the axis lengths and  $\mathbf{R} \in \mathbb{R}^{3 \times 4}$  (represented as a quaternion) for the rotation matrix  $\mathbf{R}$ . During rendering, each 3D Gaussian is projected onto the image plane, and the final pixel color is computed using alpha blending:

$$\mathbf{c}(\mathbf{x}) = \sum_{i=1}^N \mathbf{c}_i \cdot o_i \cdot T_i, \quad T_{i+1} = (1 - o_i) \cdot T_i, \quad (1)$$

where  $\mathbf{c}_i$ ,  $o_i$ , and  $T_i$  denote the color, opacity, and accumulated transmittance of the  $i$ -th Gaussian, respectively.

The adaptive density control in vanilla 3DGS, while effective, lacks robustness in certain scenarios. To address this limitation, the 3DGSMCMC framework reformulates 3DGS as a Markov Chain Monte Carlo (MCMC) process, treating each Gaussian primitive as a sample drawn from the scene's posterior distribution. Instead of stochastic gradient descent (SGD), 3DGSMCMC employs Stochastic Gradient Langevin Dynamics (SGLD) for parameter updates:

$$\begin{aligned} g_k &\leftarrow g_{k-1} - \lambda_{lr} \cdot \nabla_g \mathbb{E}_{I \sim \mathcal{I}} [\mathcal{L}_{\text{total}}(g_{k-1}; I)] + \lambda_{\text{noise}} \cdot \epsilon, \\ \epsilon &= \lambda_{lr} \cdot \sigma(-k(t-o)) \cdot \Sigma \eta, \quad \epsilon = [\epsilon_\mu, 0], \end{aligned} \quad (2)$$

where  $g_k$  denotes the Gaussian parameters at iteration  $k$ ,  $\lambda_{lr}$  the learning rate,  $\mathcal{L}_{\text{total}}$  the total loss, and  $\lambda_{\text{noise}}$  the coefficient controlling the injected noise. The term  $\epsilon$  is parameterized by the sigmoid function  $\sigma(\cdot)$  with hyperparameters  $k$  and  $t$ , the Gaussian opacity  $o$ , the covariance matrix  $\Sigma$ , and a 3D standard Gaussian random vector  $\eta$ .

When adding Gaussians, 3DGSMCMC samples the locations of the new Gaussians from the normalized opacity-based probability distribution of the current Gaussian set. For pruning Gaussians, 3DGSMCMC relocates the discarded Gaussians to the location of high-opacity Gaussians through the same opacity-based sampling. To maintain the stability of the Markov chain, the parameters of the newly generated Gaussians are computed as:

$$\begin{aligned} \Sigma_{1,\dots,N}^{\text{new}} &= (o^{\text{old}})^2 \left( \sum_{i=1}^N \sum_{k=0}^{i-1} \left( \binom{i-1}{k} \frac{(-1)^k (o^{\text{new}})^{k+1}}{\sqrt{k+1}} \right) \right)^{-2} \Sigma^{\text{old}}, \\ o_{1,\dots,N}^{\text{new}} &= \mu^{\text{old}}, \quad o_{1,\dots,N}^{\text{new}} = 1 - \sqrt[ N ]{1 - o^{\text{old}}} \end{aligned} \quad (3)$$

To promote sparsity in opacity and control the scale of the covariance matrices, 3DGSMCMC introduces two additional regularization term in loss function:

$$\begin{aligned} L_{\text{total}} &= (1 - \lambda_{ssim}) \times L_1 + \lambda_{ssim} \times L_{ssim} + \\ &\quad \lambda_o \times \sum_i |o_i|_1 + \lambda_\Sigma \times \sum_{ij} \left| \sqrt{\text{eig}_j(\Sigma_i)} \right|_1 \end{aligned} \quad (4)$$

### 3.2 Limitation of Existing Framework

3DGSMCMC formulates 3DGS optimization as a Markov Chain Monte Carlo process and achieves promising results, as the Langevin dynamics component encourages exploration of the posterior distribution. However, reconstructing complex scenes remain challenging. The posterior energy landscape is often highly multi-modal, with each mode corresponding to a different Gaussian configuration that explains the scene. When the energy barriers between modes are large, the combination of gradient guidance and Langevin noise is often insufficient to push the model out of its current local mode.

This issue is further exacerbated by the opacity-based sampling mechanism used in 3DGSMCMC. When adding or relocating Gaussians, a set of new sample positions is drawn independently and identically distributed (i.i.d.) from

a probability distribution  $\pi(x)$ , which is proportional to the normalized opacities of the current Gaussians:

$$x^{(1)}, x^{(2)}, \dots, x^{(N)} \stackrel{\text{i.i.d.}}{\sim} \pi(x) \quad (5)$$

Here each  $x^{(i)}$  corresponds to the spatial position where a new Gaussian will be placed. Although straightforward, this opacity-driven rule induces a clustering effect: new Gaussians tend to accumulate in regions that were discovered early in training. As dominant structures become highly opaque, subsequent sampling becomes more biased toward these well-reconstructed areas, leaving under-explored or geometrically complex regions insufficiently covered. From an MCMC perspective, this bias limits efficient exploration and traps the chain in a single posterior mode.

## 4 Method

To address the prevalent issues of local mode trapping and limited convergence in 3DGS optimization, we propose Opt3DGS, a novel optimization framework that combines two complementary components: an Adaptive Weighted SGLD that promotes global exploration and helps the model escape local minima in exploration stage; A Local Quasi-Newton Direction-guided Adam that refines the solution in exploitation stage with more accurate, curvature-aware updates. The following sections describe these two components in detail.

### 4.1 3DGS with Adaptive Weighted SGLD

Despite the clustering effect of opacity-based sampling (Sec. 3.2), it still offers favorable initializations for new Gaussians. Instead of modifying the sampling mechanism, we enhance the model's exploration to avoid premature convergence to suboptimal modes. A direct way to enhance exploration is to increase the Langevin noise intensity  $\lambda_{\text{noise}}$ . However, this is not robust: scene complexity varies, and excessive noise can destabilize training and impede convergence.

To address this, we introduce the flat histogram principle and propose an adaptive weighted stochastic gradient langevin dynamics(AW-SGLD) update for 3DGS. Let the configuration of Gaussian primitives for current scene follow a probability distribution  $P(g)$ , defined as:

$$P(g) \propto \exp \left( -\frac{\mathcal{L}_{\text{total}}(g)}{\tau} \right), \quad g \in \mathcal{G}, \quad (6)$$

where  $g$  denotes the current sample,  $\mathcal{G}$  is the sample space,  $\mathcal{L}_{\text{total}}$  is the total training loss (the energy function of the target distribution), and  $\tau$  is the temperature parameter.

Our objective is to construct a flattened distribution  $\rho(g)$  based on  $P(g)$  to facilitate the traversal of the sample space. To achieve this, we divide  $\mathcal{G}$  into  $m$  disjoint subregions based on the energy levels of  $\mathcal{L}_{\text{total}}(g)$ :

$$\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \dots \cup \mathcal{G}_m, \quad \mathcal{G}_n = \{g : u_{n-1} < \mathcal{L}_{\text{total}}(g) < u_n\} \quad (7)$$

where  $u_0 = -\infty$ ,  $u_m = +\infty$ , while  $u_1$  and  $u_{m-1}$  are specified by the user. Inspired by (Neal 2001), we define the flattened distribution  $\rho(g)$  as:

$$\rho(g) \propto \frac{P(g)}{\Psi^\zeta(\Theta, \mathcal{L}_{\text{total}}(g))}, \quad (8)$$

where  $\zeta > 0$  is a flattening hyperparameter controlling the degree of flattening, and  $\Psi(\Theta, \mathcal{L}_{\text{total}}(g))$  is a weighting function that takes the energy of current sample  $\mathcal{L}_{\text{total}}(g)$ , and a weight vector  $\Theta$  as inputs, and returns the corresponding flattening weight, where  $\Theta$  is defined as:

$$\Theta = \left\{ (\theta(1), \theta(2), \dots, \theta(m)) \mid 0 < \theta(1), \theta(2), \dots, \theta(m) < 1, \sum_{i=1}^m \theta(i) = 1 \right\}, \quad (9)$$

with  $\theta(i) = 1/m$  at the start of training.

To avoid gradient vanishing issues associated with the piecewise constant form of  $\Psi$ , as met in (Liang et al. 2007), we construct  $\Psi$  using a piecewise exponential interpolation function, following (Deng et al. 2020):

$$\begin{aligned} \Psi(\Theta, \mathcal{L}_{\text{total}}(g)) = \sum_{i=1}^m \mathbf{1}_{\{u_{i-1} \leq \mathcal{L}_{\text{total}}(g) \leq u_i\}} \times \\ \theta(i-1) \exp \left( (\log \theta(i) - \log \theta(i-1)) \frac{\mathcal{L}_{\text{total}}(g) - u_{i-1}}{\Delta u} \right) \end{aligned} \quad (10)$$

where  $1_A$  is the indicator function, equal to 1 when event  $A$  occurs and 0 otherwise. This formulation interpolates the discrete  $\theta(i)$  values exponentially based on the energy of the current sample  $g$ .

To derive the update rule for the flattened distribution  $\rho(g)$ , we compute the gradient:

$$\nabla_g \log \rho(g) = -\frac{\nabla_g \mathcal{L}_{\text{total}}(g)}{\tau} \times \left[ 1 + \zeta \tau \frac{\partial \log \Psi(\Theta, \mathcal{L}_{\text{total}}(g))}{\partial \mathcal{L}_{\text{total}}(g)} \right]. \quad (11)$$

The format (11) is expanded to:

$$\nabla_g \log \rho(g) = -\frac{\nabla_g \mathcal{L}_{\text{total}}(g)}{\tau} \times \left[ 1 + \zeta \tau \frac{\log \theta(J(g)) - \log(\theta(J(g)-1) \vee 1)}{\Delta u} \right] \quad (12)$$

where  $\Delta u = u_n - u_{n-1}$  for  $n \in \{2, \dots, m-1\}$  and  $J(g) \in \{1, 2, \dots, m\}$  denotes the index of the subregion containing the current sample  $g$ :

$$J(g) = \sum_{i=1}^m i \mathbf{1}_{u_{i-1} < \mathcal{L}_{\text{total}}(g) \leq u_i}. \quad (13)$$

Compared to the gradient under the original distribution, (12) introduces an additional gradient multiplier  $\nu$ :

$$\nu = 1 + \zeta \tau \frac{\log \theta(J(g)) - \log(\theta(J(g)-1) \vee 1)}{\Delta u}. \quad (14)$$

To align the update with the flattened distribution  $\rho(g)$ , the gradient multiplier is merged into the SGLD update (2):

$$g_k \leftarrow g_{k-1} - \lambda_{\text{lr}} \cdot \nu \cdot \nabla_g \mathbb{E}_{I \sim \mathcal{I}} [\mathcal{L}_{\text{total}}(g_{k-1}; I)] + \lambda_{\text{noise}} \cdot \epsilon, \quad (15)$$

Next, we update the weight vector  $\Theta$  to ensure that  $\Psi(\Theta, \mathcal{L}_{\text{total}}(g))$  produces appropriate flattening weights as training goes. Following (Deng et al. 2020), we employ a stochastic approximation (SA) approach to estimate  $\Theta$  during training. The goal of SA is to drive each  $\theta(i)$  to converge

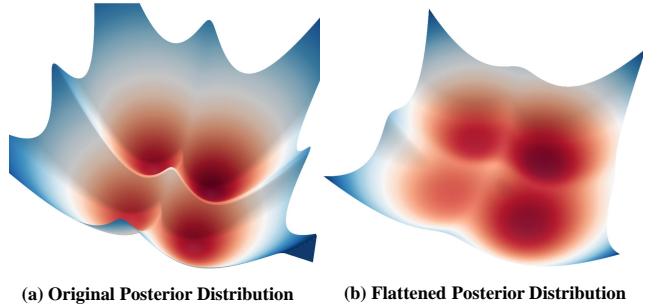


Figure 2: Original (a) and Flattened (b) posterior distribution. In the original distribution, High energy barriers between modes can trap the model in a single basin. The flattened distribution reduces these barriers, enabling free exploration across modes.

to the cumulative probability density of the corresponding subregion under the original distribution. At each iteration, before updating the Gaussian primitives, we perform:

$$\theta_k(i) = \theta_{k-1}(i) + \lambda_\theta \theta_{k-1}^\zeta (J(g_k)) \cdot (1_{i=J(g_k)} - \theta_{k-1}(i)), \quad (16)$$

where  $\lambda_\theta$  is the learning rate for updating  $\theta(i)$ . This step increases the weight of the current subregion  $J(g_k)$  while decreasing the weights of other subregions.

During exploration stage, we simultaneously update the Gaussian primitives  $g$  and the weight vector  $\Theta$  using the above rules. Consistent with 3DGSMCMC, we employ the gradients from Adam optimizer in place of raw gradients in (15) to enhance optimization stability. By flattening the posterior distribution, AW-SGLD enhances the exploration capability of the model, as illustrated in Figure 2, thereby increasing the likelihood of converging to the deepest mode—the one containing the optimal solution.

## 4.2 Local Quasi-Newton Direction-guided Adam optimizer

Although the flattened distribution based on the flat-histogram principle enhances the model’s exploration capability and helps escape from local traps, the ultimate goal of 3DGS remains to find the global optimum that minimizes the loss. Enhanced exploration alone does not guarantee precise convergence to the optimal point within a mode.

To improve convergence quality in the later stages of training, we switch to exploitation from exploration, and design a curvature-aware optimization strategy. While prior works have employed Newton’s method (Lan et al. 2025) or the Levenberg–Marquardt (LM) (Höllein et al. 2024) algorithm to accelerate optimization, these approaches require complicated calculations of the Hessian matrix or its approximation. Our objective is achieving precise convergence without incurring excessive computational overhead.

We propose a Local Quasi-Newton Direction-guided Adam Optimizer (LQNAdam). Here, “local” indicates that each Gaussian primitive is treated independently. We apply the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) (Nocedal and Wright 2006) algorithm to the posi-

| Methods     | MipNeRF360         |                    |                       | Tanks & Temples    |                    |                       | DeepBlending       |                    |                       |
|-------------|--------------------|--------------------|-----------------------|--------------------|--------------------|-----------------------|--------------------|--------------------|-----------------------|
|             | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) |
| MipNeRF     | 29.23              | 0.844              | 0.207                 | 22.22              | 0.759              | 0.257                 | 29.40              | 0.901              | 0.245                 |
| 3DGS        | 28.69              | 0.870              | 0.182                 | 23.14              | 0.841              | 0.183                 | 29.41              | 0.903              | 0.243                 |
| Scaffold-GS | 28.84              | 0.848              | 0.220                 | 23.96              | 0.853              | 0.177                 | <b>30.21</b>       | 0.906              | 0.254                 |
| FreGS       | 27.85              | 0.826              | 0.209                 | 23.96              | 0.841              | 0.183                 | 29.93              | 0.904              | <u>0.240</u>          |
| 3DHSGS      | 29.56              | 0.873              | 0.178                 | 24.49              | 0.857              | 0.169                 | 29.76              | 0.905              | 0.242                 |
| 3DGSMCMC    | 29.89              | <b>0.900</b>       | 0.190                 | 24.29              | 0.860              | 0.190                 | 29.67              | 0.900              | 0.320                 |
| SSS         | <u>29.90</u>       | 0.893              | <u>0.145</u>          | <b>24.87</b>       | <u>0.873</u>       | <b>0.138</b>          | 30.07              | <u>0.907</u>       | 0.247                 |
| Ours        | <b>29.96</b>       | 0.897              | <b>0.143</b>          | <u>24.80</u>       | <b>0.875</b>       | 0.139                 | 30.09              | <b>0.911</b>       | <b>0.229</b>          |

Table 1: Quantitative comparison between ours and baseline methods. For a fair comparison, we use the same resolution settings and maximum number of Gaussians as in 3DGSMCMC. The **best** and second-best results in each column are highlighted.

tional attributes  $\mu$  of each Gaussian primitive and estimate a quasi-Newton direction based on the past  $K$  steps. Following (Liu and Nocedal 1989), the value of  $K$  is chosen between 3 and 7 to balance computational cost and performance. This is motivated by the observation in 3DGS<sup>2</sup> (Lan et al. 2025) that positional attributes significantly influence rendering quality and that Gaussians are weakly coupled, enabling parallel estimation of local quasi-Newton directions. Details of the L-BFGS algorithm are provided in the Supplementary Material.

To ensure stable convergence, L-BFGS typically uses a line search to determine the step size. However, performing line search for each Gaussian primitive is impractical in our context. Instead, we treat the Quasi-Newton direction from L-BFGS as a pseudo-gradient and feed it into the Adam optimizer, computing the final update direction. This approach leverages Adam’s robustness while incorporating curvature-aware directions, yielding more accurate updates as the model approaches a solution. Notably, L-BFGS does not require computation of the Hessian matrix, making our optimizer compatible with various loss functions.

Let  $\mathbb{D}$  denote the quasi-Newton direction estimated by L-BFGS for a Gaussian’s positional attributes  $\mu$ . The final update direction is computed as  $Adam(\mathbb{D})$ . Within the Markov Chain Monte Carlo framework, the update rule for our Local Quasi-Newton Direction-guided Adam Optimizer is:

$$\mu_{t+1} = \mu_t - \lambda_{lr} \cdot Adam(\mathbb{D}) + \lambda_{noise} \cdot \epsilon_\mu. \quad (17)$$

In the later stages of training, we switch to the exploitation. In this stage, the L1 loss is replaced by L2 loss, and LQNAdam is adopted. We also disable the gradient multiplier  $\nu$ , allowing model updates to follow the original distribution, thereby focusing on enhancing convergence quality.

## 5 Experiments

**Implementation Details** All experiments with Opt3DGS are performed on a NVIDIA RTX 4090 GPU, with a total of 30,000 optimization iterations. The growth rate of Gaussian primitives is fixed at 5%, following the setting in 3DGSMCMC. In the Adaptive Weighted SGLD module, the posterior energy range is discretized into 200 uniform bins. For all scenes the energy interval is set to  $[0.0, 0.2]$ , except for the *train* scene in the Tanks & Temples dataset, where a wider

range of  $[0.0, 0.3]$  is used. A warm-up of 2,500 iterations is applied to stabilize energy estimates before adaptive weighting, and the flattening coefficient is fixed to  $\zeta = 0.75$  for all experiments. For the quasi-Newton updates, we employ an L-BFGS history size  $K$  of 5, and compute quasi-Newton directions for Gaussian positions in parallel on CUDA. The training process switches from the exploration phase to the exploitation phase at iteration 29,000, and the final 1,000 iterations are used for exploitation for fine convergence.

**Baseline Methods** We compare Opt3DGS with vanilla 3DGS (Kerbl et al. 2023) and several representative variants that aim to improve the optimization or representation of 3DGS: 3DHGS (Li et al. 2025), FreGS (Zhang et al. 2024), Scaffold-GS (Lu et al. 2024), 3DGSMCMC (Kheradmand et al. 2024), and SSS (Zhu et al. 2025), as well as MipNeRF (Barron et al. 2022) as a representative NeRF-based method. Among these baselines, 3DHGS and SSS improve the expressiveness of Gaussian primitives; Scaffold-GS introduces an MLP component to accelerate training and enhance quality; 3DGSMCMC employs a stochastic optimization framework based on MCMC; and FreGS applies frequency regularization to boost both convergence speed and rendering fidelity. All reported baseline results are taken from their original publications.

**Datasets and Metrics** Following prior work on 3DGS, we evaluate the proposed Opt3DGS on three widely used real-world datasets: **MipNeRF360** (Barron et al. 2022), which contains 3 outdoor scenes (*garden*, *bicycle*, *stump*) and 4 indoor scenes (*kitchen*, *bonsai*, *room*, *counter*); **DeepBlending** (Hedman et al. 2018), consisting of 2 indoor scenes (*dr-johnson* and *playroom*); and **Tanks & Temples** (Knapsch et al. 2017), with 2 outdoor scenes (*train* and *truck*). For quantitative evaluation, we adopt three widely used visual quality metrics (Zhang et al. 2018): PSNR, SSIM, and LPIPS, computed on the test images.

### 5.1 Benchmark Results

The quantitative comparison with various baselines across three benchmark datasets is summarized in Table 1. Our Opt3DGS achieves the best performance on 5 out of 9 metrics and ranks second on the remaining 4. Compared to 3DGSMCMC, which shares the same Gaussian representa-

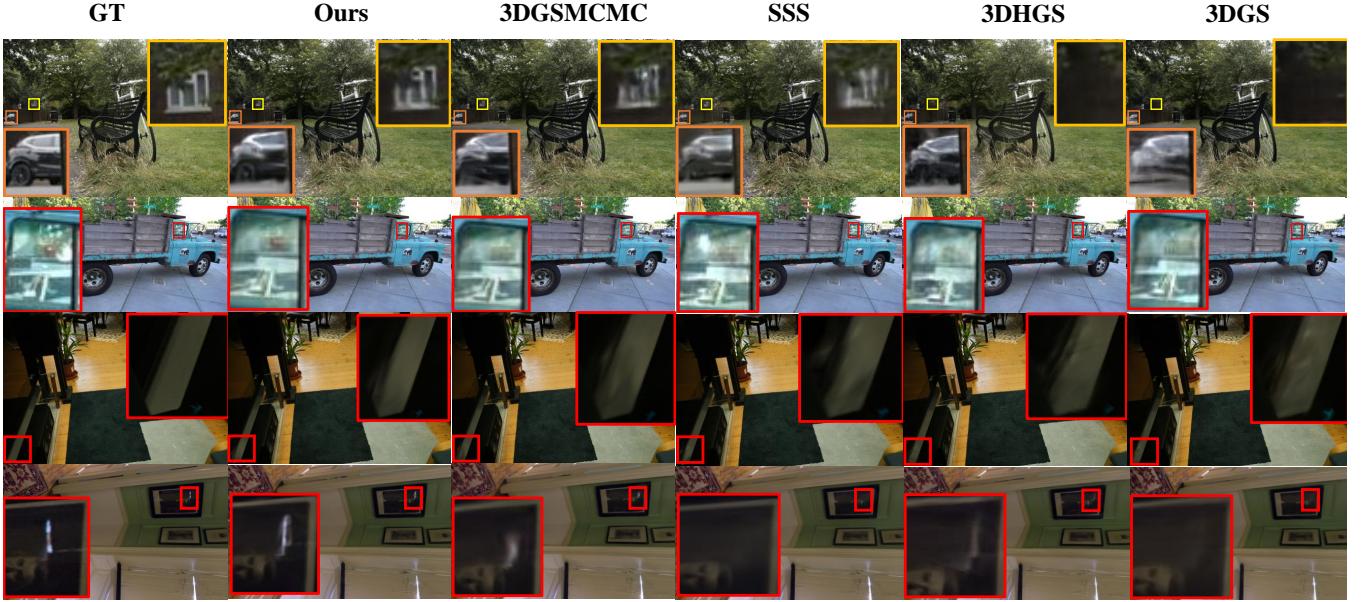


Figure 3: Visualization comparison. Our method achieves higher fidelity in challenging regions like distant and fine details.

| Methods  | MipNeRF360         |                    |                       | Tanks & Temples    |                    |                       | DeepBlending       |                    |                       |
|----------|--------------------|--------------------|-----------------------|--------------------|--------------------|-----------------------|--------------------|--------------------|-----------------------|
|          | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) |
| 3DGS     | 27.89              | 0.840              | 0.260                 | 21.93              | 0.800              | 0.270                 | 29.55              | 0.900              | 0.330                 |
| 3DGSMCMC | 29.72              | 0.890              | 0.190                 | 24.21              | 0.860              | 0.190                 | 29.71              | 0.900              | 0.320                 |
| Ours     | <b>29.78</b>       | <b>0.893</b>       | <b>0.149</b>          | <b>24.39</b>       | <b>0.865</b>       | <b>0.151</b>          | <b>29.90</b>       | <b>0.905</b>       | <b>0.236</b>          |

Table 2: Quantitative comparison between our method and baselines with random initialization. Although random initialization leads to a poor starting state and makes optimization challenging, our method achieves superior results across all metrics.

tion but differs solely in the optimization strategy, Opt3DGS achieves consistent performance gains across all metrics except for the SSIM on the MipNeRF360 dataset. On the Tanks & Temples dataset, we achieve PSNR/SSIM/LPIPS scores of 24.80 / 0.875 / 0.139, compared to 24.29 / 0.860 / 0.190 for 3DGSMCMC, representing improvements of 2.09%, 1.74%, and 26.84% respectively. Compared to the previous state-of-the-art SSS, which improves both the 3DGS representation and the training optimization, we achieve comparable or better results. On the DeepBlending dataset, we achieve 30.09 / 0.911 / 0.229, compared to 30.07 / 0.907 / 0.247 for SSS, representing improvements of 0.06%, 0.4%, and 7.2% respectively. These results confirm that better posterior exploration and convergence, rather than architectural modifications, can yield substantive performance improvements. We present qualitative comparisons of the novel view synthesis in Figure 3. We compare our method with several baselines: 3DGSMCMC, SSS, 3DHGS, and 3DGS. For a fair comparison, our method adopts the same configuration as 3DGSMCMC, and the other methods use their default settings. We show results on four novel views, where our method demonstrates superior rendering fidelity, particularly in distant background regions, fine geometric details,

and subtle scene structures that are difficult to capture.

## 5.2 Performance in Challenging Conditions

We further evaluate the performance of our optimization framework Opt3DGS under various challenging conditions.

**Random Initialization** By leveraging the exploration capability of stochastic noise, 3DGSMCMC achieves good rendering quality even without using Structure-from-Motion (SfM) initialization, instead relying on random initialization. For 3DGS task, random initialization means the model starts far from high-quality solutions, thereby significantly increasing the difficulty of finding a good solution. We report quantitative results under random initialization on all datasets in Table 2, comparing Opt3DGS with 3DGSMCMC and 3DGS. For fairness, our method uses the same random initialization scheme, training image resolution, and maximum number of Gaussians as 3DGSMCMC. Our method outperforms all baselines across all 9 metrics on the 3 datasets, showing Opt3DGS is more effective at guiding the model toward high-quality solutions even when starting from poor initial states.

| Methods                      | Train              |                    |                       |           | Truck              |                    |                       |           |
|------------------------------|--------------------|--------------------|-----------------------|-----------|--------------------|--------------------|-----------------------|-----------|
|                              | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) | Time      | PSNR( $\uparrow$ ) | SSIM( $\uparrow$ ) | LPIPS( $\downarrow$ ) | Time      |
| Baseline (3DGSMCMC)          | 22.47              | 0.830              | 0.240                 | <b>11</b> | 26.11              | 0.890              | 0.140                 | <b>22</b> |
| Baseline + AW-SGLD           | 22.74              | 0.841              | 0.180                 | 12        | 26.49              | 0.901              | 0.104                 | 22        |
| Baseline + AW-SGLD + LQNAdam | <b>23.01</b>       | <b>0.846</b>       | <b>0.176</b>          | 12        | <b>26.61</b>       | <b>0.903</b>       | <b>0.102</b>          | 23        |

Table 3: Ablation study on the Tanks & Temples dataset. AW-SGLD refers to the Adaptive Weighted SGLD component and LQNAdam denotes the Local Quasi-Newton Direction-guided Adam optimizer. Time is reported in minutes.

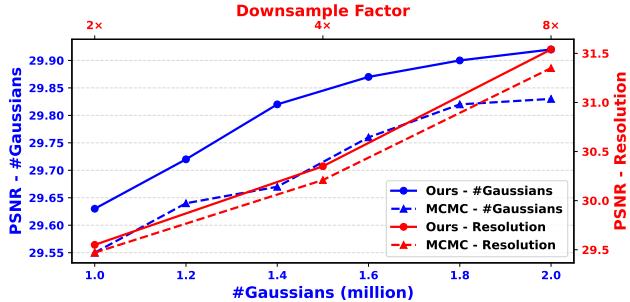


Figure 4: PSNR results on the MipNeRF dataset with different image resolutions(red) and the maximum number of Gaussians(blue).

**Higher Image Resolution** Higher input image resolution increases the difficulty of scene fitting, as finer details and higher-frequency signals demand more accurate geometry and appearance reconstruction. This makes the posterior distribution landscape more complex and raises the risk of converging to suboptimal local modes. In Figure 4, we report PSNR comparisons between 3DGSMCMC and our method at different resolutions, on the MipNeRF360 dataset. Our method consistently outperforms the base model across all three resolution settings, demonstrating its superior robustness and effectiveness under more challenging reconstruction conditions.

**Limited Number of Gaussians** When the number of available Gaussians is reduced, the model’s representational capacity decreases. We evaluate performance under various Gaussian budget constraints on the MipNeRF360 dataset as shown in Figure 4. Our method consistently outperforms the base model, 3DGSMCMC, across all settings. This demonstrates that even with limited representational capacity, our optimization framework remains effective at guiding the model to higher-quality convergence.

### 5.3 Ablation Studies

We conduct ablation experiments on the Tanks & Temples dataset in Table 3. All experimental settings (including image resolution and the number of Gaussians) are kept identical to those in 3DGSMCMC. The results show that incorporating AW-SGLD alone improves rendering quality by encouraging better exploration and reducing the risk of local entrapment, while the subsequent use of LQNAdam further refines the solution and leads to higher-quality convergence

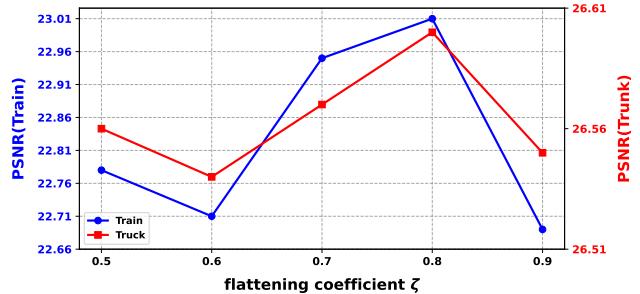


Figure 5: Ablation Study about flattening coefficient  $\zeta$  on the Tanks & Temples Dataset.

through curvature-aware updates.

The flattening coefficient  $\zeta$  is a key hyperparameter in our optimization framework, with larger values producing flatter posterior distributions. The ablation study on  $\zeta$  is presented in Figure 5. We observe that Opt3DGS performs best on the Tanks & Temples dataset when  $\zeta$  values is near 0.8. This setting generalizes well across all evaluated datasets.

## 6 Conclusion

In this paper, we present Opt3DGS, a novel and effective optimization framework for 3DGS. We decompose the training process into two stages—exploration and exploitation—and provide an analysis of the limitations of existing optimization strategies. In the exploration stage, we introduce Adaptive Weighted SGLD, which enables the model to escape local minima and increases the likelihood of reaching globally optimal solutions. In the exploitation stage, we design a Local Quasi-Newton Direction-guided Adam optimizer to achieve more accurate convergence. Our approach improves the performance of 3DGS purely through optimization enhancements, without modifying the Gaussian representation, introducing auxiliary networks, or incurring significant additional computational costs, yet it still achieves state-of-the-art rendering quality. Looking forward, the modular nature of Opt3DGS, independent of representation or architecture, makes it a promising replacement for the optimization component in various 3DGS-based systems. Its foundation in posterior distribution reshaping and curvature-aware updates also enables the extension of optimization-centric techniques to other areas of explicit differentiable rendering.

## Acknowledgements

This research was funded by the National Natural Science Foundation of China, Grant No. 42571412.

## References

- Barron, J. T.; Mildenhall, B.; Tancik, M.; Hedman, P.; Martin-Brualla, R.; and Srinivasan, P. P. 2021. Mip-NeRF: a multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 5855–5864.
- Barron, J. T.; Mildenhall, B.; Verbin, D.; Srinivasan, P. P.; and Hedman, P. 2022. Mip-NeRF 360: unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5470–5479.
- Cen, J.; Fang, J.; Yang, C.; Xie, L.; Zhang, X.; Shen, W.; and Tian, Q. 2025. Segment any 3D gaussians. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, 1971–1979.
- Chen, D.; Li, H.; Ye, W.; Wang, Y.; Xie, W.; Zhai, S.; Wang, N.; Liu, H.; Bao, H.; and Zhang, G. 2024. PGSR: planar-based gaussian splatting for efficient and high-fidelity surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*.
- Chen, Y.; Jiang, J.; Jiang, K.; Tang, X.; Li, Z.; Liu, X.; and Nie, Y. 2025. DashGaussian: optimizing 3D gaussian splatting in 200 seconds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 11146–11155.
- Deng, W.; Lin, G.; and Liang, F. 2020. A contour stochastic gradient langevin dynamics algorithm for simulations of multi-modal distributions. *Advances in neural information processing systems*, 33: 15725–15736.
- Guédon, A.; and Lepetit, V. 2024. SuGaR: surface-aligned gaussian splatting for efficient 3D mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5354–5363.
- Hedman, P.; Philip, J.; Price, T.; Frahm, J.-M.; Drettakis, G.; and Brostow, G. 2018. Deep blending for free-viewpoint image-based rendering. *ACM Transactions on Graphics (ToG)*, 37(6): 1–15.
- Höllein, L.; Božič, A.; Zollhöfer, M.; and Nießner, M. 2024. 3DGS-LM: faster gaussian-splatting optimization with Levenberg-Marquardt. *arXiv preprint arXiv:2409.12892*.
- Hou, Q.; Rauwendaal, R.; Li, Z.; Le, H.; Farhadzadeh, F.; Porikli, F.; Bourd, A.; and Said, A. 2024. Sort-free gaussian splatting via weighted sum rendering. *arXiv preprint arXiv:2410.18931*.
- Huang, B.; Yu, Z.; Chen, A.; Geiger, A.; and Gao, S. 2024. 2D gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 conference papers*, 1–11.
- Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4): 139–1.
- Kheradmand, S.; Rebain, D.; Sharma, G.; Sun, W.; Tseng, Y.-C.; Isack, H.; Kar, A.; Tagliasacchi, A.; and Yi, K. M. 2024. 3D gaussian splatting as markov chain monte carlo. *Advances in Neural Information Processing Systems*, 37: 80965–80986.
- Kingma, D. P.; and Ba, J. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Knapitsch, A.; Park, J.; Zhou, Q.-Y.; and Koltun, V. 2017. Tanks and temples: benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36(4): 1–13.
- Lan, L.; Shao, T.; Lu, Z.; Zhang, Y.; Jiang, C.; and Yang, Y. 2025. 3DGS<sup>2</sup>: near second-order converging 3D gaussian splatting. *arXiv preprint arXiv:2501.13975*.
- Li, H.; Liu, J.; Sznaier, M.; and Camps, O. 2025. 3D-HGS: 3D half-gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 10996–11005.
- Liang, F.; Liu, C.; and Carroll, R. J. 2007. Stochastic approximation in monte carlo computation. *Journal of the American Statistical Association*, 102(477): 305–320.
- Liu, D. C.; and Nocedal, J. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1): 503–528.
- Lu, T.; Yu, M.; Xu, L.; Xiangli, Y.; Wang, L.; Lin, D.; and Dai, B. 2024. Scaffold-GS: structured 3D gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 20654–20664.
- Matsuki, H.; Murai, R.; Kelly, P. H.; and Davison, A. J. 2024. gaussian splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18039–18048.
- Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2021. NeRF: representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1): 99–106.
- Müller, T.; Evans, A.; Schied, C.; and Keller, A. 2022. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4): 1–15.
- Neal, R. M. 2001. Annealed importance sampling. *Statistics and computing*, 11(2): 125–139.
- Nocedal, J.; and Wright, S. J. 2006. *Numerical optimization*. Springer.
- Pehlivan, H.; Camiletto, A. B.; Foo, L. G.; Habermann, M.; and Theobalt, C. 2025. Second-order optimization of gaussian splats with importance sampling. *arXiv preprint arXiv:2504.12905*.
- Pumarola, A.; Corona, E.; Pons-Moll, G.; and Moreno-Noguer, F. 2021. D-NeRF: neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10318–10327.
- Rota Bulò, S.; Porzi, L.; and Kortscheder, P. 2024. Revising densification in gaussian splatting. In *European Conference on Computer Vision*, 347–362. Springer.

- Svitov, D.; Morerio, P.; Agapito, L.; and Del Bue, A. 2024. Billboard splatting (BBSplat): learnable textured primitives for novel view synthesis. *arXiv preprint arXiv:2411.08508*.
- Wang, F.; and Landau, D. P. 2001. Efficient, multiple-range random walk algorithm to calculate the density of states. *Physical review letters*, 86(10): 2050.
- Yan, Z.; Low, W. F.; Chen, Y.; and Lee, G. H. 2024. Multi-scale 3D gaussian splatting for anti-aliased rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 20923–20931.
- Yang, Z.; Yang, H.; Pan, Z.; and Zhang, L. 2023. Real-time photorealistic dynamic scene representation and rendering with 4D gaussian splatting. *arXiv preprint arXiv:2310.10642*.
- Ye, Z.; Li, W.; Liu, S.; Qiao, P.; and Dou, Y. 2024. AbsGS: recovering fine details in 3D gaussian splatting. In *Proceedings of the 32nd ACM International Conference on Multimedia*, 1053–1061.
- Yu, Z.; Chen, A.; Huang, B.; Sattler, T.; and Geiger, A. 2024a. Mip-splatting: alias-free 3D gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 19447–19456.
- Yu, Z.; Sattler, T.; and Geiger, A. 2024b. Gaussian opacity fields: efficient adaptive surface reconstruction in unbounded scenes. *ACM Transactions on Graphics (ToG)*, 43(6): 1–13.
- Zhang, J.; Zhan, F.; Xu, M.; Lu, S.; and Xing, E. 2024. FreGS: 3D gaussian splatting with progressive frequency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 21424–21433.
- Zhang, K.; Riegler, G.; Snavely, N.; and Koltun, V. 2020. NeRF++: analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*.
- Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 586–595.
- Zhu, J.; Yue, J.; He, F.; and Wang, H. 2025. 3D student splatting and scooping. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 21045–21054.