
Metropolis-Hastings Sampling for 3D Gaussian Reconstruction

Hyunjin Kim¹ Haebeom Jung² Jaesik Park²

¹UC San Diego ²Seoul National University
hyk071@ucsd.edu {haebeom.jung, jaesik.park}@snu.ac.kr

Abstract

We propose an adaptive sampling framework for 3D Gaussian Splatting (3DGS) that leverages comprehensive multi-view photometric error signals within a unified Metropolis-Hastings approach. Vanilla 3DGS heavily relies on heuristic-based density-control mechanisms (e.g., cloning, splitting, and pruning), which can lead to redundant computations or premature removal of beneficial Gaussians. Our framework overcomes these limitations by reformulating densification and pruning as a probabilistic sampling process, dynamically inserting and relocating Gaussians based on aggregated multi-view errors and opacity scores. Guided by Bayesian acceptance tests derived from these error-based importance scores, our method substantially reduces reliance on heuristics, offers greater flexibility, and adaptively infers Gaussian distributions without requiring predefined scene complexity. Experiments on benchmark datasets, including Mip-NeRF360, Tanks and Temples and Deep Blending, show that our approach reduces the number of Gaussians needed, achieving faster convergence while matching or modestly surpassing the view-synthesis quality of state-of-the-art models. Our project page is available at <https://hjhyunjinkim.github.io/MH-3DGS>.

1 Introduction

Novel view synthesis (NVS) focuses on creating photo-realistic images of a 3D scene from unseen viewpoints. It is significant due to its broad applicability across various real-world scenarios. While Neural Radiance Fields (NeRF) [36] has largely advanced Novel View Synthesis by mapping 3D locations and view directions to view-dependent colors and volumetric densities, its slow rendering speed hinders real-world deployment. To address these limitations, 3D Gaussian Splatting (3DGS) [20] recently emerged, enabling real-time photo-realistic rendering by representing complex scenes with explicit 3D Gaussians, each of which is projected to the screen and defined by its position, anisotropic covariance, and opacity.

Despite these advantages, 3DGS and its extensions [18, 20, 56] rely heavily on heuristic-based adaptive density control mechanisms, such as cloning, splitting, and pruning by introducing fixed thresholds on opacity, size, and positional gradients in the view space. This leads to inflation of memory with redundant Gaussians when thresholds are loose, or to a sacrifice of fidelity when they are tight. To tackle this problem, Kheradmand et al. [22] recast 3DGS updates as Stochastic Gradient Langevin Dynamics (SGLD), framing Gaussian updates as Markov Chain Monte Carlo (MCMC) samples from an underlying probability distribution representing scene fidelity. As SGLD updates Gaussians in the direction of the image-reconstruction gradient and adds a small noise term, the procedure behaves like a gradient-based optimizer. Although it replaces heuristics with state transitions, it still fixes the total number of Gaussians upfront, limiting adaptability to scenes of varying complexity.

Motivated by these considerations, we propose an adaptive sampling framework that uniquely leverages comprehensive multi-view error signals within a unified Metropolis-Hastings (MH) [6, 35] approach. We reformulate 3D Gaussian Splatting (3DGS) as an MH sampling process and newly derive a concise, rigorously proven acceptance probability grounded in the classical MH rule. By casting the densification and pruning stages of 3DGS as a probabilistic sampling problem, our method selectively adds Gaussians in regions with high visual or geometric significance as indicated by aggregated view-consistent photometric discrepancies. By deriving Gaussian-level importance scores from opacity and multi-view error metrics, we generate Gaussian candidates through coarse-to-fine proposals, which are subsequently accepted or rejected via Bayesian acceptance tests. This probabilistic process adaptively converges to a well-balanced Gaussian distribution with only minimal reliance on heuristics. It requires no prior knowledge of scene complexity while seamlessly integrating with existing 3DGS workflows to achieve better reconstruction quality, capturing both coarse structures and fine details.

We demonstrate the effectiveness of our approach on real-world scenes from the Mip-NeRF360 [1], Tanks and Temples [23], and Deep Blending [17] datasets. Our method reduces the number of Gaussians needed to represent each scene solely through an improved sampling mechanism, while consistently matching or surpassing the view-synthesis quality of 3DGS across diverse scenes. Additionally, our approach converges faster than 3DGS-MCMC [22], achieving target PSNR values in fewer training iterations and reduced wall-clock time. By coupling principled probabilistic inference with dense multi-view error signals, our method matches the rendering accuracy of 3DGS-MCMC [22] with fewer Gaussians, demonstrating that comparable quality can be achieved with a leaner, more generalizable representation that converges more rapidly.

We summarize our contributions as follows:

- We introduce an adaptive Metropolis-Hastings (MH) framework that replaces densification heuristics in 3DGS with a closed-form Bayesian posterior for each Gaussian and a lightweight photometric surrogate that turns the MH acceptance ratio into a single logistic-voxel product.
- To this end, we derive Metropolis-Hastings transition equations that are specialized to 3DGS and prove that this MH sampler is mathematically sound, demonstrating that the heuristic densification rule in 3DGS can be recast as a principled MH update.
- Our experiments show that our framework achieves compelling view synthesis quality while converging faster.

2 Related Work

2.1 Neural 3D Scene Representation

Novel View Synthesis generates realistic images from novel viewpoints that differ from the original captures [56]. Neural Radiance Fields (NeRF) [36] pioneered this field by using MLPs to model 3D scenes from multi-view 2D images, inspiring extensions for large scenes [13, 46, 52], dynamic content [8, 14, 40, 41, 49], and pose-free reconstruction [2, 19, 27, 33, 34, 57]. However, NeRF’s extensive training and rendering time prompted numerous acceleration efforts [4, 10, 11, 37, 42, 45].

3D Gaussian Splatting (3DGS) [20] emerged as a compelling alternative, using differentiable rasterization of 3D Gaussian primitives initialized from sparse point clouds generated from Structure-from-Motion [44]. This approach enables efficient optimization and high-resolution output, spurring extensions in dynamic scenes [25, 26, 31, 51, 53], 3D generation [5, 29, 47, 48, 55], and large-scale reconstruction [21, 28, 30]. Yet, 3DGS suffers from a significant memory burden due to covariance matrices and spherical harmonics, leading to memory-efficient variants that rely on heuristic density-control rules [9, 15, 16, 24, 38, 39, 50].

Recent work has explored probabilistic sampling in 3D reconstruction: Goli et al. [12] and Bortolon et al. [3] apply Metropolis-Hastings for sample refinement and surface point selection in NeRF. While these methods use Metropolis-Hastings for specific components, our work reformulates the entire 3D reconstruction problem as a Metropolis-Hastings sampling process, replacing heuristic density controls in 3DGS with a unified probabilistic framework that enhances both efficiency and generality.

2.2 Adaptive Density Control for 3D Gaussian Splatting

Kheradmand et al. [22] recently introduced the use of Stochastic Gradient Langevin Dynamics (SGLD) for sampling and optimization in 3DGS, combined with an opacity-driven relocation strategy to shift low-opacity Gaussians towards higher-opacity regions, while enforcing a fixed limit on the total number of Gaussians. Additionally, Rota Bulò et al. [43] introduces error-based densification with perceptual error prioritization and sets a global limit on the number of Gaussians. Similarly, Ye et al. [54] addresses the limitations of gradient collision in adaptive density control by proposing a homodirectional gradient as a criterion for densification. Moreover, Mallick et al. [32] presents a score-based guided densification approach that uses training-time priors to restrict growth. More recently, Deng et al. [7] introduced deterministic geometric splitting rules for efficient density control. As 3DGS-MCMC [22] aligns most closely with our objectives and delivers the best overall reconstruction quality, it serves as our primary baseline.

In contrast, our proposed framework leverages a unified Metropolis-Hastings sampling strategy grounded in comprehensive multi-view error signals. Rather than relying on thresholds, our method computes Gaussian importance scores from aggregated multi-view photometric errors and dynamically adapts the number and distribution of Gaussians via probabilistic inference. We derive the MH acceptance rule in the context of 3DGS and prove that our importance-weighted proposals preserve detailed balance with respect to the multi-view photometric likelihood. This rigorous formulation distinguishes our framework from prior methods and enables natural scaling with scene complexity.

3 Preliminary

3.1 3D Gaussian Splatting

3D Gaussian Splatting (3DGS) [20] represents the 3D scene as a set of anisotropic 3D Gaussians, further optimized by differentiable tile rasterization. Each 3D Gaussian $\mathcal{G}$ is characterized by a position vector μ and a covariance matrix Σ :

$$\mathcal{G}(x) = e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}, \quad \Sigma = R S S^\top R^\top, \quad (1)$$

where the covariance matrix is factorized into a scaling matrix S and a rotation matrix R . The transformed 2D covariance matrix Σ' is calculated by the viewing transform W and the Jacobian of the affine transformation of the projective transformation J as $\Sigma' = J W \Sigma W^\top J^\top$. The color C of a pixel can be computed by blending $\mathcal{N}$ ordered points overlapping the pixel as $C = \sum_{i \in \mathcal{N}} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j)$, where c_i and α_i represent the view-dependent color and opacity computed from a 2D Gaussian distribution.

To achieve a high-quality representation, 3DGS employs adaptive density control that clones Gaussians in under-reconstructed areas and splits Gaussians in over-reconstructed regions. During densification, 3DGS evaluates the positional gradient $\nabla_p \mathcal{L}$ of each Gaussian, refining further if the gradient exceeds a predefined threshold τ_p . If a Gaussian surpasses the size threshold τ_s , indicating over-reconstruction, it is split into smaller Gaussians. Alternatively, under-reconstructed areas prompt cloning by shifting Gaussians along the gradient direction for better scene representation.

3.2 Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm [6, 35] is a widely used Markov Chain Monte Carlo (MCMC) method for sampling from complex, high-dimensional probability distributions. The objective is to construct a Markov chain whose stationary distribution $\pi(\mathbf{x})$ matches the target distribution, enabling efficient sampling where direct methods are impractical.

Given a target distribution $\pi(\mathbf{x})$ defined over a state space $\mathcal{X}$, the algorithm generates a sequence $\{\mathbf{x}^{(t)}\}_{t=1}^N$ of samples by iteratively proposing candidates from a proposal distribution $q(\mathbf{x}, \mathbf{y})$ and accepting or rejecting them based on an acceptance probability. At each iteration t , a new candidate $\mathbf{y}$ is proposed from the distribution $q(\mathbf{x}^{(t)}, \cdot)$. The acceptance probability is further computed as

$$\rho(\mathbf{x}^{(t)}, \mathbf{y}) = \min \left(1, \frac{\pi(\mathbf{y}) q(\mathbf{y}, \mathbf{x}^{(t)})}{\pi(\mathbf{x}^{(t)}) q(\mathbf{x}^{(t)}, \mathbf{y})} \right). \quad (2)$$

The candidate $\mathbf{y}$ is accepted with probability $\rho(\mathbf{x}^{(t)}, \mathbf{y})$; otherwise, the chain remains at the current state, $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$. The acceptance probability is designed to ensure that the Markov chain satisfies the detailed balance condition:

$$\pi(\mathbf{x})q(\mathbf{x}, \mathbf{y})\rho(\mathbf{x}, \mathbf{y}) = \pi(\mathbf{y})q(\mathbf{y}, \mathbf{x})\rho(\mathbf{y}, \mathbf{x}) \quad (3)$$

which guarantees that $\pi(\mathbf{x})$ is the stationary distribution of the Markov chain. The distribution of the generated samples converges to the target distribution $\pi(\mathbf{x})$ as the number of iterations N increases.

4 Method

Our method begins with a key design choice: unlike 3DGS, which densifies points at fixed thresholds, we employ a lean point-based sampler that keeps the most informative Gaussians, sharply reducing storage while preserving fidelity. Section 4.1 lays the theoretical groundwork for this: we extend the Metropolis–Hastings algorithm to 3DGS and rigorously derive scene-adaptive equations that generalize the traditional formulation. Section 4.2 describes how the sampler proposes candidates guided by multi-view photometric and opacity errors, and Sec. 4.3 presents the MH acceptance test with a newly derived practical expression for the MH acceptance probability. The complete algorithmic flow is in Algorithm 1 and 2 in Appendix D.

4.1 Formulation of the Metropolis-Hastings Framework

Our main objective is to eliminate heuristic dependence in adaptive density control by recasting densification and pruning as a principled sampling problem. Specifically, we interpret each modification of the Gaussian set as a proposal in a Metropolis-Hastings Markov Chain. Each step in the chain perturbs the current scene representation Θ by (i) **inserting** new Gaussians near pixels that still exhibit high multi-view error or (ii) **relocating** Gaussians whose opacity has collapsed. The proposed scene Θ' is then accepted or rejected according to a probabilistic rule that, in expectation, favors configurations that reproduce the captured images better while remaining spatially sparse.

In our formulation, a scene is described as a collection $\Theta = \{g_i\}_{i=1}^N$ of Gaussian splats, where each splat contains parameters $(\mathbf{x}_i, \Sigma_i, \mathbf{c}_i, \alpha_i)$, representing the position, covariance, color, and opacity of a Gaussian. We define the overall loss function as:

$$\mathcal{L}(\Theta) = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}} + \lambda_{\text{opacity}}\bar{\alpha} + \lambda_{\text{scale}}\bar{\Sigma} \quad (4)$$

where $\mathcal{L}_1$ and $\mathcal{L}_{\text{D-SSIM}}$ are the loss terms from the original 3DGS [20] method. The last two terms are regularizers on the mean opacity and covariance, used to penalize large or widespread opacities and large Gaussian spreads, respectively, as advocated by 3DGS-MCMC [22]. We use $\lambda = 0.2$, $\lambda_{\text{opacity}} = 0.01$ and $\lambda_{\text{scale}} = 0.01$ in our framework.

From a likelihood viewpoint, we treat the minimization of $\mathcal{L}(\Theta)$ equivalent to maximizing the likelihood of the observed image set $\mathcal{D} = \{I^{(v)}\}$. Thus, we can interpret this as:

$$\ln p(\mathcal{D} | \Theta) \approx -\mathcal{L}(\Theta) \quad (5)$$

Since likelihood terms are usually expressed with base e , we adopt the natural logarithm ($\ln$) throughout to keep the formulas clean and consistent.

Furthermore, photometric fidelity alone does not stop the optimizer from piling Gaussians onto the same patch once it converges locally. Thus, we discourage such redundancies by introducing a voxel-wise prior:

$$\ln p(\Theta) = -\lambda_v \sum_{v \in \mathcal{V}} \ln(1 + c_{\Theta}(v)) \quad (6)$$

where $c_{\Theta}(v)$ is the number of Gaussians in voxel v . This prior encourages at most one or two Gaussians per voxel. Since the cost grows logarithmically, empty cells get almost no penalty, but the penalty rises quickly once the voxel becomes crowded. Combining Eq. 4 and 6, we are able to define a negative log-posterior:

$$\mathcal{E}(\Theta) = \mathcal{L}(\Theta) + \lambda_v \sum_{v \in \mathcal{V}} \ln(1 + c_{\Theta}(v)) \quad (7)$$

Based on this derivation, we can define the posterior density based on Bayesian statistics:

$$\pi(\Theta) = \frac{1}{Z} e^{-\mathcal{E}(\Theta)}, \quad (8)$$

where Z is a normalization constant (See Appendix A for details). Our framework samples Gaussian configurations Θ in proportion to $\pi(\Theta)$ and adaptively refines the 3D Gaussian representation to cover undersampled regions. To facilitate this sampling, we construct a Markov Chain whose states are successive scene representations $\Theta_0, \Theta_1, \dots, \Theta_t$. From a given state Θ_t , we draw a proposal Θ' by adding Gaussians in high-error areas or relocating low opacity Gaussians, with probability density $q(\Theta' | \Theta_t)$. The construction of $q(\Theta' | \Theta_t)$ is detailed in Sec. 4.2.

We accept or reject Θ' according to the Metropolis-Hastings (MH) rule:

$$\rho(\Theta' | \Theta_t) = \min\left(1, \frac{\pi(\Theta') q(\Theta_t | \Theta')}{\pi(\Theta_t) q(\Theta' | \Theta_t)}\right), \quad \rho(\Theta' | \Theta_t) \in [0, 1] \quad (9)$$

where $\rho(\Theta' | \Theta_t)$ indicates the standard MH acceptance probability. At each step we draw $u \sim \mathcal{U}(0, 1)$; if $u < \rho$, the proposal is accepted and we set $\Theta_{t+1} = \Theta'$, otherwise we keep $\Theta_{t+1} = \Theta_t$. We explain how the MH rule accepts or rejects Θ' again in Sec. 4.3. Over many iterations, this Markov Chain converges toward our target distribution $\pi(\Theta)$, systematically densifying the Gaussian set, reducing loss, and relocating Gaussians that fail to improve image fidelity.

4.2 Proposal Generation

Building on the Metropolis-Hastings framework, we now specify how a candidate configuration Θ' is generated from the current state Θ . The guiding principle is to concentrate proposals in problematic regions where Θ either (1) lacks sufficient **opacity**, indicating missing or overly sparse geometry, or (2) exhibits a large **photometric error**.

4.2.1 Per-pixel importance field

At every densification step, we first choose a working view subset $\mathcal{C}_t = \{c_{t,1}, \dots, c_{t,k_t}\}$ from the full training set $\mathcal{C}$ to ensure balanced use of all viewpoints over time and broad viewing-angle coverage. Note that this selected subset $\mathcal{C}_t$ is used only for computing the importance field. The subset size is annealed according to

$$k_t = \max\left(1, \lfloor (1 - \eta_t) |\mathcal{C}| \rfloor\right), \quad \eta_t = \frac{t - t_{\min}}{t_{\max} - t_{\min}} \quad (10)$$

so that early iterations ($\eta_t \approx 0$) ensure broad coverage across diverse views, whereas we reduce the subset size to focus on specific vantage points requiring finer coverage. We use a *round-robin* schedule that walks through $\mathcal{C}$ in contiguous blocks of length k_t , as this allows every viewpoint to contribute equally to the error signal after a few iterations. For each viewpoint $c \in \mathcal{C}_t$, we compare the current prediction $\hat{I}^{(c)}$ to the ground truth image $I^{(c)}$ and record per-pixel SSIM and $\mathcal{L}_1$ errors. To construct a unified importance field for Gaussian proposal generation, we aggregate these error maps and construct view-averaged importance maps:

$$\text{SSIM}_{\text{agg}}(p) = \frac{1}{k_t} \sum_{c \in \mathcal{C}_t} [1 - \text{SSIM}(I^{(c)}(p), \hat{I}^{(c)}(p))], \quad \mathcal{L}_{1\text{agg}}(p) = \frac{1}{k_t} \sum_{c \in \mathcal{C}_t} |I^{(c)}(p) - \hat{I}^{(c)}(p)| \quad (11)$$

where k is the number of selected viewpoints, and p indicated image pixels. This aggregation produces a spatial field that captures the difficulty of reconstruction across the current viewing subset. We then feed $\text{SSIM}_{\text{agg}}(p)$ and $\mathcal{L}_{1\text{agg}}(p)$ along with the opacity $O(p)$, which is directly obtained by projecting the per-Gaussian opacity on pixel p , into our importance score. After applying robust normalization to each quantity (to mitigate outliers and match dynamic ranges), the three cues are fused through a logistic function:

$$s(p) = \sigma\left(\alpha O(p) + \beta \text{SSIM}_{\text{agg}}(p) + \gamma \mathcal{L}_{1\text{agg}}(p)\right), \quad \sigma(z) = \frac{1}{1 + e^{-z}} \quad (12)$$

where α, β, γ control the relative emphasis on opacity, structural similarity, and photometric fidelity, respectively. We use $\alpha = 0.8, \beta = 0.5, \gamma = 0.5$ in our tests. The scalar field $s \in (0, 1)$ highlights the pixels that are simultaneously under-covered and photometrically inaccurate across the chosen views.

4.2.2 Mapping Pixels to Gaussian Importance

Given the 3D position $\mathbf{x}_i \in \mathbb{R}^3$ of a Gaussian g_i , we obtain its image-plane coordinates via the calibrated projection $\Pi: \mathbb{R}^3 \rightarrow \mathbb{R}^2$. Rather than integrating the importance field over the full elliptical footprint of g_i , which requires evaluating hundreds of pixels per splat, we adopt a lightweight surrogate that uses the floor operator to sample only the pixel lying directly beneath the center of g_i :

$$I(i) = s(\lfloor \Pi(\mathbf{x}_i) \rfloor) \quad (13)$$

where $s(\cdot)$ is the logistic importance map in Eq. 12. The importance weight $I(i)$ is then normalized once per iteration to give a categorical distribution $P_{\text{pick}}(i) = I(i) / \sum_j I(j)$.

4.2.3 Proposal Distribution

Having established a Bayesian viewpoint on our 3D Gaussian representation and a per-pixel importance score $s(p)$, we now detail the proposal mechanism in our Metropolis-Hastings (MH) sampler. We maintain a Markov Chain $\Theta_0 \rightarrow \Theta_1 \rightarrow \dots$, each Θ_t comprising the current Gaussian set. From state Θ_t , we form a proposal Θ' in **two** phases.

During the **Coarse Phase**, we sample a batch $\mathcal{I}_c = \{i_1^c, \dots, i_{B_c}^c\}$ of size B_c from P_{pick} . For every index $i \in \mathcal{I}_c$, we create a new Gaussian:

$$g_i^c = (\mathbf{x}_i + \boldsymbol{\delta}_i^c, \Sigma_i, \mathbf{c}_i, \alpha_i), \quad \boldsymbol{\delta}_i^c \sim \mathcal{N}(\mathbf{0}, \sigma_{\text{coarse}}^2 \mathbf{I}) \quad (14)$$

where σ_{coarse} is chosen to be large enough to enable the sampler to explore broad coverage gaps.

In the **Fine Phase**, a second batch $\mathcal{I}_f = \{i_1^f, \dots, i_{B_f}^f\}$ is also drawn from the distribution P_{pick} . Similar to the coarse phase, we create a new Gaussian for every index in $i \in \mathcal{I}_f$:

$$g_i^f = (\mathbf{x}_i + \boldsymbol{\delta}_i^f, \Sigma_i, \mathbf{c}_i, \alpha_i), \quad \boldsymbol{\delta}_i^f \sim \mathcal{N}(\mathbf{0}, \sigma_{\text{fine}}^2 \mathbf{I}) \quad (15)$$

where $\sigma_{\text{fine}} < \sigma_{\text{coarse}}$. The smaller perturbation radius focuses on residual high-importance regions that still yield large weights $I(i)$ (and hence a large $P_{\text{pick}}(i)$) after the coarse phase fills the major coverage gaps. This allows our method to refine the geometry further using locally placed Gaussians. See Fig. 2 for a visualization of the coarse-fine proposal phase.

Finally, all the newly spawned Gaussians are combined to form the birth component Θ'_{birth} of the proposal. This is combined the current state Θ_t to yield a full proposal Θ' :

$$\Theta'_{\text{birth}} = \{g_i^c \mid i \in \mathcal{I}_c\} \cup \{g_i^f \mid i \in \mathcal{I}_f\}, \quad \Theta' = \Theta_t \cup \Theta'_{\text{birth}} \quad (16)$$

Formally, the probability of drawing the birth component is:

$$q(\Theta' \mid \Theta) = \prod_{i \in \mathcal{I}_c \cup \mathcal{I}_f} P_{\text{pick}}(i) \mathcal{N}(\mathbf{x}'_i; \mathbf{x}_i, \sigma_{\phi}^2 \mathbf{I}) \quad (17)$$

4.2.4 Relocation of Low Contributing Gaussians

In addition to densification, our framework incorporates a relocation mechanism inspired by 3DGS-MCMC [22] to reuse Gaussians with persistently low opacity. Let $\mathcal{D} = \{i \mid \alpha_i \leq \tau\}$ denote the set of Gaussians with low opacity, and $\mathcal{A} = \{j \mid \alpha_j > \tau\}$ the set of high opacity Gaussians. We consider $\tau = 0.005$. For each $g_i \in \mathcal{D}$, we sample a replacement index $j \in \mathcal{A}$ with probability

$$p(j) = \frac{\alpha_j}{\sum_{k \in \mathcal{A}} \alpha_k},$$

and update g_i 's parameters by transferring those from g_j , optionally modulated by a ratio reflecting the sampling frequency. The relocation step preserves the representational fidelity of our framework by continuously reinforcing effective Gaussians while reassigning under-contributing ones.

4.3 Metropolis-Hastings Proposal Acceptance

Now, the Metropolis-Hastings sampler must decide whether the proposal Θ' is kept or rejected. Recall the MH rule in Eq. 9. Considering the posterior density in Eq. 8, we can rewrite the rule as:

$$\rho_{\text{MH}} = \min \left\{ 1, e^{-\Delta \mathcal{E}} \frac{q(\Theta \mid \Theta')}{q(\Theta' \mid \Theta)} \right\} \quad (18)$$

Figure 1: PSNR over training iterations for our method and 3DGS-MCMC, averaged across all scenes. Dotted vertical lines indicate when each method reaches 98 % of its eventual PSNR, highlighting near-optimal convergence speed. While both methods ultimately attain comparable final PSNR, ours converges faster.

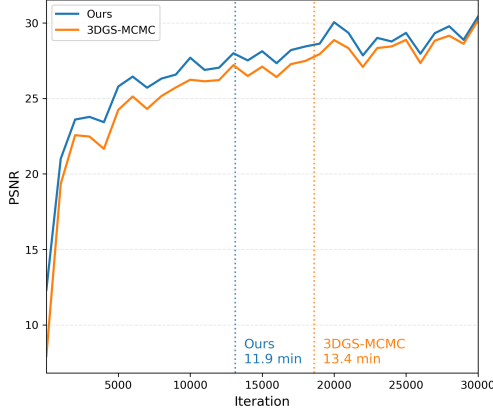
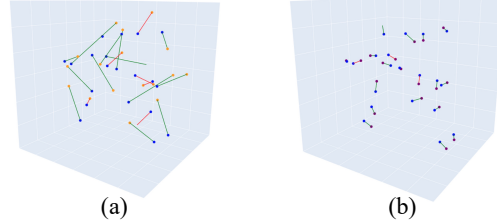


Table 1: Time to equal PSNR comparison between our method and 3DGS-MCMC, averaged across all benchmark scenes. Our method consistently reaches target PSNR thresholds faster.

Target PSNR (dB)	Time - Ours (s / mins)	Time - 3DGS-MCMC (s / mins)
21	16.30 / 0.27	17.08 / 0.28
24	61.34 / 1.02	98.38 / 1.64
27	287.01 / 4.78	341.64 / 5.69
30	851.52 / 14.19	983.05 / 16.38

Figure 2: 3D visualization of (a) coarse and (b) fine proposal stages (see Sec. 4.2.3 for details).



where $\Delta\mathcal{E} = \mathcal{E}(\Theta') - \mathcal{E}(\Theta)$. The proposal Θ' is a batch of new Gaussians, but since the perturbations are independent, we can evaluate the acceptance separately. Considering one candidate Gaussian whose center falls in voxel v' , the insertion of that particular Gaussian leads to $\Delta\mathcal{E}$ as:

$$\Delta\mathcal{E} = \Delta\mathcal{L} + \lambda_v \ln\left(\frac{1 + c_\Theta(v') + 1}{1 + c_\Theta(v')}\right) = \Delta\mathcal{L} - \ln D(v'), \quad D(v') = \frac{1}{1 + \lambda_v c_\Theta(v')} \quad (19)$$

where $\Delta\mathcal{L}$ is the photometric change and $c(v')$ is the voxel occupancy before the insertion. (See Appendix B for details.) However, evaluating $\mathcal{L}$ requires rendering every view in the set $\mathcal{C}_t$, which is computationally expensive. Thus, we use the tight empirical correlation between the importance map $I(i)$ and the loss reduction, $-\Delta\mathcal{L} \approx I(i)$, as a surrogate. (See Appendix C for details.)

Moreover, the calculation of $q(\Theta | \Theta')$ in Eq. 18 requires the enumeration of the reverse action, which is deleting the particular newborn Gaussian. As our framework does not generate such a move, we absorb the unavailable reverse-proposal density into the voxel factor $D(v') = (1 + \lambda_v c_\Theta(v'))^{-1}$, which already penalises insertions in crowded cells. Substituting both surrogates into Eq. 18 gives an upper bound $\min\{1, e^{I(i)} D(v')\}$, and mapping the exponential through the logistic $\sigma(z) = (1 + e^{-z})^{-1}$ yields a practical version of the MH rule:

$$\rho(i) = \sigma(I(i)) D(v') \quad (20)$$

Following the Metropolis-Hastings algorithm, we draw a uniform random number $u \sim \mathcal{U}(0, 1)$ for each candidate Gaussian. The proposal is accepted and added to the current set Θ_t when $u < \rho(i)$, otherwise it is rejected. Because $\rho(i) = \sigma(I(i)) D(v')$ is large only when the per-Gaussian importance $I(i)$ is high and the voxel factor $D(v')$ is close to one, Gaussians that target significant photometric errors in sparsely populated regions are accepted with high probability, whereas candidates in already crowded voxels or low-error areas are mostly discarded.

4.4 Comparison with 3DGS-MCMC [22]

Although both methods fall under the broad scope of Monte-Carlo sampling, they embody *fundamentally different* MCMC philosophies. 3DGS-MCMC treats every Gaussian parameter as a latent variable and applies SGLD: it perturbs all dimensions with small, isotropic noise and accepts every proposed move, so the chain drifts locally through the existing cloud.

In contrast, our method runs a Metropolis-Hastings birth sampler that targets only positions. We draw global proposals straight in image-space voids highlighted by a multi-view error map, then pass each

Table 2: Quantitative results of our method and state-of-the-art models evaluated on Mip-NeRF 360 [1], Tanks&Temples [23] and Deep Blending [17] datasets. We present the baseline results from 3DGS [20]. For a fair comparison, we also report the results for 3DGS obtained from our experiments (Denoted as 3DGS*), as well as those for 3DGS and 3DGS-MCMC trained with the *same number of Gaussians* as our method (denoted as 3DGS-S and 3DGS-MCMC, respectively). The unit of the number of Gaussians is million. The results are colored as **best**, **second-best**, and **third-best**. To avoid redundancy, only 3DGS* results are considered instead of 3DGS.

Dataset	Mip-NeRF 360				Tanks & Temples				Deep Blending			
Method	PSNR↑	SSIM↑	LPIPS↓	# GS↓	PSNR↑	SSIM↑	LPIPS↓	# GS↓	PSNR↑	SSIM↑	LPIPS↓	# GS↓
Plenoxels [10]	23.08	0.626	0.463	-	21.08	0.719	0.379	-	23.06	0.795	0.510	-
INGP-base [37]	25.30	0.671	0.371	-	21.72	0.723	0.330	-	23.62	0.797	0.423	-
INGP-big [37]	25.59	0.699	0.331	-	21.92	0.745	0.305	-	24.96	0.817	0.390	-
Mip-NeRF 360 [1]	27.69	0.792	0.237	-	22.22	0.759	0.257	-	29.40	0.901	0.245	-
3DGS [20]	27.21	0.815	0.214	-	23.14	0.841	0.183	-	29.41	0.903	0.243	-
3DGS* [20]	27.44	0.811	0.223	3.176	23.63	0.848	0.177	1.831	29.53	0.904	0.244	2.815
3DGS-S [20]	26.62	0.761	0.299	0.723	23.38	0.828	0.214	0.773	29.27	0.897	0.272	0.751
3DGS-MCMC [22]	27.69	0.816	0.234	0.723	24.30	0.862	0.170	0.773	29.84	0.906	0.254	0.751
Ours	27.34	0.798	0.241	0.723	23.99	0.852	0.166	0.773	30.12	0.909	0.245	0.751

candidate through a Metropolis-Hastings accept-reject test that weighs its predicted photometric gain against a sparsity prior. This decouples exploration (“where to propose”) from retention (“what to keep”), enabling long jumps to uncovered regions and the principled dismissal of redundant splats, which Langevin-noise walks with unconditional acceptance cannot achieve. Moreover, we analytically derive and rigorously prove that the 3DGS densification process can be recast as a Metropolis-Hastings sampler, establishing a principled bridge between 3DGS and our formulation.

In principle, 3DGS-MCMC is a local, always-accept SGLD chain, whereas our approach is a global, importance-driven MH chain focused on the scene’s unexplored areas. This leads to our method converging faster, as shown in Fig. 1. Measuring time to equal PSNR across all benchmark scenes, our method reaches 30 dB approximately 2.2 minutes faster than 3DGS-MCMC, with consistently faster convergence at intermediate quality thresholds, as shown in Tab. 1.

5 Experiments

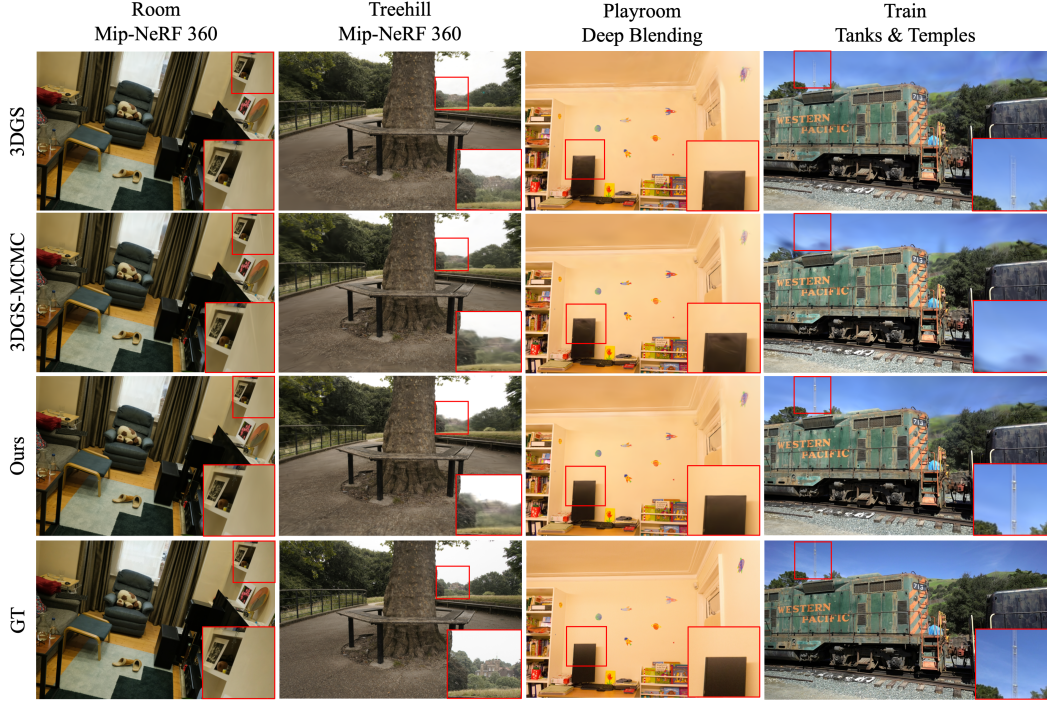
5.1 Experimental Settings

Dataset and Metrics. We perform comprehensive experiments using the same real-world datasets employed in 3DGS. Specifically, we utilize the scene-scale view synthesis dataset from Mip-NeRF360 [1], which includes nine large-scale real-world scenes comprising five outdoor and four indoor environments. Additionally, we selected two scenes each from the Tanks and Temples [23] and the Deep Blending [17] datasets, using the same scenes as in the original 3DGS. We evaluate each method using the standard evaluation metrics: peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and learned perceptual image patch similarity (LPIPS).

Implementation Details. Our framework is based on 3DGS [20] implementation. We also use the differentiable tile rasterizer implementation provided by 3DGS-MCMC [22]. We conducted experiments on an NVIDIA RTX 3090 GPU. Averaged across all datasets, our method runs at 0.0528 seconds per iteration and 26.4 minutes for 30K iterations. During densification, a normalized progress value ($\in [0, 1]$) linearly transitions parameters from coarse (larger offsets, voxel sizes, and batch sizes) to fine (smaller offsets and voxel sizes) to shift focus from broad to precise refinements. Coarse proposal scales range from 10.0 to 5.0, fine proposal scales from 2.0 to 1.0, and voxel sizes from 0.02 to 0.005. Batch sizes for coarse and fine proposals are set to 4,500 and 16,000, respectively.

Comparison with Baselines. We compare against methodologies that model large-scale scenes. We compare our approach against state-of-the-art baselines, such as Plenoxels [10], Instant-NGP [37], Mip-NeRF 360 [1], and 3DGS [20]. For the baselines aside from 3DGS, we present the original results as reported in 3DGS. Moreover, we include a comparison with 3DGS-MCMC [22], which has a similar motivation and shows state-of-the-art performance.

Figure 3: Qualitative results of our method compared to 3DGS and the corresponding ground truth image from held-out test views. As shown in the red bounding box, our method better captures the fine and coarse details that 3DGS [20] and 3DGS-MCMC [22] miss.



5.2 Results

Table 2 presents the quantitative evaluations, and Fig. 3 demonstrates the qualitative evaluations on real-world scenes. Our method consistently surpasses 3DGS performance with the same number of Gaussians and achieves or exceeds 3DGS performance with fewer points. Unlike 3DGS, which relies on fixed thresholds for densification, our framework leverages a streamlined point-based sampling strategy that drastically reduces storage requirements while preserving high-fidelity reconstructions. In particular, Tab. C in Appendix F demonstrates our approach’s consistent efficiency across varied scene complexities. We also provide the full per-scene evaluations in Appendix E.

Although 3DGS-MCMC can yield slightly higher metrics, it updates and optimizes every Gaussian parameter, which increases its inefficiency and necessitates knowing the scene’s complexity (the number of Gaussians) in advance. In contrast, our method focuses exclusively on sampling and optimizing point positions, providing a simpler yet equally accurate alternative. As evidenced by Fig. 1, our method reaches the 98 % PSNR threshold faster than 3DGS-MCMC, both in iteration and wall-clock time, demonstrating faster convergence. This shows that our framework can still achieve state-of-the-art reconstruction quality even without exhaustive parameter updates.

5.3 Ablation Study

Components for the Importance Scores. We conducted an ablation study on the key Gaussian components to identify the most influential factors in importance score calculation for accurate sampling. The results, shown in Tab. 3, support the effectiveness of our *Importance Score Computation* design. Specifically, combining SSIM, L1 loss, and opacity values yields higher PSNR scores and reduces the required number of Gaussians compared to both the baseline and variants that rely solely on photometric losses. This study underscores the need for careful factor selection.

Effectiveness of Relocation. We explore whether relocating Gaussians was effective in Tab. 5. Our experiments confirm that the relocation strategy is essential for maintaining high representational fidelity. It shows that dynamically reallocating underperforming Gaussians to areas where they contribute more effectively improves performance in both indoor and outdoor scenes.

Table 3: Ablations on the importance score components.

Method Dataset			Mip-NeRF360 [1]			Tanks&Temples [23]			Deep Blending [17]		
L1	SSIM	Opacity	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
✓	✓		27.22	0.796	0.243	23.89	0.851	0.168	29.87	0.907	0.247
✓		✓	27.30	0.796	0.242	23.92	0.851	0.169	29.85	0.907	0.246
	✓	✓	27.30	0.796	0.243	23.87	0.852	0.167	30.02	0.908	0.247
✓	✓	✓	27.34	0.798	0.241	23.99	0.852	0.166	30.12	0.909	0.245

Table 4: Ablations on the core sampling strategy: Metropolis–Hastings (MH) vs. Stochastic Gradient Langevin Dynamics (SGLD). We denote the number of Gaussians next to the method.

	Mip-NeRF360			Tanks&Temples			Deep Blending		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
3DGS + MH (0.85M)	26.87	0.782	0.260	23.66	0.843	0.173	29.46	0.898	0.258
3DGS + SGLD (0.85M)	26.80	0.789	0.250	23.47	0.841	0.178	29.45	0.898	0.259

Table 5: Ablations on relocation and the multi-view importance score. Results presented for 3DGS use the same number of Gaussians as ours.

	Mip-NeRF360			Tanks&Temples			Deep Blending		
	PSNR ↑	SSIM↑	LPIPS ↓	PSNR ↑	SSIM↑	LPIPS ↓	PSNR ↑	SSIM↑	LPIPS ↓
3DGS	26.62	0.761	0.299	23.38	0.828	0.214	29.27	0.897	0.272
Ours w/o Relocation	26.94	0.778	0.267	23.69	0.844	0.184	29.57	0.901	0.267
Ours w/o Importance Score	27.31	0.800	0.234	23.79	0.851	0.167	29.97	0.906	0.245
Ours	27.34	0.798	0.241	23.99	0.852	0.166	30.12	0.909	0.245

Contribution of Metropolis-Hastings Sampling. To measure the exact contribution of our core Metropolis-Hastings (MH) sampling strategy, we conducted an ablation study comparing our method directly against the Stochastic Gradient Langevin Dynamics (SGLD) approach from 3DGS-MCMC [22]. We implemented only the core sampling algorithms from each method while maintaining matched Gaussian counts to ensure a fair comparison. The results shown in Tab. 4 demonstrate that MH sampling consistently outperforms SGLD across all benchmark datasets, confirming the effectiveness of our sampling strategy.

Furthermore, to understand the source of our method’s effectiveness, we conducted an ablation study in Tab. 5 that isolates the contributions of the sampling process and the multi-view importance score. Our experiments demonstrate that the method retains strong performance even when the importance scores are removed. This indicates that the underlying sampling mechanism is the primary driver of our approach, while the multi-view importance scores act as an intelligent guidance mechanism.

6 Conclusion

We present an adaptive sampling framework that unifies densification and pruning for 3D Gaussian Splatting via a probabilistic Metropolis-Hastings scheme. The importance-weighted sampler automatically targets visually or geometrically critical regions while avoiding redundancy. We derive a new, concise probabilistic formulation that links proposals to scene importance, enabling faster convergence without sacrificing accuracy. Our experiments show that our method matches or surpasses state-of-the-art quality with fewer Gaussians, and our method’s simplicity invites further advances in 3D scene representation.

Limitations and Future Work. Our Metropolis-Hastings sampler, though aware of low-density regions, still prioritizes high-density areas, leading to outdoor scenes with sparser data performing worse than indoor scenes. Moreover, using multi-view photometric errors when proposing points adds roughly 5 minutes of extra training time per scene. Crafting multi-scale representations or dual-stream foreground/background samplers, and introducing faster optimization to reduce training time, would be promising directions for future research.

Acknowledgments

This work was supported by IITP grant (RS-2021-II211343: AI Graduate School Program at Seoul National Univ. (5%), RS-2023-00227993: Detailed 3D reconstruction for urban areas from unstructured images (35%), and RS-2025-02303703: Realworld multi-space fusion and 6DoF free-viewpoint immersive visualization for extended reality (60%)) funded by the Korea government (MSIT).

References

- [1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022.
- [2] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023.
- [3] Matteo Bortolon, Theodore Tsesmelis, Stuart James, Fabio Poiesi, and Alessio Del Bue. Iffnerf: Initialisation free and fast 6dof pose estimation from a single image and a nerf model. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1985–1991. IEEE, 2024.
- [4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022.
- [5] Zilong Chen, Feng Wang, Yikai Wang, and Huaping Liu. Text-to-3D using Gaussian Splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21401–21412, June 2024.
- [6] Siddhartha Chib and Edward Greenberg. Understanding the Metropolis-Hastings Algorithm. *The American Statistician*, 49(4):327–335, 1995. doi: 10.1080/00031305.1995.10476177.
- [7] Xiaobin Deng, Changyu Diao, Min Li, Ruohan Yu, and Duanqing Xu. Efficient density control for 3d gaussian splatting. *arXiv preprint arXiv:2411.10133*, 2024.
- [8] Yilun Du, Yanan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *IEEE/CVF International Conference on Computer Vision*, pages 14304–14314. IEEE Computer Society, 2021.
- [9] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, Zhangyang Wang, et al. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *Advances in Neural Information Processing Systems*, 37:140138–140158, 2024.
- [10] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance Fields Without Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5501–5510, June 2022.
- [11] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *IEEE/CVF International Conference on Computer Vision*, pages 14346–14355, 2021.
- [12] Lily Goli, Daniel Rebain, Sara Sabour, Animesh Garg, and Andrea Tagliasacchi. nerf2nerf: Pairwise registration of neural radiance fields. In *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023.
- [13] Jonathan Granskog, Till N. Schnabel, Fabrice Rousselle, and Jan Novák. Neural Scene Graph Rendering. *ACM Transactions on Graphics*, 40(4), August 2021.
- [14] Yudong Guo, Keyu Chen, Sen Liang, Yong-Jin Liu, Hujun Bao, and Juyong Zhang. Ad-nerf: Audio driven neural radiance fields for talking head synthesis. In *IEEE/CVF International Conference on Computer Vision*, pages 5784–5794, 2021.
- [15] Alex Hanson, Allen Tu, Geng Lin, Vasu Singla, Matthias Zwicker, and Tom Goldstein. Speedy-splat: Fast 3d gaussian splatting with sparse pixels and sparse primitives. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21537–21546, 2025.
- [16] Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. Pup 3d-gs: Principled uncertainty pruning for 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5949–5958, 2025.

- [17] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep Blending for Free-viewpoint Image-based Rendering. *37(6):257:1–257:15*, 2018.
- [18] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. In *SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery, 2024. doi: 10.1145/3641519.3657428.
- [19] Yoonwoo Jeong, Seokjun Ahn, Christopher Choy, Anima Anandkumar, Minsu Cho, and Jaesik Park. Self-calibrating neural radiance fields. In *IEEE/CVF International Conference on Computer Vision*, pages 5846–5854, 2021.
- [20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4), July 2023.
- [21] Bernhard Kerbl, Andreas Meuleman, Georgios Kopanas, Michael Wimmer, Alexandre Lanvin, and George Drettakis. A Hierarchical 3D Gaussian Representation for Real-Time Rendering of Very Large Datasets. *ACM Transactions on Graphics*, 43(4), July 2024.
- [22] Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Yang-Che Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3d gaussian splatting as markov chain monte carlo. *Advances in Neural Information Processing Systems*, 37:80965–80986, 2024.
- [23] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction. *ACM Transactions on Graphics*, 36(4), 2017.
- [24] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3D Gaussian Representation for Radiance Field. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21719–21728, 2024.
- [25] Junoh Lee, ChangYeon Won, Hyunjun Jung, Inhwan Bae, and Hae-Gon Jeon. Fully explicit dynamic gaussian splatting. *Advances in Neural Information Processing Systems*, 37:5384–5409, 2024.
- [26] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime Gaussian Feature Splatting for Real-Time Dynamic View Synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8508–8520, June 2024.
- [27] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021.
- [28] Jiaqi Lin, Zhihao Li, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Jiayue Liu, Yangdi Lu, Xiaofei Wu, Songcen Xu, Youliang Yan, and Wenming Yang. VastGaussian: Vast 3D Gaussians for Large Scene Reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5166–5175, June 2024.
- [29] Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align Your Gaussians: Text-to-4D with Dynamic 3D Gaussians and Composed Diffusion Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8576–8588, June 2024.
- [30] Yang Liu, Chuanchen Luo, Lue Fan, Naiyan Wang, Junran Peng, and Zhaoxiang Zhang. Citygaussian: Real-time high-quality large-scale scene rendering with gaussians. In *European Conference on Computer Vision*, pages 265–282. Springer, 2025.
- [31] Zhicheng Lu, Xiang Guo, Le Hui, Tianrui Chen, Min Yang, Xiao Tang, Feng Zhu, and Yuchao Dai. 3d geometry-aware deformable gaussian splatting for dynamic view synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8900–8910, 2024.
- [32] Saswat Subhajiya Mallick, Rahul Goel, Bernhard Kerbl, Markus Steinberger, Francisco Vicente Carrasco, and Fernando De La Torre. Taming 3dgs: High-quality radiance fields with limited resources. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024.
- [33] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021.
- [34] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. In *IEEE/CVF International Conference on Computer Vision*, pages 6351–6361, 2021.
- [35] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6): 1087–1092, 1953.

- [36] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *European Conference on Computer Vision*, 2020.
- [37] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4):1–15, 2022.
- [38] Simon Niedermayr, Josef Stumpffegger, and Rüdiger Westermann. Compressed 3D Gaussian Splatting for Accelerated Novel View Synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10349–10358, June 2024.
- [39] Panagiotis Papantonakis, Georgios Kopanas, Bernhard Kerbl, Alexandre Lanvin, and George Drettakis. Reducing the Memory Footprint of 3D Gaussian Splatting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 7(1), May 2024.
- [40] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021.
- [41] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Transactions on Graphics*, 40(6), dec 2021.
- [42] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021.
- [43] Samuel Rota Bulò, Lorenzo Porzi, and Peter Kotschieder. Revising densification in gaussian splatting. In *European Conference on Computer Vision*, pages 347–362. Springer, 2024.
- [44] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3D. In *ACM Transactions on Graphics*, volume 25, pages 835–846. 2006.
- [45] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5459–5469, 2022.
- [46] Matthew Tancik, Vincent Casser, Xinchun Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable Large Scene Neural View Synthesis. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, June 2022.
- [47] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2024.
- [48] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation. In *International Conference on Learning Representations*, 2024.
- [49] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021.
- [50] Henan Wang, Hanxin Zhu, Tianyu He, Runsen Feng, Jiajun Deng, Jiang Bian, and Zhibo Chen. End-to-end rate-distortion optimized 3d gaussian representation. In *European Conference on Computer Vision*, pages 76–92. Springer, 2024.
- [51] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20310–20320, June 2024.
- [52] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. Bungeenerf: Progressive neural radiance field for extreme multi-scale scene rendering. In *European Conference on Computer Vision*, pages 106–122. Springer, 2022.
- [53] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20331–20341, 2024.

- [54] Zongxin Ye, Wenyu Li, Sidun Liu, Peng Qiao, and Yong Dou. Absgs: Recovering fine details in 3d gaussian splatting. In *ACM International Conference on Multimedia*, pages 1053–1061, 2024.
- [55] Taoran Yi, Jiemin Fang, Junjie Wang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. GaussianDreamer: Fast Generation from Text to 3D Gaussians by Bridging 2D and 3D Diffusion Models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6796–6807, June 2024.
- [56] Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19447–19456, 2024.
- [57] Jiahui Zhang, Fangneng Zhan, Yingchen Yu, Kunhao Liu, Rongliang Wu, Xiaoqin Zhang, Ling Shao, and Shijian Lu. Pose-free neural radiance fields via implicit pose regularization. In *IEEE/CVF International Conference on Computer Vision*, pages 3534–3543, 2023.

A Extended Derivation for Eq. 8

In Bayesian statistics, the posterior distribution of the parameters Θ given data $\mathcal{D}$ is defined as:

$$p(\Theta | \mathcal{D}) = \frac{p(\mathcal{D} | \Theta) p(\Theta)}{\underbrace{\int p(\mathcal{D} | \Theta) p(\Theta) d\Theta}_Z}, \quad (21)$$

where $p(\mathcal{D} | \Theta)$ is the likelihood, $p(\Theta)$ is the prior, and Z is the marginal likelihood.

In Eq. 7, we introduced a negative log-posterior

$$\mathcal{E}(\Theta) = \mathcal{L}(\Theta) + \lambda_v \sum_{v \in \mathcal{V}} \ln(1 + c_\Theta(v)) \quad (22)$$

In Eq. 5, we found out that $\ln p(\mathcal{D} | \Theta) \approx -\mathcal{L}(\Theta)$. Combining these two, we can derive:

$$\mathcal{E}(\Theta) = -\ln p(\mathcal{D} | \Theta) - \ln p(\Theta) \quad (23)$$

If we exponentiate $-\mathcal{E}(\Theta)$:

$$e^{-\mathcal{E}(\Theta)} = p(\mathcal{D} | \Theta) p(\Theta), \quad (24)$$

we can derive the unnormalized posterior density. To turn this into a proper probability distribution, we normalize by $Z = \int e^{-\mathcal{E}(\Theta)} d\Theta$, yielding:

$$\pi(\Theta) = \frac{1}{Z} e^{-\mathcal{E}(\Theta)} = \frac{p(\mathcal{D} | \Theta) p(\Theta)}{Z} = p(\Theta | \mathcal{D}) \quad (25)$$

As $\pi(\Theta)$ matches the posterior distribution form in Eq. 21, we can define it as the posterior density.

B Extended Derivation for Eq. 19

Below is the derivation for Eq. 19. Recall Eq. 7, which defines the negative log-posterior, also known as the energy function:

$$\mathcal{E}(\Theta) = \mathcal{L}(\Theta) + \lambda_v \sum_{v \in \mathcal{V}} \ln(1 + c_\Theta(v)) \quad (26)$$

Thus $\Delta\mathcal{E} = \mathcal{E}(\Theta') - \mathcal{E}(\Theta)$ can be calculated as:

$$\begin{aligned} \Delta\mathcal{E} &= \mathcal{E}(\Theta') - \mathcal{E}(\Theta) \\ &= \mathcal{L}(\Theta') + \lambda_v \sum_{v \in \mathcal{V}} \ln(1 + c_{\Theta'}(v)) - (\mathcal{L}(\Theta) + \lambda_v \sum_{v \in \mathcal{V}} \ln(1 + c_\Theta(v))) \\ &= (\mathcal{L}(\Theta') - \mathcal{L}(\Theta)) + \lambda_v \sum_{v \in \mathcal{V}} [\ln(1 + c_{\Theta'}(v)) - \ln(1 + c_\Theta(v))] \end{aligned}$$

As we considered the case of having added exactly one Gaussian into voxel v' , we can define $c_{\Theta'}(v)$, the number of Gaussians in v' after the move, as:

$$c_{\Theta'}(v') = c_\Theta(v') + 1 \quad (27)$$

Moreover, the sum collapses to a single term for voxel v' . Thus, continuing our derivation:

$$\begin{aligned} \Delta\mathcal{E} &= (\mathcal{L}(\Theta') - \mathcal{L}(\Theta)) + \lambda_v \sum_{v \in \mathcal{V}} [\ln(1 + c_\Theta(v') + 1) - \ln(1 + c_\Theta(v'))] \\ &= \Delta\mathcal{L} + \lambda_v \ln \left(\frac{1 + c_\Theta(v') + 1}{1 + c_\Theta(v')} \right) \\ &= \Delta\mathcal{L} - \ln \left(\frac{1 + c_\Theta(v')}{1 + c_\Theta(v') + 1} \right)^{\lambda_v} \end{aligned}$$

Setting $\left(\frac{1 + c_\Theta(v')}{1 + c_\Theta(v') + 1} \right)^{\lambda_v}$ as $D(v')$, we can derive:

$$\Delta\mathcal{E} = \Delta\mathcal{L} - \ln D(v') \quad (28)$$

Now set $x = \frac{1}{1+c_{\Theta}(v')}$. We can use the first-order Taylor series expansion (also known as the first-order Maclaurin expansion) to approximate

$$\lambda_v \ln(1+x) \approx \lambda_v x = \frac{\lambda_v}{1+c_{\Theta}(v')} \quad (29)$$

We can further approximate as follows:

$$\begin{aligned} \ln D(v') &= -\lambda_v \ln \left(1 + \frac{1}{1+c_{\Theta}(v')} \right) \\ &= -\lambda_v \ln(1+x) \approx -\lambda_v x \\ D(v') &\approx e^{-\lambda_v x} \end{aligned} \quad (30)$$

Using the Maclaurin expansion, we can approximate:

$$e^{-a} = \frac{1}{e^a} = \frac{1}{1+a+\frac{a^2}{2}+O(a^3)} \approx \frac{1}{1+a} \quad (31)$$

Thus we can approximate $e^{-\lambda_v x}$ as $\frac{1}{1+\lambda_v x}$

$$D(v') \approx e^{-\lambda_v x} \approx \frac{1}{1+\frac{\lambda_v}{1+c_{\Theta}(v')}} \approx \frac{1}{1+\lambda_v c_{\Theta}(v')} \quad (32)$$

C Extended Derivation for Surrogate

This section explains why we approximate $-\Delta\mathcal{L}$ to $I(i)$.

Let's say we insert a tiny splat $\{\varepsilon g_i\}$ to the current state Θ . We'll call this state Θ' . By the first-order term in the multivariate Taylor expansion in the direction of g_i :

$$\Delta\mathcal{L} = \mathcal{L}(\Theta') - \mathcal{L}(\Theta) \approx \left\langle \nabla_{g_i} \mathcal{L}(\Theta), \varepsilon g_i \right\rangle \quad (33)$$

where $\nabla_{g_i} \mathcal{L}(\Theta)$ is the gradient of the loss w.r.t. the Gaussian's parameters and the inner product measures the first-order change when adding εg_i .

Since g_i only affects $p_i = \lfloor \Pi(\mathbf{x}_i) \rfloor$, we can show:

$$-\Delta\mathcal{L} \propto \alpha O(p_i) + \beta \text{SSIM}_{\text{agg}}(p_i) + \gamma \mathcal{L}_{\text{agg}}(p_i) = z_i \quad (34)$$

As stated in Eq. 12 and 13, we set:

$$I(i) = \sigma(z_i) = \frac{1}{1+e^{-z_i}} \quad (35)$$

The Maclaurin series for $\sigma(z) = \frac{1}{1+e^{-z}}$ around 0 is:

$$\begin{aligned} \sigma(z) &= \sigma(0) + \sigma'(0)z + \frac{\sigma''(0)}{2!}z^2 + \frac{\sigma^{(3)}(0)}{3!}z^3 + \dots \\ &= \frac{1}{2} + \frac{1}{4}z - \frac{1}{48}z^3 + \dots \approx \frac{1}{2} + \frac{1}{4}z \end{aligned}$$

Thus, $\sigma(z_i) \approx \frac{1}{2} + \frac{1}{4}z_i$. Since our Metropolis-Hastings rule uses only relative magnitudes of $\sigma(z_i)$, the constant and the scale are absorbed, yielding $\sigma(z_i) \propto z_i$. Finally, if we put everything together:

$$-\Delta\mathcal{L} \approx z_i \approx \sigma(z_i) = I(i) \quad (36)$$

D Algorithms

Our algorithms for our coarse-fine Metropolis-Hastings sampling framework and overall pipeline are summarized in Alg. 1 and 2.

Algorithm 1 Coarse–Fine Metropolis–Hastings Sampling for 3DGS

Require: Scene $\Theta = \{g_i\}$ where $g_i = (\mathbf{x}_i, \Sigma_i, c_i, \alpha_i)$
View set $\mathcal{C}$; iteration t , total steps $t_{\max}$
Voxel size v ; density penalty λ_v
Proposal stds (σ_c, σ_f) ; batch sizes (B_c, B_f)
Importance weights (α, β, γ) ; opacity threshold τ

// Select diverse view subset for error aggregation
1: $\mathcal{C}_t \leftarrow \text{VIEWSUBSET}(\mathcal{C}, t, t_{\max})$

// Compute multi-view error maps (SSIM, L1)
2: $\text{SSIM}_{\text{agg}}, \mathcal{L}1_{\text{agg}} \leftarrow \frac{1}{|\mathcal{C}_t|} \sum_{c \in \mathcal{C}_t} [1 - \text{SSIM}(I^{(c)}, \hat{I}^{(c)}), |I^{(c)} - \hat{I}^{(c)}|]$

// Relocate Gaussians with low opacity
3: **for** $g_i \in \Theta$ **where** $\alpha_i \leq \tau$ **do**
4: $\text{RELOCATE}(g_i)$
5: **end for**

// Compute importance scores per Gaussian
6: **for** $g_i \in \Theta$ **do**
7: $p_i \leftarrow \Pi(\mathbf{x}_i)$ ▷ Project center to image plane
8: $O(p_i) \leftarrow \alpha_i$ ▷ Use projected Gaussian opacity
9: $S(i) \leftarrow \alpha O(p_i) + \beta \text{SSIM}_{\text{agg}}(p_i) + \gamma \mathcal{L}1_{\text{agg}}(p_i)$
10: $I(i) \leftarrow \sigma(S(i))$
11: **end for**
12: $c(\cdot) \leftarrow \text{VOXELCOUNTS}(\Theta, v)$

// Metropolis–Hastings insertion (Coarse $\rightarrow$ Fine)
13: **for** $(\sigma, B) \in \{(\sigma_c, B_c), (\sigma_f, B_f)\}$ **do**
14: Sample $\mathcal{I}$ of size B from Θ with $\Pr(i) \propto I(i)$
15: **for** $i \in \mathcal{I}$ **do**
16: $\mathbf{x}' \leftarrow \mathbf{x}_i + \mathcal{N}(0, \sigma^2 \mathbf{I})$
17: $v' \leftarrow \text{VOXELINDEX}(\mathbf{x}', v)$
18: $D(v') \leftarrow \frac{1}{1 + \lambda_v c(v')}$
19: $\rho \leftarrow I(i) \cdot D(v')$ ▷ Acceptance probability
20: **if** $\text{UNIFORM}(0, 1) < \rho$ **then**
21: $\Theta \leftarrow \Theta \cup \{(\mathbf{x}', \Sigma_i, c_i, \alpha_i)\}$
22: **end if**
23: **end for**
24: **end for**

25: **return** Θ

Algorithm 2 Overall pipeline of Metropolis–Hastings 3DGS

1: **Input:** M, S, C, A (SfM Points, Covariances, Colors, Opacities); image dimensions w, h .
2: **Output:** Optimized and densified attributes M, S, C, A .
3: $i \leftarrow 0$ ▷ Iteration count
4: **while** not converged **do**
5: $\{V, \hat{I}\} \leftarrow \text{SAMPLETRAININGVIEW}()$ ▷ Camera V and image
6: $I \leftarrow \text{RASTERIZE}(M, S, C, A, V)$
7: $L \leftarrow \text{LOSS}(I, \hat{I})$
8: $S_{\text{SSIM}, i} \leftarrow \text{SSIM}(I, \hat{I})$
9: $S_{\text{L1}, i} \leftarrow \text{L1}(I, \hat{I})$
10: $[M, S, C, A] \leftarrow \text{ADAM}(\nabla L)$ ▷ Update attributes
11: **if** $\text{ISREFINEMENTITERATION}(i)$ **then**
12: **for** Current Scene Θ **do**
13: $\text{METROPOLIS_HASTINGS_SAMPLING}(M, S, C, A)$
14: **end for**
15: **end if**
16: $i \leftarrow i + 1$
17: **end while**
18: **return** M, S, C, A

E Per-Scene Results

In this section, we provide the full per-scene evaluations. We evaluated our method, 3DGS [20] and 3DGS-MCMC [22] on various datasets and scenes. We provide the per-scene results for Mip-NeRF 360 [1], Tanks & Temples [23] and Deep Blending [17] datasets in Tab. A and B. All evaluations have been conducted on a NVIDIA GeForce RTX 3090.

Table A: Full per-scene results evaluated on Mip-NeRF 360 dataset.

Type of Dataset		Indoor				Outdoor					
Scene		Bonsai	Counter	Kitchen	Room	Bicycle	Flowers	Garden	Stump	Treehill	Average
3DGS [20]	PSNR	32.14	28.98	31.25	31.65	25.10	21.33	27.31	26.64	22.54	27.44
	SSIM	0.940	0.906	0.931	0.927	0.747	0.589	0.857	0.770	0.635	0.811
	LPIPS	0.206	0.202	0.117	0.197	0.244	0.359	0.122	0.216	0.347	0.223
	Train (mm:ss)	28:04	43:25	31:05	26:29	42:41	31:21	43:16	34:00	30:53	34:34
	# of Gaussians	1249672	1174274	1763905	1478813	5728256	3480107	5667876	4494105	3522801	3175479
	Storage (MB)	296.56	277.73	419.18	349.76	1354.80	823.09	1340.52	1062.91	833.18	750.86
3DGS-S [20]	PSNR	31.37	28.84	30.91	31.04	23.55	20.10	26.27	24.84	22.64	26.62
	SSIM	0.935	0.909	0.924	0.918	0.616	0.488	0.794	0.677	0.590	0.761
	LPIPS	0.205	0.196	0.134	0.214	0.410	0.478	0.230	0.381	0.444	0.299
	Train (mm:ss)	26:42	28:04	33:59	25:22	25:54	23:58	28:17	23:02	28:00	27:02
	# of Gaussians	777782	698871	623992	764539	741748	695495	811695	697792	690517	722492
	Storage (MB)	183.96	165.29	147.58	181.02	175.43	164.49	191.98	165.04	163.32	170.90
3DGS-MCMC [22]	PSNR	32.48	29.27	31.41	32.15	25.13	21.64	26.71	27.21	23.09	27.69
	SSIM	0.949	0.920	0.930	0.933	0.745	0.595	0.832	0.791	0.648	0.816
	LPIPS	0.176	0.177	0.126	0.186	0.275	0.377	0.170	0.247	0.370	0.234
	Train (mm:ss)	32:08	37:27	30:25	35:37	31:56	25:39	24:16	23:50	26:01	29:42
	# of Gaussians	777782	698871	623992	764539	741748	695495	811695	697792	690517	722492
	Storage (MB)	183.96	165.29	147.58	181.02	175.43	164.49	191.98	165.04	163.32	170.90
Ours	PSNR	32.52	29.30	31.35	32.15	24.53	20.99	26.49	25.95	22.80	27.34
	SSIM	0.950	0.918	0.930	0.934	0.711	0.556	0.819	0.743	0.621	0.798
	LPIPS	0.171	0.174	0.123	0.179	0.289	0.384	0.185	0.279	0.384	0.241
	Train (mm:ss)	41:05	46:59	41:20	46:31	34:58	35:13	31:36	32:46	37:05	38:37
	# of Gaussians	777782	698871	623992	764539	741748	695495	811695	697792	690517	722492
	Storage (MB)	183.96	165.29	147.58	181.02	175.43	164.49	191.98	165.04	163.32	170.90

Table B: Full per-scene results evaluated on Tanks & Temples and Deep Blending datasets.

Dataset		Tanks&Temples			Deep Blending		
Scene		Train	Truck	Average	Dr Johnson	Playroom	Average
3DGS [20]	PSNR ($\uparrow$)	21.89	25.37	23.63	29.08	29.97	29.53
	SSIM ($\uparrow$)	0.814	0.882	0.848	0.901	0.907	0.904
	LPIPS ($\downarrow$)	0.207	0.147	0.177	0.244	0.244	0.244
	Train (mm:ss)	13:22	19:09	16:15	29:26	26:20	27:53
	# of Gaussians	1085904	2575516	1830710	3308886	2320688	2814787
	Storage (MB)	256.83	609.14	432.99	782.59	548.87	665.73
3DGS-S [20]	PSNR ($\uparrow$)	21.80	24.95	23.38	28.99	29.54	29.27
	SSIM ($\uparrow$)	0.791	0.865	0.828	0.892	0.901	0.897
	LPIPS ($\downarrow$)	0.245	0.183	0.214	0.280	0.263	0.272
	Train (mm:ss)	19:16	18:05	18:41	20:18	20:23	20:21
	# of Gaussians	788942	756104	772523	766469	735409	750939
	Storage (MB)	186.60	178.83	182.72	181.28	173.93	177.61
3DGS-MCMC [22]	PSNR ($\uparrow$)	22.58	26.02	24.30	29.37	30.30	29.84
	SSIM ($\uparrow$)	0.834	0.889	0.862	0.902	0.910	0.906
	LPIPS ($\downarrow$)	0.196	0.143	0.170	0.260	0.248	0.254
	Train (mm:ss)	15:50	15:45	15:48	26:28	23:39	25:03
	# of Gaussians	788942	756104	772523	766469	735409	750939
	Storage (MB)	186.60	178.83	182.72	181.28	173.93	177.61
Ours	PSNR ($\uparrow$)	22.44	25.54	23.99	29.77	30.47	30.12
	SSIM ($\uparrow$)	0.821	0.883	0.852	0.904	0.913	0.909
	LPIPS ($\downarrow$)	0.198	0.134	0.166	0.249	0.240	0.245
	Train (mm:ss)	20:38	19:40	20:09	32:27	29:54	31:10
	# of Gaussians	788942	756104	772523	766469	735409	750939
	Storage (MB)	186.60	178.83	182.72	181.28	173.93	177.61

F Comparison of the Number of Gaussians

Table C contrasts the number of Gaussians needed by our approach and by 3DGS for each scene. In every case, our method attains equal or superior rendering quality while requiring substantially fewer Gaussians.

Table C: Comparison of our method and 3DGS on the number of Gaussian center points used to represent scenes in the Mip-NeRF 360 [1], Tanks & Temples [23], and Deep Blending [17] datasets.

Dataset	Mip-NeRF 360									Tanks&Temples		Deep Blending	
Method	Bonsai	Counter	Kitchen	Room	Bicycle	Flowers	Garden	Stump	Treehill	Train	Truck	Dr Johnson	Playroom
3DGS [20]	1,249,672	1,174,274	1,763,905	1,297,736	4,888,552	2,887,138	5,667,876	4,324,049	3,224,118	1,085,904	2,580,408	3,308,886	2,320,688
Ours	777,782	698,871	623,992	765,370	741,748	695,495	811,695	697,792	690,517	788,942	756,104	766,469	735,409

G Broader Impacts

As our method focuses on the core and theoretical problem of 3D Scene Reconstruction, our work has the potential for positive societal impact on various downstream applications using 3D Gaussian Splatting [20]. As we reduce the reliance on heuristics and the number of Gaussians used in a scene, we expect our work to enhance many downstream applications that require less memory usage. However, as we did not introduce any fundamental new content generation capabilities through our work, there is little potential for our work to have a negative societal impact beyond the ethical considerations already present in the field of 3D reconstruction.

H Dataset Licenses

We used the following datasets:

- Mip-NeRF360 [1]: The license term is unavailable. Available at <https://jonbarron.info/mipnerf360/>
- Tanks and Temples [23]: Published under the Creative Commons Attribution 4.0 International (CC BY 4.0) License. Available at <https://www.tanksandtemples.org/license/>
- Deep Blending [17]: The license term is unavailable. Available at <http://visual.cs.ucl.ac.uk/pubs/deepblending/>

NeurIPS Paper Checklist

1. Claims

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

Answer: [\[Yes\]](#)

Justification: We write our contributions and scope in the Abstract and Introduction in Sec. 1.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

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

Answer: [\[Yes\]](#)

Justification: We discuss our limitations in the Conclusion.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

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

Answer: [\[Yes\]](#)

Justification: We provide it in Sec. 4 and Appendix A, B and C

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

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

Answer: [Yes]

Justification: We provide the details of our experiments in Sec. 5

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

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

Answer: [Yes]

Justification: We will release the code at <https://github.com/hjhyunjinkim/MH-3DGS>. The datasets used in this paper are open sourced, details are provided at Appendix H.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

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

Answer: [Yes]

Justification: We provide detailed experiment setups in Sec. 5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

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

Answer: [No]

Justification: Due to our limited computational resources, we were unable to report error bars.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

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

Answer: [Yes]

Justification: This information is provided in Sec. 5.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

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

Answer: [Yes]

Justification: : This work conforms with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

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

Answer: [Yes]

Justification: Discussed in Appendix G.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

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

Answer: [NA]

Justification: NA

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

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

Answer: [Yes]

Justification: We have properly cited all sources. We also cited the dataset licenses in Appendix H.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset’s creators.

13. **New assets**

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

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

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

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

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [NA]

Justification: LLM was used for writing and editing purposes.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.