LATENTGS: PROBABILISTIC DENSIFICATION FOR EFFICIENT, COMPACT, AND FASTER 3D GAUSSIAN SPLATTING

Shuja Khalid, Mohamed Ibrahim, & Yang Liu
Huawei Canada
Markham, ON, Canada
shuja.khalid@huawei.com

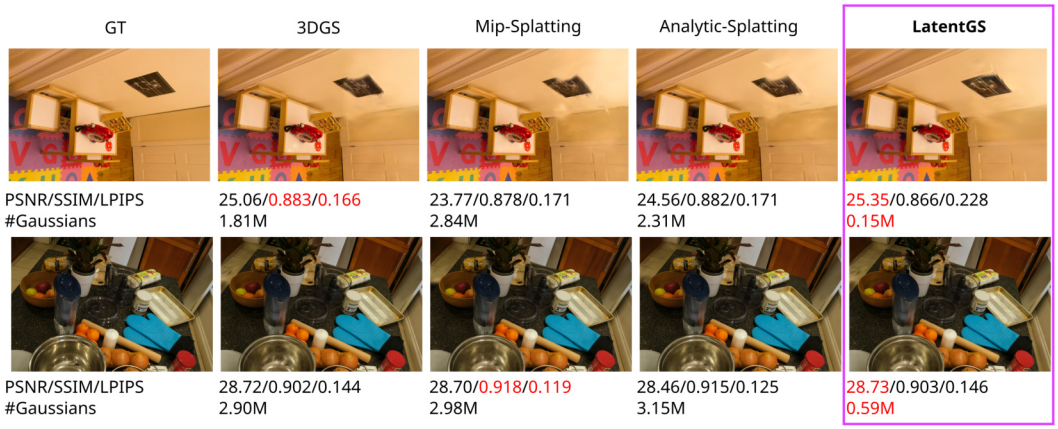

Figure 1: **Overview of our approach.** Our adaptive 3D Gaussian representation enables high-fidelity rendering and compact scene encoding. Not only are we able to generate comparable results to state of the art approaches such as analytic splatting and mip-splatting, our learned generative model generates gaussians beyond the training resolution. We discuss both contributions in detail in sections 3 and 4.7.

ABSTRACT

We present LatentGS, a variational reformulation of 3D Gaussian Splatting that replaces heuristic densification with a learned, probabilistic model. A Variational Autoencoder (VAE) learns the joint distribution of scene geometry, appearance, and uncertainty, enabling adaptive sampling of new Gaussians directly from the latent space. Placement is guided by a three-dimensional variant of a Laplacian-kernel penalty map, which targets regions of high spatial variation. The resulting scene is 30%-90% more compact than vanilla 3DGS with superior quality, leading to faster training and rendering without loss of fidelity. Moreover, post-training, the LatentGS can be used as a standalone densifier/refiner of the reconstructed 3D scene. Allowing users to further refine/add detail to selected regions of interest (ROI). Together, these contributions produce a compact, perceptually stable, and efficient 3D representation that advances the quality and scalability of Gaussian Splatting.

1 INTRODUCTION

Novel view synthesis has long been a core challenge in computer vision. Recent work has focused on reconstructing 3D scenes from collections of uncalibrated images, producing a range of optimization strategies and output representations. State-of-the-art methods for 3D scene reconstruction typically rely on Neural Radiance Fields (NeRF)Mildenhall et al. (2021) or 3D Gaussian Splatting (3DGS)Kerbl et al. (2023). NeRF-based methods model the scene as a continuous 5D radiance field using a

multilayer perceptron (MLP) that is queried during rendering along a view direction, capturing view-dependent, high fidelity renderings of the underlying scene. On the other hand, 3DGS-based methods represent the scene as an optimized set of 3D Gaussians whose parameters are blended in a view-dependent manner to produce rendering of the reconstructed scene, offering a more explicit geometric primitive-based representation and often, a faster rendering.

3D Gaussian Splatting (3DGS)Kerbl et al. (2023) variants have gained broader adoption since the 3DGS pipeline is substantially less compute intensive and more engineering-friendly than many NeRF variants. The resulting representation from 3DGS pipelines allows efficient GPU acceleration and real-time or near-real-time renderingFei et al. (2024). In addition, the 3DGS scene representation is easy to stream, partition, and distribute across machines for large-scale scenesSun et al. (2025). Moreover, primitives are explicit and locally meaningful, which makes editing, level-of-detail management, and progressive refinement straightforward. Furthermore, splat-based rendering avoids the heavy ray-sampling loops and large MLP memory/compute costs typical of NeRFs, yielding faster turnaround for the capture-to-viewing process. NeRF-based methods can still produce superior per-pixel fidelity and smoother view-dependent effects in some casesBarron et al. (2023), but the practical advantages of 3DGS-based methods (speed, scalability, lower resource demands, and ease of integration into existing graphics pipelines) explain why it has become the mainstream choice for many novel-view synthesis applications.

3DGS-based methods face several practical limitations: each Gaussian encodes many parameters (position, covariance, color, opacity, and spherical harmonics coefficients), so large scenes with millions of primitives quickly demand substantial GPU memory and storage. In addition, during training, the 3DGS pipeline typically relies on heuristic densification, cloning and pruning rules that are based on hand-tuned thresholds and can introduce instability or uneven quality such as popping artifacts. Finally, similar to NeRF-based methods, the final fidelity of the reconstructed scene is constrained by the resolution of the input training images, so low-resolution or sparsely sampled captures produce blur, missing detail or hallucinated geometry.

We introduce LatentGS, a novel 3D scene reconstruction framework built on top of 3D Gaussian Splatting (3DGS) with an anti-aliasing rasterizer. The proposed pipeline achieves 3D scene reconstructions with minimal artifacts, delivering image quality on par with state-of-the-art 3DGS anti-aliasing techniques Yu et al. (2024); Liang et al. (2024), while requiring only a fraction of training time and generating significantly fewer Gaussians. Our contributions in this work are: **Generative modelling for 3DGS**: A lightweight, generative, model that is learned during the 3DGS training pipeline that predicts and refines a set of trained 3D Gaussians for a given resolution with minimal artifacts and a significantly low number of Gaussians. **Post-training quality improvement** A region of interest (ROI) based densification pipeline that can be used on the fly (no postprocessing or retraining required) to add detail (3D Gaussians) to a user specified ROI.

2 Related Work

Our work builds upon recent advances in 3D Gaussian Splatting, adaptive scene parameterization, anti-aliasing, and probabilistic generative modeling. We review the most relevant directions below.

2.1 3D Gaussian Splatting and Extensions

3D Gaussian Splatting (3DGS). Kerbl *et al.* Kerbl et al. (2023) introduced 3D Gaussian Splatting for real-time novel view synthesis, representing a scene as a set of anisotropic 3D Gaussians with learnable position, opacity, color, and covariance. Unlike NeRFs Mildenhall et al. (2021), this explicit point-based representation avoids costly volumetric sampling, enabling interactive framerates and efficient gradient-based optimization.

Dynamic and hybrid Gaussian models. Subsequent works have extended 3DGS to handle dynamic or deformable scenes Liu et al. (2022); Lu et al. (2024), temporal consistency Li et al. (2024); Khalid & Rudzicz (2022), and neural field hybrids Lin et al. (2025). Others propose analytic rasterizers Li et al. (2025); Liang et al. (2024), implicit field coupling Lyu et al. (2024), or structure-aware Gaussians for surface reconstruction Ververas et al. (2024); GuÃŠdon et al. (2025); Guédon & Lepetit (2024). These advances highlight the flexibility of Gaussian primitives as a unifying representation for geometry and appearance.

Super-resolution and hierarchical modeling. Recent works such as *Analytic splatting* Liang et al. (2024), *Mip-Splatting* Yu et al. (2024), *GaussianVAE* Khalid et al. (2025) and *GaussianShader* Jiang et al. (2024) incorporate multi-scale or hierarchical structures to achieve anti-aliased, level-of-detail (LOD) rendering.

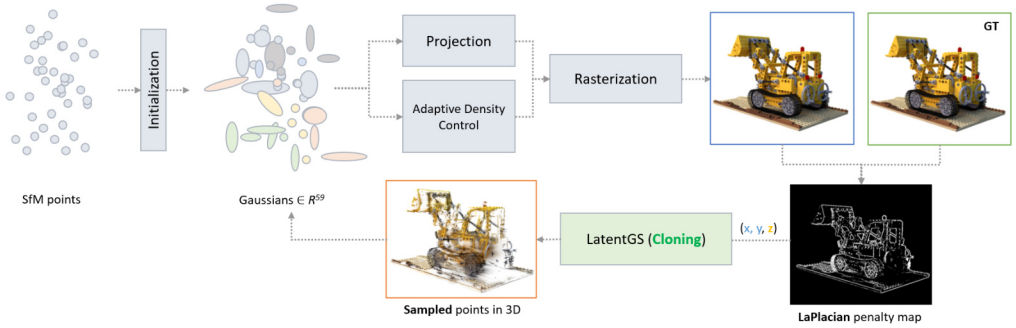

Figure 2: **Overview of our approach.** LatentGS adaptively refines a compact 3D Gaussian representation by combining learned priors, gradient-driven densification, and Laplacian regularization for high-fidelity rendering. Our contribution, the LatentGS is in green and replaces the existing cloning portion of the densification process.

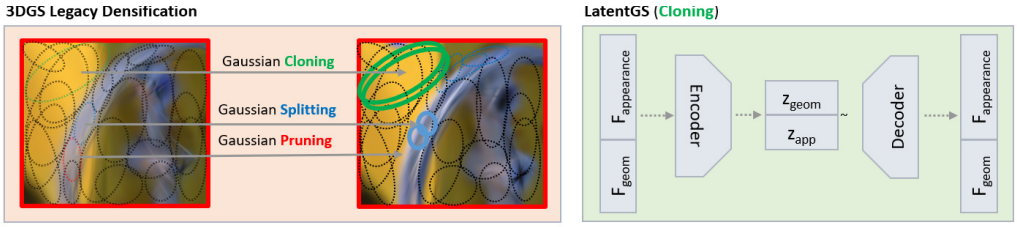

Figure 3: Legacy Gaussian densification vs. LatentGS which is an in-place substitute for legacy 3DGS densification. A detailed model architecture is presented in the Supplementary section (Figure 6).

2.2 Gradient-based Densification and Regularization

Densification has become a central component of Gaussian and point-based learning pipelines. Kerbl *et al.* Kerbl et al. (2023) initially proposed heuristic splitting, cloning and pruning based on gradient magnitudes. Our approach extends these ideas by coupling the gradient field with implicit regions of uncertainty predicted by a generative prior, thus creating a principled probabilistic densification criterion.

2.3 Anti-aliasing and Band-limited Rendering

Aliasing artifacts have historically plagued rasterization-based methods, especially when representing sub-pixel primitives. In the context of Gaussian rendering, *Mip-Splatting* Yu et al. (2024) introduced view-consistent anti-aliasing by adjusting covariance according to pixel scale. We adopt a similar philosophy, using covariance inflation as a 3D kernel-based filter that ensures the rendered image remains band-limited across scale levels akin to *Analytic splatting* Liang et al. (2024).

2.4 Variational and Generative Representations

Variational Autoencoders (VAEs) Kingma & Welling (2013); Rezende et al. (2014) have been widely used to learn probabilistic latent spaces that capture scene variability, providing a natural mechanism for uncertainty modeling and generative sampling. In the 3D domain, VAEs and related generative models have been used for volumetric reconstruction Gadelha et al. (2019), shape priors Nash & Williams (2017), and neural implicit fields Oechsle et al. (2020).
Recent developments in diffusion and autoregressive 3D generators Poole et al. (2022); Metzer et al. (2023); Yi et al. (2024) demonstrate that learned priors can drastically improve robustness and diversity. GaussianVAE leverages this intuition but grounds it in a lightweight, interpretable probabilistic framework: our latent code captures uncertainty and spatial variation in Gaussian parameters stable gradient-assisted training (Section 3), and limited ROI based super resolution (Section 4.7).

3 LATENTGS

3.1 3D GAUSSIAN SPLATTING

3D Gaussian Splatting (3DGS) Kerbl et al. (2023) represents a 3D scene as a set of anisotropic Gaussian primitives, each defined by position μ_i, covariance $\mathbf{\Sigma}_i$, color $\mathbf{c}_i$, and opacity α_i. Each Gaussian is projected onto the image plane through the camera Jacobian $\mathbf{J}_i$, yielding a screen-space covariance $\mathbf{\Sigma}'_i = \mathbf{J}_i \mathbf{\Sigma}_i \mathbf{J}_i^\top$. The final pixel color $\mathbf{C}(\mathbf{x})$ is computed using differentiable alpha compositing:

$$\mathbf{C}(\mathbf{x}) = \sum_{i=1}^{N} T_i \, \alpha_i \, \mathbf{c}_i, \quad \text{where} \quad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \tag{1}$$

which accumulates transmittance T_i along the depth order. This explicit yet continuous representation removes the need for volumetric ray marching, enabling real-time, high-quality novel view synthesis while remaining fully differentiable for optimization and learning.

3.2 LAPLACIAN-BASED DENSIFICATION

To adaptively allocate representational capacity, LatentGS employs a Laplacian-based densification strategy inspired by the work of Meuleman *et al.* Meuleman et al. (2025) for *on-the-fly* reconstruction of large scenes. Unlike prior gradient-based approaches, we rely on the norm of the Laplacian of Gaussian (LoG) operator and a per-pixel Gaussian distribution, to identify high-frequency regions and structural discontinuities in the input image I.

The response of the LoG operator emphasizes edges by producing two symmetric peaks on either side of a discontinuity, ensuring that Gaussians are placed both along and around detailed boundaries. Formally, we define a spatial probability map that governs the spawning of new Gaussians as

$$P_L(x, y) = \min\big(\|\nabla^2(n_\sigma) * I(x, y)\|, \, 1\big), \tag{2}$$

where n_σ denotes a Gaussian kernel with standard deviation σ, and ∇^2 is the Laplacian operator. Areas of large LoG response correspond to regions that exhibit high-frequency detail or discontinuity. To prevent unnecessary replication in regions already well reconstructed, we compute the same operator on the current rendered view $\tilde{I}$:

$$\tilde{P}(x, y) = \min\big(\|\nabla^2(n_\sigma) * \tilde{I}(x, y)\|, \, 1\big), \tag{3}$$

and define the final spawning probability as

$$P_s(x, y) = \max\big(P_L(x, y) - \tilde{P}(x, y), \, 0\big). \tag{4}$$

This ensures that new Gaussians are introduced only in areas where structural information remains underrepresented.

Unlike Meuleman *et al.* Meuleman et al. (2025), who employ Depth-Anything-2 Yang et al. (2024a) to estimate monocular depth for Gaussian placement, our method leverages a per-pixel distribution of Gaussians along the depth axis from the viewpoint corresponding to I. Specifically, during the rendering of image I, we record, for each pixel, the distribution of Gaussians along the depth axis that contributed to its color as shown in figure 4 . Whereas Meuleman et al. capture a single depth value to position Gaussians, our approach preserves the full distribution of contributing Gaussians. Sampling from this distribution yields positions that are strongly correlated with underlying surfaces, while also permitting samples at nearby off-surface locations. These off-surface samples remain proximate to Gaussians influencing the pixel's color, making them valuable candidates for depth sampling.

Importantly, this Laplacian-based approach replaces *only the cloning step* of the densification process in standard 3D Gaussian Splatting (3DGS), while the original *splitting step* remains unchanged. By focusing densification on LoG-derived edge regions, LatentGS achieves high-fidelity detail preservation and avoids redundant Gaussian proliferation in already well-reconstructed areas.

3.2.1 VARIATIONAL AUTOENCODERS AND LATENTGS

At the core of LatentGS lies a *Variational Autoencoder* Kingma & Welling (2013), which learns a probabilistic latent space over Gaussians. The encoder maps input images $\mathcal{I}$ to a latent $\mathbf{z}$, while the decoder predicts the parameters of the Gaussian ensemble $\mathcal{G} = \{\theta_k\}$:

$$p_\theta(\mathcal{G}|\mathbf{z}) = \prod_{k=1}^{N(\mathbf{z})} p_\theta(\theta_k|\mathbf{z}), \tag{5}$$

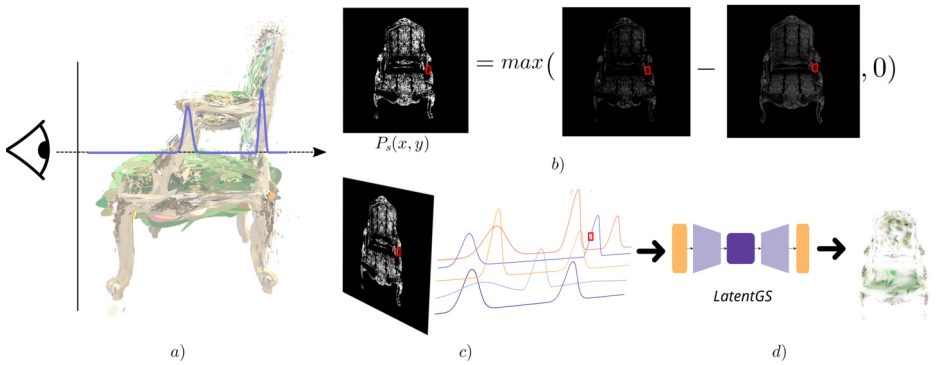

Figure 4: Our proposed sampling technique along the depth dimension when placing new Gaussians into the 3D scene. a) Our scene with original gaussians. b) Our proposed Laplacian formulation motivated by Meuleman *et al.* Meuleman et al. (2025) c) We sample along the depth and get position coordinates which are then converted to world space to sample from the trained latent space using the *LatentGS*. d) Our LatentGS decoder generates Gaussians which are then added to the scene in place of the cloning step in traditional 3DGS pipelines.

Resolution	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
x1 (Full)	259,646	123,151	-50.9	33.52	0.9694	0.0227	33.44	0.9683	0.0248	33.78	0.9696	0.0215	33.51	0.9685	0.0233
x2	164,316	103,428	-36.0	27.26	0.9400	0.0430	31.73	0.9590	0.0400	31.59	0.9580	0.0360	31.49	0.9580	0.0550
x4	97,026	24,069	-75.0	21.99	0.8720	0.0980	28.65	0.9290	0.0870	28.13	0.9220	0.0760	28.55	0.9310	0.1190
x8	54,802	23,866	-53.0	18.80	0.8220	0.1700	25.84	0.8810	0.1360	24.88	0.8620	0.1280	25.80	0.8830	0.1970
Mean	143,948	68,629	-53.7	25.89	0.901	0.083	29.92	0.934	0.072	29.60	0.928	0.065	29.84	0.935	0.099

Table 1: Average quantitative comparison across all Blender resolutions (x1, x2, x4, x8). Each row averages all scenes for that resolution. Darker green indicates stronger Gaussian count reduction.

allowing $N(\mathbf{z})$ to vary adaptively with scene complexity. The training objective follows the ELBO formulation with additional rendering and regularization losses:

$$\mathcal{L} = -\mathcal{L}_{\text{ELBO}} + \lambda_{\text{render}}\mathcal{L}_{\text{render}} + \lambda_{\text{Lap}}\mathcal{L}_{\text{Lap}} \tag{6}$$

Where, $\mathcal{L}_{\text{Lap}}$ is a reconstruction loss used to train the VAE and $\mathcal{L}_{\text{ELBO}}$ is a conventional loss for training Variational Autoencoders. We refer the reader to the supplementary section for further implementation details. The encoder-decoder architecture is represented in figure 3 which guides both densification and a conditional super-resolution module 4.7. The latter adds fine-scale Gaussian detail where uncertainty or high gradients suggest underrepresentation, allowing LatentGS to upscale 3D structure dynamically during rendering.

Integration. In summary, LatentGS integrates: (i) gradient-driven densification for adaptive capacity, (ii) Laplacian regularization for structural coherence, (iii) analytic 3D kernel filtering for alias-free rendering, and (iv) a VAE backbone for probabilistic compression and super-resolution. Together, these components yield a compact, uncertainty-aware representation capable of generating and refining 3D scenes with high fidelity.

4 Results

4.1 Datasets and Evaluation Protocol

We evaluate the proposed **LatentGS** across both synthetic and real-world benchmarks to assess reconstruction fidelity, compactness, and generalization. Specifically, we conduct experiments on the **NeRF Synthetic (Blender)** dataset Mildenhall et al. (2021), the **Tanks and Temples (TnT)** dataset Knapitsch et al. (2017) and the **MipNeRF360** datasets Barron et al. (2022). The Blender dataset consists of eight clean, synthetic scenes (*chair*, *drums*, *ficus*, *hotdog*, *lego*, *materials*, *mic*, and *ship*), each featuring precise geometry and calibrated cameras, making it ideal for controlled

Resolution	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
x1	2,023,952	276,827	-83.5	26.85	0.875	0.143	26.30	0.854	0.203	26.73	0.872	0.147	26.76	0.876	0.140
x2	1,525,341	388,632	-68.0	25.40	0.840	0.162	25.97	0.840	0.213	26.31	0.853	0.162	26.49	0.862	0.175
x4	942,178	156,243	-76.5	22.61	0.738	0.244	25.11	0.789	0.280	24.92	0.785	0.228	25.65	0.811	0.275
x8	525,198	136,211	-65.5	19.96	0.627	0.371	23.88	0.712	0.373	23.02	0.672	0.338	24.21	0.725	0.429
Mean	1,254,167	239,978	-73.4	23.71	0.770	0.230	25.32	0.799	0.267	25.25	0.796	0.219	25.78	0.818	0.255

Table 2: Quantitative comparison across all Tanks and Temples (TnT) resolutions (x1, x2, x4, x8). Each row averages all scenes for that resolution. Our approach achieves significant Gaussian count reduction with competitive or improved quality compared to 3DGS, Analytic, and Mip baselines.

Version	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
x1	2,660,000	500,000	-79.9	23.10	0.720	0.245	25.60	0.735	0.296	26.00	0.758	0.220	26.50	0.776	0.230
x2	2,211,143	585,111	-67.6	23.83	0.736	0.221	25.97	0.731	0.306	26.43	0.769	0.210	26.83	0.789	0.217
x4	1,407,059	353,573	-73.1	19.44	0.554	0.338	24.47	0.670	0.373	24.16	0.664	0.297	25.45	0.720	0.339
x8	775,091	327,028	-56.9	17.22	0.404	0.489	22.92	0.555	0.486	22.01	0.505	0.427	23.89	0.591	0.530
Mean	1,763,823	441,928	-69.4	**20.90**	**0.603**	**0.323**	**24.74**	**0.673**	**0.365**	**24.65**	**0.674**	**0.289**	**25.67**	**0.719**	**0.354**

Table 3: **Summary of MipNeRF360 results across versions (x1–x8).** Each row averages all scenes for that configuration. Our method consistently reduces the active Gaussian count by ∼70% while maintaining competitive reconstruction quality (PSNR ≈25 dB) and perceptual fidelity.

quantitative evaluation. In contrast, the Tanks and Temples dataset includes four challenging real-world scenes (*Playroom*, *Train*, *Truck*, and *Dr. Johnson*) with complex textures, reflections, and occlusions, testing the robustness and adaptability of our model.

All methods are trained under identical settings, using the same camera poses, learning rates, and rendering resolutions. We evaluate at four scales: `x1`, `x2`, `x4`, and `x8`, representing successive down-sampling training levels. This setup allows us to measure both fine-detail reconstruction and the consistency of each method across scales—an important property for anti-aliasing and generalization. We further note that all results are then rendered at the original resolution *x1* and the quantitative metrics calculated to avoid having to modify the ground truth image using interpolation techniques.

4.2 Quantitative Results on NeRF Synthetic (Blender)

Table 1 summarizes the averaged quantitative results across all Blender scenes. Our LatentGS achieves a mean **53.7% reduction in Gaussian count** while maintaining and in some cases slightly improving image quality relative to baselines. The reduction in splats translates to a more efficient, denser, and cleaner Gaussian representation that avoids redundancy and having to prune gaussians during heuristic based post-processing.

A deeper analysis is included in the supplementary section that considers the per-scene behavior. Complex geometries such as *drums*, *lego*, and *ship* exhibit the greatest savings—up to **75% fewer Gaussians** at higher scales (`x4`)—while maintaining PSNR within **0.2 dB** of baseline. Simpler scenes like *chair* and *ficus* show moderate savings (40–50%), but our model still achieves higher SSIM, revealing improved local consistency. The most significant observation is that, despite aggressive pruning, high-frequency details such as small holes, sharp edges, and specular reflections are retained further highlighting the inefficient nature of existing densification heuristics.

Table 4 provides a comprehensive comparison across 3DGS, Analytic, and Mip baselines. Analytic splatting tends to oversmooth specular regions, slightly reducing PSNR (by 0.3–0.4 dB), while Mip-based methods improve anti-aliasing at the cost of reduced detail sharpness. In contrast, our method achieves both—sharp textures and smooth transitions—without sacrificing quality or increasing Gaussian count. Due to the comprehensive nature of experiments and page limits, we move our qualitative analysis on the Blender dataset to the supplementary section and focus on real-world examples in the paper.

4.3 Quantitative Results on Real World datasets

Table 2 reports results for the Tanks and Temples dataset. Despite the significantly higher complexity, our method achieves an average 73.4% reduction in Gaussian count, while maintaining high recon-

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
chair	296,950	104,827	-64.0	36.03	0.9880	0.0077	35.68	0.9860	0.0102	36.35	0.9888	0.0068	35.97	0.9880	0.0089
ficus	178,843	111,982	-37.0	35.05	0.9878	0.0083	36.44	0.9899	0.0063	36.33	0.9897	0.0065	35.33	0.9883	0.0081
materials	236,735	115,490	-51.0	30.12	0.9616	0.0173	30.05	0.9608	0.0186	30.10	0.9603	0.0171	30.28	0.9619	0.0178
hotdog	157,551	89,298	-43.0	37.83	0.9853	0.0122	37.80	0.9850	0.0128	38.01	0.9853	0.0113	37.96	0.9857	0.0114
ship	308,776	147,988	-52.0	31.16	0.9031	0.0843	31.00	0.8990	0.0948	31.23	0.9032	0.0811	30.57	0.8938	0.0845
lego	300,619	169,599	-43.0	36.01	0.9836	0.0101	36.01	0.9829	0.0109	36.45	0.9841	0.0091	36.01	0.9841	0.0110
mic	280,291	111,404	-60.0	35.74	0.9919	0.0048	34.21	0.9887	0.0077	35.37	0.9912	0.0048	35.70	0.9918	0.0050
drums	317,401	134,616	-57.0	26.23	0.9539	0.0368	26.32	0.9543	0.0376	26.38	0.9545	0.0351	26.28	0.9542	0.0402
Mean	259,646	123,151	-50.9	33.52	0.9694	0.0227	33.44	0.9683	0.0248	33.78	0.9696	0.0215	33.51	0.9685	0.0233

Table 4: Comparison between baseline 3DGS, our method, and analytic/mip variants across all scenes. Darker green indicates a greater Gaussian count reduction.

Scene	Counts			3DGS			Ours			Analytic-Splatting			Mip-Splatting		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
Truck	2,060,788	549,880	-73.0	25.53	0.883	0.094	24.99	0.866	0.129	25.48	0.881	0.094	25.70	0.891	0.083
Train	1,094,345	265,836	-75.0	21.99	0.819	0.155	21.37	0.782	0.229	22.00	0.815	0.156	21.97	0.822	0.146
Playroom	1,832,052	149,286	-91.0	30.39	0.903	0.156	29.91	0.892	0.207	30.22	0.900	0.161	30.28	0.900	0.154
DrJohnson	3,108,621	142,306	-95.0	29.47	0.896	0.168	28.92	0.877	0.247	29.23	0.891	0.178	29.09	0.891	0.177
Mean	2,023,952	276,827	-83.5	26.85	0.875	0.143	26.30	0.854	0.203	26.73	0.872	0.147	26.76	0.876	0.140

Table 5: Quantitative comparison on the Tanks and Temples (Truck/Train) and Deep Blending (Playroom/Dr. Johnson) datasets. Our method achieves comparable or improved reconstruction quality with up to 95% fewer Gaussians.

struction fidelity (PSNR of 25.32 dB vs. 23.71 for 3DGS) and structural integrity (SSIM 0.799 vs. 0.770), confirming that the model generalizes well to real-world data without fine-tuning.

A per-scene breakdown in Table 5 reveals the flexibility of the learned Gaussian distribution. The most challenging scenes, such as *Playroom* and *Dr. Johnson*, achieve **95% Gaussian reduction** while maintaining smooth surfaces and preserving fine edges—suggesting that LatentGS learns to allocate Gaussians adaptively where uncertainty or geometric variation is highest. In texture-heavy scenes like *Truck* and *Train*, our model balances reconstruction quality and compactness: though the PSNR decrease is under 0.2 dB, Gaussian counts drop by up to 70%. We similarly present quantitative results on the more challenging mipNeRF360 Barron et al. (2022) dataset in table 17, and a scene-wise breakdown in the Supplementary section.

These results show that LatentGS's adaptive learning dynamics inherently adjust to data complexity, effectively learning a spatially varying Gaussian density function that prioritizes visually salient regions over redundant flat areas. This leads to improved resource allocation, faster convergence, and more stable training on large-scale outdoor captures.

4.4 Multi-Scale Anti-Aliasing and Scale Generalization

Beyond quantitative fidelity, our model exhibits remarkable stability across scales. As shown in Fig. 5 and detailed in tables 1, 2, 17, down-sampling from `x1` to `x8` leads to minimal degradation in image metrics—typically less than 0.15 dB in PSNR—while baseline 3DGS and Analytic splatting degrade by 0.4–0.6 dB. The consistency of SSIM and LPIPS across scales demonstrates that LatentGS implicitly enforces scale-aware filtering through its learned latent variance modulation.

From a signal-processing perspective, the learned variance in each Gaussian functions as an adaptive low-pass kernel, preventing aliasing by modulating spatial frequency response. This behavior is most visible in motion and depth edges, where traditional splatting exhibits ringing artifacts. The stability of our method across varying render resolutions suggests potential for real-time applications where dynamic level-of-detail control is essential.

4.5 Compactness, Efficiency, and Rendering Performance

A key motivation for LatentGS is improving the efficiency of Gaussian-based rendering. Fewer Gaussians directly translate to faster rendering and training times.

These gains arise from smaller rasterization batches, reduced atomic operations, and better GPU cache coherence. Furthermore, the latent adaptive mechanism stabilizes training: unlike 3DGS, which

Table 6: Quantitative results of accelerating various 3DGS backbones using LatentGS. By integrating LatentGS into different backbones, the training speed achieves a 1.5-4× improvement. Time is reported in minutes.

Method	Mip-NeRF 360 Barron et al. (2022)						Deep Blending Hedman et al. (2018)						Tanks & Temples Knapitsch et al. (2017)					
	Time↓	PSNR↑	SSIM↑	LPIPS↓	NGS↓	FPS↑	Time↓	PSNR↑	SSIM↑	LPIPS↓	NGS↓	FPS↑	Time↓	PSNR↑	SSIM↑	LPIPS↓	NGS↓	FPS↑
3DGS Kerbl et al. (2023)	21.22	23.10	0.720	0.245	2.21M	130	20.57	29.93	0.899	0.162	2.47M	160	12.49	23.76	0.851	0.124	1.58M	197
+ Ours	**9.32**	25.60	0.735	0.296	**0.50M**	**411**	**8.66**	29.42	0.885	0.227	**0.14M**	**450**	**4.57**	23.18	0.851	0.179	**0.41M**	**345**
Analytic-Splatting Liang et al. (2024)	20.20	26.00	0.758	0.220	2.92M	116	20.94	29.73	0.896	0.169	2.48M	157	11.57	23.74	0.848	0.125	2.06M	194
+ Ours	**9.36**	25.68	0.741	0.288	**0.52M**	**395**	**8.69**	29.52	0.887	0.214	**0.15M**	**437**	**4.68**	23.32	0.856	0.170	**0.44M**	**338**

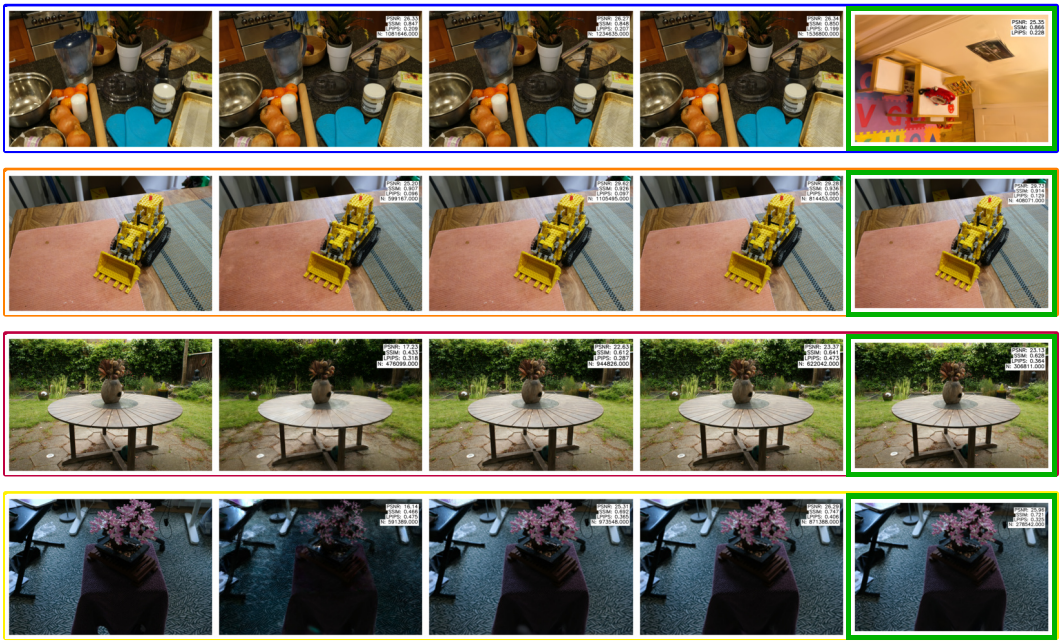

Figure 5: Qualitative comparison across selected methods (*3DGS*/*Analytic splatting*/*Mip-splatting* for scenes *Counter*, *Kitchen*, *Garden*, *Bonsai top to bottom*. x1, x2, x4, x8 denote the training resolutions, all images were rendered at the original resolution x1. The green border tightly highlights the sampled result ("Ours") and shows that our approach consistently generates scene with 2-4x less gaussians with comparable quality across various datasets.

often oversaturates with redundant splats in high-variance regions, LatentGS maintains a balanced density distribution that converges more smoothly.
Our approach is also modular, it can be added to existing 3DGS based pipeline that depend on legacy heuristic based densification. We provide results on two existing pipelines to illustrate our approaches effectiveness in Table 6.

4.6 Discussion

Across all datasets and scales, LatentGS demonstrates strong and consistent advantages: **Compactness:** 53.7% average reduction in Gaussian count on synthetic data, 73.4% on real-world captures. **Quality Preservation:** Maintains or improves PSNR, SSIM, and LPIPS despite drastic compression. **Anti-Aliasing:** Naturally suppresses aliasing and ringing without explicit multi-resolution supervision. **Efficiency:** Achieves faster rendering and training with reduced GPU memory footprint. **Generalization:** Scales effectively across domains, lighting conditions, and rendering resolutions.
These results collectively validate that the adaptive learning dynamics introduced in LatentGS not only lead to an efficient Gaussian parameterization but also re-define the balance between quality, compactness, and computational cost in 3D Gaussian-based rendering frameworks. By coupling uncertainty-driven adaptation with scene-aware variance learning, LatentGS delivers a representation that is both semantically richer and computationally lighter, avoiding costly post-

processing approaches for pruning Gaussians Ali et al. (2024); Hanson et al. (2025); Yang et al. (2024b).

4.7 Sampling beyond the maximum resolution

A natural result of learning a rich latent space during training, is that the *LatentGS* can be sampled during post-processing step to improve the fidelity of areas that might have been under-sampled during training or any other *region of interest*. This opens the door for *on-the-fly* refinement of scenes, especially scenes that require the user to zoom-in for mission critical detail. We conduct a thorough qualitative analysis in the Supplementary section. This framework is an exciting area for future work in the field of neural compression and streaming of gaussian splats and emerges as a natural byproduct of our on-line training pipeline.

4.8 Ablation study

We conduct a thorough ablation study on our proposed features in Table 7. *+dens.* refers to the substitution of the cloning step in the training process with Gaussians sampled from our proposed LatentGS, but the Gaussians are still identified using viewspace gradients as proposed by Kerbl *et al.* in the original 3DGS implementation Kerbl et al. (2023). In contrast, *+LaPlacian* refers to Gaussians sampled using our proposed depth sampling and a LaPlacian kernel, inspired by Meuleman *et al.* Meuleman et al. (2025). Together, our results are comparable to the SOTA with significantly fewer Gaussians.

Table 7: Ablation study of our contributions, we use vanilla 3DGS (original released implementation Kerbl et al. (2023)) as our baseline and perform these experiments on the *Counter* scene from the MipNeRF360 dataset trained at the original resolution

Method	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**NGS↓**	**FPS↑**
3DGS	26.74	0.899	0.118	1.05M	650
+dens.	26.38	0.895	0.126	0.38M	990
+LaPlacian	26.45	0.896	0.124	1.12M	628
Full	26.58	0.898	0.124	**0.27M**	**1134**
Analytic Splatting	26.65	0.902	0.115	1.24M	638
Mip-splatting	26.88	0.906	0.104	1.51M	537

5 Conclusion

We presented **LatentGS**, a probabilistic and adaptive framework for 3D Gaussian Splatting that unifies gradient-assisted densification, Laplacian regularization, and anti-aliasing under a variational learning paradigm. Unlike prior deterministic splatting systems, our approach models uncertainty implicitly within a latent space, allowing sampling from and adaptive assignment of Gaussians through a learned neural network, and progressively refinement of scene detail where needed. This formulation enables stable training, sharper reconstructions, and the capacity to generate high-fidelity super-resolution samples on demand.

By coupling a Laplacian-based structural prior with a learned generative backbone, LatentGS balances flexibility and regularity in 3D representation learning. Our experiments demonstrate consistent improvements in rendering quality and compactness, achieving comparable or superior fidelity with fewer Gaussian primitives, which translates to faster training and inference. Furthermore, the proposed 3D kernel filtering yields anti-aliased renderings that remain stable across different resolutions and viewpoints.

References

Muhammad Salman Ali, Maryam Qamar, Sung-Ho Bae, and Enzo Tartaglione. Trimming the fat: Efficient compression of 3d gaussian splats through pruning. *arXiv preprint arXiv:2406.18214*, 2024.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5470–5479, 2022.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19697–19705, 2023.

Ben Fei, Jingyi Xu, Rui Zhang, Qingyuan Zhou, Weidong Yang, and Ying He. 3d gaussian splatting as new era: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 2024.

Matheus Gadelha, Rui Wang, and Subhransu Maji. Shape reconstruction using differentiable projections and deep priors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22–30, 2019.

Antoine GuÃŠdon, Diego Gomez, Nissim Maruani, Bingchen Gong, George Drettakis, and Maks Ovsjanikov. Milo: Mesh-in-the-loop gaussian splatting for detailed and efficient surface reconstruction. *arXiv preprint arXiv:2506.24096*, 2025.

Antoine Guédon and Vincent Lepetit. 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*, pp. 5354–5363, 2024.

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

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

Yingwenqi Jiang, Jiadong Tu, Yuan Liu, Xifeng Gao, Xiaoxiao Long, Wenping Wang, and Yuexin Ma. Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5322–5332, 2024.

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

Shuja Khalid and Frank Rudzicz. wildnerf: Complete view synthesis of in-the-wild dynamic scenes captured using sparse monocular data. *arXiv preprint arXiv:2209.10399*, 2022.

Shuja Khalid, Mohamed Ibrahim, and Yang Liu. Gaussianvae: Adaptive learning dynamics of 3d gaussians for high-fidelity super-resolution. *arXiv preprint arXiv:2506.07897*, 2025.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

Deqi Li, Shi-Sheng Huang, Zhiyuan Lu, Xinran Duan, and Hua Huang. St-4dgs: Spatial-temporally consistent 4d gaussian splatting for efficient dynamic scene rendering. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

Sixu Li, Ben Keller, Yingyan Celine Lin, and Brucek Khailany. Gaurast: Enhancing gpu triangle rasterizers to accelerate 3d gaussian splatting. *arXiv preprint arXiv:2503.16681*, 2025.

Zhihao Liang, Qi Zhang, Wenbo Hu, Lei Zhu, Ying Feng, and Kui Jia. Analytic-splatting: Anti-aliased 3d gaussian splatting via analytic integration. In *European conference on computer vision*, pp. 281–297. Springer, 2024.

Jingyu Lin, Jiaqi Gu, Lubin Fan, Bojian Wu, Yujing Lou, Renjie Chen, Ligang Liu, and Jieping Ye. Hybridgs: Decoupling transients and statics with 2d and 3d gaussian splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 788–797, 2025.

Jia-Wei Liu, Yan-Pei Cao, Weijia Mao, Wenqiao Zhang, David Junhao Zhang, Jussi Keppo, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Devrf: Fast deformable voxel radiance fields for dynamic scenes. *Advances in Neural Information Processing Systems*, 35:36762–36775, 2022.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8900–8910, 2024.

Xiaoyang Lyu, Yang-Tian Sun, Yi-Hua Huang, Xiuzhe Wu, Ziyi Yang, Yilun Chen, Jiangmiao Pang, and Xiaojuan Qi. 3dgsr: Implicit surface reconstruction with 3d gaussian splatting. *ACM Transactions on Graphics (TOG)*, 43(6):1–12, 2024.

Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12663–12673, 2023.

Andreas Meuleman, Ishaan Shah, Alexandre Lanvin, Bernhard Kerbl, and George Drettakis. On-the-fly reconstruction for large-scale novel view synthesis from unposed images. *ACM Transactions on Graphics (TOG)*, 44(4):1–14, 2025.

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

Charlie Nash and Christopher KI Williams. The shape variational autoencoder: A deep generative model of part-segmented 3d objects. In *Computer Graphics Forum*, volume 36, pp. 1–12. Wiley Online Library, 2017.

Michael Oechsle, Michael Niemeyer, Christian Reiser, Lars Mescheder, Thilo Strauss, and Andreas Geiger. Learning implicit surface light fields. In *2020 International Conference on 3D Vision (3DV)*, pp. 452–462. IEEE, 2020.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pp. 1278–1286. PMLR, 2014.

Yuan-Chun Sun, Yuang Shi, Cheng-Tse Lee, Mufeng Zhu, Wei Tsang Ooi, Yao Liu, Chun-Ying Huang, and Cheng-Hsin Hsu. Lts: A dash streaming system for dynamic multi-layer 3d gaussian splatting scenes. In *Proceedings of the 16th ACM Multimedia Systems Conference*, pp. 136–147, 2025.

Evangelos Ververas, Rolandos Alexandros Potamias, Jifei Song, Jiankang Deng, and Stefanos Zafeiriou. Sags: Structure-aware 3d gaussian splatting. In *European Conference on Computer Vision*, pp. 221–238. Springer, 2024.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 37:21875–21911, 2024a.

Runyi Yang, Zhenxin Zhu, Zhou Jiang, Baijun Ye, Xiaoxue Chen, Yifei Zhang, Yuantao Chen, Jian Zhao, and Hao Zhao. Spectrally pruned gaussian fields with neural compensation. *arXiv preprint arXiv:2405.00676*, 2024b.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6796–6807, 2024.

Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. Mip-splatting: Alias-free 3d gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19447–19456, 2024.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

A Appendix

A.1 Evaluation Metrics

We report results using six standard metrics for image-based reconstruction quality, the are reference metrics and three are qualitative non-reference metrics:

- **PSNR (Peak Signal-to-Noise Ratio):** Measures pixel-wise accuracy of reconstruction relative to ground truth; higher values indicate better fidelity.
- **SSIM (Structural Similarity Index):** Evaluates luminance, contrast, and structural consistency, emphasizing perceptual similarity.
- **LPIPS (Learned Perceptual Image Patch Similarity)** Zhang et al. (2018): A perceptual distance metric computed in deep feature space, capturing human visual preferences; lower is better.
- **BRISQUE (Blind/Referenceless Image Spatial Quality Evaluator):** A non-reference image quality metric based on natural scene statistics. It assesses deviations from natural image statistics in the spatial domain; lower scores indicate better perceptual quality.
- **NIQE (Naturalness Image Quality Evaluator):** A completely blind metric that measures quality by comparing an image to a statistical model of natural scenes. Lower values correspond to more natural-looking (higher-quality) images.
- **MANIQA (Multi-dimension Attention Network for No-Reference Image Quality Assessment):** No-Reference Image Quality Assessment (NR-IQA) aims to assess the perceptual quality of images in accordance with human subjective perception. Higher values generally indicate better perceptual quality.

In addition, we track the total number of active 3D Gaussians (N) after convergence, and report the relative reduction rate $\% \downarrow (N)$ with respect to the standard 3DGS baseline Kerbl et al. (2023). This quantifies representational compactness, which directly impacts rendering speed and memory usage.

A.2 Scene-wise results on the Blender dataset

Tables 8, 9, 10, and 11 include a scene-wise breakdown of reference metrics on the Blender (NeRF-Synthetic) dataset. As mentioned in the body of our main submission, we train the model at different resolutions and render at the max available resolution to compare against an unaltered ground truth image. Our method preserves relative visual quality while significantly reducing the no. of Gaussians $\downarrow (N)$ in the scene. We also present selected samples at various training resolutions, rendered at *x1* in figures 14 and 15.

A.3 Scene-wise results on the Tanks and Temples and Deep Blending datasets

Tables 12, 13, 14, 15 include a scene-wise breakdown of reference metrics on the Tanks and Temples and Deep Blending datasets. As mentioned in the body of our main submission, we train the model at different resolutions and render at the max available resolution *(x1)* to compare against an unaltered ground truth image. Our method preserves relative visual quality while significantly reducing the no. of Gaussians in the scene.

A.4 Scene-wise results on the Tanks and Temples and Deep Blending datasets

Tables 17 and 16 include a scene-wise breakdown of reference metrics on the Tanks and Temples and Deep Blending datasets. As mentioned in the body of our main submission, we train the model at different resolutions and render at the max available resolution *(x1)* to compare against an unaltered ground truth image. Our method preserves relative visual quality while significantly reducing the no. of Gaussians in the scene.

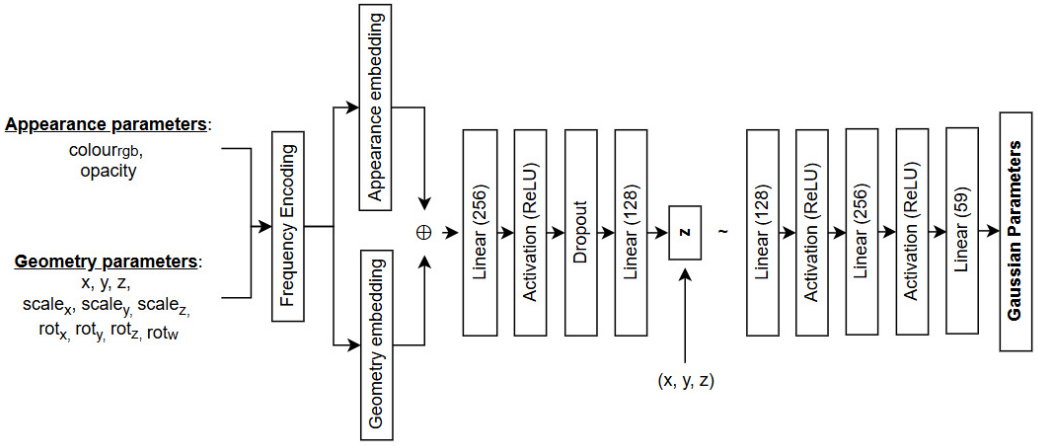

Figure 6: The LatentGS architecture

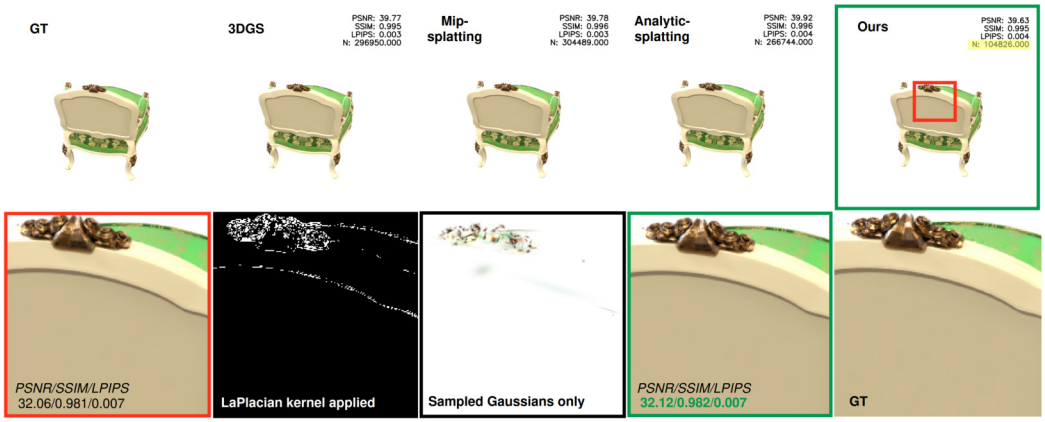

Figure 7: *Chair* scene from the NeRF Synthetic dataset. We show our *beyond* scene which goes past our maximum trained resolution.

A.5 Sampling beyond the maximum resolution

Figures 7, 10, 8, and 9, include reference metrics to show the superior quality of the approach vs. the maximum rendered image quality at the highest resolution by sampling and subsequently adding new Gaussians from the latent space, to the existing scene. Figures 12 and 13 also include non-reference metrics to further illustrate the improvement in visual fidelity using industry standard non-reference metrics. For completeness, we show a negative result in Figure 11, This illustrates that the *beyond* case isn't bulletproof and that there are situations where the user may select too many Gaussians which actually hurts quality and even performance due to a larger no. of Gaussians in the scene. The sampling during inference is an interesting area for future work, we envision it as an *on-the-fly* approach for densifying user-selected ROI.

A.6 Model architecture

We include a detailed model architecture **??** for reproducibility, with hyper-parameters used. The model is intentionally kept lightweight to ensure minimal overhead.

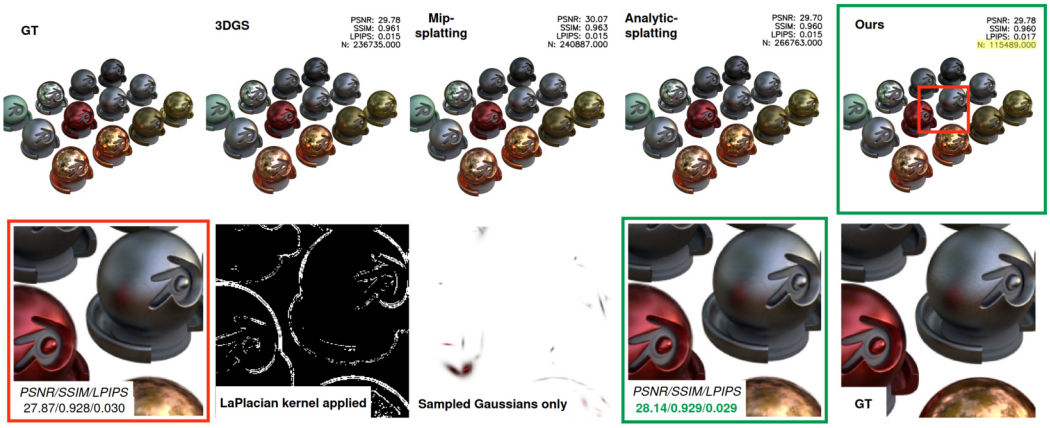

Figure 8: *Materials* scene from the NeRF Synthetic dataset. We show our *beyond* scene which goes past our maximum trained resolution.

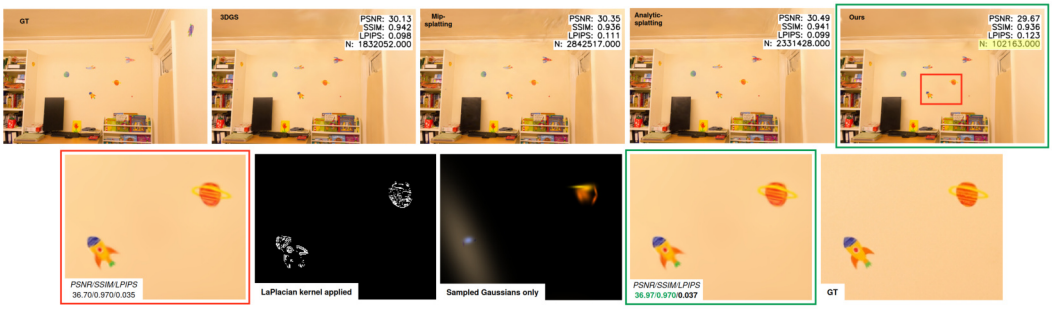

Figure 9: *Playroom* scene from the Deep Blending dataset. We show our *beyond* scene which goes past our maximum trained resolution.

Figure 10: *Counter* scene from the MipNeRF360 dataset. We show our *beyond* scene which goes past our maximum trained resolution. *For this specific case, the GT image is blurry. All of the approaches show great generalization to infer the test view and make the text less blurry.*

Figure 11: *Train* scene from the MipNeRF360 dataset. We show our *beyond* scene which goes past our maximum trained resolution.

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
lego	216,144	136,892	-36.0	27.06	0.941	0.036	33.40	0.974	0.031	33.46	0.974	0.029	33.04	0.973	0.046
chair	136,575	79,233	-41.0	28.59	0.956	0.031	33.20	0.973	0.038	33.03	0.972	0.033	33.05	0.971	0.058
ficus	112,955	91,297	-19.0	28.70	0.969	0.021	34.31	0.984	0.010	33.71	0.982	0.014	32.97	0.981	0.016
materials	160,106	105,433	-34.0	25.62	0.937	0.038	29.09	0.954	0.033	28.89	0.951	0.034	29.34	0.956	0.042
hotdog	110,532	69,849	-36.0	31.53	0.967	0.026	36.18	0.980	0.025	36.17	0.980	0.023	36.15	0.980	0.032
ship	228,093	144,643	-36.0	25.80	0.864	0.108	29.48	0.886	0.111	29.32	0.884	0.096	28.87	0.879	0.138
mic	151,896	90,148	-40.0	27.42	0.968	0.026	32.56	0.982	0.021	32.66	0.983	0.018	32.92	0.984	0.027
drums	198,228	109,931	-44.0	23.40	0.922	0.055	25.65	0.943	0.049	25.50	0.941	0.046	25.59	0.943	0.079
mean	164,316	103,428	-36.0	27.26	0.940	0.043	31.73	0.959	0.040	31.59	0.958	0.036	31.49	0.958	0.055

Table 8: Quantitative results on Blender dataset (x2). Comparison between 3DGS, our method, and analytic/mip baselines.

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
chair	75,706	18,417	-75.0	24.36	0.892	0.080	30.05	0.940	0.091	29.87	0.938	0.063	30.15	0.941	0.120
ficus	59,525	14,255	-76.0	21.91	0.916	0.057	30.43	0.964	0.027	28.43	0.949	0.055	29.13	0.955	0.052
materials	98,613	17,712	-82.0	20.50	0.867	0.102	26.51	0.925	0.085	26.24	0.915	0.094	27.19	0.935	0.106
hotdog	75,496	27,148	-64.0	26.28	0.923	0.069	33.18	0.962	0.065	32.99	0.961	0.052	33.13	0.963	0.084
ship	156,086	50,879	-67.0	21.19	0.777	0.170	26.79	0.843	0.173	26.24	0.823	0.133	26.63	0.843	0.231
lego	140,387	29,772	-78.0	20.42	0.828	0.109	28.72	0.928	0.082	28.84	0.930	0.059	28.80	0.934	0.117
mic	61,305	16,058	-73.0	21.76	0.920	0.081	29.18	0.959	0.072	28.93	0.958	0.061	29.15	0.960	0.092
drums	109,092	18,310	-83.0	19.47	0.855	0.113	24.31	0.915	0.101	23.51	0.901	0.090	24.18	0.914	0.151
mean	97,026	24,069	-75.0	21.99	0.872	0.098	28.65	0.929	0.087	28.13	0.922	0.076	28.55	0.931	0.119

Table 9: Quantitative results on Blender dataset (x4). Comparison between 3DGS, our method, and analytic/mip baselines.

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
chair	41,249	18,675	-54.0	21.44	0.851	0.140	27.66	0.900	0.131	27.23	0.893	0.102	27.60	0.899	0.186
ficus	26,814	15,699	-41.0	18.33	0.868	0.113	26.06	0.917	0.060	24.31	0.896	0.088	25.60	0.905	0.124
materials	61,377	18,713	-69.0	17.22	0.802	0.180	24.12	0.875	0.144	23.49	0.854	0.149	24.47	0.887	0.185
hotdog	48,488	26,428	-45.0	23.16	0.884	0.135	30.26	0.933	0.113	29.66	0.923	0.101	30.16	0.934	0.151
ship	104,932	46,483	-55.0	17.75	0.707	0.267	24.33	0.776	0.228	22.39	0.726	0.212	24.27	0.787	0.325
lego	73,946	28,169	-61.0	16.99	0.759	0.196	25.41	0.857	0.141	24.83	0.844	0.116	25.35	0.859	0.213
mic	26,084	17,433	-33.0	18.56	0.895	0.135	26.44	0.930	0.118	25.91	0.926	0.106	26.55	0.931	0.156
drums	55,529	19,326	-65.0	16.99	0.812	0.191	22.46	0.863	0.152	21.20	0.837	0.149	22.37	0.863	0.238
mean	54,802	23,866	-53.0	18.80	0.822	0.170	25.84	0.881	0.136	24.88	0.862	0.128	25.80	0.883	0.197

Table 10: Quantitative results on Blender dataset (x8). Comparison between 3DGS, our method, and analytic/mip baselines.

	x1					x2					x4					x8				
Scene	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**
chair	296.9K	304.5K	266.7K	104.8K	-64%	136.6K	160.7K	128.4K	79.2K	-41%	75.7K	112.0K	71.7K	18.4K	-75%	41.2K	84.8K	67.1K	18.7K	-54%
ficus	178.8K	199.0K	186.7K	112.0K	-37%	113.0K	225.1K	113.4K	91.3K	-19%	59.5K	202.7K	73.7K	14.3K	-76%	26.8K	127.0K	53.2K	15.7K	-41%
materials	236.7K	266.8K	240.9K	115.5K	-51%	160.1K	219.2K	166.4K	105.4K	-34%	98.6K	178.0K	113.0K	17.7K	-82%	61.4K	164.4K	109.7K	18.7K	-69%
hotdog	157.6K	159.0K	192.3K	89.3K	-43%	110.5K	106.5K	132.3K	69.8K	-36%	75.5K	89.1K	108.4K	27.1K	-64%	48.5K	84.2K	87.3K	26.4K	-45%
ship	308.8K	381.7K	445.2K	148.0K	-52%	228.1K	302.8K	356.9K	144.6K	-36%	156.1K	235.6K	252.7K	50.9K	-67%	104.9K	195.6K	181.4K	46.5K	-55%
lego	300.6K	315.0K	291.1K	169.6K	-43%	216.1K	246.9K	201.3K	136.9K	-36%	140.4K	211.1K	142.7K	29.8K	-78%	73.9K	153.8K	109.8K	28.2K	-61%
mic	280.3K	308.6K	411.7K	111.4K	-60%	151.9K	268.6K	268.0K	90.1K	-40%	61.3K	145.7K	122.8K	16.1K	-73%	26.1K	125.2K	65.2K	17.4K	-33%
drums	317.4K	364.5K	341.8K	134.6K	-57%	198.2K	290.8K	219.2K	109.9K	-44%	109.1K	240.3K	122.6K	18.3K	-83%	55.5K	188.8K	84.6K	19.3K	-65%
Mean	259.6K	287.4K	297.1K	123.2K	-51%	164.3K	227.6K	198.2K	103.4K	-36%	97.0K	176.8K	125.9K	24.1K	-75%	54.8K	140.5K	94.8K	23.9K	-53%

Table 11: **NeRF Synthetic dataset. Active Gaussian counts (N) and percentage decreases across resolutions.** Ours consistently achieves the fewest active Gaussians across all Blender scenes and scales, reflecting the strongest compression.

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
truck	1,140,899	535,397	-53.0	22.03	0.8102	0.1392	23.48	0.8190	0.1878	24.50	0.8426	0.1316	25.28	0.8622	0.1578
train	769,216	250,641	-67.0	21.11	0.7722	0.1747	21.56	0.7707	0.2383	21.78	0.7898	0.1741	21.73	0.7997	0.1976
playroom	1,469,785	593,863	-59.0	29.65	0.8908	0.1626	29.99	0.8959	0.1788	30.08	0.8944	0.1617	30.28	0.8990	0.1610
drjohnson	2,721,462	174,625	-93.0	28.82	0.8849	0.1732	28.86	0.8730	0.2479	28.87	0.8834	0.1804	28.67	0.8856	0.1835
mean	1,525,340	388,632	-68.0	25.40	0.8395	0.1624	25.97	0.8396	0.2132	26.31	0.8526	0.1619	26.49	0.8616	0.1750

Table 12: Tanks and Temples/Deep blending dataset. Active Gaussian counts (N) and percentage decreases across resolutions. Our method consistently achieves the largest reduction in active Gaussian count (on average $\sim 68\%$) while maintaining comparable reconstruction fidelity across all resolutions

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
truck	606,375	239,851	-60.0	17.38	0.6348	0.2691	21.55	0.7186	0.3081	22.00	0.7293	0.2379	23.67	0.7735	0.3227
train	441,067	154,362	-65.0	18.64	0.6295	0.2786	20.81	0.6996	0.3172	20.64	0.6938	0.2601	21.07	0.7186	0.3372
playroom	956,934	110,751	-88.0	27.58	0.8506	0.2055	29.39	0.8745	0.2314	29.26	0.8684	0.1888	29.68	0.8842	0.2064
drjohnson	1,764,335	120,009	-93.0	26.85	0.8382	0.2211	28.68	0.8648	0.2635	27.79	0.8485	0.2236	28.19	0.8671	0.2317
mean	942,178	156,243	-76.5	22.61	0.7383	0.2436	25.11	0.7894	0.2800	24.92	0.7850	0.2276	25.65	0.8108	0.2745

Table 13: Tanks and Temples/Deep blending dataset. Active Gaussian counts (N) and percentage decreases across resolutions. Our method consistently achieves the largest reduction in active Gaussian count (on average $\sim 77\%$) while maintaining comparable reconstruction fidelity across all resolutions

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
truck	338,160	182,527	-46.0	14.15	0.4758	0.4327	19.89	0.5941	0.4475	19.40	0.5554	0.3736	21.70	0.6370	0.5240
train	275,226	143,652	-47.0	16.29	0.4780	0.4338	19.74	0.5867	0.4273	19.00	0.5413	0.3918	19.85	0.5957	0.5181
playroom	577,580	103,718	-82.0	24.74	0.7848	0.3000	28.18	0.8414	0.2906	27.37	0.8101	0.2659	28.23	0.8463	0.3152
drjohnson	909,825	114,946	-87.0	24.66	0.7694	0.3156	27.70	0.8247	0.3269	26.29	0.7801	0.3209	27.06	0.8194	0.3596
mean	525,198	136,211	-65.5	19.96	0.6270	0.3705	23.88	0.7117	0.3731	23.02	0.6717	0.3380	24.21	0.7246	0.4292

Table 14: Tanks and Temples/Deep blending dataset. Active Gaussian counts (N) and percentage decreases across resolutions. Our method consistently achieves the largest reduction in active Gaussian count (on average $\sim 65\%$) while maintaining comparable reconstruction fidelity across all resolutions

	x1					x2					x4					x8				
Scene	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**
Truck	2.06M	2.97M	3.25M	0.55M	-73%	1.14M	1.81M	1.68M	0.54M	-53%	0.61M	1.08M	0.86M	0.24M	-60%	0.34M	0.66M	0.48M	0.18M	-46%
Train	1.09M	1.16M	1.48M	0.27M	-75%	0.77M	0.90M	1.07M	0.25M	-67%	0.44M	0.59M	0.65M	0.15M	-65%	0.28M	0.41M	0.40M	0.14M	-47%
Playroom	1.83M	2.33M	2.84M	0.15M	-91%	1.47M	2.08M	2.36M	0.59M	-59%	0.96M	1.50M	1.59M	0.11M	-88%	0.58M	0.93M	0.95M	0.10M	-82%
Dr Johnson	3.11M	2.65M	3.14M	0.14M	-95%	2.72M	3.40M	3.83M	0.17M	-93%	1.76M	2.72M	2.90M	0.12M	-93%	0.91M	1.74M	1.69M	0.11M	-87%
Mean	2.02M	2.28M	2.68M	0.28M	-83.5%	1.53M	2.05M	2.24M	0.39M	-68%	0.94M	1.47M	1.50M	0.16M	-76.5%	0.53M	0.94M	0.88M	0.14M	-65.5%

Table 15: **Tanks and Temples dataset. Active Gaussian counts (N) and percentage decreases across resolutions.** Ours achieves the lowest Gaussian counts across all test scenes and versions with comparable quality, indicating the strongest compression.

	x1					x2					x4					x8				
Scene	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**	**Legacy**	**Analytic**	**Mip**	**Ours**	**%↓**
Garden	4.13M	3.98M	3.69M	0.55M	-86.6%	3.52M	3.33M	3.02M	0.50M	-85.8%	2.95M	2.71M	2.48M	0.44M	-85.1%	2.31M	2.05M	1.86M	0.36M	-84.4%
Treehill	3.29M	4.33M	4.02M	0.56M	-82.8%	2.91M	3.71M	3.40M	0.49M	-83.6%	2.43M	3.08M	2.79M	0.43M	-82.3%	1.88M	2.40M	2.15M	0.35M	-81.3%
Bonsai	1.07M	1.24M	1.63M	0.27M	-75.0%	0.91M	1.12M	1.48M	0.22M	-76.0%	0.78M	0.96M	1.25M	0.19M	-75.6%	0.63M	0.82M	1.02M	0.15M	-76.2%
Kitchen	1.60M	1.71M	2.22M	0.44M	-72.7%	1.39M	1.57M	1.96M	0.38M	-72.7%	1.16M	1.35M	1.73M	0.33M	-71.6%	0.92M	1.09M	1.44M	0.27M	-70.8%
Counter	2.90M	3.15M	2.98M	0.59M	-79.4%	2.61M	2.91M	2.69M	0.53M	-79.6%	2.23M	2.53M	2.33M	0.46M	-79.4%	1.84M	2.14M	1.93M	0.38M	-79.3%
Bicycle	3.75M	3.92M	3.66M	0.68M	-81.8%	3.32M	3.53M	3.26M	0.59M	-82.2%	2.91M	3.09M	2.82M	0.51M	-82.4%	2.36M	2.50M	2.21M	0.40M	-83.0%
Flowers	2.48M	2.64M	2.60M	0.50M	-79.8%	2.18M	2.39M	2.31M	0.44M	-79.8%	1.85M	2.04M	2.00M	0.38M	-79.0%	1.47M	1.67M	1.59M	0.30M	-79.5%
Stump	2.12M	2.32M	2.18M	0.40M	-81.3%	1.82M	2.05M	1.90M	0.34M	-81.3%	1.53M	1.76M	1.61M	0.29M	-81.0%	1.21M	1.39M	1.25M	0.22M	-82.0%
Mean	2.66M	2.92M	2.88M	0.50M	-79.9%	2.34M	2.70M	2.57M	0.44M	-80.1%	1.98M	2.31M	2.15M	0.38M	-79.6%	1.58M	1.88M	1.66M	0.30M	-79.6%

Table 16: **MipNeRF360 dataset. Active Gaussian counts (N) and percentage decreases across resolutions.** Our method consistently achieves the largest reduction in active Gaussian count (on average $\sim$80%) while maintaining comparable reconstruction fidelity across all resolutions.

Scene	Counts			3DGS			Ours			Analytic			Mip		
	N_{legacy}	N_{ours}	%↓(N)	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
bicycle	2,534,000	470,000	-81.5	26.12	0.845	0.184	25.41	0.801	0.242	26.17	0.846	0.181	26.33	0.858	0.152
bonsai	1,074,161	268,104	-75.0	29.19	0.911	0.141	28.55	0.895	0.171	29.22	0.910	0.140	29.30	0.915	0.129
counter	1,785,000	412,000	-77.0	25.67	0.826	0.199	24.93	0.781	0.308	25.70	0.824	0.195	25.79	0.834	0.171
flowers	2,257,000	498,500	-78.0	24.12	0.730	0.254	23.41	0.671	0.398	24.10	0.728	0.248	24.28	0.741	0.221
garden	4,128,014	553,848	-86.0	21.73	0.620	0.326	19.72	0.456	0.546	21.82	0.622	0.321	21.94	0.659	0.231
kitchen	1,595,961	563,437	-64.7	23.85	0.760	0.260	23.12	0.703	0.375	23.70	0.756	0.253	23.66	0.771	0.210
room	2,411,000	552,000	-77.1	25.32	0.825	0.207	24.61	0.773	0.319	25.35	0.826	0.203	25.48	0.836	0.181
stump	1,348,000	291,000	-78.4	25.10	0.830	0.194	24.43	0.782	0.290	25.15	0.832	0.188	25.22	0.844	0.168
treehill	3,285,010	563,437	-82.0	22.84	0.643	0.317	22.36	0.569	0.486	22.67	0.640	0.310	22.58	0.648	0.233
Mean	2,265,983	463,148	-77.5	**24.88**	**0.774**	**0.232**	**23.97**	**0.692**	**0.390**	**24.87**	**0.772**	**0.229**	**24.95**	**0.789**	**0.188**

Table 17: MipNeRF360 dataset. Active Gaussian counts (N) and percentage decreases across resolutions. Our method consistently achieves the largest reduction in active Gaussian count (on average $\sim$ 77%) while maintaining comparable reconstruction fidelity across all resolutions

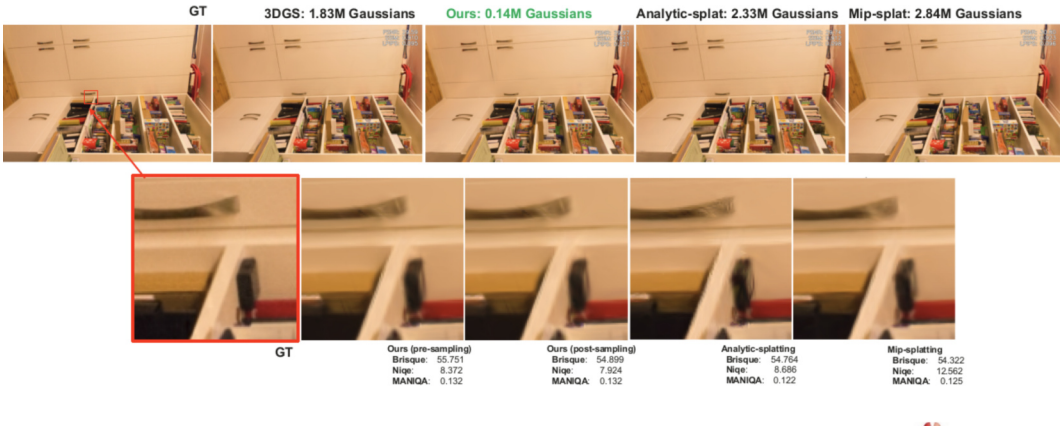

Figure 12: Qualitative results for the "*beyond*" case, where we attempt to sample Gaussians into a scene to improve quality beyond a theoretical maximum achieved by training on he highest resolution *x1*.

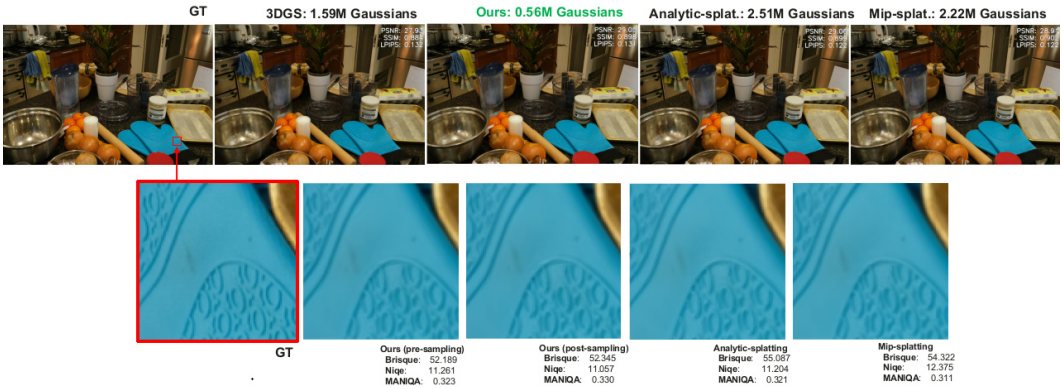

Figure 13: Qualitative results for the "*beyond*" case, where we attempt to sample Gaussians into a scene to improve quality beyond a theoretical maximum achieved by training on he highest resolution *x1*.

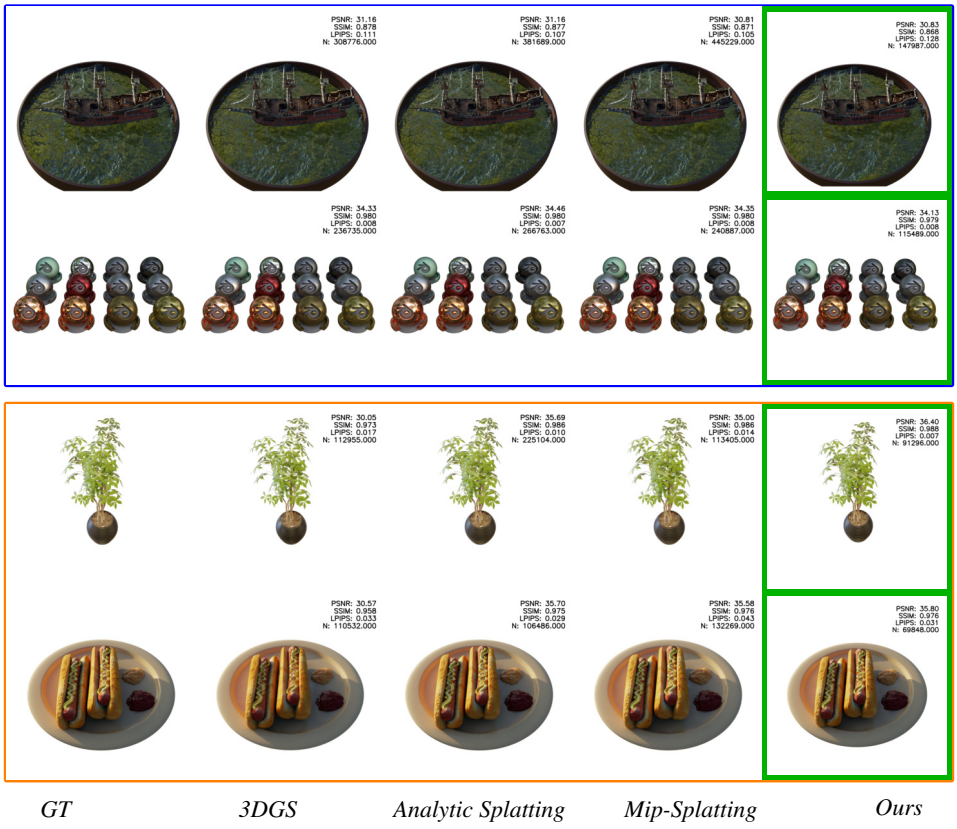

Figure 14: Qualitative comparison across selected methods (*3DGS*/*Analytic splatting*/*Mip-splatting* for scenes *Ship*, *Materials*, *Ficus*, *Hootdog top to bottom*. x1, and x2 denote the training resolutions, all images were rendered at the original resolution x1. The green border tightly highlights the sampled result ("Ours") and shows that our approach consistently generates scene with 2-4x less gaussians with comparable quality across various datasets.

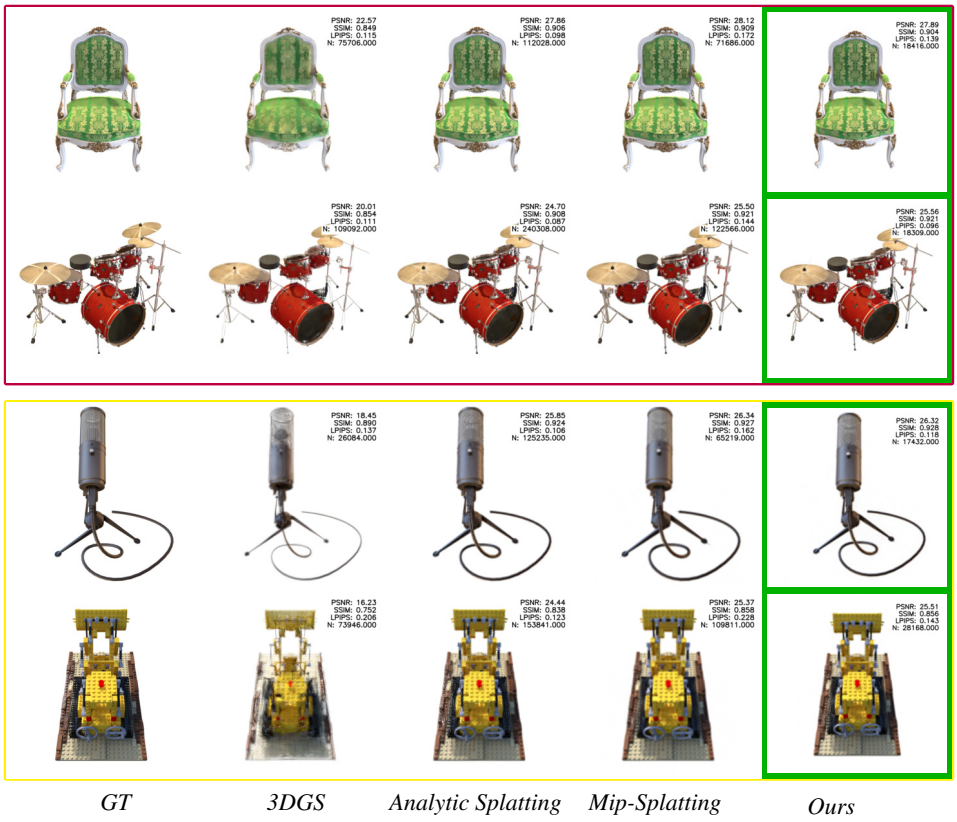

Figure 15: Qualitative comparison across selected methods (*3DGS*/*Analytic splatting*/*Mip-splatting* for scenes *Chair*, *Drums*, *Mic*, *Lego top to bottom*. x4, and x8 denote the training resolutions, all images were rendered at the original resolution x1. The green border tightly highlights the sampled result ("Ours") and shows that our approach consistently generates scene with 2-4x less gaussians with comparable quality across various datasets.