VANISHGS: DUAL-PATH MODELING OF PRIMITIVE DISAPPEARANCE FOR PRUNING 3D GAUSSIAN SPLATTING

Anonymous authors
Paper under double-blind review

ABSTRACT

While 3D Gaussian Splatting (3DGS) enables high-quality real-time novel view synthesis, its reliance on millions of Gaussian primitives leads to substantial storage and computational overhead. Existing post-pruning methods typically estimate primitive importance from rendering contribution, heuristic attributes, or generic parameter sensitivity, which do not directly measure the rendering distortion caused by removing a Gaussian, potentially leading to inaccurate importance estimation. In this paper, we propose **VanishGS**, a post-pruning framework that estimates Gaussian importance by the rendering distortion induced by its disappearance. To capture this effect, we model Gaussian disappearance through two paths, *opacity dissolution* and *geometric collapse*, characterizing its rendering impact from both appearance and geometric perspectives. We estimate the directional distortion along these pruning perturbations using the Gauss–Newton matrix and efficiently approximate the required diagonal entries through randomized projections. By measuring the cost of removing each Gaussian, VanishGS provides an importance criterion that is better aligned with the pruning objective. Extensive experiments across multiple benchmarks and pruning ratios demonstrate that VanishGS consistently preserves higher rendering quality. Moreover, VanishGS can be integrated into existing post compression pipelines, further demonstrating its generality as a plug-and-play pruning strategy. The code is included in *Supplementary Material*.

1 INTRODUCTION

3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) has emerged as a transformative technique for novel view synthesis (NVS) (Barron et al., 2022), achieving an unprecedented balance between real-time rendering speed and photorealistic fidelity (Chen et al., 2026b; 2025b). 3DGS represents a scene using an explicit set of millions of Gaussian primitives. However, due to its large parameter scale and weak structural constraints during optimization, this explicit representation comes at a substantial storage cost, often requiring hundreds of megabytes for a single scene (Duisterhof et al., 2024; Chen et al., 2025a; 2024). Such a memory footprint severely limits deployment on resource-constrained devices (Navaneet et al., 2024; Huang et al., 2025; Lei et al., 2025; Jiang et al., 2024). Therefore, reducing the number of redundant Gaussian primitives while preserving rendering quality remains a critical challenge for efficient 3DGS representation (Kong et al., 2025; Ali et al., 2024; Papantonakis et al., 2024; Chen et al., 2026a).

Pruning is a key strategy for reducing redundancy in 3DGS while preserving its real-time rendering efficiency (Hanson et al., 2025a; Lee et al., 2025). Recent approaches prune Gaussian primitives according to estimated importance. Early strategies, including opacity-based pruning in vanilla 3DGS (Kerbl et al., 2023), rely on individual primitive attributes, while EAGLES (Girish et al., 2024), LightGaussian (Fan et al., 2024), and MesonGS (Xie et al., 2024) incorporate richer rendering statistics to characterize primitive contribution. More recent methods further improve importance estimation through learned predictors (Yang et al., 2026), second-order parameter sensitivity (Hanson et al., 2025b), or analytically approximated Hessian-based importance (Chen et al., 2026d).

These methods substantially improve Gaussian importance estimation from different perspectives. While rendering participation provides a useful indication of how much a primitive contributes to the current image formation process, for pruning, a more direct quantity is the rendering distortion caused when the primitive disappears entirely. A Gaussian may participate strongly in rendering yet

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

cause little distortion when removed if its contribution is compensated by the content revealed behind it. We therefore view Gaussian importance from the consequence of disappearance: a primitive is important if its removal causes substantial rendering degradation, and redundant if it can vanish with little visual impact. This perspective further raises a key question: *how should the disappearance of a Gaussian be represented in its parameter space?*

Motivated by this view, we propose **VanishGS**, which models Gaussian disappearance through two physical paths: *opacity dissolution* to suppress its radiometric presence, and *geometric collapse* to eliminate its spatial support. By estimating the rendering distortion associated with these two paths and combining them into a unified score, VanishGS measures how costly it is for a Gaussian to vanish. As shown in Fig. 1, VanishGS consistently achieves higher PSNR across different pruning levels, with its advantage becoming increasingly pronounced as the pruning ratio increases.

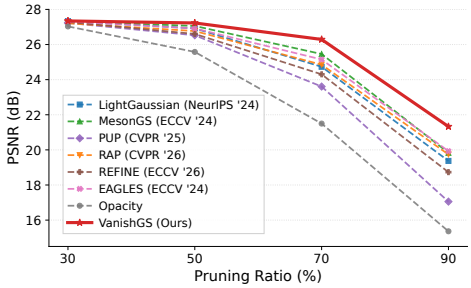


Figure 1: Rendering Quality under different Pruning Ratios on Mip-NeRF 360.

Our main contributions are summarized as follows.

(**1**) We formulate Gaussian importance from the distortion of primitive disappearance and connect pruning importance to disappearance-inducing parameter perturbations.

(**2**) We propose a dual-path pruning metric that models Gaussian disappearance through complementary *opacity dissolution* and *geometric collapse*, and estimates their associated rendering distortion using an efficient Gauss–Newton approximation.

(**3**) Extensive experiments across multiple benchmarks and pruning ratios demonstrate that VanishGS preserves superior rendering quality under matched pruning budgets compared with existing 3DGS pruning methods.

2 Related Work

Compression of 3D Gaussian Representations. Existing compression approaches mainly reduce this redundancy from three complementary perspectives. One line of work focuses on parameter and attribute compression, exploiting statistical redundancy among Gaussian attributes through scalar or vector quantization, entropy coding, compact latent representations, and reduced spherical harmonics (SH) representations (Navaneet et al., 2024; Niedermayr et al., 2024; Papantonakis et al., 2024; Chen et al., 2026c; Liu et al., 2025). These methods primarily reduce the storage cost of each primitive while largely preserving the original set of Gaussians. Another line introduces structured representations, such as anchors, hash grids, hierarchical contexts, and octrees, to organize spatially correlated primitives and enable parameter sharing or more effective entropy coding (Chen et al., 2024; Lu et al., 2024; Wang et al., 2024; Ren et al., 2025; Morgenstern et al., 2024; Liu et al., 2024). Such approaches alter the representation structure to improve coding efficiency. A third direction, Gaussian compactification, reduces redundancy by directly optimizing the number and spatial distribution of primitives. This includes improved densification strategies during training, post-training primitive pruning, and Gaussian merging or aggregation (Girish et al., 2024; Fan et al., 2024; Xie et al., 2024; Fang & Wang, 2024; Mallick et al., 2024; Wang et al., 2025; Zhang et al., 2024; Bai et al., 2026). Unlike attribute compression, reducing the number of primitives simultaneously decreases storage, memory consumption, and rasterization workload. Our work focuses on post-training Gaussian pruning, where the central problem is how to reliably estimate the importance of each primitive before removal.

Gaussian Pruning and Importance Estimation. Gaussian pruning removes primitives according to an importance score, and different methods mainly differ in how this score is calculated. A simple baseline uses opacity as the pruning criterion (Kerbl et al., 2023). Subsequent methods incorporate information more closely related to the rendering process. EAGLES (Girish et al., 2024), LightGaussian (Fan et al., 2024), and MesonGS (Xie et al., 2024) combine Gaussian attributes with visibility, transmittance, ray intersections, geometric extent, or accumulated rendering contributions to obtain more reliable importance estimates. These methods are efficient and effective, but mainly

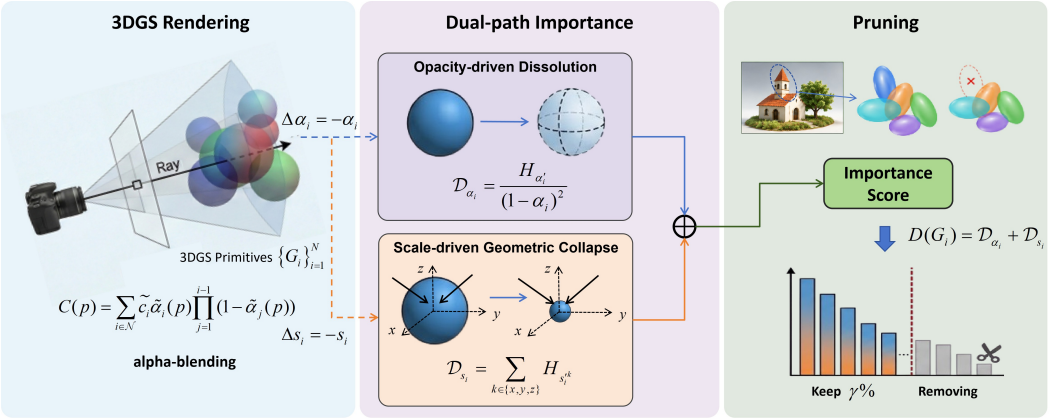


Figure 2: Overview of VanishGS, which models Gaussian disappearance through opacity dissolution and geometric collapse.

characterize the contribution of a Gaussian under the current rendering configuration. More recent approaches establish a closer connection between importance and rendering quality. PUP (Hanson et al., 2025b) evaluates the sensitivity of reconstruction error to Gaussian spatial parameters using a second-order approximation. RAP (Yang et al., 2026) instead learns to predict importance from Gaussian attributes and local neighborhood statistics, avoiding expensive multi-view rendering during inference. REFINE (Chen et al., 2026d) derives a rendering-free analytical approximation of removal-induced perceptual error using rendering-aware second-order information. Beyond direct pruning, GHAP (Wang et al., 2025) reduces the primitive population through global Gaussian aggregation rather than independent removal.

Existing methods progressively incorporate rendering contribution, parameter sensitivity, or learned priors into Gaussian importance estimation. However, these criteria do not directly measure how the disappearance of a primitive affects the quality of the rendered image. This motivates us to characterize Gaussian importance from the perspective of the rendering distortion induced by primitive disappearance.

3 Proposed Method

In this section, we first formulate Gaussian importance based on the rendering distortion caused by pruning and derive an efficient approximation for evaluating this distortion (Sec. 3.1). We then model pruning through two physically motivated parameter changes, *opacity dissolution* and *geometric collapse*, and combine their distortion estimates into a unified pruning metric (Sec. 3.2). An overview of the proposed VanishGS framework is illustrated in Fig. 2.

3.1 Rethinking 3DGS Pruning via Rendering Distortion

A 3D scene in 3DGS is represented by a set of N Gaussian primitives $\mathcal{G} = \{G_i = (\mu_i, s_i, q_i, c_i, \alpha_i)\}_{i=1}^{N}$. Each primitive is defined by its center $\mu_i \in \mathbb{R}^3$, spatial scaling $s_i \in \mathbb{R}^3$, rotation $q_i \in \mathbb{R}^4$, color SH coefficients $c_i \in \mathbb{R}^{3\times 16}$, and opacity $\alpha_i \in [0, 1]$. During rendering, these primitives are projected and depth-sorted to perform alpha-blending along each pixel ray p, computing the final color $C(p)$ via the accumulated transmittance $T_i(p)$ (Kopanas et al., 2021), formulated as

$$C(p) = \sum_{i\in\mathcal{N}} \tilde{c}_i \tilde{\alpha}_i(p) T_i(p), T_i(p) = \prod_{j=1}^{i-1} (1 - \tilde{\alpha}_j(p)), \tag{1}$$

where $\mathcal{N}$ is the depth-sorted set of primitives overlapping pixel p, $\tilde{c}_i$ is the color computed via SH, and $\tilde{\alpha}_i(p)$ is the density evaluated in pixel p multiplied by the standalone opacity α_i.

The effect of pruning on the rendered appearance can be quantified by

$$\mathcal{D}(\Delta\mathcal{G}) = \frac{1}{V}\sum_{v=1}^{V}\frac{1}{P_v}\left\|\mathbf{c}_v(\mathcal{G}+\Delta\mathcal{G}) - \mathbf{c}_v(\mathcal{G})\right\|_2^2, \tag{2}$$

where $\Delta\mathcal{G}$ denotes a parameter perturbation induced by pruning. For viewpoint v, $\mathbf{c}_v(\mathcal{G}) = \mathrm{vec}(C_v(\mathcal{G})) \in \mathbb{R}^{P_v}$ is the vectorized rendered RGB image, where $P_v = 3h_v w_v$ is the number of rendered RGB values. V denotes the number of available viewpoints of the scene.

When $\Delta\mathcal{G}$ corresponds to pruning an individual Gaussian G_i, $\mathcal{D}(\Delta\mathcal{G})$ quantifies the rendering degradation caused by its removal. This naturally provides a measure of primitive importance: a larger distortion indicates a more important Gaussian, whereas a smaller distortion suggests greater redundancy and thus a better pruning candidate.

To efficiently approximate this distortion, we locally linearize the rendering function as

$$\mathbf{c}_v(\mathcal{G}+\Delta\mathcal{G}) \approx \mathbf{c}_v(\mathcal{G}) + \mathbf{J}_v\Delta\mathcal{G},\ \mathbf{J}_v = \frac{\partial\mathbf{c}_v}{\partial\mathcal{G}}, \tag{3}$$

where $\mathbf{J}_v$ denotes the rendering Jacobian. Substituting Eq. (3) into Eq. (2) yields

$$\mathcal{D}(\Delta\mathcal{G}) \approx \Delta\mathcal{G}^T\mathbf{H}\Delta\mathcal{G}, \tag{4}$$

where

$$\mathbf{H} = \frac{1}{V}\sum_{v=1}^{V}\frac{1}{P_v}\mathbf{J}_v^T\mathbf{J}_v \tag{5}$$

is the aggregated Gauss–Newton matrix (Evtushenko, 1985) over all viewpoints.

For a 3DGS model containing millions of parameters, constructing $\mathbf{H}$ is computationally prohibitive. We therefore estimate only its diagonal entries using randomized Rademacher projections (Bekas et al., 2007). Specifically, for each viewpoint v, we sample a random vector $\mathbf{z}_v \in \{-1,+1\}^{P_v}$ satisfying $\mathbb{E}[\mathbf{z}_v\mathbf{z}_v^T] = \mathbf{I}$ and construct

$$\phi_v = \frac{1}{\sqrt{P_v}}\mathbf{z}_v^T\mathbf{c}_v(\mathcal{G}). \tag{6}$$

Its gradient with respect to the Gaussian parameters is

$$\mathbf{g}_v = \nabla_{\mathcal{G}}\phi_v = \frac{1}{\sqrt{P_v}}\mathbf{J}_v^T\mathbf{z}_v. \tag{7}$$

According to the Hutchinson identity (Hutchinson, 1989), the element-wise squared gradient provides an unbiased estimator of the diagonal Gauss–Newton terms:

$$\mathbb{E}_{\mathbf{z}}\left[\mathbf{g}_v \odot \mathbf{g}_v\right] = \frac{1}{P_v}\,\mathrm{diag}\left(\mathbf{J}_v^T\mathbf{J}_v\right). \tag{8}$$

By averaging the independent estimates over V viewpoints, we obtain

$$\hat{\mathbf{h}} = \frac{1}{V}\sum_{v=1}^{V}\mathbf{g}_v \odot \mathbf{g}_v,\ \mathbb{E}[\hat{\mathbf{h}}] = \mathrm{diag}(\mathbf{H}). \tag{9}$$

This approximation can be efficiently evaluated using standard forward and backward rendering operations, without constructing the rendering Jacobian or the full Gauss–Newton matrix. We next construct parameter perturbations that are physically aligned with Gaussian pruning, as detailed in Sec. 3.2.

3.2 Dual-path Pruning Metric

We model pruning through two physically motivated parameter changes: *opacity dissolution* and *geometric collapse*.

Opacity-driven Dissolution. This path simulates removal by making the Gaussian *completely* transparent. We keep the geometry and color unchanged and apply a perturbation $\Delta\alpha_i = -\alpha_i$. The corresponding rendering distortion is derived as:

$$\mathcal{D}_{\alpha_i} \approx H_{\alpha_i}(\Delta\alpha_i)^2 = H_{\alpha_i}{\alpha_i}^2, \tag{10}$$

where H_{α_i} denotes the diagonal Gauss–Newton term associated with opacity α_i. In 3DGS, opacity is optimized through an unconstrained latent variable $\alpha'_i \in \mathbb{R}$, with $\alpha_i = \frac{1}{1+\exp(-\alpha'_i)}$. Applying the chain rule to the rendering Jacobian gives

$$H_{\alpha'_i} = \left(\frac{\partial \alpha_i}{\partial \alpha'_i}\right)^2 H_{\alpha_i}. \tag{11}$$

Since $\frac{\partial \alpha_i}{\partial \alpha'_i} = \alpha_i(1-\alpha_i)$, we have

$$H_{\alpha_i} = \frac{H_{\alpha'_i}}{\alpha_i^2(1-\alpha_i)^2}. \tag{12}$$

Substituting Eq. (12) into Eq. (10) yields

$$\mathcal{D}_{\alpha_i} \approx \frac{H_{\alpha'_i}}{(1-\alpha_i)^2}. \tag{13}$$

This term estimates the rendering distortion induced when pruning G_i through opacity dissolution.

Scale-driven Geometric Collapse. The second path simulates Gaussian removal by collapsing its spatial support while retaining its opacity. For each principal axis $k \in \{x, y, z\}$, we apply $\Delta s_i^k = -s_i^k$. The corresponding rendering distortion is approximated as

$$\mathcal{D}_{s_i} \approx \sum_{k\in\{x,y,z\}} H_{s_i^k}(\Delta s_i^k)^2 = \sum_{k\in\{x,y,z\}} H_{s_i^k}(s_i^k)^2, \tag{14}$$

where $H_{s_i^k}$ denotes the diagonal Gauss–Newton term associated with the physical scale along axis k.

3DGS parameterizes scale using the latent log-scale variable $s_i'^k$ through $s_i^k = \exp(s_i'^k)$. Applying the chain rule to the rendering Jacobian gives

$$H_{s_i'^k} = \left(\frac{\partial s_i^k}{\partial s_i'^k}\right)^2 H_{s_i^k} = (s_i^k)^2 H_{s_i^k}. \tag{15}$$

Therefore, Eq. (14) simplifies to

$$\mathcal{D}_{s_i} \approx \sum_{k\in\{x,y,z\}} H_{s_i'^k}. \tag{16}$$

Thus, this term estimates the rendering distortion induced when pruning G_i through geometric collapse.

Unified Dual-path Primitive Importance. Opacity dissolution and geometric collapse capture two complementary ways in which a Gaussian can cease contributing to rendering. We therefore combine their distortion estimates into a unified importance score:

$$D(G_i) = \mathcal{D}_{\alpha_i} + \mathcal{D}_{s_i} = \frac{H_{\alpha'_i}}{(1-\alpha_i)^2} + \sum_{k\in\{x,y,z\}} H_{s_i'^k}. \tag{17}$$

A larger $D(G_i)$ indicates that suppressing the Gaussian through these pruning-aligned parameter changes is expected to induce greater rendering distortion. Given a target pruning ratio γ, we rank all Gaussians according to $D(G_i)$ and remove the lowest-scoring γ fraction.

4 Experiments

4.1 Experiment Settings

Datasets. We evaluated our method on the same challenging real-world scenes as 3DGS. We used all nine scenes from the Mip-NeRF 360 dataset (Barron et al., 2022), which contains five outdoor and four indoor scenes. Additionally, two outdoor scenes, `truck` and `train`, were taken from the Tanks & Temples dataset (Knapitsch et al., 2017), and two indoor scenes, `drjohnson` and

Table 1: Bjontegaard delta comparisons of VanishGS against different pruning baselines.

VanishGS (Ours) vs. Baseline	Mip-NeRF 360 (Barron et al., 2022)		Deep Blending (Hedman et al., 2018)		Tanks & Temples (Knapitsch et al., 2017)	
	BD-Rate (%) ↓	BD-PSNR (dB) ↑	BD-Rate (%) ↓	BD-PSNR (dB) ↑	BD-Rate (%) ↓	BD-PSNR (dB) ↑
vs. LightGaussian (Fan et al., 2024) [NeurIPS]	-36.200	1.410	-36.862	0.865	-40.918	1.148
vs. MesonGS (Xie et al., 2024) [ECCV]	-24.895	0.854	-27.717	0.626	-38.976	1.003
vs. PUP (Hanson et al., 2025b) [CVPR]	-48.751	2.643	-35.846	1.109	-68.900	4.061
vs. RAP (Yang et al., 2026) [CVPR]	-35.184	1.287	-25.656	0.515	-38.555	0.939
vs. REFINE (Chen et al., 2026d) [ECCV]	-45.654	2.087	-47.461	1.445	-54.572	2.018
vs. EAGLES (Girish et al., 2024) [ECCV]	-30.129	1.042	-28.157	0.645	-45.015	1.291
vs. Opacity (Kerbl et al., 2023) [TOG]	-60.617	4.393	-62.159	3.578	-66.992	3.478

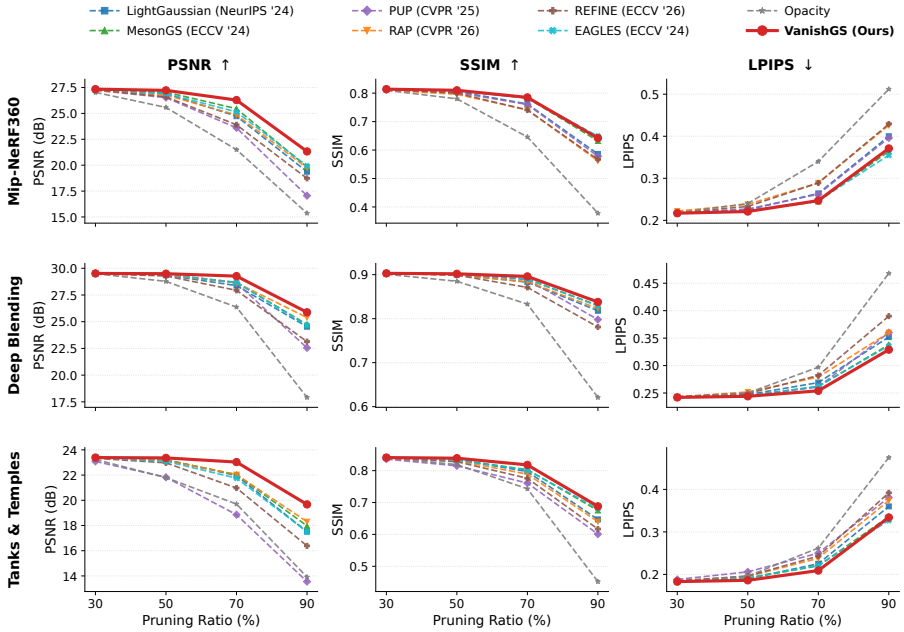


Figure 3: Rendering quality under different pruning ratios. Detailed numerical results are provided in Table A1 of the subsequent *Appendix*.

`playroom`, were taken from the Deep Blending dataset (Hedman et al., 2018). For consistency, we used the COLMAP camera poses provided by the original 3DGS (Kerbl et al., 2023).

Implementation Details. Our VanishGS is a plug-and-play, post-pruning pipeline that can be seamlessly applied to any pre-trained 3DGS model. To highlight the effectiveness of the evaluation metric itself, unless otherwise specified, our comparisons regarding pruning effects were conducted under a *zero-shot* condition. That is, after removing primitives based on the algorithm, strictly no subsequent fine-tuning rendering optimization was performed.

Baseline Methods. We compared our VanishGS with seven representative and state-of-the-art 3DGS pruning methods, including Opacity (Kerbl et al., 2023), EAGLES (Girish et al., 2024), LightGaussian (Fan et al., 2024), MesonGS (Xie et al., 2024), PUP (Hanson et al., 2025b), RAP (Yang et al., 2026) and REFINE (Chen et al., 2026d).

4.2 Experimental Results

Comparisons of Rendering Fidelity. We evaluated different methods under pruning ratios from 30% to 90% using PSNR, SSIM, and LPIPS (Zhang et al., 2018). We further reported BD-Rate and BD-PSNR to assess overall performance across the full pruning range.

Fig. 3 shows the quality–pruning curves on Mip-NeRF 360, Deep Blending, and Tanks & Temples. While different methods performed similarly under mild pruning, their performance gaps widened substantially as the pruning ratio increased. VanishGS consistently exhibited a slower PSNR drop



Figure 4: Visual comparison after 90% pruning. Zooming in for details. We also refer the reviewers to the video demo contained in the supplementary material for more visual results.

and higher SSIM and lower LPIPS at high pruning ratios, especially beyond 70%. This suggests that our pruning criterion better identifies primitives whose removal causes large rendering distortion, thereby preserving more visually important Gaussians under a constrained budget.

Table 2: Quantitative comparison of different pruning methods with fine-tuning. The best and second-best results are highlighted in red and blue, respectively.

Dataset	Method	Ratio = 0.5			Ratio = 0.7			Ratio = 0.9		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Mip-NeRF 360	LightGaussian [NeurIPS'24]	27.487	0.815	0.216	27.266	0.808	0.230	25.537	0.739	0.319
	MesonGS [ECCV'24]	27.488	0.816	0.215	27.364	0.813	0.222	26.099	0.771	0.284
	PUP [CVPR'25]	27.461	0.815	0.216	27.265	0.810	0.225	25.825	0.768	0.284
	RAP [CVPR'26]	27.459	0.813	0.222	27.236	0.803	0.244	25.898	0.738	0.335
	REFINE [ECCV'26]	27.432	0.814	0.219	27.094	0.802	0.243	25.233	0.731	0.339
	EAGLES [ECCV'24]	27.474	0.815	0.216	27.319	0.812	0.224	26.140	0.776	0.280
	Opacity [TOG'23]	27.359	0.813	0.221	26.706	0.785	0.265	25.135	0.708	0.362
	VanishGS (Ours)	27.481	0.816	0.215	27.464	0.814	0.223	26.808	0.776	0.286
Tanks & Temples	LightGaussian [NeurIPS'24]	23.637	0.845	0.181	23.526	0.835	0.198	22.433	0.777	0.279
	MesonGS [ECCV'24]	23.626	0.844	0.182	23.506	0.838	0.194	22.644	0.794	0.259
	PUP [CVPR'25]	23.467	0.839	0.187	23.023	0.825	0.205	21.464	0.770	0.275
	RAP [CVPR'26]	23.665	0.843	0.185	23.592	0.833	0.206	22.945	0.787	0.280
	REFINE [ECCV'26]	23.600	0.842	0.184	23.377	0.828	0.205	21.816	0.743	0.316
	EAGLES [ECCV'24]	23.611	0.844	0.183	23.435	0.836	0.197	22.447	0.791	0.259
	Opacity [TOG'23]	23.360	0.842	0.185	23.115	0.828	0.210	22.117	0.769	0.300
	VanishGS (Ours)	23.630	0.845	0.180	23.613	0.842	0.189	23.241	0.807	0.251
Deep Blending	LightGaussian [NeurIPS'24]	29.555	0.903	0.241	29.508	0.901	0.250	28.660	0.882	0.294
	MesonGS [ECCV'24]	29.579	0.904	0.240	29.550	0.903	0.245	28.939	0.889	0.278
	PUP [CVPR'25]	29.568	0.904	0.240	29.562	0.905	0.242	29.188	0.897	0.266
	RAP [CVPR'26]	29.566	0.903	0.244	29.561	0.902	0.254	29.198	0.891	0.289
	REFINE [ECCV'26]	29.546	0.903	0.242	29.517	0.901	0.252	27.923	0.865	0.323
	EAGLES [ECCV'24]	29.581	0.904	0.240	29.563	0.904	0.244	28.996	0.891	0.276
	Opacity [TOG'23]	29.584	0.903	0.243	29.508	0.900	0.254	28.843	0.888	0.286
	VanishGS (Ours)	29.549	0.903	0.240	29.582	0.905	0.242	29.487	0.902	0.261

Table 1 further summarizes the overall quality–model size trade-off. VanishGS achieved negative BD-Rate and positive BD-PSNR against all baselines. Averaged over the three datasets, it reduced BD-Rate by 30.53%, 33.13%, 34.43%, and 37.99% over MesonGS, RAP, EAGLES, and LightGaussian, respectively, with corresponding BD-PSNR gains of 0.83, 0.91, 0.99, and 1.14 dB. Across all

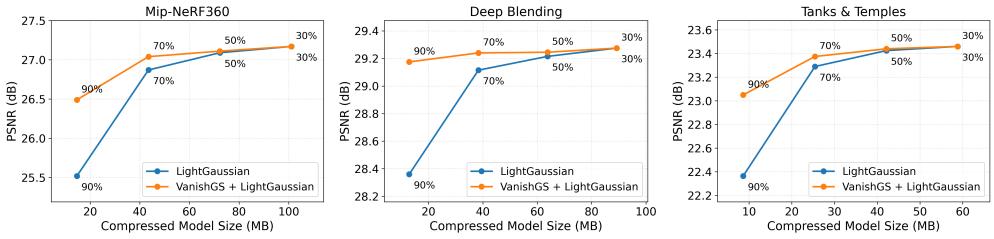
comparisons, the BD-Rate reduction reached up to 68.90%, while the BD-PSNR gain reached 4.39 dB, demonstrating a consistent quality advantage across different model sizes. Fig. 4 provides visual comparisons under 90% pruning.

Comparisons with Fine-tuning. Since post-pruning fine-tuning is often used to recover rendering quality, we further evaluated all pruning methods under the same 5k fine-tuning schedule to ensure a fair comparison. Table 2 reports the results. At a moderate pruning ratio of 0.5, most methods achieved comparable reconstruction quality, suggesting that fine-tuning can largely compensate for pruning distortion when sufficient Gaussian redundancy remains. As the pruning ratio increased, however, the impact of the initial pruning decision became more pronounced. At a ratio of 0.7, VanishGS achieved the highest PSNR on all three datasets. Under the more aggressive ratio of 0.9, it outperformed the strongest baseline by 0.668 dB, 0.289 dB, and 0.296 dB on Mip-NeRF 360, Deep Blending, and Tanks & Temples, respectively, while maintaining competitive or superior SSIM and LPIPS. These results show that VanishGS maintained consistent advantages before and after fine-tuning, particularly under aggressive pruning, demonstrating the robustness of its pruning strategy under post-pruning optimization.

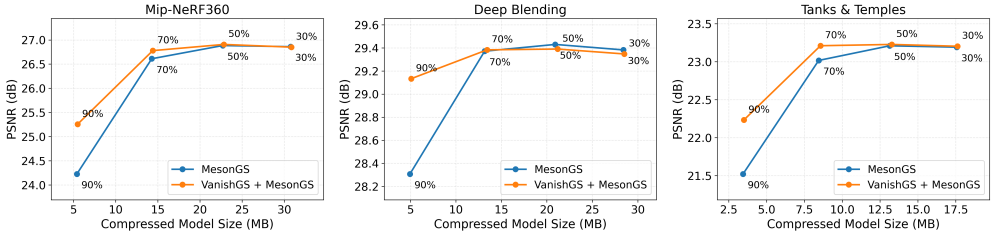
Comparisons of Running Time and Rendering Speed. We further compared the pruning time and rendering speed of different methods at a 90% pruning ratio on Mip-NeRF 360. As shown in Table 3, VanishGS required 27.03 s for pruning, which was higher than lightweight heuristic methods such as REFINE, but remained lower than PUP. Importantly, the resulting pruned model achieved 403.36 FPS, outperforming several representative pruning baselines, including LightGaussian, MesonGS, PUP, and EAGLES. These results indicate that VanishGS incurred moderate additional pruning overhead for more reliable importance estimation while still preserving high rendering efficiency after pruning.

Table 3: Pruning time and rendering speed on Mip-NeRF 360 at a 90% pruning ratio.

Method	PSNR ↑	Time (s) ↓	FPS ↑
LightGaussian	19.372	22.68	386.57
MesonGS	19.835	21.17	358.19
PUP	17.054	42.42	399.40
RAP	19.739	11.99	459.20
REFINE	18.731	0.01	410.35
EAGLES	19.930	17.53	348.42
Opacity	15.363	–	496.58
VanishGS (Ours)	21.329	27.03	403.36



(a) Integration with LightGaussian.



(b) Integration with MesonGS.

Figure 5: Rate–distortion comparisons when replacing the native pruning criterion in existing compression pipelines with VanishGS. Detailed results are provided in Tables A2 and A3 of *Appendix*.

Generality and Extensibility. To validate VanishGS as a plug-and-play pruning strategy, we integrated it into existing compression pipelines and a multi-round pruning framework, replacing only the original pruning module while keeping all other configurations and hyperparameters unchanged. As shown in Fig. 5, the rate–distortion curves used for the BD analysis shift upward after replacing the original pruning module with VanishGS. In particular, the VanishGS-integrated variants lie above the original LightGaussian and MesonGS pipelines across most pruning levels, indicating higher

rendering quality at comparable model sizes. This advantage becomes more pronounced under aggressive pruning, especially at a 90% pruning ratio, demonstrating that VanishGS can be effectively integrated into existing compression pipelines without modifying their remaining components.

We further evaluated VanishGS using the multi-round pruning-and-fine-tuning framework of PUP (Hanson et al., 2025b). Following the original protocol, we sequentially pruned 80% and 50% of the remaining Gaussians, with fine-tuning performed after each round, resulting in an overall 90% pruning ratio. Table 4 shows that replacing the original PUP pruning strategy with VanishGS improved PSNR by 0.18, 0.58, and 0.56 dB on Mip-NeRF 360, Tanks & Temples, and Deep Blending, respectively, together with consistent improvements in SSIM and LPIPS. These results demonstrate that VanishGS remains effective beyond one-shot pruning and generalizes well to iterative pruning settings.

Table 4: Comparison under two-round prune-and-fine-tuning at a 90% pruning ratio.

Dataset	Method	PSNR ↑	SSIM ↑	LPIPS ↓
Mip-NeRF 360	PUP	26.67	0.786	0.272
	VanishGS (Ours)	**26.85**	**0.788**	**0.271**
Tanks & Temples	PUP	22.72	0.801	0.244
	VanishGS (Ours)	**23.30**	**0.810**	**0.242**
Deep Blending	PUP	28.85	0.881	0.301
	VanishGS (Ours)	**29.41**	**0.899**	**0.267**

Ablation Study. To analyze the effectiveness of the proposed dual-path design, Table 5 and Fig. 6 examine the contribution of each pruning path on Mip-NeRF 360 under a 90% pruning ratio. Removing the scale-based geometric path led to the largest performance degradation, with PSNR decreasing from 21.329 to 20.338 dB, indicating that geometric collapse provides a strong cue for identifying redundant Gaussians. Using the geometric path alone (w/o opacity) already achieved competitive performance, but remained inferior to the full model across all three metrics. Incorporating the opacity path further improved PSNR by 0.094 dB and SSIM by 0.007, while reducing LPIPS from 0.375 to 0.371. The visual comparisons in Fig. 6 are also consistent with these quantitative trends, further supporting the effectiveness of combining both pruning paths. Overall, these results show that the two paths capture complementary aspects of pruning-induced degradation, with geometric collapse providing the primary signal and opacity dissolution offering additional cues for more reliable importance estimation.

Table 5: Ablation of VanishGS on Mip-NeRF 360 at a 90% pruning ratio.

Variant	PSNR↑	SSIM↑	LPIPS↓
w/o scale	20.338	0.631	0.379
w/o opacity	21.235	0.635	0.375
VanishGS (Ours)	**21.329**	**0.642**	**0.371**



Figure 6: Visual results of the ablation study.

5 CONCLUSION

In this work, we presented VanishGS, a post-pruning framework that estimates Gaussian importance by the rendering distortion induced by its disappearance. VanishGS models Gaussian removal through two disappearance paths, *opacity dissolution* and *geometric collapse*, and estimates the resulting directional distortion using a Gauss–Newton matrix. To enable efficient evaluation, we further approximate the required diagonal entries through randomized projections. Extensive experiments across multiple benchmarks and pruning ratios demonstrated that VanishGS preserved higher rendering quality, with more pronounced advantages under aggressive pruning. Moreover, VanishGS can be integrated into existing post compression pipelines, demonstrating its generality as a plug-and-play pruning strategy. Despite its effectiveness, VanishGS still requires multi-view differentiable rendering and relies on a diagonal Gauss–Newton matrix, motivating future work on more efficient view sampling and structured distortion estimation. Overall, these results highlight the importance of directly measuring the rendering distortion caused by Gaussian removal and aligning importance estimation with the actual pruning objective.

Reproducibility Statement

We provide detailed descriptions of the proposed method and pruning procedure in Secs. 3.1 and 3.2, and report the experimental settings and evaluation protocols in Sec. 4.1. Complete quantitative results and additional qualitative comparisons are provided in the Appendix. The supplementary material includes an anonymized implementation of VanishGS together with instructions for environment setup and usage, as well as an interactive video demo for additional visual comparisons. For importance estimation, only training camera views are used, and test views are not involved in the pruning process.

AI Use Statement

In this work, we used generative AI tools to assist with language polishing, grammatical correction, manuscript organization, and LaTeX formatting. We also used generative AI tools to review the presentation of experimental results, plotting code, and illustrative figures for clarity and consistency. All AI-assisted suggestions were manually reviewed and revised by the authors before being incorporated into the manuscript. The proposed method, mathematical formulation, experimental implementation, experimental results, and scientific conclusions were developed and validated by the authors. We take full responsibility for the final content of this work, including all text, claims, code, figures, and experimental results.

Ethics Statement

This work focuses on efficient pruning for 3D Gaussian Splatting and does not involve human-subject studies, collection of personally identifiable information, or the release of new sensitive data. All experiments were conducted using publicly available benchmark datasets and established evaluation protocols. VanishGS is designed as a model pruning and compression technique and does not introduce additional mechanisms for acquiring or inferring sensitive information. Nevertheless, like other 3D reconstruction and novel-view synthesis techniques, the underlying representations may inherit general concerns related to privacy, unauthorized scene reconstruction, or misuse of captured visual data. These considerations are not specific to the proposed pruning method, but should be taken into account when deploying 3D reconstruction systems in real-world applications. We are not aware of additional ethical risks introduced specifically by VanishGS.

References

Muhammad Salman Ali, Maryam Qamar, Sung Ho Bae, and Enzo Tartaglione. Trimming the fat: Efficient compression of 3D Gaussian splats through pruning. In *BMVC*, 2024.

Tian Bai, Zheng Qiu, Haojie Chen, and Ziyang Dai. Plug-and-play optimization for 3d gaussian splatting compression: Distribution regularization, probabilistic pruning and detail compensation. In *AAAI*, volume 40, pp. 2372–2380, 2026.

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, pp. 5470–5479, 2022.

Costas Bekas, Effrosyni Kokiopoulou, and Yousef Saad. An estimator for the diagonal of a matrix. *Applied Numerical Mathematics*, 57(11–12):1214–1229, 2007.

Jianing Chen, Zehao Li, Yujun Cai, Hao Jiang, Chengxuan Qian, Juyuan Kang, Shuqin Gao, Honglong Zhao, Tianlu Mao, and Yucheng Zhang. HAIF-GS: Hierarchical and induced flow-guided Gaussian splatting for dynamic scene. In *NeurIPS*, 2025a.

Jiarui Chen, Yikeng Chen, Yingshuang Zou, Ye Huang, Peng Wang, Yuan Liu, Yujing Sun, and Wenping Wang. MEGS^2: Memory-efficient gaussian splatting via spherical gaussians and unified pruning. In *ICLR*, 2026a.

Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. HAC: Hash-grid assisted context for 3D Gaussian splatting compression. In *ECCV*, pp. 422–438, 2024.

Yihang Chen, Qianyi Wu, Mengyao Li, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Fast feedforward 3D Gaussian splatting compression. In *ICLR*, 2025b.

Yihang Chen, Mengyao Li, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. PCGS: Progressive compression of 3D Gaussian splatting. In *AAAI*, volume 40, pp. 3111–3119, 2026b.

Yihang Chen, Qianyi Wu, Mengyao Li, Weiyao Lin, Junhui Hou, Mehrtash Harandi, and Jianfei Cai. Feedforward compression of static and streamable 3D Gaussian splatting. *IEEE TCSVT*, 2026c.

Zhang Chen, Shuai Wan, Mengting Yu, Fuzheng Yang, and Junhui Hou. REFINE: Super-efficient 3d gaussian splatting pruning via rendering-free primitive importance. In *ECCV*, volume 17027, pp. 131–147, 2026d.

Bardienus P. Duisterhof, Mandi Zhao, Yunchao Yao, Jia-Wei Liu, Jenny Seidenschwarz, Mike Zheng Shou, Deva Ramanan, Shuran Song, Stan Birchfield, Bowen Wen, and Jeffrey Ichnowski. DeformGS: Scene flow in highly deformable scenes for deformable object manipulation. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2024.

Yurij G Evtushenko. *Numerical optimization techniques*. Springer-Verlag, 1985.

Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. LightGaussian: Unbounded 3D Gaussian compression with 15x reduction and 200+ FPS. In *NeurIPS*, volume 37, pp. 140138–140158, 2024.

Guangchi Fang and Bing Wang. Mini-splatting: Representing scenes with a constrained number of Gaussians. In *ECCV*, pp. 165–181, 2024.

Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. Eagles: Efficient accelerated 3D Gaussians with lightweight encodings. In *ECCV*, pp. 54–71, 2024.

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 *CVPR*, 2025a.

Alex Hanson, Allen Tu, Vasu Singla, Mayuka Jayawardhana, Matthias Zwicker, and Tom Goldstein. PUP 3D-GS: Principled uncertainty pruning for 3D Gaussian splatting. In *CVPR*, pp. 5949–5958, 2025b.

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

He Huang, Wenjie Huang, Qi Yang, Yiling Xu, and Zhu Li. A hierarchical compression technique for 3D Gaussian splatting compression. In *ICASSP*, pp. 1–5, 2025.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Yuheng Jiang, Zhehao Shen, Penghao Wang, Zhuo Su, Yu Hong, Yingliang Zhang, Jingyi Yu, and Lan Xu. HiFi4G: High-fidelity human performance rendering via compact Gaussian splatting. In *CVPR*, pp. 19734–19745, 2024.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM TOG*, 42(4):139:1–139:14, 2023.

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

Hanyang Kong, Xingyi Yang, and Xinchao Wang. Efficient Gaussian splatting for monocular dynamic scene rendering via sparse time-variant attribute modeling. In *AAAI*, volume 39, pp. 4374–4382, 2025.

Georgios Kopanas, Julien Philip, Thomas Leimkühler, and George Drettakis. Point-based neural rendering with per-view optimization. *Comput. Graph. Forum*, 40(4):29–43, 2021.

Joo Chan Lee, Jong Hwan Ko, and Eunbyung Park. Optimized minimal 3D Gaussian splatting. In *NeurIPS*, 2025.

Jiahui Lei, Yijia Weng, Adam W Harley, Leonidas Guibas, and Kostas Daniilidis. MoSca: Dynamic Gaussian fusion from casual videos via 4D motion scaffolds. In *CVPR*, pp. 6165–6177, 2025.

Lei Liu, Zhenghao Chen, and Dong Xu. 3D Gaussian splatting data compression with mixture of priors. In *ACM MM*, pp. 8341–8350, 2025.

Xiangrui Liu, Xinju Wu, Pingping Zhang, Shiqi Wang, Zhu Li, and Sam Kwong. CompGS: Efficient 3D scene representation via compressed Gaussian splatting. In *ACM MM*, pp. 2936–2944, 2024.

Tao Lu, Mulin Yu, Linning Xu, Yuanbo Xiangli, Limin Wang, Dahua Lin, and Bo Dai. Scaffold-GS: Structured 3D Gaussians for view-adaptive rendering. In *CVPR*, pp. 20654–20664, 2024.

Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Francisco Vicente Carrasco, Markus Steinberger, and Fernando De La Torre. Taming 3DGS: High-quality radiance fields with limited resources. In *SIGGRAPH Asia 2024*, pp. 1–11, 2024.

Wieland Morgenstern, Florian Barthel, Anna Hilsmann, and Peter Eisert. Compact 3D scene representation via self-organizing Gaussian grids. In *ECCV*, pp. 18–34, 2024.

KL Navaneet, Kossar Pourahmadi Meibodi, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. CompGS: Smaller and faster Gaussian splatting with vector quantization. In *ECCV*, pp. 330–349, 2024.

Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. Compressed 3D Gaussian splatting for accelerated novel view synthesis. In *CVPR*, pp. 10349–10358, 2024.

Panagiotis Papantonakis, Georgios Kopanas, Bernhard Kerbl, Alexandre Lanvin, and George Drettakis. Reducing the memory footprint of 3D Gaussian splatting. In *ACM SIGGRAPH*, 2024.

Kerui Ren, Lihan Jiang, Tao Lu, Mulin Yu, Linning Xu, Zhangkai Ni, and Bo Dai. Octree-GS: Towards consistent real-time rendering with LOD-structured 3D Gaussians. *IEEE TPAMI*, 2025.

Tao Wang, Mengyu Li, Geduo Zeng, Cheng Meng, and Qiong Zhang. Gaussian herding across pens: An optimal transport perspective on global gaussian reduction for 3DGS. In *NeurIPS*, 2025.

Yufei Wang, Zhihao Li, Lanqing Guo, Wenhan Yang, Alex C. Kot, and Bihan Wen. ContextGS: Compact 3D Gaussian splatting with anchor level context model. In *NeurIPS*, volume 37, 2024.

Shuzhao Xie, Weixiang Zhang, Chen Tang, Yunpeng Bai, Rongwei Lu, Shijia Ge, and Zhi Wang. MesonGS: Post-training compression of 3D Gaussians via efficient attribute transformation. In *ECCV*, pp. 434–452, 2024.

Kaifa Yang, Qi Yang, Yiling Xu, and Zhu Li. Rap: Fast feedforward rendering-free attribute-guided primitive importance score prediction for efficient 3d gaussian splatting processing. In *CVPR*, pp. 33323–33332, 2026.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pp. 586–595, 2018.

Zhaoliang Zhang, Tianchen Song, Yongjae Lee, Li Yang, Cheng Peng, Rama Chellappa, and Deliang Fan. Lp-3dgs: learning to prune 3d gaussian splatting. In *NeurIPS*, volume 37, pp. 122434–122457, 2024.

A APPENDIX

Detailed Zero-shot Results. Table A1 reports the complete quantitative results of different pruning methods without fine-tuning. These results provide the detailed numerical values corresponding to the pruning-quality curves in Fig. 3 and the BD-Rate/BD-PSNR comparisons reported in Table 1 of the main paper.

Table A1: Quantitative comparison of different pruning methods without post-pruning fine-tuning. The best and second-best results are highlighted in red and blue, respectively.

Dataset	Method	Ratio = 0.3			Ratio = 0.5			Ratio = 0.7			Ratio = 0.9		
		PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Mip-NeRF 360	LightGaussian	27.333	0.814	0.217	26.919	0.806	0.225	24.733	0.762	0.264	19.372	0.586	0.400
	MesonGS	27.327	0.814	0.217	27.057	0.809	0.222	25.458	0.783	0.245	19.835	0.632	0.364
	PUP	27.231	0.812	0.218	26.527	0.802	0.227	23.607	0.760	0.262	17.054	0.578	0.396
	RAP	27.211	0.810	0.222	26.767	0.796	0.238	24.851	0.741	0.289	19.739	0.562	0.426
	REFINE	27.286	0.813	0.218	26.603	0.800	0.233	24.341	0.741	0.289	18.731	0.567	0.429
	EAGLES	27.314	0.814	0.217	26.916	0.809	0.223	25.148	0.783	0.247	19.930	0.649	0.355
	Opacity	27.031	0.810	0.218	25.579	0.780	0.240	21.498	0.646	0.340	15.363	0.378	0.512
	VanishGS (Ours)	27.344	0.814	0.217	27.228	0.810	0.221	26.290	0.785	0.247	21.329	0.642	0.371
Tanks & Temples	LightGaussian	23.383	0.841	0.183	23.236	0.835	0.190	21.956	0.797	0.225	17.514	0.647	0.360
	MesonGS	23.379	0.841	0.184	23.181	0.834	0.191	22.042	0.803	0.220	17.975	0.675	0.336
	PUP	23.083	0.836	0.188	21.826	0.815	0.206	18.858	0.761	0.250	13.549	0.601	0.381
	RAP	23.375	0.841	0.184	23.176	0.831	0.195	22.036	0.789	0.238	18.300	0.640	0.374
	REFINE	23.329	0.839	0.185	22.970	0.828	0.196	20.977	0.776	0.243	16.384	0.617	0.392
	EAGLES	23.375	0.841	0.184	23.093	0.833	0.192	21.754	0.802	0.221	17.503	0.681	0.327
	Opacity	23.238	0.840	0.183	21.801	0.819	0.196	19.701	0.743	0.262	13.912	0.452	0.475
	VanishGS (Ours)	23.390	0.841	0.183	23.363	0.839	0.186	23.032	0.818	0.209	19.673	0.688	0.334
Deep Blending	LightGaussian	29.509	0.903	0.243	29.360	0.900	0.247	28.379	0.883	0.269	24.533	0.818	0.352
	MesonGS	29.526	0.903	0.242	29.432	0.902	0.245	28.676	0.890	0.262	24.742	0.830	0.338
	PUP	29.512	0.903	0.242	29.401	0.901	0.245	28.635	0.889	0.261	22.550	0.798	0.360
	RAP	29.511	0.903	0.243	29.368	0.899	0.252	28.650	0.884	0.279	25.387	0.823	0.360
	REFINE	29.492	0.903	0.243	29.278	0.899	0.249	27.936	0.871	0.282	23.136	0.781	0.390
	EAGLES	29.524	0.903	0.242	29.439	0.902	0.245	28.671	0.890	0.262	24.664	0.829	0.336
	Opacity	29.500	0.901	0.243	28.775	0.885	0.250	26.370	0.833	0.297	17.914	0.621	0.468
	VanishGS (Ours)	29.518	0.903	0.242	29.496	0.902	0.244	29.267	0.896	0.254	25.880	0.838	0.329

Table A2: Detailed results of integrating VanishGS into the LightGaussian compression pipeline.

Prune	Pruning Metric	Mip-NeRF 360				Deep Blending				Tanks & Temples			
		Size (MB)↓	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓	PSNR↑	SSIM↑	LPIPS↓
30%	LightGaussian	101.08	27.167	0.807	0.229	89.23	29.274	0.902	0.250	58.81	23.464	0.842	0.186
	+ VanishGS (Ours)	101.05	27.166	0.807	0.229	89.22	29.273	0.902	0.249	58.80	23.464	0.842	0.186
50%	LightGaussian	72.31	27.095	0.805	0.234	63.82	29.211	0.901	0.252	42.15	23.423	0.839	0.190
	+ VanishGS (Ours)	72.28	27.115	0.806	0.232	63.81	29.240	0.901	0.251	42.14	23.440	0.840	0.189
70%	LightGaussian	43.50	26.868	0.797	0.248	38.37	29.114	0.897	0.262	25.43	23.288	0.829	0.207
	+ VanishGS (Ours)	43.49	27.039	0.803	0.240	38.37	29.240	0.901	0.253	25.41	23.378	0.836	0.198
90%	LightGaussian	14.66	25.521	0.742	0.324	12.91	28.359	0.879	0.303	8.63	22.368	0.778	0.281
	+ VanishGS (Ours)	14.66	26.493	0.769	0.297	12.91	29.176	0.898	0.270	8.63	23.046	0.805	0.253

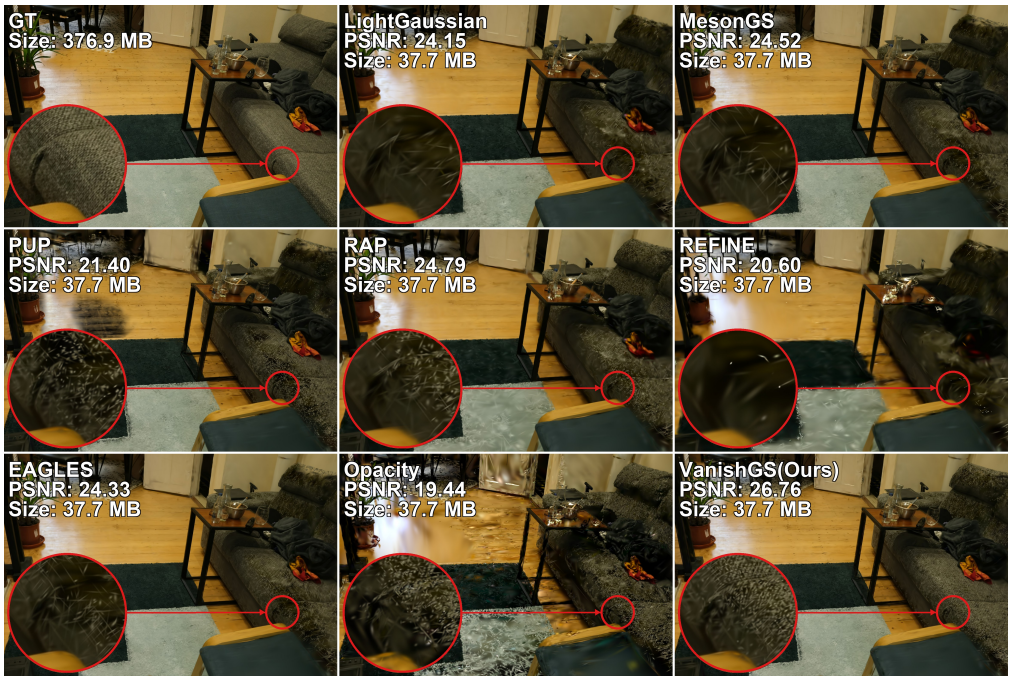
Table A3: Detailed results of integrating VanishGS into the MesonGS compression pipeline.

Prune	Pruning Metric	Mip-NeRF 360				Deep Blending				Tanks & Temples			
		Size (MB)↓	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓	PSNR↑	SSIM↑	LPIPS↓	Size (MB)↓	PSNR↑	SSIM↑	LPIPS↓
30%	MesonGS	30.74	26.861	0.795	0.246	28.41	29.383	0.899	0.256	17.56	23.192	0.830	0.201
	+ VanishGS (Ours)	30.83	26.851	0.795	0.246	28.52	29.348	0.899	0.256	17.62	23.204	0.831	0.201
50%	MesonGS	22.70	26.884	0.796	0.245	20.92	29.431	0.900	0.256	13.13	23.210	0.829	0.203
	+ VanishGS (Ours)	22.83	26.907	0.796	0.244	21.20	29.391	0.898	0.256	13.27	23.228	0.831	0.199
70%	MesonGS	14.28	26.611	0.790	0.254	13.16	29.372	0.897	0.263	8.45	23.017	0.818	0.218
	+ VanishGS (Ours)	14.40	26.779	0.791	0.254	13.43	29.384	0.899	0.258	8.56	23.211	0.825	0.209
90%	MesonGS	5.41	24.228	0.719	0.332	4.95	28.308	0.877	0.305	3.45	21.523	0.753	0.294
	+ VanishGS (Ours)	5.48	25.255	0.733	0.326	5.09	29.134	0.891	0.285	3.50	22.234	0.767	0.284

Integration with Existing Compression Pipelines. Tables A2 and A3 report the detailed numerical results corresponding to the rate–distortion comparisons with LightGaussian and MesonGS shown

in Figs. 5a and 5b of the main paper, respectively. For both pipelines, we replaced only the original primitive pruning criterion with VanishGS while keeping all subsequent recovery and compression components unchanged. All results were averaged over the scenes in each dataset.

Additional Visual Comparisons. In addition to the representative visual comparison presented in Fig. 4 of the main paper, Fig. A1 presents additional qualitative results across different scenes under a 90% pruning ratio. These results complement the visual examples in the main paper with additional scene-level and close-up comparisons.



(a) room



(b) truck.

Figure A1: Visual comparison after 90% pruning. Zooming in for details.