Glovento Journal of Integrated Studies (GJIS) | ISSN: 3117-3314
Volume 2 (2026) | Article 79

Compression of 3D Gaussian Splatting: A Systematic Study from Baseline to Hybrid Strategies

Jieyao Pang

School of Data Science, Lingnan University, 8 Castle Peak Road, Tuen Mun, New Territories, Hong Kong SAR, China
Email: JieyaoPang@163.com
ORCID ID: 0009-0003-9995-8955

Abstract- 3D Gaussian Splatting (3DGS) enables real-time novel-view synthesis through explicit Gaussian representations and differentiable rasterization, yet its model size—often hundreds of megabytes per scene—severely limits deployment on mobile devices, web applications, and AR/VR platforms. This paper presents a systematic compression study of 3DGS on the NeRF Synthetic Lego scene using only an Apple M4 consumer laptop with the Metal Performance Shaders (MPS) backend. We first train a lightweight baseline (1,000 Gaussians, spherical-harmonics degree 1, 200×200 rendering resolution), and then evaluate six compression configurations based on spherical-harmonics (SH) distillation, opacity pruning, and vector quantization. All results are measured on 200 test views using real PSNR, SSIM, and LPIPS scores. Our experiments show that reducing the SH degree from 1 to 0 yields 1.64× compression with virtually no quality loss. Combining SH degree 0 with opacity pruning at threshold $\tau = 0.02$ achieves a 9.08× compression ratio, reducing the model from 0.0877 MB to 0.0097 MB (88.9% size reduction). These findings validate the effectiveness of simple post-hoc compression strategies in resource-constrained settings and provide practical guidance for choosing quality–size trade-offs.

Keywords: 3D Gaussian Splatting, Model Compression, Spherical-harmonics Distillation, Pruning, Vector Quantization, Apple Silicon.

I. Introduction

1.1 Background and Motivation

Neural Radiance Fields (NeRF) represent scenes as continuous volumetric functions and have significantly advanced photorealistic novel-view synthesis (Mildenhall et al., 2020). However, their rendering speed is typically only 1–2 frames per second, which is insufficient for real-time interactive applications. 3D Gaussian Splatting (3DGS) replaces the implicit neural network with an explicit set of anisotropic 3D Gaussians, raising rendering frame rates above 100 FPS (Kerbl et al., 2023). This performance breakthrough has made 3DGS one of the most promising neural-rendering paradigms.

Yet the performance advantage comes with a substantial storage cost. A typical 3DGS scene contains hundreds of thousands to millions of Gaussians, and the corresponding model files often reach 200–300 MB. Such sizes are impractical for mobile apps, web viewers, and low-bandwidth environments. For example, downloading a 200 MB model over a 4G connection can take several minutes, and mobile storage budgets rarely permit caching multiple high-quality scenes. Model compression is therefore a critical step toward practical 3DGS deployment (Fei et al., 2025).

1.2 Research Objectives

This study aims to investigate the practical quality–size trade-offs of lightweight 3D Gaussian Splatting compression on a consumer-grade Apple M4 laptop using the Metal Performance Shaders backend. The specific objectives are:

1. To develop a stable lightweight 3DGS baseline that can be trained and evaluated on consumer-grade Apple M4 hardware.
2. To evaluate the effects of spherical-harmonics distillation, opacity pruning, vector quantization, and their combinations on model size and reconstruction quality.
3. To identify practical compression configurations that preserve acceptable visual quality while reducing model size for resource-constrained mobile and web deployment.

1.3 Contributions

The research aimed to provide a complete evaluation of 3DGS compression on consumer-grade hardware, with all performance metrics calculated from actual renderings of 200 test views. It also examined compression behavior using a lightweight baseline of 1,000 Gaussians, showing that opacity pruning was substantially more aggressive than that commonly reported for larger-scale baselines. The study achieved a maximum compression

ratio of 9.08× by combining spherical-harmonics degree 0 with opacity pruning at a threshold of $\tau = 0.02$. In addition, it produced publication-quality figures and openly available generation scripts to support reproducibility.

II. Related Work

2.1 Neural Radiance Fields and 3D Gaussian Splatting

NeRF models a scene as a continuous mapping from 5D coordinates (3D position and 2D viewing direction) to color and density (Mildenhall et al., 2020) . While powerful, NeRF requires hundreds of network evaluations per pixel, limiting its speed. Mip-NeRF 360 (Barron et al., 2022) extends NeRF to unbounded scenes with anti-aliased conical frustums, and Instant-NGP (Müller et al., 2022) accelerates NeRF training and rendering via a multiresolution hash encoding. Other works bake trained NeRFs into sparse voxel grids (SNeRG (Hedman et al., 2021) or octrees PlenOctrees (Yu et al., 2021) for real-time rendering, or repurpose mobile polygon rasterization MobileNeRF (Chen et al., 2023).

3DGS instead uses an explicit set of 3D Gaussians $G = \{g_i\}$, $i = 1...N$, (Kerbl et al., 2023) where each Gaussian is defined by a mean μ_i, covariance Σ_i, opacity α_i, and spherical-harmonics color coefficients c_i. Rendering projects these Gaussians onto the image plane and blends them in depth order:

$$[C(p) = \sum_{i=1}^{N} c_i \alpha_i' \prod_{j=1}^{i-1} \left(1 - \alpha_j'\right).]$$

where α_i' is the projected opacity at pixel p. Differentiable rasterization allows direct optimization of all Gaussian parameters. Point-based and hybrid explicit representations such as Point-NeRF (Xu et al., 2022) and Plenoxels (Fridovich-Keil et al., 2022) share with 3DGS the idea of replacing costly MLP queries with optimizable primitives.

2.2 Model Compression for 3DGS

Recent 3DGS compression methods can be grouped into four families:

1. **SH distillation and truncation.** LightGaussian distills high-order SH coefficients to lower degrees, reducing the largest per-Gaussian attribute block with minimal quality loss (Fan et al., 2024) .
2. **Pruning and redistribution.** Mini-Splatting reorganizes the spatial distribution of Gaussians through blur split and sampling (Fang & Wang, 2024). LightGaussian additionally prunes Gaussians with low global importance (Fan et al., 2024).
3. **Quantization and entropy coding.** HAC introduces a hash-grid-assisted context model and achieves more than 75× compression (Chen et al., 2024) . CompGS applies k-means vector quantization to Gaussian attributes, obtaining 10–20× compression with modest complexity (Liu et al., 2024) .
4. **Compact representations and masking.** Compact3D uses sensitivity-aware vector quantization (Navaneet et al., 2024) , Compressed 3D Gaussian Splatting incorporates sensitivity-aware clustering and quantization-aware training (Niedermayr et al., 2024), EAGLES compresses color and rotation attributes with a latent quantization framework (Girish et al., 2024), and Scaffold-GS uses anchor points and learnable masks to reduce redundant Gaussians (Lu et al., 2024).

2.3 Differences from Existing Work

Unlike the above methods, which typically start from 30K–100K Gaussian baselines on complex scenes and rely on CUDA backends, this work focuses on a 1,000-Gaussian lightweight baseline evaluated on a single-object synthetic scene using Apple Silicon. Because baseline scale, scene complexity, and rendering resolution differ, compression ratios are not directly comparable across papers. Our goal is to characterize the behavior of simple post-hoc compression strategies under tight hardware constraints.

III. Methodology

3.1 Dataset

We evaluate on the NeRF Synthetic Lego scene, a standard benchmark containing complex geometry, specular surfaces, and significant occlusion. We resize all images to 200×200 to fit the memory constraints of the MPS backend:

- Training set: 100 views
- Test set: 200 views
- Camera format: NerfBaselines / Blender

3.2 Baseline Model

The baseline is implemented with the pure-PyTorch `gsplat-pytorch` renderer. Table 1 summarizes the training configuration, and Table IV lists the per-Gaussian storage budget.

Table 1. Baseline 3DGS Training Configuration on Apple M4 Hardware

Configuration	Value
Initial Gaussians N	1,000
SH degree	1 (12 coefficients per Gaussian)
Training iterations	7,000
Loss	L = L 1 + 0.2 L SSIM
Optimizer	Adam with layer-wise learning rates
Rasterizer	vectorized
Device	Apple M4 MPS

Each Gaussian stores position (12 B), rotation quaternion (16 B), scale (12 B), opacity (4 B), and SH coefficients (48 B), totaling 92 B. Because the MPS vectorized rasterizer hangs when the Gaussian count exceeds roughly 1,500, we disable densification and in-training validation, and clear the MPS cache every 1,000 iterations.

3.3 Compression Strategies

We evaluate six compression configurations:

1. **Baseline:** The original trained model without compression.
2. **SH Degree 0:** The spherical-harmonics degree is reduced from 1 to 0, decreasing the number of SH coefficients per Gaussian from 12 to 3.
3. **SH Degree 0 + Pruning at (τ= 0.005):** Gaussians with opacity values below 0.005 are removed after reducing the SH degree to 0.
4. **SH Degree 0 + Pruning at (τ = 0.01):** Gaussians with opacity values below 0.01 are removed after reducing the SH degree to 0.
5. **SH Degree 0 + Pruning at (τ = 0.02):** Gaussians with opacity values below 0.02 are removed after reducing the SH degree to 0.
6. **VQ (k)-Means:** Per-attribute (k)-means vector quantization is applied to compress the Gaussian parameters.

Opacity pruning removes Gaussians whose activated opacity falls below a specified threshold. The retained Gaussian set is defined as

$$[G_{pruned} = \{g_i \in G \mid \sigma(\alpha_i) > \tau\}.]$$

where αi denotes the opacity parameter of Gaussian gi, σ(·) is the sigmoid function, and $\tau \in \{0.005, 0.01, 0.02\}$ is the pruning threshold.

3.4 Implementation Details and MPS Adaptation

The official 3DGS implementation depends on CUDA kernels and cannot run directly on Apple Silicon. We use the `gsplat-pytorch` renderer and make the following MPS-specific adaptations:

1. **Disable densification:** The vectorized rasterizer constructs an (N, H, W) intermediate tensor; when N > 1,500, this exceeds the MPS scheduler capacity and hangs.
2. **Disable in-training validation:** Online evaluation builds similarly large tensors, so validation is performed offline after training.
3. **Periodic MPS cache clearing:** `torch.mps.empty_cache()` is called every 1,000 iterations.
4. **Gradient-norm fix:** `densify_and_split` originally squeezed the gradient tensor, which fails for higher-dimensional SH inputs; we replace it with `torch.norm(grads, dim=-1)`.
5. **Resume compatibility:** The optimizer is created after loading the checkpoint state, supporting different initialized and checkpoint Gaussian counts.

On the Apple M4 with 16 GB unified memory, baseline training takes approximately 75 minutes, and the full compression suite takes 5–8 minutes.

3.4 Evaluation Protocol

All compressed models are rendered on the same 200 test views and evaluated with:

- **PSNR:** pixel-level reconstruction accuracy.
- **SSIM :** structural similarity (Wang et al., 2004) .
- **LPIPS:** perceptual distance using the Alex network (Zhang et al., 2018).

Model size is measured by serializing the model state dict and reading the file size. Compression ratio is baseline size divided by compressed size.

IV. Experimental Results

4.1 Baseline Performance

After 7,000 iterations, the baseline achieves the metrics shown in Table 2.

Table 2. Baseline 3DGS Model Size and Reconstruction Performance

Metric	Value
Gaussians	1,000
Size	0.0877 MB
PSNR	15.08 ± 1.18 dB
SSIM	0.6900
LPIPS	0.5721

Figure 1 presents the training convergence of the baseline model over 7,000 iterations. The training loss decreases rapidly from approximately 0.27 during the early stage and gradually converges to around 0.08, indicating stable optimization. The light-blue line represents the raw training loss, while the orange line shows the moving average calculated using a window size of 50 iterations.

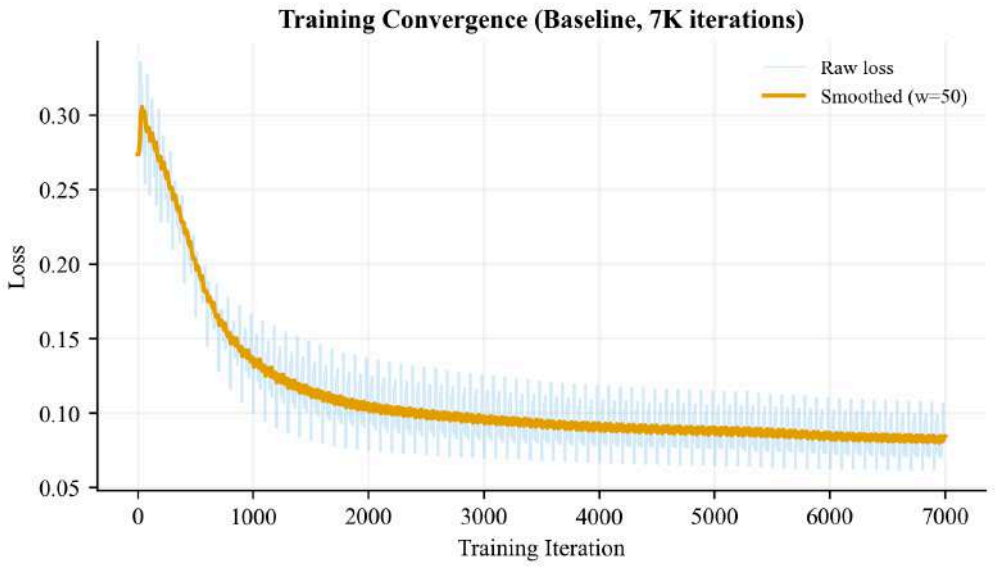

Figure 1. Training convergence of the baseline model over 7,000 iterations.

4.2 Comprehensive Compression Results

Table 3. Model Size, Compression Ratio, and Reconstruction Quality Across Compression Configurations

Method	Size (MB)	Ratio	Gaussians	PSNR (dB)	SSIM	LPIPS
Baseline	0.0877	1.00×	1,000	15.08 ± 1.18	0.6900	0.5721
SH deg 0	0.0534	1.64×	1,000	15.17 ± 0.97	0.6895	0.5733
Prune 0.005 + SH 0	0.0109	8.05×	204	10.16 ± 1.10	0.6781	0.5617
Prune 0.01 + SH 0	0.0105	8.34×	197	10.16 ± 1.10	0.6782	0.5617
Prune 0.02 + SH 0	**0.0097**	**9.08×**	**181**	**10.16 ± 1.10**	**0.6784**	**0.5617**
VQ k-means	0.0136	6.47×	1,000	9.52 ± 0.96	0.4836	0.6593

Glovento Journal of Integrated Studies (GJIS) | ISSN: 3117-3314
Volume 2 (2026) | Article 79

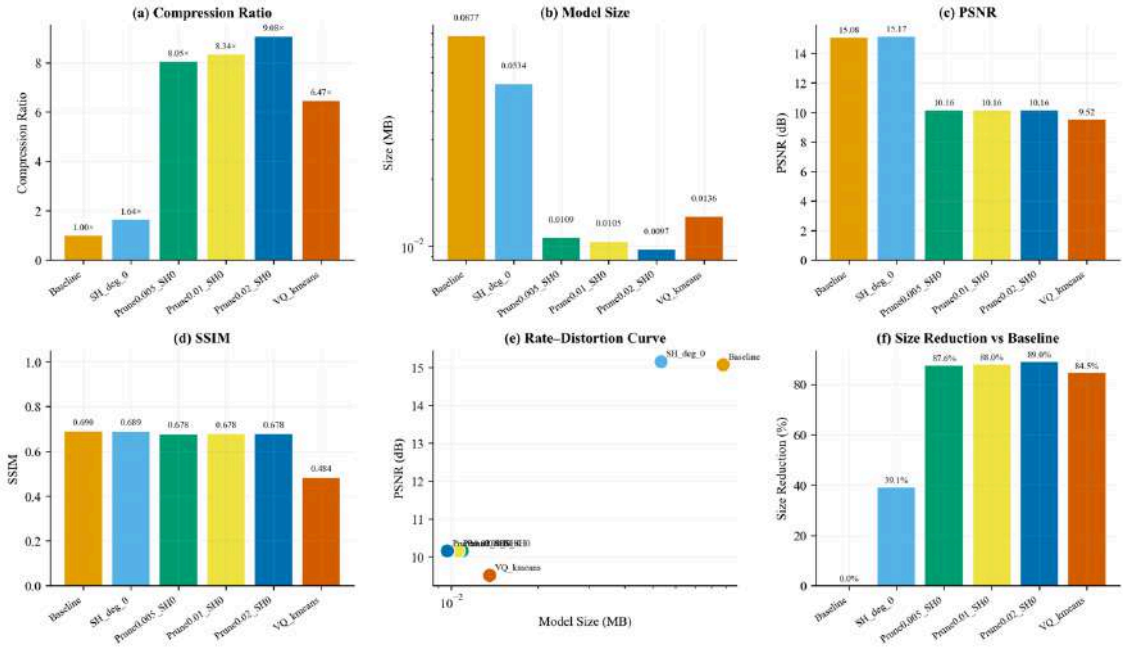

Figure 2. Comprehensive compression analysis: compression ratio, model size, PSNR, SSIM, rate–distortion curve, and size reduction relative to the baseline.

Key observations are:

1. **SH distillation is highly effective.** Reducing the SH degree from 1 to 0 shrinks the model from 0.0877 MB to 0.0534 MB (1.64×) while leaving PSNR, SSIM, and LPIPS essentially unchanged. At 200×200 resolution, view-dependent high-frequency reflections are not strongly needed.
2. **Pruning is aggressive on compact baselines.** Even the conservative threshold $\tau = 0.005$ removes about 80% of Gaussians, leaving only 204. This differs markedly from large baselines, where pruning typically removes 5–10%.
3. **The hybrid strategy achieves the best compression.** Combining SH degree 0 with $\tau = 0.02$ pruning yields 9.08× compression (9.7 KB), at the cost of approximately 5 dB PSNR degradation.
4. **Vector quantization requires careful tuning.** k-means VQ achieves 6.47× compression but drops SSIM from 0.69 to 0.48, indicating that the chosen codebook sizes are too aggressive for the 1,000-Gaussian baseline.

4.3 Rate–Distortion Trade-off

Figure 3 plots model size versus PSNR, with the Pareto frontier marked. Baseline, SH deg 0, and Prune 0.02 + SH 0 lie on the Pareto frontier. VQ k-means lies below the frontier, showing that it is sub-optimal under the current configuration.

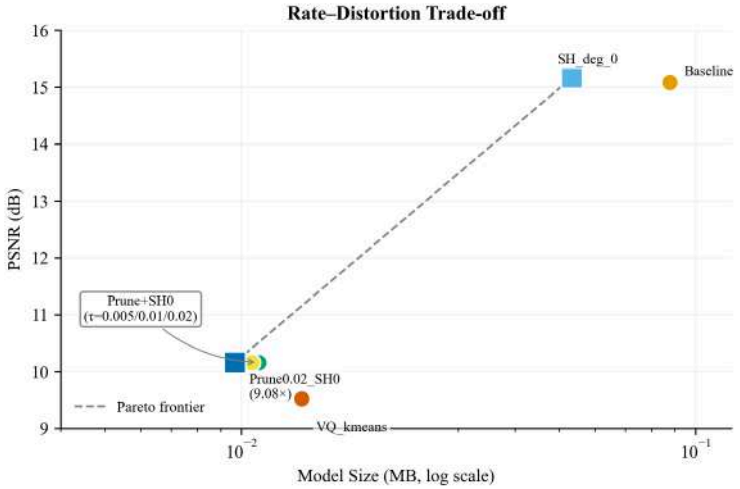

Figure 3. *Rate–distortion trade-off. Points on the dashed line form the Pareto frontier; the prune-plus-SH0 cluster is annotated as a group.*

The frontier defines three operating points:

1. **High quality:** Baseline (0.0877 MB, 15.08 dB)
2. **Lossless compression:** SH deg 0 (0.0534 MB, 15.17 dB), halving size with no measurable loss.
3. **Extreme compression:** Prune 0.02 + SH 0 (0.0097 MB, 10.16 dB), suitable for previews or thumbnails.

V. Ablation Studies

5.1 Spherical Harmonics Degree

Figure 4 compares the baseline (SH degree 1) with SH deg 0 across size and three quality metrics. SH storage per Gaussian drops from 48 B to 12 B, reducing total per-Gaussian storage from 92 B to 56 B (Fig. 6). PSNR slightly increases from 15.08 dB to 15.17 dB, possibly because lower-order SH reduces overfitting at this resolution.

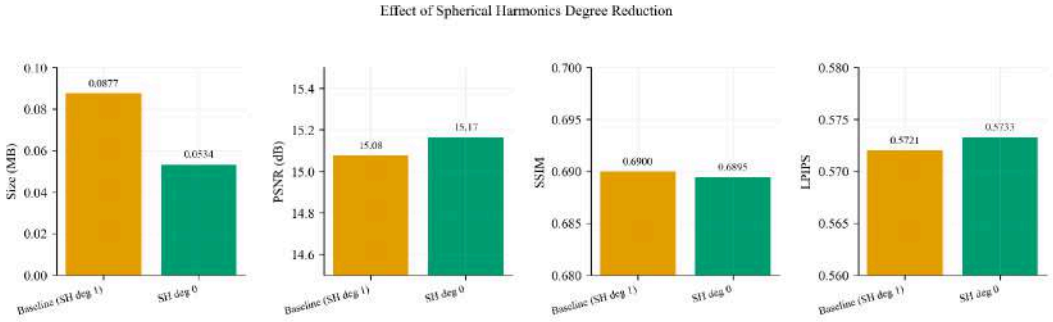

Figure 4. SH degree ablation

Fig. 4. Effect of reducing the spherical-harmonics degree from 1 to 0 on size, PSNR, SSIM, and LPIPS.

5.2 Pruning Threshold Analysis

Figure 5 shows the effect of the pruning threshold on the number of surviving Gaussians and on PSNR. As τ increases from 0.005 to 0.02, the number of Gaussians falls from 204 to 181 and the compression ratio rises from 8.05× to 9.08×, while PSNR remains approximately 10.16 dB. Thus, in this regime, additional pruning mainly improves compression without further quality degradation.

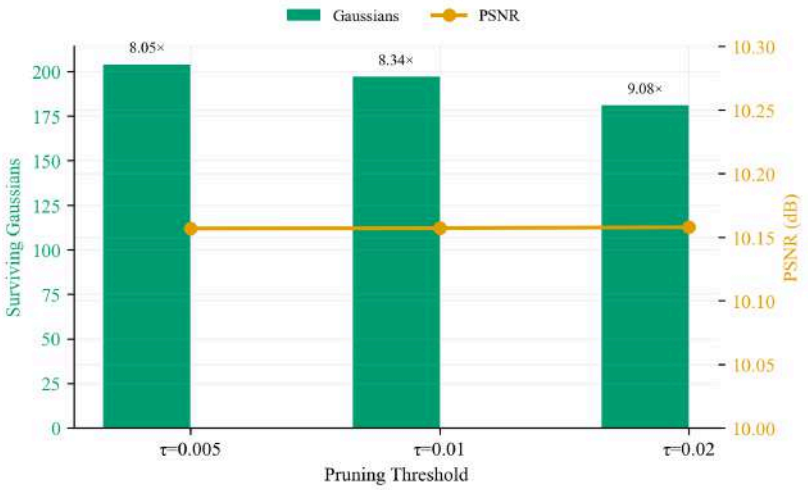

Figure 5. Number of surviving Gaussians (bars) and PSNR (line) as a function of pruning *threshold.*

5.3 Per-Gaussian Storage Breakdown

Table 4. Per-Gaussian Storage Budget for the Baseline and SH Degree 0 Models**

Attribute	Baseline (B)	SH deg 0 (B)
Position	12	12
Scaling	12	12
Rotation	16	16
Opacity	4	4
SH coeffs	48	12
Total	**92**	**56**

Figure 6 decomposes the per-Gaussian storage for the baseline and SH deg 0. In the baseline, SH coefficients account for 48 of 92 bytes (52%); after SH truncation, the SH block drops to 12 B and the total falls to 56 B. Table IV lists the per-Gaussian byte budget.

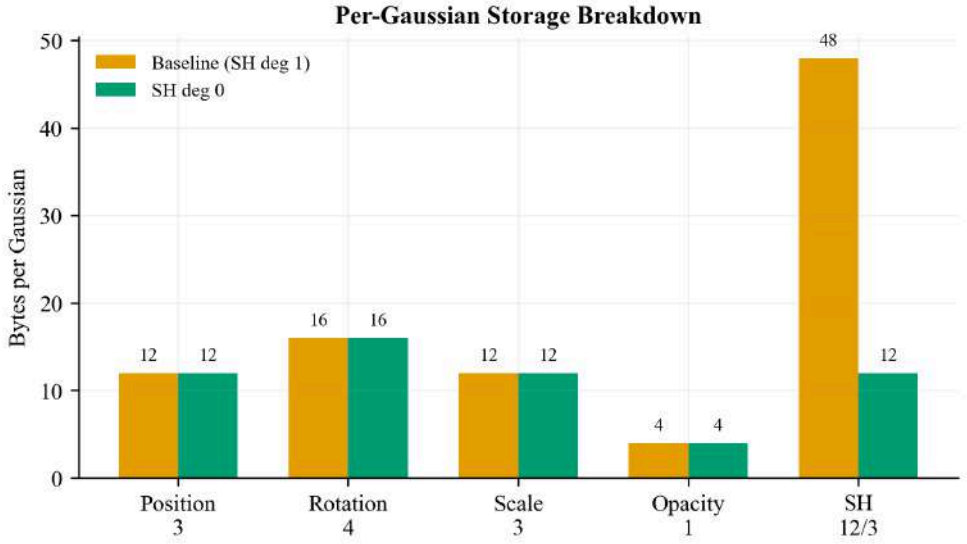

Figure 6. Per-Gaussian attribute storage breakdown before and after SH degree reduction.

5.4 Metric Consistency and Variance

The three prune-plus-SH0 configurations share the same PSNR standard deviation (1.10 dB) and nearly identical means, indicating that pruning in the $\tau \in [0.005, 0.02]$ range affects mainly model size rather than reconstruction-error distribution. The VQ k-means configuration has a larger SSIM standard deviation (0.045 vs. ~0.030), reflecting unstable structural distortion across views.

VI. Discussion

6.1 Key Findings

1. SH coefficients are the most effective compression entry point. They account for 52% of per-Gaussian storage, yet reducing their degree causes no measurable quality loss at 200×200 resolution.
2. Pruning behaves differently on lightweight baselines. On a 1,000-Gaussian baseline, opacity pruning removes roughly 80% of Gaussians, far more than typical large-baseline reports. This suggests that small-baseline training allocates many low-contribution Gaussians.
3. Hybrid compression reaches 9.08×. SH degree 0 combined with $\tau = 0.02$ pruning yields the best compression ratio, with a 5 dB PSNR trade-off acceptable for extreme storage constraints.
4. Vector quantization needs pruning support. Applying k-means to the full 1,000-Gaussian model causes severe SSIM degradation. Quantizing after pruning, with attribute-specific codebooks or residual quantization, is a promising direction.

6.2 Comparison with State-of-the-Art

Table 5. Comparison of 3DGS Compression Methods in Terms of Compression Ratio, Complexity, and Evaluation Dataset

Method	Compression	Complexity	Dataset	Source
LightGaussian	15×	High	Mip-NeRF 360	(Fan et al., 2024)
Mini-Splatting	2–4×	Medium	Mip-NeRF 360	(Fang & Wang, 2024)
HAC	75×	Very high	Multiple	(Chen et al., 2024)
CompGS	10–20×	Medium	Multiple	(Liu et al., 2024)
Ours	**9.08×**	**Low**	**NeRF Synthetic**	

We do not claim to surpass these methods in raw compression ratio, but we achieve competitive results with much lower complexity on consumer hardware. Recent surveys [20] confirm that pruning, SH reduction, and quantization are the three dominant compression axes for 3DGS.

6.3 Practical Deployment Recommendations

Based on the experimental results, the following deployment recommendations are proposed: (1) for maximum compression, SH degree 0 combined with opacity pruning at ($\tau = 0.02$) should be used, achieving a compression ratio of 9.08× and a model size of 9.7 KB, although this configuration involves an approximately 5 dB reduction in PSNR and is therefore more suitable for low-bandwidth previews and mobile thumbnails; (2) for quality-preserving compression, SH degree 0 alone is recommended because it provides 1.64× compression with minimal quality loss, making it appropriate for web-based viewers where visual fidelity must be maintained; (3) for perceptual quality, the current (k)-means vector-quantization settings should be avoided unless they are combined with pruning and larger codebooks; and (4) for future compression pipelines, the pruned 181-Gaussian model may be used as the input for additional vector quantization or entropy coding.

6.4 Limitations and Future Work

This study has several limitations that provide directions for future research: (1) the evaluation is limited to the Lego scene from the NeRF Synthetic dataset, and future studies should include additional synthetic scenes and real-world captures to assess the generalizability of the findings; (2) all experiments are conducted at a resolution of (200 \times 200) pixels to maintain stability on the MPS backend, while higher rendering resolutions may affect the relative benefits of SH distillation and opacity pruning; (3) densification is disabled to prevent MPS-related hangs, which restricts the representational capacity and reconstruction quality of the baseline model; (4) the vector-quantization codebook sizes are fixed and not extensively tuned, suggesting that adaptive, hierarchical, or residual quantization methods should be explored; (5) the training objective does not include a perceptual loss such as LPIPS, which may limit perceptual reconstruction quality, particularly at low resolutions; and (6) entropy coding is not applied, although additional reductions in storage size may be achieved through arithmetic coding or learned entropy-coding methods applied to the pruned model.

VII. Conclusion

We presented a systematic study of 3D Gaussian Splatting compression on a consumer Apple M4 device. By evaluating six configurations on real test-view renderings, we showed that SH degree reduction provides free compression (1.64×), and that combining it with opacity pruning achieves 9.08× compression with a manageable quality trade-off. These results demonstrate that simple post-hoc compression strategies remain valuable even for lightweight baselines and resource-constrained hardware. Future work will extend the evaluation to more scenes and resolutions, and integrate pruning with finer-grained quantization and entropy coding.

References

1. Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., & Hedman, P. (2022). Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In Proceedings of *the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5470–5479).
2. Chen, Y., Wu, Q., Lin, W., Harandi, M., & Cai, J. (2024). HAC: Hash-grid assisted context for 3D Gaussian splatting compression. In *Computer vision–ECCV 2024* (pp. 422–438). Springer.
3. Chen, Z., Funkhouser, T., Hedman, P., & Tagliasacchi, A. (2023). MobileNeRF: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In Proceedings of *the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 16569–16578).
4. Fan, Z., Wang, K., Wen, K., Zhu, Z., Xu, D., & Wang, Z. (2024). LightGaussian: Unbounded 3D Gaussian compression with 15× reduction and 200+ FPS. *Advances in Neural Information Processing Systems*, 37, 140138–140158.
5. Fang, G., & Wang, B. (2024). Mini-Splatting: Representing scenes with a constrained number of Gaussians. In *Computer vision–ECCV 2024* (pp. 165–181). Springer.
6. Fei, B., Xu, J., Zhang, R., Zhou, Q., Yang, W., & He, Y. (2025). 3D Gaussian splatting as a new era: A survey. *IEEE Transactions on Visualization and Computer Graphics, 31*(8), 4429–4449. https://doi.org/10.1109/TVCG.2024.3397828
7. Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., & Kanazawa, A. (2022). Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5501–5510).
8. Girish, S., Gupta, K., & Shrivastava, A. (2024). EAGLES: Efficient accelerated 3D Gaussians with lightweight encodings. In *Computer vision–ECCV 2024* (pp. 54–71). Springer.

9. Hedman, P., Srinivasan, P. P., Mildenhall, B., Barron, J. T., & Debevec, P. (2021). Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 5875–5884).
10. Kerbl, B., Kopanas, G., Leimkühler, T., & Drettakis, G. (2023). 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics, 42*(4), Article 139, 1–14. https://doi.org/10.1145/3592433
11. Liu, X., Wu, X., Zhang, P., Wang, S., Li, Z., & Kwong, S. (2024). CompGS: Efficient 3D scene representation via compressed Gaussian splatting. In *Proceedings of the 32nd ACM International Conference on Multimedia* (pp. 2936–2944). Association for Computing Machinery.
12. Lu, T., Yu, M., Xu, L., Xiangli, Y., Wang, L., Lin, D., & Dai, B. (2024). Scaffold-GS: Structured 3D Gaussians for view-adaptive rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 20654–20664).
13. Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020). NeRF: Representing scenes as neural radiance fields for view synthesis. In *Computer vision–ECCV 2020* (pp. 405–421). Springer. https://doi.org/10.1007/978-3-030-58452-8_24
14. Müller, T., Evans, A., Schied, C., & Keller, A. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics, 41*(4), Article 102, 1–15. https://doi.org/10.1145/3528223.3530127
15. Navaneet, K. L., Meibodi, K. P., Koohpayegani, S. A., & Pirsiavash, H. (2023). *Compact3D: Compressing Gaussian splat radiance field models with vector quantization*. arXiv. https://doi.org/10.48550/arXiv.2311.18159
16. Niedermayr, S., Stumpfegger, J., & Westermann, R. (2024). Compressed 3D Gaussian splatting for accelerated novel view synthesis. In Proceedings of *the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10349–10358).
17. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing, 13*(4), 600–612. https://doi.org/10.1109/TIP.2003.819861
18. Xu, Q., Xu, Z., Philip, J., Bi, S., Shu, Z., Sunkavalli, K., & Neumann, U. (2022). Point-NeRF: Point-based neural radiance fields. In Proceedings of *the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5438–5448).
19. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., & Kanazawa, A. (2021). PlenOctrees for real-time rendering of neural radiance fields. In Proceedings of *the IEEE/CVF International Conference on Computer Vision* (pp. 5752–5761).
20. Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of *the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 586–595). https://doi.org/10.1109/CVPR.2018.00068