

L-GGSC: Learnable Graph-based Gaussian Splatting Compression

Akihiro Kuwabara¹, Hinata Kirihara¹, Sorachi Kato¹, Toshiaki Koike-Akino², Takuya Fujihashi¹

¹The University of Osaka, Japan

²Mitsubishi Electric Research Laboratories (MERL), USA

{kuwabara.akihiro, kirihara.hinata, kato.sorachi, fujihashi.takuya}@ist.osaka-u.ac.jp

koike@merl.com

Abstract

3D Gaussian Splatting (GS) has emerged as a method that achieves high-quality 3D scene representation and fast rendering, with applications in various fields. However, the substantial storage requirements of complex scenes limit its practical deployment on resource-constrained platforms. In this paper, we propose a novel method, namely learnable graph-based GS compression (L-GGSC). L-GGSC introduces a parameterized graph shift operator and a systematic parameter reduction strategy to optimize the hyperparameter search space. Evaluations on three 3D GS datasets using the typical parameter of the graph shift operators demonstrate that the parameterized graph shift operator of the proposed L-GGSC has the potential to simultaneously improve data size and rendering quality against the regular graph Laplacian matrix.

1. Introduction

3D Gaussian Splatting (GS) [8] has attracted significant attention for its effective representation and rendering of 3D scenes. It is used in various applications such as 3D reconstruction and simultaneous localization and mapping (SLAM) [1, 5, 13, 21]. Unlike Neural Radiance Fields (NeRF) [14] with implicit representations, 3D GS explicitly reconstructs 3D scenes using a collection of 3D Gaussians. Each Gaussian consists of geometric and appearance components, including a 3D center position, a scale vector, a rotation, spherical harmonic (SH) coefficients for view-dependent color and an opacity. These attributes enable 3D GS to reconstruct scenes with high visual fidelity.

However, despite its capabilities, 3D GS has substantial storage and computational requirements. Storing all primitive attributes with full precision often re-

sults in files that are several gigabytes in size for complex scenes. This limits its applicability to resource-constrained platforms, such as mobile devices. Therefore, compression techniques are essential for reducing data redundancy and enabling effective storage and transmission.

3D GS compression approaches can be broadly categorized into two distinct classes: generative and traditional compression methods. Generative methods are designed to construct more compact scene representations by optimizing 3D GS parameters under specific constraints or by learning a compact parameter representation. To obtain the compact representation, some studies consider that many Gaussian primitives share similar attributes, and thus they use codebooks based on K-means for quantization [4, 11, 15–17]. Other studies aim to use lightweight multi-layer perceptrons (MLPs) for generating Gaussian attributes such as color and shape from inputs including embeddings or structural features [2, 6, 12, 19].

On the other hand, traditional compression methods convert the learned 3D Gaussians into a bitstream suitable for storage and transmission by applying signal-processing-based compression techniques. A pioneering work is Graph-based Gaussian Splatting Compression (GGSC) [22], which utilizes Graph Signal Processing (GSP). They regard the Gaussian primitives as a graph signal and define a graph Fourier transform (GFT) based on the graph. Since GFT realizes frequency conversion in the graph domain, the subsequent quantization and coding operations yield a compact representation of the 3D GS. However, GGSC still has two key issues. The first issue is that GGSC only uses the standard graph shift operator for GFT. It is well-known that the decorrelation performance of GFT depends heavily on the graph shift operator used. The second issue is the large search space required to find an optimal graph shift operator for

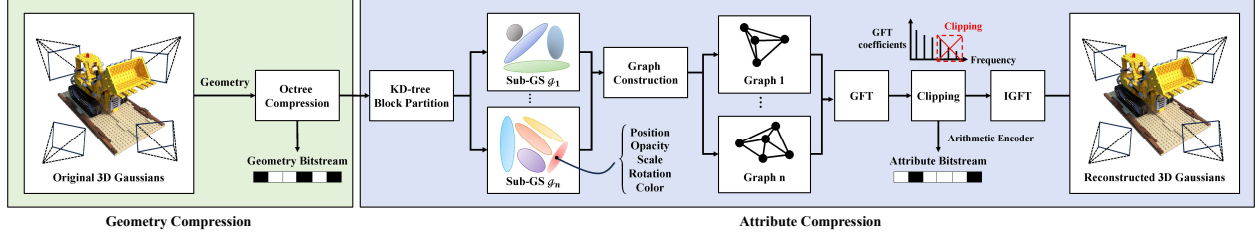


Figure 1. Overview of the proposed scheme.

each 3D GS compression. Some studies have designed a seven-parameter graph shift operator [3, 9]. However, this approach requires significant computational overhead.

This paper proposes a novel graph-based GS compression method, namely, Learnable-GGSC (L-GGSC). In contrast to the pioneering GGSC work, the main contributions of the proposed L-GGSC are two-fold. The first contribution is to introduce the parameterized graph shift operator to realize learnable graph-based GS compression. To the best of our knowledge, this is the first study to make the graph shift operator learnable for 3D GS compression. The second contribution is a novel parameter reduction strategy that quickly optimizes the graph shift operator for the desired 3D GS. The original graph shift operator has seven parameters, but we reduced it to four by analyzing the effect of the parameters on the GFT basis.

Evaluations on three 3D GS datasets using the typical parameter of the graph shift operators demonstrate that the proposed L-GGSC scheme achieves better reconstruction quality than GGSC at low bit rates, thereby realizing more efficient data compression for 3D scenes. In addition, through comprehensive experiments with varying clipping ratios and quantization precisions, we reveal that the optimal graph shift operator varies depending on the specific attribute and compression level.

2. Proposed scheme

2.1. Overview

Fig. 1 shows the overview of the proposed scheme. We consider that the encoder operates on a 3D GS sample comprising N primitives, each characterized by attributes such as spherical harmonics (SH) coefficients, opacity, scale, and rotation. The proposed scheme performs sequential compression of geometry and attributes. For geometry compression, an octree-based method is employed to reduce the number of primitives and encode their positions into a compact bitstream.

Based on the positions of the remaining primitives, the encoder constructs a graph structure and derives graph basis functions using adjacency and degree matrices. Each attribute is then projected into the frequency domain by computing the corresponding GFT coefficients using the derived basis functions. These GFT coefficients are subsequently clipped, quantized, and entropy-coded for efficient compression. At the decoder side, the positions of the primitives are first reconstructed from the geometry bitstream. Using the decoded positions, the graph basis matrix is reconstructed, and the GFT coefficients are inverse-transformed to decode the attribute vectors of the SH coefficients, opacity, scale, and rotation. Finally, the decoder renders the 3D scene using the reconstructed geometry and attribute information.

2.2. Sub-GS division

An original GS sample may have tens of thousands to millions of primitives. Since the computational cost of graph signal processing increases significantly with the number of primitives, we first partition the original graph signal into smaller subsets. To achieve balanced partitioning, the proposed scheme utilizes a kd-tree-based strategy, which tends to distribute a similar number of primitives across sub-GS compared to octree-based partitioning. Specifically, the full set of Gaussian primitives $\mathcal{G}$ is divided into n subgroups based on the spatial locations of their centers:

$$\mathcal{G} = \mathcal{G}_1^{(m_1)} \oplus \mathcal{G}_2^{(m_2)} \oplus \dots \oplus \mathcal{G}_n^{(m_n)}, \quad (1)$$

where $\mathcal{G}_i^{(m_i)}$ denotes the i -th sub-GS containing m_i primitives. In our implementation, we set an upper bound of $m_i \leq 200$ to ensure that each subgraph remains computationally tractable for GFT-based attribute compression.

2.3. Graph construction and GFT

The encoder compresses each attribute vector using a GFT derived from the compressed positions of the primitives. To this end, 3D GS is modeled as a weighted,

undirected graph signal $\mathcal{G} = (\mathbf{V}, \mathcal{E}, \mathbf{W})$, where $\mathbf{V}$ and $\mathcal{E}$ denote the sets of vertices (primitives) and edges, respectively, and $\mathbf{W}$ is the adjacency matrix with positive weights. The element $W_{i,j}$ of $\mathbf{W}$ represents the weight of the edge connecting vertex i and vertex j . Each element $W_{i,j}$ of the adjacency matrix is defined by the following equation:

$$W_{i,j} = \exp\left(-\frac{\|\mathbf{p}_i - \mathbf{p}_j\|_2^2}{\kappa_p}\right), \quad (2)$$

where κ_p is a parameter representing the variance of the distance between two primitives. The degree matrix $\mathbf{D}$ is derived from the adjacency matrix $\mathbf{W}$ as follows:

$$\mathbf{D} = \text{diag}(D_1, D_2, \dots, D_N), \quad D_i = \sum_{n=1}^{\hat{N}} W_{i,n}, \quad (3)$$

where $\hat{N}$ is the number of primitives after geometry compression. The GFT basis functions are derived from the graph shift operator $\mathbf{L}$, which is parameterized by the adjacency matrix $\mathbf{W}$ and the degree matrix $\mathbf{D}$. $\mathbf{L}$ is formulated using seven hyperparameters as follows [3]:

$$\mathbf{L} = m_1 \mathbf{D}_a^{e_1} + m_2 \mathbf{D}_a^{e_2} \mathbf{W}_a \mathbf{D}_a^{e_3} + m_3 \mathbf{I}, \quad (4)$$

where $\mathbf{W}_a = \mathbf{W} + a\mathbf{I}$ is the adjusted adjacency matrix, $\mathbf{D}_a$ is the corresponding degree matrix, and $\mathbf{I}$ is the $\hat{N} \times \hat{N}$ identity matrix. The hyperparameter set $\mathbf{m} = (m_1, m_2, m_3, e_1, e_2, e_3, a) \in \mathbb{R}^7$ controls the construction of the graph shift operator.

The graph shift operator $\mathbf{L}$ is generally real and symmetric, and therefore admits an orthogonal set of eigenvectors with real, non-negative eigenvalues. The eigenvalue decomposition of $\mathbf{L}$ is given by:

$$\mathbf{L} = \Phi \Delta \Phi^{-1}, \quad (5)$$

where Φ is the matrix of eigenvectors, and Δ is a diagonal matrix of the corresponding eigenvalues. The GFT coefficients $\mathbf{f}$, which represent the frequency components of an attribute vector $\mathbf{s}$, are computed by projecting $\mathbf{s}$ onto the eigenbasis:

$$\mathbf{f} = \mathbf{s} \Phi. \quad (6)$$

Most of the resulting coefficients $\mathbf{f}$ are small and close to zero, especially on the high-frequency components. Here, the eigenvectors corresponding to the large eigenvalues represent the high-frequency components in the graph domain. Before quantization, the high-frequency components of the GFT coefficients are clipped to realize compression without introducing noticeable distortion. We define a clipping ratio $r_a \in (0, 1]$ where

$(1 - r_a) \cdot \hat{N}$ higher-frequency coefficients in $\mathbf{f}$ for attribute a are set to 0 for clipping. The clipped GFT coefficients are then quantized and entropy-coded to generate the compressed bitstream.

At the decoder side, entropy decoding and dequantization are performed to obtain an estimate of the GFT coefficients, denoted as $\hat{\mathbf{f}}$. The inverse GFT (IGFT) is then applied to reconstruct the attribute vector $\hat{\mathbf{s}}$ using the decoded GFT coefficients and the eigenvectors:

$$\hat{\mathbf{s}} = \hat{\mathbf{f}} \Phi^{-1}. \quad (7)$$

2.4. Compression

The proposed scheme adopts a uniform quantization to convert the floating-point values of the GFT coefficients $\mathbf{f}$ in each attribute vector into integer values as suggested by [18]. Specifically, let $f_i^{(a)}$ represents i -th floating-point value of attribute vector a and $\hat{f}_i^{(a)}$ represents the corresponding integer value.

$$\hat{f}_i^{(a)} = \left\lfloor \frac{(f_i^{(a)} - f_{\min}^{(a)}) * (2^{q_a} - 1)}{\max(f_{\max}^{(a)} - f_{\min}^{(a)})} + \frac{1}{2} \right\rfloor, \quad (8)$$

where q_a is a bit depth for attribute a 's quantization, $f_{\min}^{(a)}$ and $f_{\max}^{(a)}$ are minimum and maximum floating-point value of attribute a , $\max(\cdot)$ is the maximum value across all the attributes, and $\lfloor x \rfloor$ is the floor function and gives the greatest integer less than or equal x . After quantization, we use adaptive arithmetic coding to compress the integer value of the attributes.

2.5. Parameter reduction

To effectively minimize traffic while preserving the quality of 3D GS-based scene reconstruction, it is essential to optimize the hyperparameter set $\mathbf{m} = (m_1, m_2, m_3, e_1, e_2, e_3, a)$ of graph shift operator and quantization parameter q , under the given bandwidth constraint. Assuming M candidate values per parameter, an exhaustive search would require evaluating M^8 combinations, which is computationally expensive and inefficient.

To mitigate this complexity, the proposed method reduces the parameter space by identifying and retaining only those parameters that significantly impact rate-distortion performance in the context of 3D GS compression. An important insight arises from the spectral properties of the graph shift operator: the eigenvectors—used as the basis for the GFT—are invariant to the scalar term $m_3 \mathbf{I}$. We simplify the formulation by setting $m_3 = 0$, resulting in a six-parameter formulation:

$$\mathbf{L}_{6\text{params}} = m_1 \mathbf{D}_a^{e_1} + m_2 \mathbf{D}_a^{e_2} \mathbf{W}_a \mathbf{D}_a^{e_3}. \quad (9)$$

Table 1. Quantitative comparison between GGSC and L-GGSC for various datasets.

Bitrate	Method	Tanks & Temples [10]: train				DeepBlending [7]: playroom				Synthetic-NeRF [14]: lego			
		Size (MB) ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Size (MB) ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Size (MB) ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Low	GGSC (L_{reg})	25.86	16.05	0.492	0.489	34.21	19.46	0.747	0.434	7.47	24.03	0.830	0.154
	L-GGSC (L_{rw})	15.88	16.15	0.492	0.480	27.83	21.85	0.769	0.416	6.96	24.50	0.845	0.146
	L-GGSC (L_{comb})	15.72	16.12	0.491	0.480	28.41	22.18	0.768	0.411	6.98	24.23	0.835	0.160
High	GGSC (L_{reg})	90.58	21.06	0.768	0.246	122.51	29.00	0.884	0.269	30.26	27.81	0.947	0.050
	L-GGSC (L_{rw})	81.10	20.40	0.759	0.255	131.46	28.45	0.879	0.274	33.24	27.71	0.947	0.050
	L-GGSC (L_{comb})	84.09	20.37	0.757	0.256	137.63	28.29	0.879	0.274	33.08	27.69	0.947	0.050

We further reduce dimensionality by normalizing m_1 with respect to m_2 as $m'_1 = \frac{m_1}{m_2}$, yielding a five-parameter operator:

$$L_{5\text{params}} = \frac{L_{6\text{params}}}{m_2} = m'_1 D_a^{e_1} + D_a^{e_2} W_a D_a^{e_3}. \quad (10)$$

To ensure that L remains a real symmetric matrix—required for defining a GFT with orthogonal eigenvectors—we ensure the condition $e_2 = e_3$. We define a consolidated exponent $e_+ = \frac{e_2 + e_3}{2}$, which leads to the final four-parameter formulation:

$$L_{4\text{params}} = m'_1 D_a^{e_1} + D_a^{e_+} W_a D_a^{e_+}. \quad (11)$$

In summary, the proposed scheme reduces the search space for finding an effective graph basis Φ —that yields favorable rate-distortion trade-offs in 3D GS compression—to four hyperparameters $\mathbf{m}' = (m'_1, e_1, e_+, a)$ and one quantization parameter q .

3. Experiments

3.1. Settings

Datasets and metrics: For a comprehensive comparison against GGSC [22], we select three scenes from different datasets: “train” from Tanks & Temples [10], “playroom” from Deep Blending [7] as real-world datasets, and “lego” from Synthetic NeRF [14] as a synthetic dataset. We used the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM) [20], Learned Perceptual Image Patch Similarity (LPIPS) [23], and compressed data size in megabytes (MB) as performance metrics.

Baseline and graph shift operators: We consider GGSC as the baseline for the graph-based 3D GS compression. GGSC and the proposed L-GGSC utilize a graph shift operator to derive the basis functions for the GFT. In this paper, we use the seven-parameter L-GGSC for comparison. Here, the graph shift operator L in L-GGSC is formulated using the adjacency matrix W , the degree matrix D , and seven hyperparameters



Figure 2. Visual snapshots of baseline GGSC and the proposed L-GGSC for the “train” scene.

$\mathbf{m} = (m_1, m_2, m_3, e_1, e_2, e_3, a)$, as defined in Eq. (4). In this paper, we specifically investigate the following three types of graph shift operators by using different combinations of hyperparameters:

- **Regular L_{reg} :** This operator corresponds to the hyperparameter setting of $\mathbf{m} = (1, -1, 0, 1, 0, 0, 0)$ and is described as $D - W$. This operator is used as the graph shift operator in GGSC.
- **Random walk L_{rw} :** This operator corresponds to the hyperparameter setting of $\mathbf{m} = (0, -1, 1, 0, -1, 0, 0)$ and is described as $I_n - D^{-1}W$.
- **Combination L_{comb} :** It corresponds to the hyperparameter setting of $\mathbf{m} = (0, -1, 1, 0, -\frac{1}{2}, -\frac{1}{2}, 0)$ and is described as $I_n - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}$.

We note that the proposed L-GGSC, using the four hyperparameters $L_{4\text{params}}$ defined in Eq. (11), can be used by optimizing the hyperparameters based on an objective function. Further analysis of the rate-distortion performance and optimization speed reduction based on the proposed reduced hyperparameters is left as future work.

Compression settings: The clipping ratio r_a and bit depth for each attribute q_a are parameters used to evaluate rate-distortion performance. To ensure a fair com-

parison, the specific parameter settings for low bit rate and high bit rate environments were aligned with those utilized in the GGSC implementation. Specifically, the frequency clipping ratios for SH coefficients, opacity, scale, and rotation attributes were defined as r_C, r_O, r_S, r_R , respectively, and these were set within the range of 0.4 to 1.0. The bit depth for these attributes was also defined as q_C, q_O, q_S, q_R , respectively, and these were set within the range of 4 to 8.

Implementation detail: All the evaluations exhibited in this paper are performed with an Intel(R) Xeon(R) Silver4108 CPU@ 1.80 GHz, an NVIDIA Quadro GV100 GPU, and PyTorch 2.1.2 with Python 3.10.

3.2. Rate-distortion performance

We evaluate the rate-distortion performance of the proposed scheme under low bit rate and high bit rate settings. Table 1 shows the compressed data size and reconstruction quality of the desired 3D GS for GGSC and the proposed L-GGSC. At a low bit rate, L-GGSC outperforms GGSC by simultaneously achieving reduced data sizes and better reconstruction quality on the “train”, “playroom”, and “lego” scenes. This demonstrates the effectiveness of parameterizing the graph shift operator, especially at low bit rates.

Fig. 2 shows the visual snapshots of the rendered 3D GS for GGSC and the proposed L-GGSC. Here, we use the “train” scene from the Tanks & Temples dataset. The GGSC method exhibits noticeable blurring, particularly on the “WESTERN PACIFIC” text and the “713” number on the train. In contrast, the proposed L-GGSC can reduce the blur effect and render these details more clearly with up to a 39% reduction in data size.

3.3. Effect of graph shift operators on attributes

This section evaluates the impact of different graph shift operators on the compression performance of individual 3D GS attributes, considering varying frequency clipping ratios and quantization precisions. Figs. 3 and 4 illustrate the PSNR of the rendered 3D GS as a function of the clipping ratio and bit depth for each attribute, respectively. It can be seen that the performance degradation due to compression of the opacity and rotation attributes is smaller than that due to the SH coefficients and scale attributes. Furthermore, it is demonstrated that for SH and scale attributes, the optimal graph shift operator varies significantly with the compression level. These findings suggest that no single graph shift operator is universally optimal. Thus, the graph-based GS compression should find the most effective hyperparameters for the graph shift operator, depending on the specific attribute and desired compression level.

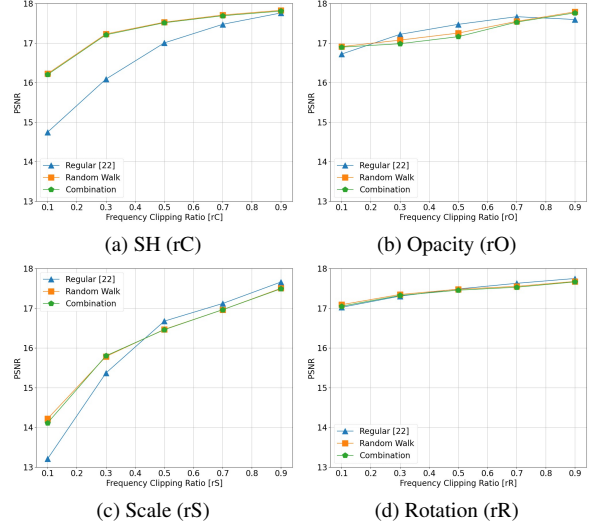


Figure 3. PSNR as a function of the frequency clipping ratio for different attributes.

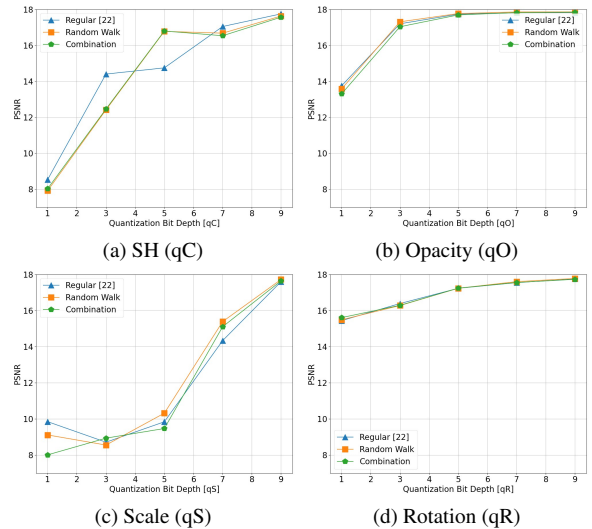


Figure 4. PSNR as a function of the quantization bit depth for different attributes.

4. Conclusion and future work

This paper proposes a novel graph-based GS compression method, L-GGSC. The proposed L-GGSC introduces a parameterized graph shift operator and a systematic parameter reduction strategy. Experiments on three datasets demonstrate that L-GGSC efficiently reduces storage cost while maintaining high rendering quality. In future work, we will investigate learning the four hyperparameters to achieve better rate-distortion and optimization convergence.

Acknowledgment

This work was supported by JST-ASPIRE Grant Number JPMJAP2432 and JSPS KAKENHI Grant Number JP22H03582.

References

- [1] Guikun Chen and Wenguan Wang. A survey on 3d gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024. [1](#)
- [2] Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Hac: Hash-grid assisted context for 3d gaussian splatting compression. In *European Conference on Computer Vision*, pages 422–438. Springer, 2024. [1](#)
- [3] George Dasoulas, Johannes Lutzeyer, and Michalis Vazirgiannis. Learning parametrised graph shift operators. In *International Conference on Learning Representations (ICLR)*, pages 1–17, 2021. Virtual. [2](#), [3](#)
- [4] Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, De-jia Xu, Zhangyang Wang, et al. Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. *Advances in neural information processing systems*, 37:140138–140158, 2024. [1](#)
- [5] 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. [1](#)
- [6] Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. Eagles: Efficient accelerated 3d gaussians with lightweight encodings. In *European Conference on Computer Vision*, pages 54–71. Springer, 2024. [1](#)
- [7] 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. [4](#)
- [8] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), 2023. [1](#)
- [9] Hinata Kirihaara, Shoichi Ibuki, Takuya Fujihashi, Toshiaki Koike-Akino, and Takashi Watanabe. Point cloud geometry compression using parameterized graph fourier transform. In *Proceedings of the 2024 SIGCOMM Workshop on Emerging Multimedia Systems*, page 52–57, 2024. [2](#)
- [10] 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. [4](#)
- [11] Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3d gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21719–21728, 2024. [1](#)
- [12] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20654–20664, 2024. [1](#)
- [13] Hidenobu Matsuki, Riku Murai, Paul H.J. Kelly, and Andrew J. Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18039–18048, 2024. [1](#)
- [14] 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. [1](#), [4](#)
- [15] K Navaneet, Kossar Pourahmadi Meibodi, Soroush Abbasi Koohpayegani, and Hamed Pirsiavash. Compact3d: Compressing gaussian splat radiance field models with vector quantization. *arXiv preprint arXiv:2311.18159*, 4, 2023. [1](#)
- [16] Simon Niedermayr, Josef Stumpffegger, and Rüdiger Westermann. Compressed 3d gaussian splatting for accelerated novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10349–10358, 2024.
- [17] Panagiotis Papantonakis, Georgios Kopanas, Bernhard Kerbl, Alexandre Lanvin, and George Drettakis. Reducing the memory footprint of 3d gaussian splatting. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 7(1):1–17, 2024. [1](#)
- [18] VVM. Call for proposals on static polygonal mesh coding, Alliance for Open Media. document: VVM-2023-004o-v2 (2023). [3](#)
- [19] Yufei Wang, Zhihao Li, Lanqing Guo, Wenhan Yang, Alex Kot, and Bihan Wen. Contextgs: Compact 3d gaussian splatting with anchor level context model. *Advances in neural information processing systems*, 37: 51532–51551, 2024. [1](#)
- [20] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. [4](#)
- [21] Tong Wu, Yu-Jie Yuan, Ling-Xiao Zhang, Jie Yang, Yan-Pei Cao, Ling-Qi Yan, and Lin Gao. Recent advances in 3d gaussian splatting. *Computational Visual Media*, 10(4):613–642, 2024. [1](#)
- [22] Qi Yang, Kaifa Yang, Yuke Xing, Yiling Xu, and Zhu Li. A benchmark for gaussian splatting compression and quality assessment study. In *Proceedings of the 6th ACM International Conference on Multimedia in Asia*, pages 1–8, 2024. [1](#), [4](#)
- [23] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features

as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018. [4](#)