Clustered Codebook Quantization for 2D Gaussian-based Image Compression

Runze Cheng*
University College London
London, United Kingdom
runze.cheng.23@ucl.ac.uk

Yicheng Zhan
University College London
London, United Kingdom
yicheng.zhan.21@ucl.ac.uk

Josef Spjut
NVIDIA
Durham, USA
jspjut@nvidia.com

Kaan Akşit
University College London
London, United Kingdom
kaanaksit@kaanaksit.com

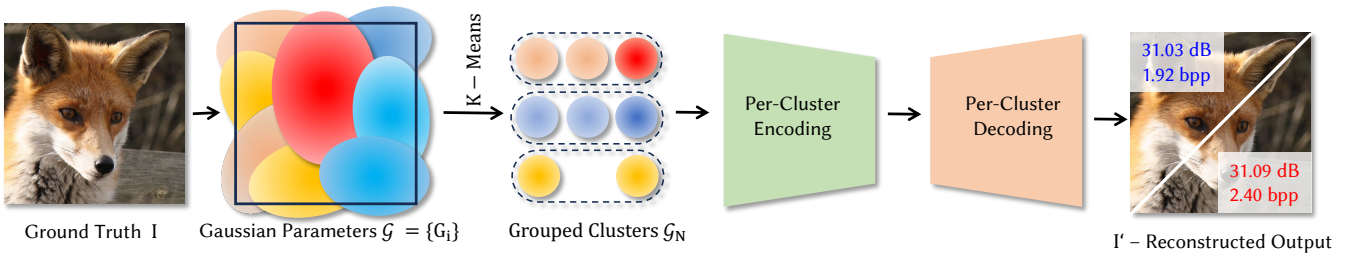

Figure 1: Overview of our Cluster-Guided Vector Quantization (CGVQ) pipeline for 2D Gaussian-based Image Compression. Given a ground truth image I, a set of 2D Gaussian primitives $\mathcal{G} = \{G_i\}$ is first fitted to the image. K-Means clustering partitions $\mathcal{G}$ into N groups $\mathcal{G}_N$ based on appearance and anisotropy similarity. Per-cluster codebooks are then trained and used to encode each group independently. At decoding, each cluster is reconstructed via multiple codebooks lookup and all clusters are composed for rendering, producing the reconstructed output I' with Bit-Per-Pixel (bpp) at 1.92 and the baseline GaussianImage (GI) [Zhang et al. 2025a] output at bpp=2.40 under similar reconstruction fidelity (PSNR = 31.0XdB). Source image © Animal Faces.

Abstract

Gaussian-based image representations effectively model image content using compact parametric primitives while preserving high visual fidelity [Zhang et al. 2025a], yet storing a large number of floating-point parameters per primitive degrades rate-distortion efficiency at higher fidelity targets. To improve the rate-distortion performance in Gaussian representation, we present our **CGVQ**, a Gaussian primitive based image compression method. Our key idea is to partition Gaussian parameters further into homogeneous groups prior to quantization, enabling higher compression efficiency and accurate parameter reconstruction. In practice, our extensive experiments show that **CGVQ** decreases the bpp by 20%↓ with respect to our baseline **GI** [Zhang et al. 2025a], while maintaining on-par visual quality.

CCS Concepts

• **Computing methodologies** → **Image compression**; *Rendering*.

Keywords

Image Compression, 2D Gaussian Splatting, K-Means, Vector Quantizer

*Primary contributor to this work

SIGGRAPH Posters '26, Los Angeles, CA, USA

ACM ISBN 979-8-4007-2548-7/26/07
https://doi.org/10.1145/3799825.3818700

ACM Reference Format:
Runze Cheng, Yicheng Zhan, Josef Spjut, and Kaan Akşit. 2026. Clustered Codebook Quantization for 2D Gaussian-based Image Compression. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters (SIGGRAPH Posters '26), July 19–23, 2026, Los Angeles, CA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3799825.3818700

1 Introduction

Image compression is essential for real-time rendering & VR/AR [Walton et al. 2021] and high-density media storage [Wang et al. 2010], demanding realistic visual fidelity alongside computational efficiency. While traditional codecs like JPEG and PNG provide robust baselines, deep learning architectures [Walton et al. 2021] have been proposed to further improve the boundaries of rate-distortion performance. More Recently, have been posed as a promising alternative for compact image representation [Zhang et al. 2025a,b]. By modeling local color and structural details with anisotropic Gaussian primitives, Gaussian-based image representation enables flexible, resolution-independent rendering and ultra-fast, GPU-friendly inference [Huang et al. 2024; Mescheder et al. 2026; Zhang et al. 2025a]. However, this flexibility incurs a storage cost. Each 2D Gaussian primitive consists of multiple unconstrained floating-point parameters (e.g., positions, covariance, and color). Accurately capturing high-frequency textures requires dense splat allocations, bloating the parameter count and degrading compression efficiency compared to SOTA VAE-based codecs [van den Oord et al. 2018; Wu et al. 2024]. To address this, **GI** [Zhang et al. 2025a] proposed to use Vector Quantization (VQ) and Residual Vector Quantization (RQ) to compress these floating-point attributes into discrete

codebooks. Yet, forcing a capacity-constrained global codebook to map the immense variance of natural image parameters inevitably leads to quantization errors and visual artifacts. **This limitation demands the need for a representation that preserves an accurate reconstruction of Gaussians while constraining parameter entropy to improve codebook efficiency.**

We propose to leverage a Cluster-based Codebook Quantizing method, **CGVQ**. Based on the principle: for a fixed codebook capacity, narrower parameter distributions are likely to achieve better quantization [Han et al. 2016]. We therefore apply K-Means to divide 2D Gaussian primitives into homogeneous clusters and train cluster-specific codebooks for position, rotation-scale, and color, followed by efficient lookup table-based decoding. The localized multi-codebook design improves reconstruction quality with only a marginal compression overhead, achieving lower distortion than baseline models within specific bitrate regimes. Our main contributions are summarized as follows:

- We propose **CGVQ**, which partitions 2D Gaussian primitives into low-variance groups before quantization.
- We show that localized codebooks provide a better compression–fidelity trade-off than a global codebook baseline, improving PSNR/SSIM at comparable bitrate. This highlights the potential of utilizing K-Means clustering as a promising direction for future codebook quantization optimization.

Our method **CGVQ** is inspired and extended based on **GI** [Zhang et al. 2025a]. The code is available at this repository.

2 Method

Following **GI**, we represent the ground truth I as a set of N 2D Gaussian primitives G_i: $\mathcal{G} = \{G_i\}_{i=1}^{N}$, $G_i = \{\boldsymbol{\mu}_i, \mathbf{RS}_i, \mathbf{c}_i\}$, where $\boldsymbol{\mu}_i \in \mathbb{R}^2$ is the 2D position, $\mathbf{RS}_i \in \mathbb{R}^{2\times 2}$ is the combined rotation-scale matrix, and $\mathbf{c}_i \in \mathbb{R}^3$ is the weighted color feature [Zhang et al. 2025a]. In practice, the rotation-scale matrix RS_i is parameterized by two learnable attributes: a scale vector $\mathbf{s}_i \in \mathbb{R}^2$ and a rotation parameter $\mathbf{r}_i \in \mathbb{R}^1$. The matrix used for rendering is then obtained by a deterministic mapping $\mathbf{RS}_i = F(\mathbf{r}_i, \mathbf{s}_i)$, where $F(\cdot)$ constructs the corresponding rotation-scale matrix from the rotation and scale parameters.

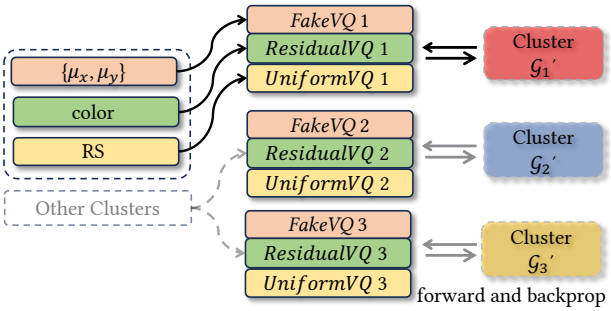

Figure 2: Codebook training and per-cluster compression.

To enable efficient per-cluster quantization, we group Gaussian primitives by feature similarity prior to codebook training. For each Gaussian primitive G_i, we construct a clustering feature vector by concatenating its rotation, scale, and RGB color attributes: $\mathbf{f}_i = [\mathbf{r}_i, \mathbf{s}_i, \mathbf{c}_i] \in \mathbb{R}^6$, where $\mathbf{r}_i \in \mathbb{R}^1$ denotes the rotation parameter, $\mathbf{s}_i \in \mathbb{R}^2$ denotes the two scale parameters, and $\mathbf{c}_i \in \mathbb{R}^3$ denotes the RGB color feature. The 2D position $\boldsymbol{\mu}_i$ is deliberately excluded, as compression quality shows a high sensitivity to the spatial accuracy. We then apply K-Means on the normalized features $\{\hat{\mathbf{f}}_i\}$ to partition $\mathcal{G}$ into K disjoint clusters by minimizing the total intra-cluster variance:

$$\{\hat{C}_k\}_{k=1}^{K} \leftarrow \underset{\{C_k\}}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{i \in C_k} \|\hat{\mathbf{f}}_i - \boldsymbol{m}_k\|^2, \tag{1}$$

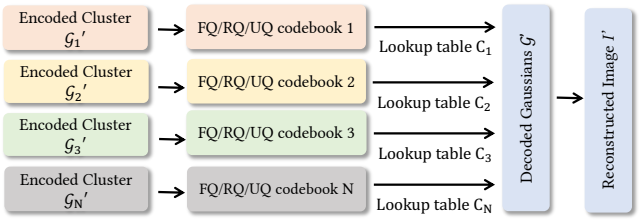

Figure 3: Encoded cluster dequantization pipeline.

where $\boldsymbol{m}_k$ is the centroid of cluster C_k. The number of clusters K is selected manually with K between the range of 4 to 16, based on the number of primitives, to keep the balance between reconstruction fidelity and compression efficiency. Clustering is performed once after the initial training phase converges, and the resulting cluster assignment $\{k_i\}$ is shared across all three parameter-specific quantizers, ensuring consistent grouping throughout the entire compression pipeline. As shown in Fig. 2, for each cluster C_k, we train three independent codebooks, one per parameter type, to exploit the distinct statistical properties of each attribute: For each cluster, we model three components: (1) **position** $\boldsymbol{\mu}$ using 16-bit floating-point QAT with a straight-through FakeVQ estimator, (2) **rotation-scale RS** using a Uniform Vector Quantization (UQ) codebook, and (3) **color c** using a Residual Vector Quantization (RQ) codebook for fine-grained residual refinement. All codebooks are trained end-to-end by minimizing a joint loss combining image reconstruction and vector quantization terms: $\mathcal{L} = \mathcal{L}_{\text{recon}}(\hat{I}, I) + \mathcal{L}_{\text{vq}}$ where $\mathcal{L}_{\text{recon}}$ is a weighted combination of a pixel-wise L1 loss in image space and a perceptual L1 loss in feature space, with $\lambda = 0.7$, and $\mathcal{L}_{\text{vq}}$ is the commitment loss that regularizes encoder outputs toward their assigned codebook entries across all active codebooks.

As illustrated in Figure 3, each compressed cluster C_k is decoded independently by looking up its three dedicated codebooks—FP16 for position $\boldsymbol{\mu}$, UQ for rotation-scale **RS**, and RQ for color **c**—using the stored quantization indices $z_i^{\mathbf{RS}}$ and $z_i^{\mathbf{c}}$ to reconstruct each primitive $\hat{G}_i$. The decoded primitives from all K clusters are then composed into a single Gaussian set $\mathcal{G}' = \bigcup_{k=1}^{K} \hat{C}_k$ for rendering. To further reduce bitrate, we also apply Partial Bits Back Encoding [Zhang et al. 2025a] to the quantization index tables.

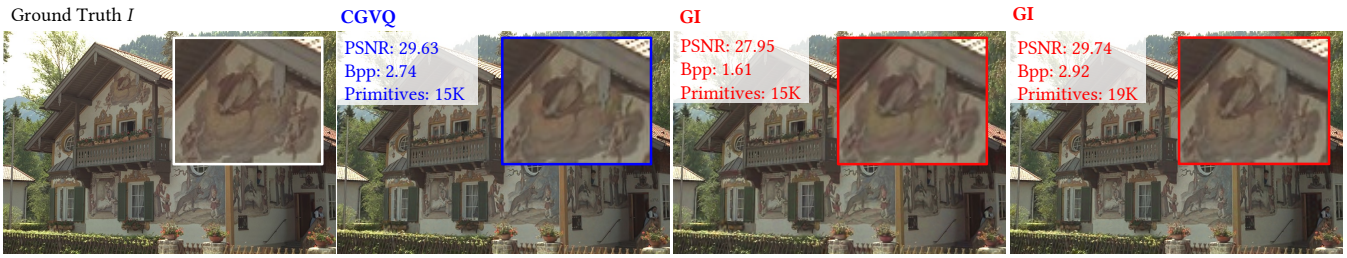

Figure 4: Qualitative Comparison of Reconstruction Fidelity. From left to right: (1) Ground Truth I; (2) CGVQ reconstruction with 15K primitives; (3) GI baseline with 15K primitives; (4) GI baseline with 19K primitives. Source image © Kodak.

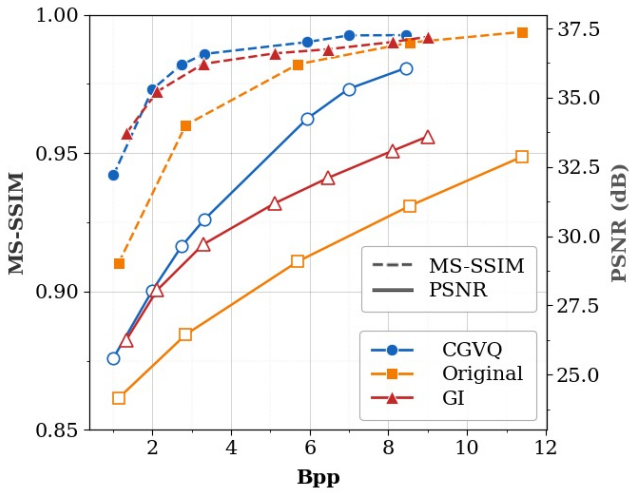

Figure 5: Visual Quality: Rate–distortion performance in terms of PSNR and SSIM versus Bpp on the Kodak dataset, where original refers to uncompressed Gaussian representation.

3 Evaluation

In our experiments, K is fixed to 16, which provides a suitable trade-off for scenes containing 10K to 25K Gaussian primitives. **Rate-Distortion Efficiency:** Compared with the GI baseline [Zhang et al. 2025a], CGVQ consistently demonstrates improved rate-distortion performance on the Kodak dataset. As shown in Fig. 5, CGVQ achieves higher PSNR and MS-SSIM than GI over the evaluated bitrate range, indicating better reconstruction quality at comparable bitrates. This advantage becomes more evident in the high-quality regime. In particular, for reconstruction quality above approximately 30 dB, CGVQ maintains a clear gain in PSNR while also preserving stronger structural similarity, suggesting a more favorable rate-distortion trade-off overall. **Parameter Efficiency:** We further assess parameter efficiency under a fixed Gaussian primitives. As shown in Fig. 4, with the same 15K Gaussian primitives, CGVQ improves PSNR by 1.68 dB↑ relative to GI. In addition, CGVQ at 15K points attains visual quality comparable to GI at 19K points, corresponding to a 21%↓ reduction in primitive count. The result of

Table 1: Performance: Encoding time and decoding FPS under different cluster settings (primitives 20K)

K	PSNR↑	Enc. FPS↑	Dec. FPS↑
1	29.33	2.91×10^{-2}	133.3
4	30.71	5.92×10^{-3}	74.6
8	30.90	3.58×10^{-3}	53.5
16	31.18	1.9×10^{-3}	33.3

CGVQ stems from the principle that narrower parameter distributions are inherently easier to quantize accurately. By partitioning 2D Gaussian primitives into K homogeneous clusters based on appearance and anisotropy similarity, we effectively constrain the variance of parameters within each group. This allows our localized multi-codebook design to capture high-frequency details more precisely than global codebook approaches. **Performance Trade Off:** Shown in Table 1 the number of clusters reduces both encoding and decoding speed, with encoding FPS dropping from 2.91×10^{-2} to 1.9×10^{-3} and decoding FPS from 133.3 to 33.3 as the cluster count increases from 1 to 16. This shows that finer clustering introduces a clear runtime overhead, highlighting a trade-off between compression performance and computational efficiency. To conclude, CGVQ improves 2D Gaussian-based image compression by enabling more efficient quantization through cluster-specific codebooks, while introducing a clear trade-off between compression quality and computational efficiency.

References

Song Han, Huizi Mao, and William J. Dally. 2016. Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). http://arxiv.org/abs/1510.00149

Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2024. 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 32, 11 pages. doi:10.1145/3641519.3657428

Lars Mescheder, Wei Dong, Shiwei Li, Xuyang Bai, Marcel Santos, Peiyun Hu, Bruno Lecouat, Mingmin Zhen, Amaël Delaunoy, Tian Fang, Yanghai Tsin, Stephan R. Richter, and Vladlen Koltun. 2026. Sharp Monocular View Synthesis in Less Than a Second. arXiv:2512.10685 [cs.CV] https://arxiv.org/abs/2512.10685

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2018. Neural Discrete Representation Learning. arXiv:1711.00937 [cs.LG] https://arxiv.org/abs/1711.00937

David R. Walton, Rafael Kuffner Dos Anjos, Sebastian Friston, David Swapp, Kaan Akşit, Anthony Steed, and Tobias Ritschel. 2021. Beyond blur: real-time ventral metamers for foveated rendering. 40, 4, Article 48 (July 2021), 14 pages. doi:10.

1145/3450626.3459943

Chaoli Wang, Hongfeng Yu, and Kwan-Liu Ma. 2010. Application-Driven Compression for Visualizing Large-Scale Time-Varying Data. *IEEE Computer Graphics and Applications* 30, 1 (2010), 59–69. doi:10.1109/MCG.2010.3

Pingyu Wu, Kai Zhu, Yu Liu, Liming Zhao, Wei Zhai, Yang Cao, and Zheng-Jun Zha. 2024. Improved Video VAE for Latent Video Diffusion Model. arXiv:2411.06449 [cs.CV] https://arxiv.org/abs/2411.06449

Xinjie Zhang, Xingtong Ge, Tongda Xu, Dailan He, Yan Wang, Hongwei Qin, Guo Lu, Jing Geng, and Jun Zhang. 2025a. GaussianImage: 1000 FPS Image Representation and Compression by 2D Gaussian Splatting. In *Computer Vision – ECCV 2024*, Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (Eds.). Springer Nature Switzerland, Cham, 327–345.

Yunxiang Zhang, Bingxuan Li, Alexandr Kuznetsov, Akshay Jindal, Stavros Diolatzis, Kenneth Chen, Anton Sochenov, Anton Kaplanyan, and Qi Sun. 2025b. Image-GS: Content-Adaptive Image Representation via 2D Gaussians. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers (SIGGRAPH Conference Papers '25)*. Association for Computing Machinery, New York, NY, USA, Article 102, 11 pages. doi:10.1145/3721238.3730596