INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC 1/SC 29/WG 4
MPEG VIDEO CODING

ISO/IEC JTC 1/SC 29/WG 4 **m68229**
July 2024, Sapporo

Title: [INVR]EE2.2: Compression of 4D Gaussian Splatting based on Video Codec and Gaussian Pruning
Source: Sungkyunkwan University (SKKU)
Authors: Jaeyeol Choi, Jun-Hyeong Park, Jong-Beom Jeong, Yeong Gyu Kim, Eun-Seok Ryu (SKKU)

Abstract

This document suggests a compression pipeline for 4D gaussian splatting (4D-GS)[1], a 3D Gaussian splatting model used to represent dynamic scenes. Feature embedding for spatial and temporal input data is compressed using quantization and VVC codec. Experiments are also conducted to apply inter-coding by temporally linking multiple feature grids in the same voxel. Additionally, the significance score-based pruning technique from LightGaussian[2] is modified to compress the canonical 3DGS, which corresponds to the deformed positions, rotations, and scaling values that consist 4D-GS.

1 Introduction

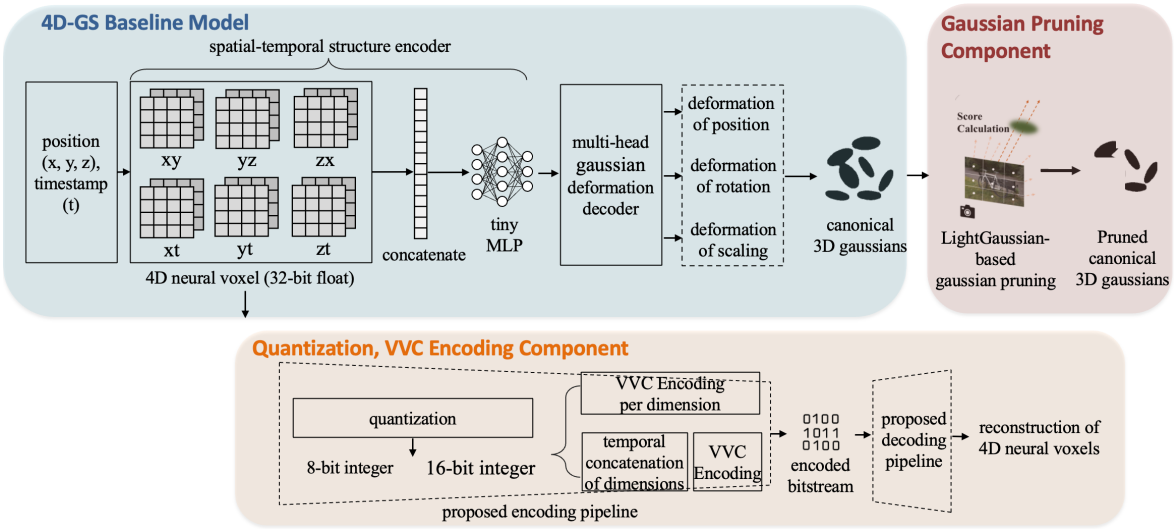

Figure 01. Proposed components of 4D-GS compressing pipeline

Recently, standardization work for 3D spatial representations enabling free-viewpoint rendering has been emerging. Among the models for implementing 3D-INVR, 3D gaussian splatting (3DGS)[3] is notable for its real-time rendering and high-quality novel view synthesizing performance. However, 3DGS requires more than 100,000 gaussians to represent a scene, resulting in file size significantly larger than NeRF[4]-based methods, posing limitations in storage and transmission. To address this, this contribution introduces a compression technique for the most cited 4D-GS model[1] used in dynamic scene representation. Fig 01. shows the two key components of the proposed 4D-GS compression: (1) application of video codec on feature embedding, (2) gaussian pruning based on projection contribution. This compression method reduces the model size by over 35% compared to the original 4D-GS, while nearly maintaining the quality of novel view synthesis for dynamic 3D spatial representations.

2 4D Gaussian Splatting Compression

2.1 Explanation of 4D Gaussian Splatting

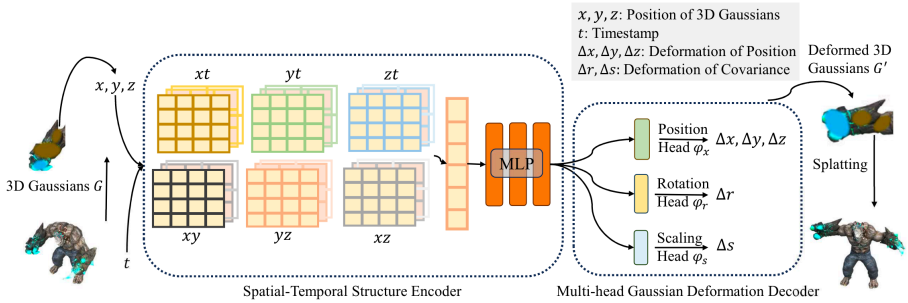

Figure 02. 4D gaussian splatting network [1]

The 4D-GS[1] model, introduced at CVPR 2024, consists of two main components: a spatial-temporal structure encoder and a Gaussian deformation decoder. The spatial-temporal structure encoder processes the 3D spatial coordinates and time value of the input points to obtain embeddings. It includes multi-resolution xy, yz, zx, xt, yt, and zt voxels for feature representation, followed by multiplication, concatenation scheme, and a tiny MLP that encodes the final embedding. The Gaussian deformation decoder consists of MLPs that output the deformation of position, rotation, and scaling corresponding to the canonical 3DGS from the embeddings. In essence, the 4D-GS[1] model maps coordinates of all timestamp to a single canonical 3DGS network.

2.2 Compression of 4D Neural Voxel based on Quantization and Video Codec

As shown in Fig 01, the spatial-temporal structure encoder maps the 4D input, consisting of position and time values, to 6 × L trainable voxel grids. This is because there are six types of 3D voxel grid (${}_4C_2 = 6$) and each type has L resolution levels. Each voxel is structured as a 3D tensor, which can be interpreted as having planes of size height × width in H dimensions. Consequently, the shape of the 4D neural voxel can be defined as [6 × L, H, height, width].

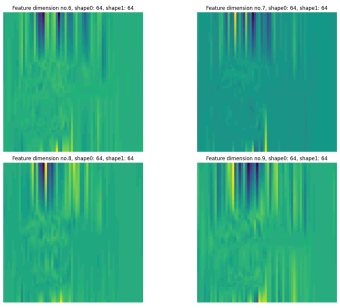

Figure 03. Visualization of some trained feature grids of *Mirror* sequence

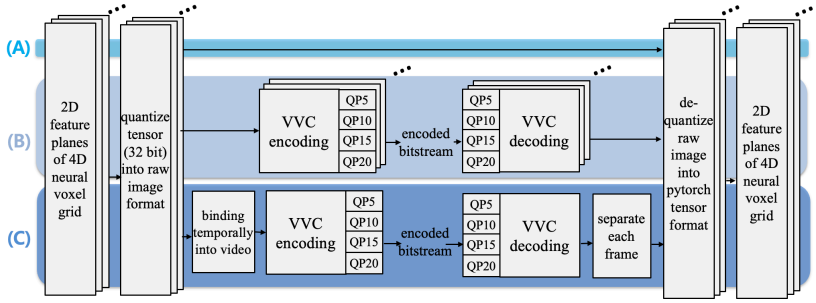

Figure 04. Experimental pipeline

Fig. 03 visualizes the values of feature grids in specific voxels from the trained 4D-GS using *Mirror* sequence. The observation showed that features in adjacent positions within a plane have similar values, and features in similar positions across multiple dimensions also have similar values. Based on this observation, we applied a video codec to compress multiple feature planes. Specifically, we experimented with connecting multiple feature planes of the same voxel along the time axis to apply inter coding. This approach is expected to improve the compression ratio while reducing the number of required decoders.

$$\hat{x} = \frac{2^n - 1}{M - m} \times (x - m)$$

$\hat{x}$: Value after quantization M : Maximum value

n : Target bit m : Minimum value

Fig. 04 illustrates the experimental flow to verify the effectiveness of this method. Experiment (A) only involves quantization and restoration, where 32-bit floats were converted to 8-bit or 16-bit as per the above formula. This not only reduce data size but also made the data suitable for video codec application. Following this, we compared the results of applying intra video codec to each individual feature grid (Experiment B) and connecting feature grids of multiple dimensions as frames in a video to apply the video codec (Experiment C).

2.3 Compression of Canonical 3DGS based on Gaussian Pruning

This section presents the compression method for the canonical 3DGS, one of the components of 4D-GS, as shown on the right side of Fig. 01.

$$\mathrm{GS_j} = \sum_{i=1}^{MHW} \mathbb{1}(\mathrm{G}(\boldsymbol{X}_j), r_i) \cdot \sigma_j \cdot \gamma(\boldsymbol{\Sigma}_j)$$

M : number of gaussians $\quad \sigma_j$: opacity of guassian j

H,W : height, width $\quad \gamma(\boldsymbol{\Sigma}_j)$: 3D volume of guassian j

LightGaussian[2] proposed three methods for compressing trained 3DGS[3], one of which is the Gaussian pruning and recovery process. Instead of simply pruning Gaussians which have low opacity, it calculates a global significance score ($\mathrm{GS_j}$) for each Gaussian based on how much it intersects with rays originating from training views, combining this with opacity and 3D volume to perform pruning. This contribution implemented Gaussian pruning and recovery within the 4D-GS training and rendering pipeline, making it applicable to prune based on deformed positions, deformed rotations, and deformed scaling. As a result, it reduces the size of the canonical 3DGS in 4DGS, which constitutes about 50% of the 4D-GS model.

3 Experiments

This section evaluates the performance of the two proposed 4D-GS compression components from Section 2 through experiments. The baseline used was the original 4D-GS without any compression. The experimental conditions were as follows:

- (A): 8-bit and 16-bit quantization applied
- (B): 16-bit quantization followed by VVC codec applied to individual 2D planes
- (C): 16-bit quantization followed by VVC codec applied to multiple planes forming a 3D voxel, connected along the time axis
- (D): (B) + Gaussian pruning based on LightGaussian module
- (E): (C) + Gaussian pruning based on LightGaussian module

The experiments were conducted on six dynerf[5] datasets. These datasets were chosen because they are commonly used in dynamic multiview radiance fields research, making them ideal for comparison with other methods. The 4D-GS was trained and rendered using the default configuration. When using the LightGaussian module, the pruning percentage was set to 0.2. For video codec compression, the reference software VTM for the VVC codec was used, setting the feature grid to the YUV400 format with only the Y component. In the case of inter coding, the low delay mode was used, with one I-frame and the rest set as P-frames.

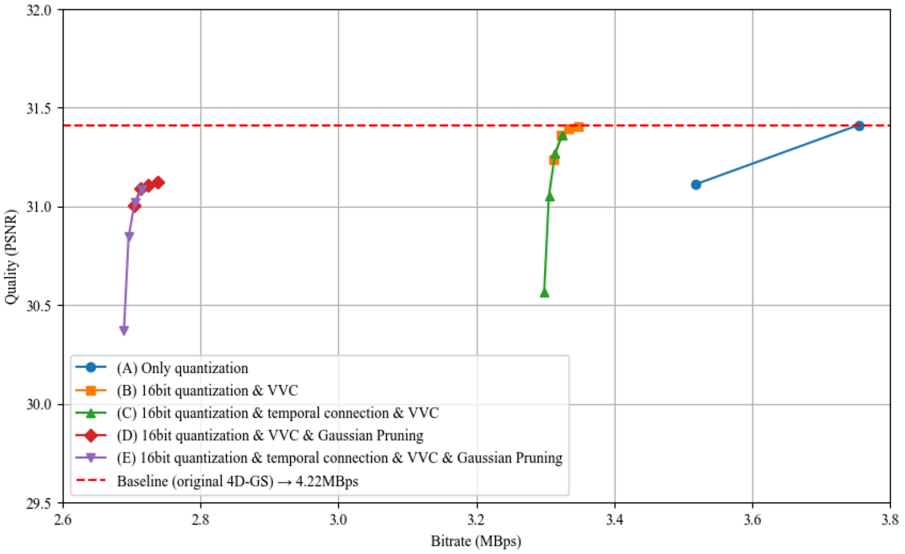

Figure 05. RD-curve result (PSNR)

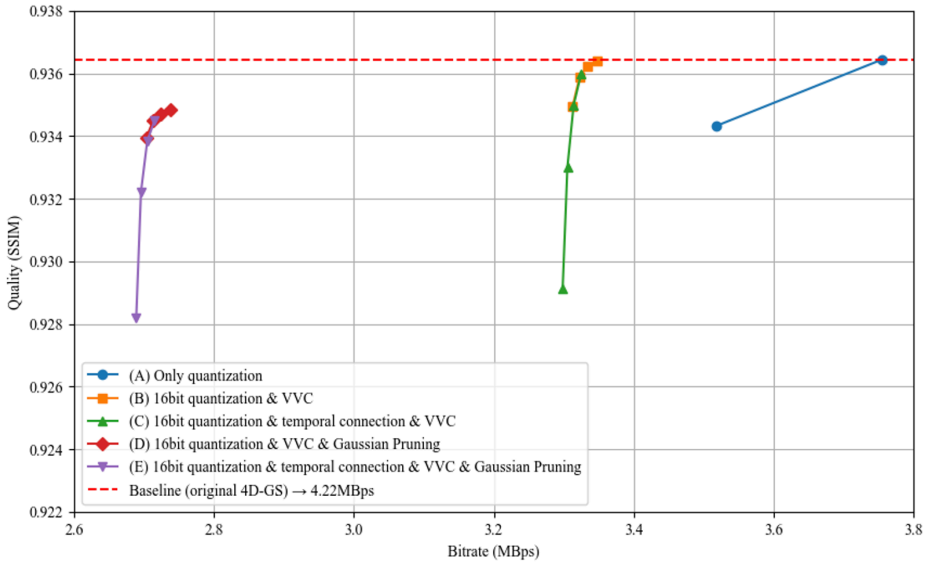

Figure 06. RD-curve result (SSIM)

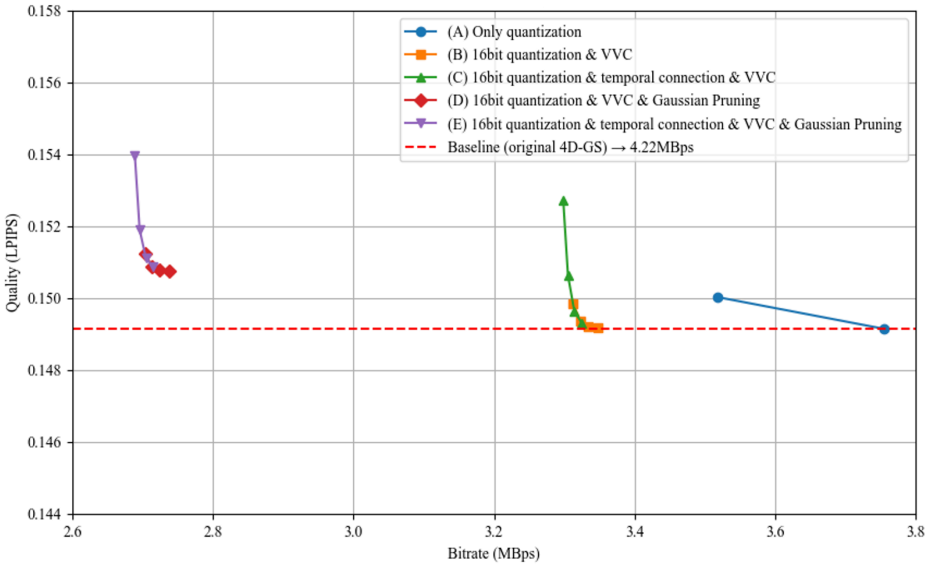

Figure 07. RD-curve result (LPIPS)

Figures 5, 6, and 7 show the average bitrate and rendering quality across the six datasets for each experimental condition. Analyzing the results, the baseline 4D-GS without compression had a bitrate of 4.22 MB per second at 30fps. Compressing the 4D neural voxel reduced the bitrate to 3.3 MBps, and further applying Gaussian pruning reduced it to 2.7 MBps, achieving a 36% reduction in size. In terms of quality, performing quantization alone on the 4D neural voxel (Experiment A) resulted in no quality loss compared to the baseline. Applying the VVC codec or Gaussian pruning to the 4D neural voxel maintained quality close to the original while improving compression rates. However, when inter coding was applied by connecting planes temporally, there was a quality degradation despite the advantage in reduced number of decoders.

Table 01. Comparison of novel view rendering quality and model file size

	PSNR	**SSIM**	**LPIPS**	# of gaussians in canonical 3DGS	feature voxel bitrate (MBps)	**full model bitrate (MBps)**
Baseline (original 4D-GS)	31.41	0.9364	0.1492	124504.5	0.95	**4.22**
Ours (High)	31.12	0.9348	0.1508	124504.5	0.04	**3.32**
Ours (Low)	31.01	0.9339	0.1512	99603.2	0.03	**2.69**

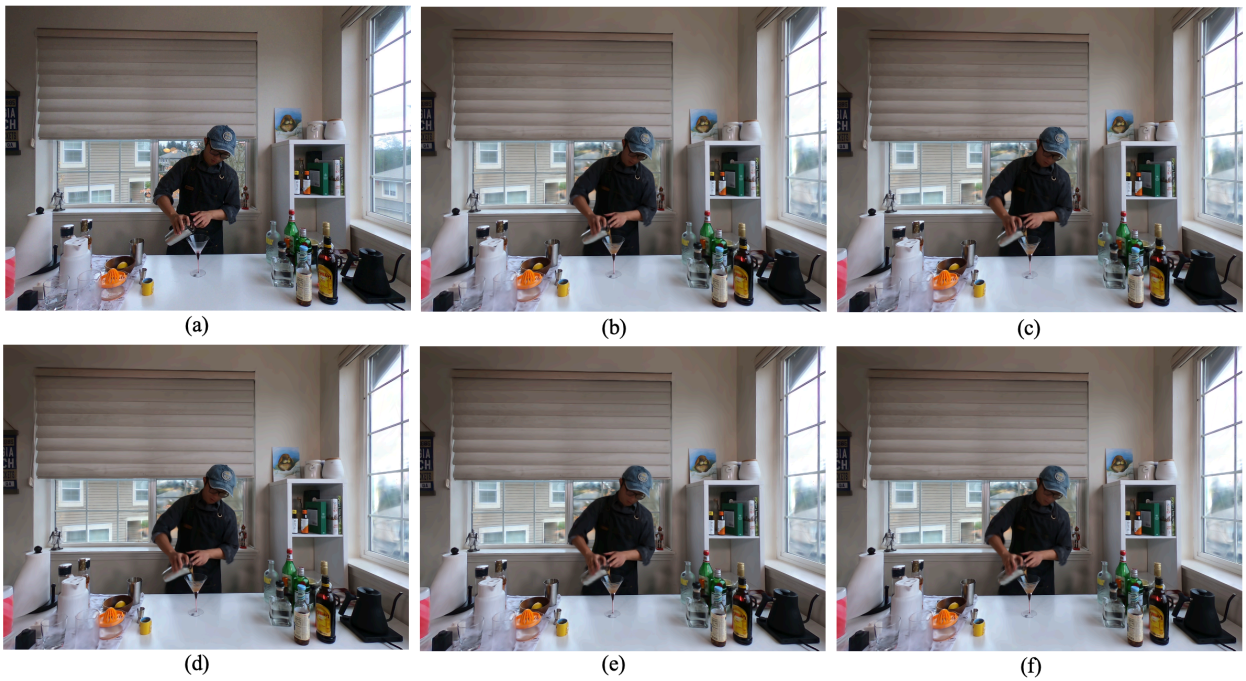

Figure 08. Rendered test views (a): ground truth, (b): baseline (original 4D-GS), (c): quantization & VVC, (d): quantization & temporal connection & VVC, (e): gaussian pruning & quantization & VVC, (f): gaussian pruning & quantization & temporal connection & VVC

Fig. 07 displays the performance changes compared to the baseline for two experimental conditions. Ours (high) refers to when only feature voxel compression is applied. The performance difference compared to the uncompressed state is very low. Ours (low) refers to when both inter coding of feature voxels and LightGaussian-based Gaussian pruning are applied. Fig. 08 shows the results of the *coffee_martini* sequence. Even after compression and reconstruction, non-Lambertian effects and shadows are well preserved, although there were some limitations in restoring motion areas.

4 Conclusion

In this document, we introduced a method to effectively reduce the bitrate of 4D-GS while maintaining test view rendering performance and shared the experimental results. Moving forward, we will compare this proposed pipeline with other 3D-INVR methods under the INVR common test conditions (CTC). We recommend modularizing and applying the coding-based feature encoding and global significance score-based Gaussian pruning methods for future 3DGS-based INVR activities.

5 References

[1] Wu, Guanjun, et al. "4d gaussian splatting for real-time dynamic scene rendering." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024.

[2] Fan, Zhiwen, et al. "Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps." arXiv preprint arXiv:2311.17245, 2023.

[3] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian Splatting for Real-Time Radiance Field Rendering," ACM Transactions on Graphics, vol. 42, no. 4, 2023.

[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: representing scenes as neural radiance fields for view synthesis," Commun. ACM, vol. 65, no. 1, pp. 99–106, Jan. 2022.

[5] Li, Tianye, et al. "Neural 3d video synthesis from multi-view video." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.