CAT-3DGS Pro: A New Benchmark for Efficient 3DGS Compression

Yu-Ting Zhan[1], He-bi Yang[1], Cheng-Yuan Ho[1], Jui-Chiu Chiang[2], and Wen-Hsiao Peng[1]
[1]National Yang Ming Chiao Tung University, Taiwan
[2]National Chung Cheng University, Taiwan

***Abstract*—3D Gaussian Splatting (3DGS) has shown immense potential for novel view synthesis. However, achieving rate-distortion-optimized compression of 3DGS representations for transmission and/or storage applications remains a challenge. CAT-3DGS introduces a context-adaptive triplane hyperprior for end-to-end optimized compression, delivering state-of-the-art coding performance. Despite this, it requires prolonged training and decoding time. To address these limitations, we propose CAT-3DGS Pro, an enhanced version of CAT-3DGS that improves both compression performance and computational efficiency. First, we introduce a PCA-guided vector-matrix hyperprior, which replaces the triplane-based hyperprior to reduce redundant parameters. To achieve a more balanced rate-distortion trade-off and faster encoding, we propose an alternate optimization strategy (A-RDO). Additionally, we refine the sampling rate optimization method in CAT-3DGS, leading to significant improvements in rate-distortion performance. These enhancements result in a 46.6% BD-rate reduction and 3× speedup in training time on BungeeNeRF, while achieving 5× acceleration in decoding speed for the *Amsterdam* scene compared to CAT-3DGS.**

***Index Terms*—3D Gaussian Splatting, Rate-Distortion Optimization**

I. Introduction

3D Gaussian Splatting (3DGS) [1] has emerged as a promising 3D scene representation, excelling in novel view synthesis within differentiable rendering frameworks due to its real-time, high-quality rendering. Despite its strengths, the redundancy inherent in 3DGS has spurred research efforts to develop more compact representations [2]–[13]. However, the efficient transmission of compressed 3DGS remains an overlooked challenge, requiring entropy coding and rate-distortion optimization.

Recently, the rate-distortion-optimized compression for 3DGS [14]–[20] started to gain attention. Its aim is to balance the bit rate and rendering quality in an end-to-end manner. Specifically, HAC [15], ContextGS [17], HEMGS [19], and CAT-3DGS [18] utilize the ScaffoldGS [7] representation, adopting an anchor-based approach where each anchor point encodes a group of Gaussian primitives into a latent feature vector and signals their geometry offsets relative to the anchor position, as well as a 6-dimensional scaling factor. The latent feature vector is further quantized by a scalar quantizer for entropy coding. These early works leverage the hyperprior idea from learned image compression [21] to model their coding probabilities. In forming the hyperprior, HAC [15] constructs multi-scale binary hash grids. ContextGS learns a quantized feature for each anchor. HEMGS adopts the pre-trained PointNet++ and an instance-aware hash grid to generate the hyperprior. Notably, to utilize spatial redundancy for better coding efficiency, both ContextGS and HEMGS additionally resort to spatial autoregressive contextual coding. ContextGS hierarchically organizes anchor points into three successive coding levels, while HEMGS compresses anchor points in raster order by referring to their decoded neighbors. Rather than explicitly organizing anchor points, CAT-3DGS projects them onto multi-scale triplanes as the hyperprior, and encodes these triplanes using spatial autoregressive models. Although achieving state-of-the-art coding results, CAT-3DGS exhibits long training and decoding time.

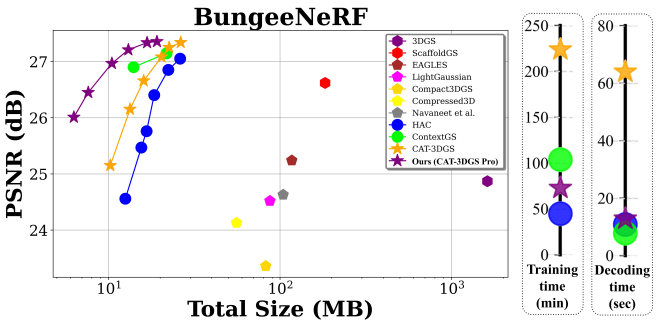

Fig. 1: Comparison of rate-distortion performance, training time on the BungeeNeRF dataset, and decoding time for the *Amsterdam* scene.

This paper introduces CAT-3DGS Pro, an enhanced version of CAT-3DGS that addresses its long training time and high decoding complexity while boosting its coding performance through an improved training strategy. The key advancements include the adoption of a PCA-guided vector-matrix (VM) hyperprior, which reduces the computational complexity of CAT-3DGS while retaining its capability to represent the 3D space effectively. This VM hyperprior significantly accelerates both training and decoding processes. To further increase parallelism, we additionally divide a high-resolution hyperprior plane into 4 smaller ones for separate entropy encoding/decoding. We also implement an alternative optimization technique to strike a better balance between the rate-distortion performance and encoding speed. These improvements culminate in state-of-the-art compression performance, as shown in Figure1. Remarkably, CAT-3DGS Pro achieves a 5× speedup in decoding speed on the *Amsterdam* scene and a 3× speedup in training time on BungeeNeRF compared to CAT-3DGS.

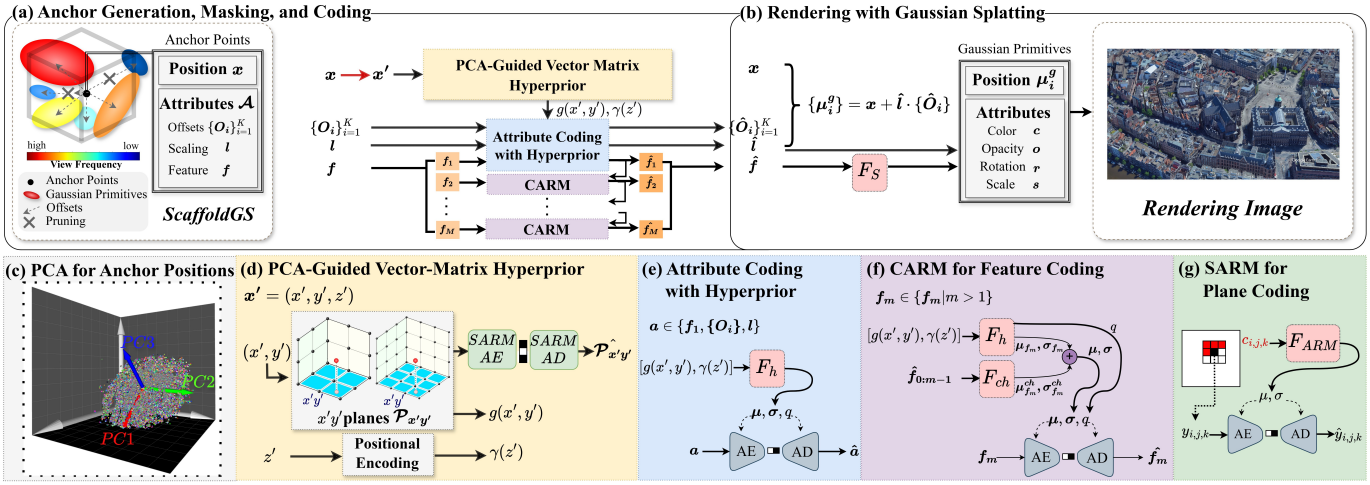

Fig. 2: System overview of our CAT-3DGS Pro. CARM: Channel-wise Autoregressive Models. SARM: Spatial Autoregressive Models.

II. Proposed Method: CAT-3DGS Pro

Figure 2 depicts the architecture of our CAT-3DGS Pro. The encoding process for a 3D scene begins with constructing a Scaffold-GS representation based on anchor points. These anchor points are characterized by their positions $\boldsymbol{x} \in \mathbb{R}^3$ and attributes $\mathcal{A}$, which include the latent features $\boldsymbol{f} \in \mathbb{R}^{50}$, offsets $\{\boldsymbol{O_i} \in \mathbb{R}^3\}_{i=1}^K$, and scaling factors $\boldsymbol{l} \in \mathbb{R}^6$ (part (a)). To entropy encode/decode the attributes of the anchors, a PCA-guided VM hyperprior (parts (c)(d)) is queried using the PCA-transformed coordinates $\boldsymbol{x'} = (x', y', z')$ of an anchor point to retrieve $g(x', y'), \gamma(z')$ for decoding the probability distributions of its attributes (part (e)). For encoding/decoding the latent features $\boldsymbol{f}$, we additionally leverage a channel-wise contextual coding scheme [18] to exploit the *intra correlation* among their components (part (f)). The VM hyperprior consists of multi-scale planes $\mathcal{P}_{x'y'}$ aligned with the two principal components bearing the largest eigenvalues, alongside a positionally encoded vector in the direction of the remaining principal component. These multi-scale planes $\mathcal{P}_{x'y'}$ also require encoding/decoding. They are quantized and encoded using a spatial autoregressive model (part (g)). In summary, the compressed bitstream for CAT-3DGS Pro includes $\mathcal{P}_{x'y'}$, the anchors' attributes $\mathcal{A}$ and positions $\boldsymbol{x}$, the binary mask, and the network weights. The binary mask implements a view frequency-aware masking mechanism [18] to skip Gaussian primitives with minimal impact on rendering quality. The anchors' positions $\boldsymbol{x}$ and network weights are represented in 16-bit and 32-bit floating-point formats, respectively, while the binary mask undergoes entropy coding. The rendering with Gaussian splatting simply follows Scaffold-GS [7].

A. PCA-Guided Vector-Matrix (VM) Hyperprior

For entropy encoding/decoding anchor attributes, we employ a PCA-guided VM hyperprior to predict their probability distributions. In Figure 2 (c), PC1, PC2, and PC3 are the three principal components extracted from anchors' locations $\boldsymbol{x}$, where the majority of anchors are predominantly distributed along PC1 and PC2, which have the highest eigenvalues. With CAT-3DGS Pro, we reduce the multi-scale triplane structure of CAT-3DGS to multi-scale planes $\mathcal{P}_{x'y'}$ oriented along PC1 and PC2. We hypothesize that the attributes of most anchors can already be predicted well by querying and decoding the corresponding latent features $g(x', y')$ derived from these multi-scale planes (i.e. dense-grid representations). These planes of various scales ensure that both the coarse and fine details are well captured. To distinguish between the $x'y'$-planes of different scales, we augment $\boldsymbol{\mathcal{P}}_{x'y'}$ with an upsampling scale r, denoted as $\boldsymbol{\mathcal{P}}_{r,x'y'} \in \mathbb{R}^{ch \times rB \times rB}$, where ch denotes the number of channels, r represents the upsampling scales, and B indicates the spatial resolution of the $x'y'$ planes at the base scale (i.e., $r = 1$). To address the information not captured by the multi-scale planes, we further introduce a positionally encoded vector $r(z')$ that takes the z' coordinate along PC3 as input. The attributes $\mathcal{A}$ of anchor points are treated as a vector-valued function defined in 3D space. Conceptually, our hyperprior VM approximates this function by a vector-matrix decomposition.

To retrieve the plane feature $g(x', y')$ for an anchor point at $\boldsymbol{x'}$, the coordinates (x', y') are first projected onto each 2D plane $\boldsymbol{\mathcal{P}}_{r,x'y'}$. The resulting projected 2D coordinates are denoted by $\pi_r(x', y')$. If $\pi_r(x', y')$ produces fractional coordinates, interpolation is performed between the nearest integer grid points using an interpolation kernel ψ. In symbols, we have $\psi(\boldsymbol{\mathcal{P}}_{r,x'y'}, \pi_r(x', y'))$. The interpolated features obtained from all planes are then concatenated to yield $g(x', y')$:

$$g(x', y') = \bigcup_r \psi(\boldsymbol{\mathcal{P}}_{r,x'y'}, \pi_r(x', y')). \tag{1}$$

Likewise, to get $\gamma(z')$, we transform z' into a sequence of sinsodial Fourier features $\gamma(z')$ across $L = 10$ frequencies:

$$\gamma(z') = \left(\sin(2^0 \pi z', \cos(2^0 \pi z'), \ldots, \sin(2^{L-1} \pi z'), \cos(2^{L-1} \pi z')\right). \tag{2}$$

Unlike the costly dense-grid representation for $\boldsymbol{\mathcal{P}}_{x'y'}$, which can effectively represent high-frequency signals, this Fourier feature vector is designed to represent smoother signal variations along the z' axis.

Finally, to entropy encode (or decode) the quantized attributes $\hat{\boldsymbol{a}} \in \{\hat{\boldsymbol{f}}_{\boldsymbol{1}}, \{\hat{\boldsymbol{O}}_{\boldsymbol{i}}\}, \hat{\boldsymbol{l}}\})$, we concatenate the plane feature $g(x', y')$ and Fourier features $\gamma(z')$. These concatenated features are then fed into the MLP decoder F_h to predict their means, variances, and quantization step size. That is, $(\boldsymbol{\mu}, \boldsymbol{\sigma}, q) = F_h([g(x', y'), \gamma(z')])$ (Figure 2 (e)), where $[\cdot]$ denotes the concatenation operation. Notably, each of these attributes is assumed to follow a Gaussian distribution, with their coding probabilities given by

$$p(\hat{\boldsymbol{a}}|g(x', y'), \gamma(z'))) = \int_{\hat{\boldsymbol{a}}-\frac{q}{2}}^{\hat{\boldsymbol{a}}+\frac{q}{2}} \mathcal{N}(\boldsymbol{a}; \boldsymbol{\mu}, \boldsymbol{\sigma})\, d\boldsymbol{a}. \tag{3}$$

B. Spatial Autoregressive Models (SARM) for Plane Coding

Given that $\boldsymbol{\mathcal{P}}_{r,x'y'}$ functions as part of the hyperprior, and are designed to capture the *inter-correlations* among the attributes of anchor points in 3D space, we apply a spatial autoregressive model F_{ARM} to encode the multi-scale planes in $\boldsymbol{\mathcal{P}}_{r,x'y'}$. The model F_{ARM} is shared across scales r and channels ch. For entropy encoding (and decoding) a grid point $y_{i,j,k}$ in $\boldsymbol{\mathcal{P}}_{r,x'y'}$ within channel k, a context is constructed from the decoded grid points in its neighborhood, defined as $c_{i,j,k} = [\hat{y}_{i-1,j-1:j+1,k}; \hat{y}_{i,j-1,k}]$. Using this context, F_{ARM} predicts the Laplace parameters $(\mu_{i,j,k}, \sigma_{i,j,k})$ that model the distribution of $y_{i,j,k}$ (Figure 2 (g)).

C. Rate-distortion Optimization

The training of CAT-3DGS Pro involves minimizing the rate-distortion cost $L_{\text{Scaffold}} + \lambda_r L_{\text{rate}}$ together with a masking loss $\lambda_m L_m$:

$$L = L_{\text{Scaffold}} + \lambda_r L_{\text{rate}} + \lambda_m L_m, \tag{4}$$

where we follow Scaffold-GS [7] to evaluate L_{Scaffold}, which includes the distortion between the original and rendered images as well as a regularization term applied to the scales s of Gaussian primitives. L_m is the mask loss adopted from CAT-3DGS [18] to regularize the view frequency-aware masking. L_{rate} indicates the number of bits needed to signal the hyperprior and the anchors' attributes:

$$L_{\text{rate}} = \frac{1}{N(50+6+3K)}(L_{\text{rate}}^{\boldsymbol{\mathcal{A}}} + \lambda_{\text{tri}} L_{\text{rate}}^{\boldsymbol{\mathcal{P}}}), \tag{5}$$

where $L_{\text{rate}}^{\boldsymbol{\mathcal{A}}} = -\sum_{\hat{\boldsymbol{a}}} log_2\, p(\hat{\boldsymbol{a}})$ is the estimated bit rate of the anchors' attributes, $L_{\text{rate}}^{\boldsymbol{\mathcal{P}}} = -\sum_{\hat{y}} log_2\, p(\hat{y})$ is the hyperprior's bit rate, and $N(50+6+3K)$ is the total number of the parameters of anchors' attributes. Following Scaffold-GS, we set $K = 10$.

Recognizing that the estimation of the bit rate in SARM significantly prolongs the training time, we propose an alternate rate-distortion optimization strategy, termed A-RDO. Inspired by NVRC [22], A-RDO evaluates the rate term $L_{\text{rate}}^{\boldsymbol{\mathcal{P}}}$ periodically with an interval of $T = 4$ steps, while the other loss terms are evaluated at every step. Our A-RDO significantly reduces training time while achieving better rate-distortion performance (Secs. III-C1 and III).

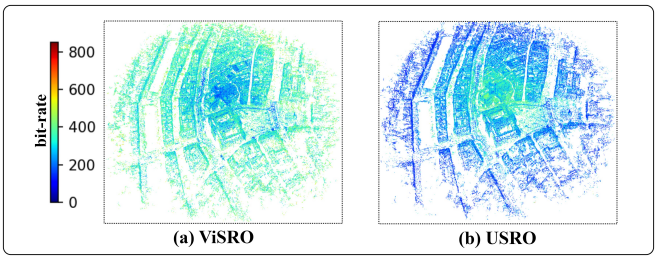

Fig. 3: Visualization of the bit-rate distribution on the *Amsterdam* scene for ViSRO and USRO. The bit rate indicates the number of bits needed to represent the anchor attributes of each anchor point.

Additionally, we observe that the training strategy in CAT-3DGS, known as Visible Sampling Rate Optimization (ViSRO), randomly samples 5% of the anchor points that are visible in the training views to estimate the bit rate of the anchor attributes $L_{\text{rate}}^{\boldsymbol{\mathcal{A}}}$. This approach results in a higher bit rate for representing the attributes of anchor points located in less frequently observed (peripheral) regions, which significantly limits the rate-distortion performance (Figure 3 (a)). It is crucial to note that the compressed bitstream must encode all the anchor points, whether they are located in peripheral or central regions. To tackle this problem, we have revised the training strategy to evaluate the rate term independently of an anchor point's visibility.The bit rate estimation is no longer restricted to visible anchor points. This new training strategy is called Uniform Sampling Rate Optimization (USRO). Figure 3 compares the bit-rate distributions of ViSRO and USRO. The results demonstrate that with USRO, central anchor points utilize a relatively higher bit rate than peripheral ones, leading to significantly improved rate-distortion performance (Sec. III-D2).

III. Experimental Results

A. Implementation Details

This section summarizes the implementation details. First, the spatial resolution B of the base-scale plane ($r = 1$) is determined proportionally based on the number of anchor points obtained after 10k training iterations, with $B = 128$ for the highest anchor count in all training scenes and $B = 64$ for the lowest one. Our multi-scale planes consists of only two scales, $r = 1, 2$, each with $ch = 8$, as opposed to 72 used in CAT-3DGS. To accelerate the decoding process, the high-resolution plane $\boldsymbol{\mathcal{P}}_{2,x'y'}$ is split into four non-overlapping, low-resolution sub-planes, each having a spatial resolution the same as $\boldsymbol{\mathcal{P}}_{1,x'y'}$. This setup results in a total of five planes that are decoded in parallel. For rate-distortion optimization, the rate parameter λ_r ranges from 0.002 to 0.04, and from 0.002 to 0.03 for BungeeNeRF, with λ_{tri} fixed at 10 and $\lambda_m = \max(10^{-3}, 0.3 \cdot \lambda_r)$.

B. Rate-Distortion Comparison

To demonstrate the rate-distortion performance of CAT-3DGS Pro, we compare it against several rate-distortion-

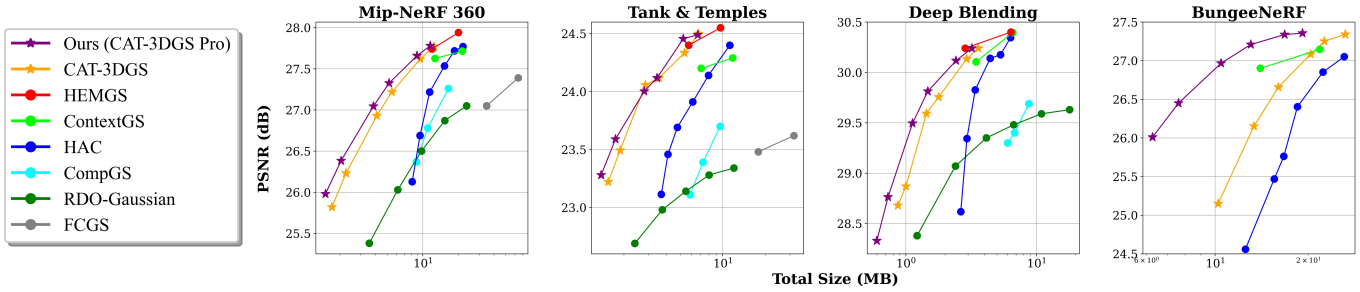

Fig. 4: Rate-distortion comparison of our CAT-3DGS Pro and several baseline methods.

TABLE I: Training time comparison on BungeeNeRF.

Method	CAT-3DGS	CAT-3DGS Pro	HAC	ContextGS
Time (min)	224.0	73.5	45.3	104.2

TABLE II: Comparison of decoding time and rendering throughput.

Method	Scene	Anchor Count (K)	Dec. Time (sec) ↓			Rendering Speed (FPS) ↑
			$\mathcal{P}$	$\mathcal{A}$	Total	
CAT-3DGS	*room*	36.5	11.9	2.3	14.2	127.0
	Amsterdam	533.0	47.4	17.0	64.4	83.4
CAT-3DGS Pro	*room*	30.8	1.0	1.1	2.1	132.7
	Amsterdam	421.6	2.2	10.9	13.1	110.0
HAC	*room*	228.8	-	-	6.6	103.1
	Amsterdam	483.5	-	-	11.3	77.9
ContextGS	*room*	119.4	-	-	2.6	92.2
	Amsterdam	360.3	-	-	8.4	113.1

optimized 3DGS compression schemes [14]–[20]. Following the common test protocol, we evaluate their coding performance on real-world scenes, including Mip-NeRF 360 [23], Tanks & Temples [24], Deep Blending [25], and BungeeNeRF [26]. As shown in Figure 4, our CAT-3DGS Pro achieves the state-of-the-art rate-distortion performance, demonstrating significant improvements over CAT-3DGS primarily due to the USRO training strategy (Sec. III-D2). While HEMGS performs comparably to our method at high rates, it requires the computationally expensive pre-trained PointNet++ for hyperprior modeling.

C. Complexity Comparison

1) Training Time: Table I compares the training time of several rate-distortion-optimized methods using BungeeNeRF, which is a particularly challenging large-scale dataset. Remarkably, CAT-3DGS Pro achieves a 3.1× speedup over CAT-3DGS. This improvement comes from two key advancements. First, A-RDO reduces the training time of CAT-3DGS (with the triplane hyperprior) by 25.6%, due to a reduced frequency of bit rate evaluations for SARM. Second, replacing the triplane hyperprior with our VM hyperprior further cuts training time by 55.8%, primarily due to lower encoding complexity and a significant reduction in hyperprior parameters from 17M to 0.6M.

2) Decoding Time and Rendering Throughput: Table II compares the decoding time and rendering throughput for two scenes: *Amsterdam* in BungeeNeRF, which has a high anchor count, and *room* in Mip-NeRF360, which features a low anchor count. In terms of decoding time, CAT-3DGS Pro achieves approximately a 5× speedup over CAT-3DGS on the *Amsterdam* scene, and demonstrates comparable decoding speed to HAC and ContextGS. However, CAT-3DGS Pro achieves much better rate-distortion performance than HAC and ContextGS. Additionally, CAT-3DGS Pro has dramatically reduced the decoding time for the multi-scale planes ($\mathcal{P}$) from 47.4s to 2.2s. This improvement is attributed to the VM-based hyperprior representation and the parallel decoding of the multi-scale planes. Furthermore, the VM-based hyperprior features a smaller memory footprint, allowing more anchor points to be decoded in a batch. This accounts for the decoding speedup in the attribute part ($\mathcal{A}$). Lastly, CAT-3DGS Pro achieves a higher rendering throughput than CAT-3DGS, as fewer anchor points are required to maintain similar rendering quality, primarily due to the USRO scheme. It tends to regularize the rate loss term by masking high-bit-rate anchor points while ensuring a minimal impact on distortion.

TABLE III: Analysis of A-RDO and the VM hyperprior on the *Amsterdam* scene.

Hyperprior	A-RDO	PSNR (dB)	Size (MB)		
			$\mathcal{P}$	$\mathcal{A}$	Total
Triplane (ch=72)		27.3	0.1	19.2	23.8
Triplane (ch=72)	✓	27.5	0.8	18.4	23.7
Triplane (ch=8)	✓	27.6	0.5	18.8	23.6
Ours, VM (ch=8)	✓	27.6	0.3	18.8	23.5
M (ch=8)	✓	27.4	0.2	20.0	24.6

D. Ablation Study

1) Impact of Hyperprior and A-RDO on Compression Performance: We investigate the impact of A-RDO and the VM hyperprior on compression performance, using the *Amsterdam* scene in BungeeNeRF as a representative example. Similar trends are also observed across the other scenes. As shown in Table III, applying A-RDO to CAT-3DGS Pro with a triplane-based hyperprior (Row 1 → Row 2) has a minimal effect on the total file size, but improves the rendering quality (PSNR). Interestingly, A-RDO increases the size of the multi-scale planes ($\mathcal{P}$) from 0.1M to 0.8M, but it does not significantly reduce the attribute size ($\mathcal{A}$). This inspires us to take the opposite approach by reducing the number of channels in the triplane hyperprior from 72 to 8 (Row 2 → Row 3). This

TABLE IV: BD-rate savings of USRO, with ViSRO as anchor.

Scheme	Mip-NeRF 360	Tanks & Temples	Deep Blending	BungeeNeRF
USRO	-15.9%	-9.8%	-2.5%	-45.6%

change results in negligible changes in both the rendering quality and total file size. However, it helps reduce the decoding and training complexity due to the lower channel count. Furthermore, replacing the triplane with our VM hyperprior (Row 3 $\rightarrow$ Row 4) producs a similar result, underscoring the advantage of adopting a simple hyperprior design. However, omitting the $\gamma(z')$ (denoted as M ($ch = 8$)) produces a noticeable rate-distortion loss.

2) Sampling Rate Optimization: Table IV presents the BD-rate savings of CAT-3DGS Pro under Uniform Sampling Rate Optimization (USRO). The results show that USRO consistently improves the rate-distortion performance across all the datasets. On the BungeeNeRF dataset, it achieves a 45.6% BD-rate reduction, due to its training views being concentrated in a small central region. Recall that with USRO, less-viewed anchor points are also taken into account during the training process.

IV. Conclusion

CAT-3DGS Pro extends CAT-3DGS with an enhanced and accelerated rate-distortion-optimized 3DGS compression framework, achieving state-of-the-art coding performance with faster training and decoding. Through the proposed PCA-guided vector-matrix hyperprior, which simplifies the triplane structure in CAT-3DGS, we significantly reduce parameter redundancy and coding complexity while preserving a similar ability to model the distribution of anchor attributes. Additionally, the alternative optimization strategy (A-RDO) enables a more balanced rate-distortion trade-off and accelerates training. Lastly, our approach further enhances compression performance by refining the sampling rate optimization.

For future work, since current methods primarily focus on the entropy coding of anchor attributes while position information is simply stored in a 16-bit floating-point format, developing a unified compression framework for both position and attribute data in the ScaffoldGS representation is a promising approach to further improving rate-distortion performance.

References

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

[2] M. S. Ali, M. Qamar, S.-H. Bae, and E. Tartaglione, "Trimming the Fat: Efficient Compression of 3D Gaussian Splats through Pruning," 2024, *arXiv:2406.18214*. [Online]. Available:https://arxiv.org/abs/2406.18214

[3] K. L. Navaneet, K. Pourahmadi Meibodi, S. A. Koohpayegani, and H. Pirsiavash, "Compact3D: Compressing Gaussian Splat Radiance Field Models with Vector Quantization," 2023. *arXiv:2311.18159*. [Online]. Available:https://arxiv.org/abs/2311.18159.

[4] Z. Fan, K. Wang, K. Wen, Z. Zhu, D. Xu, and Z. Wang, "LightGaussian: Unbounded 3D Gaussian Compression with 15x Reduction and 200+ FPS," 2023.

[5] S. Girish, K. Gupta, and A. Shrivastava, "EAGLES: Efficient Accelerated 3D Gaussians with Lightweight EncodingS," in *Proc. European Conference on Computer Vision*, 2024.

[6] J. C. Lee, D. Rho, X. Sun, J. H. Ko, and E. Park, "Compact 3D Gaussian Representation for Radiance Field," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR)*, 2024, pp. 21719–21728.

[7] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai, "Scaffold-GS: Structured 3D Gaussians for View-Adaptive Rendering," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR)*, 2024, pp. 20654–20664.

[8] W. Morgenstern, F. Barthel, A. Hilsmann, and P. Eisert, "Compact 3D Scene Representation via Self-Organizing Gaussian Grids," 2023, *arXiv:2312.13299*. [Online]. Available: https://arxiv.org/abs/2312.13299

[9] S. Niedermeier, J. Stumpfegger, and R. Westermann, "Compressed 3D Gaussian Splatting for Accelerated Novel View Synthesis," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 10349—10358.

[10] K. Ren, L. Jiang, T. Lu, M. Yu, L. Xu, Z. Ni, and B. Dai, "Octree-GS: Towards Consistent Real-time Rendering with LOD-Structured 3D Gaussians," 2024, *arXiv:2403.17898*.

[11] X. Sun, J. C. Lee, D. Rho, J. H. Ko, U. Ali, and E. Park, "F-3DGS: Factorized Coordinates and Representations for 3D Gaussian Splatting," 2024, *arXiv:2405.17083*. [Online]. Available: https://arxiv.org/abs/2405.17083.

[12] S. Shin, J. Park, and S. Cho, "Locality-aware Gaussian Compression for Fast and High-quality Rendering," 2025, *arXiv:2501.05757*. [Online]. Available: https://arxiv.org/abs/2501.05757.

[13] J. Cao, V. Goel, C. Wang, A. Kag, J. Hu, S. Korolev, C. Jiang, S. Tulyakov, and J. Ren, "Lightweight Predictive 3D Gaussian Splats," 2024. *arXiv:2406.19434*. [Online]. Available: https://arxiv.org/abs/2406.19434

[14] H. Wang, H. Zhu, T. He, R. Feng, J. Deng, J. Bian, and Z. Chen, "End-to-end Rate-Distortion Optimized 3D Gaussian Representation," 2024, *arXiv:2406.01597*. [Online]. Available: https://arxiv.org/abs/2406.01597

[15] Y. Chen, Q. Wu, W. Lin, M. Harandi, and J. Cai, "HAC: Hash-grid Assisted Context for 3D Gaussian Splatting Compression," 2024, *arXiv:2403.14530*. [Online]. Available: https://arxiv.org/abs/2403.14530

[16] X. Liu, X. Wu, P. Zhang, S. Wang, Z. Li, and S. Kwong, "CompGS: Efficient 3D Scene Representation via Compressed Gaussian Splatting," 2024, *arXiv:2404.09458*. [Online]. Available: https://arxiv.org/abs/2404.09458

[17] Y. Wang, Z. Li, L. Guo, W. Yang, A. C. Kot, and B. Wen, "ContextGS: Compact 3D Gaussian Splatting with Anchor Level Context Model," 2024, *arXiv:2405.20721*. [Online]. Available: https://arxiv.org/abs/2405.20721

[18] Y.-T. Zhan, C.-Y. Ho, H. Yang, Y.-H. Chen, J. C. Chiang, Y.-L. Liu, and W.-H. Peng, "CAT-3DGS: A Context-Adaptive Triplane Approach to Rate-Distortion-Optimized 3DGS Compression," in *Proc. Thirteenth International Conference on Learning Representations (ICLR)*, 2025.

[19] L. Liu, Z. Chen, W. Jiang, W. Wang,and D. Xu, "HEMGS: A Hybrid Entropy Model for 3D Gaussian Splatting Data Compression," 2024, *arXiv:2411.18473*. [Online]. Available: https://arxiv.org/abs/2411.18473

[20] Y. Chen, Q. Wu, M. Li, W. Lin, M. Harandi, and J. Cai, "Fast Feed-forward 3D Gaussian Splatting Compression," 2024, *arXiv:2410.08017*. [Online]. Available: https://arxiv.org/abs/2410.08017

[21] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," 2018. [Online]. Available: https://arxiv.org/abs/1802.01436

[22] H. M. Kwan, G. Gao, F. Zhang, A. Gower, and D. Bull, "NVRC: Neural Video Representation Compression," 2024. [Online]. Available: https://arxiv.org/abs/2409.07414

[23] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields," in *Proc. IEEE/CVF conference on computer vision and pattern recognition(CVPR)*, 2022, pp. 5470–5479.

[24] A. Knappitsch, J. Park, Q. Zhou, and V. Lempitsky, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics (ToG)*. vol. 36, no. 4, 2017, pp. 1-–13.

[25] T. Leimkühler, J. Philip, T. Price, J.-M. Frahm, G. Drettakis, and G. Brostow, "Deep blending for free-viewpoint image-based rendering," *ACM Transactions on Graphics (ToG)*, vol. 37, no. 6, 2018, pp. 1-–15.

[26] Y. Xiangli, L. Xu, X. Pan, N. Zhao, A. Rao, C. Theobalt, B. Dai, and D. Lin, "BungeeNeRF: Progressive Neural Radiance Field for Extreme Multi-scale Scene Rendering," in *Proc. European Conference on Computer Vision (ECCV)*, 2022, pp. 106–122.