Supplementary Materials UniPCGC: Towards Practical Point Cloud Geometry Compression via An Efficient Unified Approach

Kangli Wang[1]**, Wei Gao**[1,2*]

[1]Guangdong Provincial Key Laboratory of Ultra High Definition Immersive Media Technology, Shenzhen Graduate School,
Peking University, Shenzhen 518055, China
[2]Peng Cheng Laboratory, Shenzhen, China
kangliwang@stu.pku.edu.cn, gaowei262@pku.edu.cn

Network Architecture Details

Feature Extraction Layer (FEL)

We utilize the FEL for feature extraction, which is similar to the work in (Wang et al. 2022). For dense object point clouds, sparse convolution with $k = 3$ has proven effective in feature extraction. However, the distances within the groups increase with finer grouping. To address this, we expand the receptive field by setting $k = 5$ to extract features. To balance the cost associated with using $k = 5$, we set $c = 16$ instead of 32, achieving a trade-off between performance and complexity. In FEL, we employ stacked Inception ResNet (IRN) for feature extraction. IRN has demonstrated powerful feature extraction capabilities in (Wang et al. 2021). The detailed structure of FEL and IRN are illustrated in Figure 2.

Upsampling Layer (USL)

In this layer, we apply a transposed convolution (GenerativeConvolutionTranspose) with $k = 2$ and $s = 2$. This operation allows one POV to be refined into eight PPOV.The proposed USL is illustrated in Figure 2.

Occupancy Probability Generation (OPG)

This layer is usually located after the FEL and is utilized to calculate the occupancy probability for each voxel. It typically comprises of three convolution layers followed by a sigmoid activation layer. The detailed architecture of OPG is illustrated in Figure 2.

One-Stage Lossless Coder (OLC)

OLC is commonly used for cross-scale probability estimation, utilizing the prior information of POV from $N-1$ scale to directly upsample to the N scale in a single-stage. Due to its single-stage estimation, it achieves faster speed but relatively lower compression performance. The detailed structure of OLC is illustrated in Figure 2.

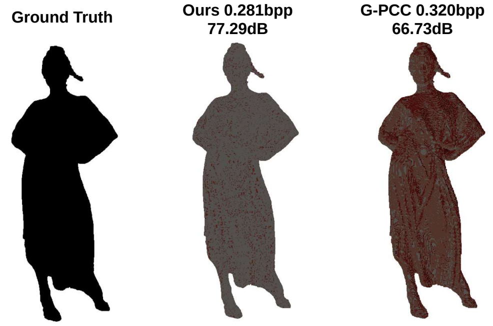

Figure 1: The color error map of lossy compression. The color error map depicts the point-to-point distortion of the reconstructed point cloud, where the color becomes more noticeable as the distortion increases.

Lossless Compression Results

Compression Results

Compared to G-PCC, we achieves a compression ratio (CR)-gain of 19.6%. Compared to SparsePCGC, we achieve a CR-gain of 5.6% when training dataset is ShapeNet, as shown in Table 1.

To compare with Voxeldnn, Octattention and ECM-OPCC methods, we train UELC using 8iVFB and MVUB and the test results are shown in Table 2. We use Voxeldnn, MsVoxelDNN, SparsePCGC, Octattention, ECM-OPCC and GPCC v23 as anchor. UELC demonstrates improved performance on the 8iVFB testset compared to SparsePCGC, with a 8.1% improvement. It also outperforms Voxeldnn (Nguyen et al. 2021a) with a 17.4% improvement, MsVoxelDNN (Nguyen et al. 2021b) with a 35.2% improvement, Octattention(Fu et al. 2022) with a 21.2% improvement, GPCC with a significant 29.2% improvement and ECM-OPCC (Jin et al. 2024) with a 11.3% improvement. Since it is difficult to re-implement the above methods, we directly cite the performance and runtime from (Wang et al. 2022; Jin et al. 2024).

[*]Corresponding Author: Wei Gao.

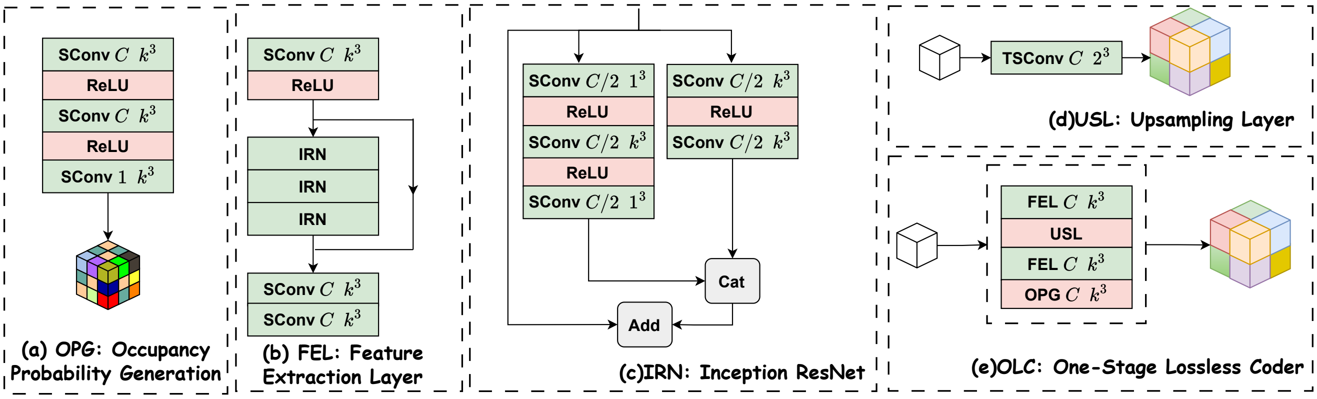

Figure 2: Illustration of the basic blocks. (a) OPG: Occupancy Probability Generation. (b) FEL: Feature Extraction Layer. (c) IRN: Inception ResNet. (d) USL: Upsampling Layer. (e) OLC: One-Stage Lossless Coder.

Additionally, we evaluated the compression capabilities of each technique within the proposed UELC framework on the sparse LiDAR point clouds of the KITTI dataset. Sequences 00-11 were utilized as the training set, while sequence 12 was reserved for testing to assess the efficacy of these methods. Our findings indicate that the proposed strategy remains applicable and effective for lower-level LiDAR point clouds. However, we observed that the non-uniform grouping scheme fails at higher levels. We attribute this performance degradation to the exceedingly sparse distribution of the point clouds in these upper layers.

Complexity Analysis

According to Table 2, our method demonstrates remarkably low complexity, making it highly attractive for practical applications. Our framework is lightweight with a model size of only 2.3MB, and it runs on an RTX4080 GPU in less than 1 second. SparsePCGC is tested using an RTX2080 GPU in the paper (Wang et al. 2022) and exhibited faster speeds compared to other models (Nguyen et al. 2021a,b; Fu et al. 2022). We re-implemented SparsePCGC and test UniPCGC and SparsePCGC using an RTX4080. The test results indicate that our method achieves faster execution speeds. GPCC belongs to traditional methods, so we do not calculate the model size. Additionally, the code for ECM-OPCC (Jin et al. 2024) is not open-source, and accessible weights are unavailable, hence the model size of ECM-OPCC is not displayed in Table 2.

Lossy Compression Results

Compression Results

Table 4 and Table 2 in the main paper presents the comparison between our approach and other methods including the non-learning method G-PCC v23 (Cao et al. 2021) and learning-based methods. Test dataset are longdress vox10 1300, redandblack vox10 1550, soldier vox10 0690, loot vox10 1200, queen 0200, basketball player vox11 0200, and dancer vox11 0001. Using D1 and D2 as distortion metrics, our method shows an average Bjontegaard Delta (BD)-rate improvement of 93.30% and 88.47% over the latest G-PCC v23. We also demonstrate superior performance compared to learning-based methods. VRCM outperforms SparsePCGC (Wang et al. 2022) by 14.02% and 12.20% in BD-rate, PCGCv2 (Wang et al. 2021) by 49.75% and 42.00% in BD-rate, GeoCNNv2 (Quach, Valenzise, and Dufaux 2020) by 75.94% and 70.43% in BD-rate, and ADL-PCC (Pereira 2021) by 77.93% and 76.36% in BD-rate, as shown in Tabel 4 and Figure 3-9. Moreover, The color error map of lossy compression is shown in Figure 1. The color error map depicts the point-to-point distortion of the reconstructed point cloud, where the color becomes more noticeable as the distortion increases.

Complexity Analysis

As shown in Table 3, our method exhibits remarkably low decoding complexity, which is comparable to the decoding time of SparsePCGC (Wang et al. 2022). Additionally, our model size is also very small, indicating that our framework is lightweight in terms of storage requirements.

Variable Rate and Complexity Support

By setting different values for λ and n, fine-grained rate modulation can be achieved. When $\lambda \in (-20, 3)$ and $n = 0, 1, 2$, continuous rate variation can be achieved, as shown in Figure 10. Due to GPU memory limitations (less than 16GB), we use only two values of λ during model training, which already encompass a wide range of rates. If more values of λ are used during model training, the RD performance is expected to improve further.

By allocating different computational resources to rate with varying compression difficulties, we aim to optimize computational efficiency. Guided by this principle, we introduce a dynamic network (Liu et al. 2017) in the Point Cloud Geometry Compression (PCGC) domain. We allocate higher complexity to higher rates to achieve lower distortion, while allocating fewer computational resources for lower rates. By setting different λ, we can achieve inference with variable complexity. First, based on the maximum λ_{max} during training, we can calculate the percentage q of voxels that need to be allocated computation, and select the voxels that need to

Table 1: Quantitative performance gains to the G-PCC and SparsePCGC. Compression ratio (CR) gain is measured by bits per point (bpp).

Point Clouds	**lossless**							
	G-PCC		**SparsePCGC**		**Ours**		**CR-gain (G-PCC)**	**CR-gain (SparsePCGC)**
	bpp	**Time (s) Enc/Dec**	**bpp**	**Time (s) Enc/Dec**	**bpp**	**Time (s) Enc/Dec**		
basketball_player_vox11	0.646	8.64/5.18	0.501	1.51/1.47	0.470	1.32/1.30	-27.2%	-6.2%
dancer_vox11	0.629	7.40/4.57	0.502	1.38/1.34	0.466	1.14/1.13	-25.9%	-7.2%
longdress_vox10	0.740	2.25/1.52	0.639	0.56/0.50	0.610	0.66/0.48	-17.6%	-4.5%
loot_vox10	0.702	2.09/1.42	0.606	0.54/0.49	0.581	0.50/0.45	-17.2%	-4.1%
queen_0200	0.604	2.60/1.71	0.556	0.65/0.59	0.496	0.56/0.51	-17.8%	-10.8%
redandblack_vox10	0.823	1.99/1.36	0.708	0.62/0.50	0.686	0.61/0.48	-25.9%	-3.1%
soldier_vox10	0.737	2.86/1.93	0.643	0.66/0.62	0.619	0.60/0.57	-19.1%	-4.2%
Average	0.697	3.97/2.53	0.594	0.84/0.79	**0.561**	**0.77/0.70**	**-19.6%**	**-5.6%**

Table 2: Performance of **lossless methods** on the 8iVFB testset under the same training conditions. The performance numbers of the compared methods are directly cited from (Wang et al. 2022; Jin et al. 2024; Gao et al. 2022). Due to the lack of the same GPU device as (Wang et al. 2022; Jin et al. 2024) for runtime testing, UniPCGC and SparsePCGC are tested using RTX 4080 GPU and intel i5-13600KF CPU for a fair runtime comparison (Mark with *).

Point clouds	**Ours**	**GPCC**	**SparsePCGC**	**VoxelDNN**	**MsVoxel**	**OctAttention**	**ECM**
Red&black	0.59	0.82	0.64	0.67	0.87	0.73	0.66
Loot	0.49	0.69	0.53	0.58	0.73	0.62	0.55
Thaidancer	0.51	0.70	0.56	0.68	0.85	0.65	0.58
Boxer	0.45	0.65	0.49	0.55	0.70	0.59	0.51
Average Bpp	**0.51**	0.72	0.56	0.62	0.79	0.65	0.58
Ours CR Gain	**0.0%**	-29.2%	-8.1%	-17.4%	-35.2%	-21.2%	-11.3%
Enc time(s)	*0.56	*1.99	*0.71/2.09	885	54	1.06	1.92
Dec time(s)	*0.57	*1.49	*0.88/1.91	640	58	1229	19.5
Test Device	RTX4080	RTX4080	RTX4080/2080	RTX2080	RTX2080	RTX2080	RTX3090
Model Size(M)	2.3	/	5.0	3.4	13.0	18.1	/

Table 3: Performance comparison of **lossy compression**. Decoding time, model size, and RD-Rate gain are all displayed. UniPCGC, SparsePCGC (Wang et al. 2022), PCGCv2 (Wang et al. 2021), ADL-PCC (Pereira 2021), GeoCNNv2 (Quach, Valenzise, and Dufaux 2020) and G-PCC are all compared.

	Ours	**PCGCv2**	**SparsePCGC**	**ADL-PCC**	**GeoCNNv2**	**G-PCC**
Dec time(s)	0.74	0.53	0.73	14.63	58	1.16
Test Device	RTX4080	RTX4080	RTX4080	RTX2080 Ti	RTX2080 Ti	RTX4080
Model Size(M)	7.6	3.2	9.5	32.8	18.8	/
Ours BD(D1)-Rate Gain	/	-49.75%	-14.02%	-77.93%	-75.94%	-93.30%
Ours BD(D2)-Rate Gain	/	-42.00%	-12.20%	-76.36%	-70.43%	-88.47%

Table 4: BD-Rate gains measured using both D1 and D2 for the UniPCGC against the G-PCC v23 for lossy coded dense point clouds

Dense Point Clouds	G-PCC	
	D1	**D2**
longdress_vox10_1300	-92.41%	-87.08%
loot_vox10_1200	-92.97%	-87.79%
red&black_vox10_1550	-91.33%	-85.13%
soldier_vox10_0690	-91.65%	-86.14%
queen_0200	-93.51%	-88.79%
player_vox11_200	-95.77%	-92.99%
dancer_vox11_0001	-95.43%	-91.40%
Average	**-93.30%**	**-88.47%**

Table 5: The ablation experiments of VRCM on the selected dense point clouds. UELC represents Uneven 8-Stage Lossless Coder, VR represents Variable Rate Module, VC represents Variable Complexity Module, and ✓denotes the usage of that method. D1-Gain represents the BD(D1)-rate gain over the baseline, D2-Gain represents the BD(D2)-rate gain over the baseline.

Method	**UELC**	**VR**	**VC**	**D1-Gain**	**D2-Gain**
Base				0.0%	0.0%
UELC	✓			-2.03%	-2.09%
VRM	✓	✓		-8.20%	-7.07%
VRCM	✓	✓	✓	**-14.02%**	**-12.20%**

be calculated based on q and correlation (Ali et al. 2024), as shown in the following equation:

$$\begin{aligned} q &= \lambda/\lambda_{max}, \\ N_{need} &= N_{sum} \times q, \end{aligned} \tag{1}$$

where N_{need} is the number of voxels that need to be calculated and N_{sum} is the total number of voxels. As a result, our model achieves faster inference speed at low rate. As shown in Figure 10, compared to static models, our dynamic network model achieves encoding acceleration of $59.6\%, 66.9\%, 58.5\%, 62.5\%$ at the lowest rate and decoding acceleration of $33.4\%, 34.6\%, 33.3\%, 33.5\%$ in loot_1200, red&black_1550, soldier_0690 and longdress_1300 respectively.

Ablation Studies

In lossy compression, UniPCGC utilizes UELC and VRCM techniques. In this section, we primarily investigate the effectiveness of these employed techniques. UELC introduce an efficient non-uniform grouping context scheme in lossless compression. At the decoding side, we combine the coordinates decoded by UELC with the decoded features from the entropy model. Therefore, the stronger the compression performance of UELC, the better the RD performance of lossy compression. We use SparsePCGC as baseline. Combining the baseline with UELC results in BD-rate gains of 2.03% and 2.09%. Building upon this, the introduction of the variable rate control module not only brings about BD-rate gains of 8.20% and 7.07%, but also enables the model to have the capability of variable rate encoding. On the basis of UELC and VR, further gains of 14.02% and 12.20% in BD-rate are achieved by introducing dynamic sparse convolution. This demonstrates the high efficiency of our proposed VRCM and UELC.

References

Ali, M. S.; Kim, Y.; Qamar, M.; Lim, S.-C.; Kim, D.; Zhang, C.; Bae, S.-H.; and Kim, H. Y. 2024. Towards efficient image compression without autoregressive models. *Advances in Neural Information Processing Systems*, 36.

Cao, C.; Preda, M.; Zakharchenko, V.; Jang, E. S.; and Zaharia, T. 2021. Compression of Sparse and Dense Dynamic Point Clouds—Methods and Standards. *Proceedings of the IEEE*, 109(9): 1537–1558.

Fu, C.; Li, G.; Song, R.; Gao, W.; and Liu, S. 2022. Octattention: Octree-based large-scale contexts model for point cloud compression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 625–633.

Gao, W.; Ye, H.; Li, G.; Zheng, H.; Wu, Y.; and Xie, L. 2022. OpenPointCloud: An open-source algorithm library of deep learning based point cloud compression. In *Proceedings of the 30th ACM international conference on multimedia*, 7347–7350.

Jin, Y.; Zhu, Z.; Xu, T.; Lin, Y.; and Wang, Y. 2024. ECM-OPCC: Efficient Context Model for Octree-Based Point Cloud Compression. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7985–7989.

Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; and Zhang, C. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, 2736–2744.

Nguyen, D. T.; Quach, M.; Valenzise, G.; and Duhamel, P. 2021a. Learning-based lossless compression of 3d point cloud geometry. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4220–4224. IEEE.

Nguyen, D. T.; Quach, M.; Valenzise, G.; and Duhamel, P. 2021b. Multiscale deep context modeling for lossless point cloud geometry compression. *arXiv preprint arXiv:2104.09859*.

Pereira, A. G. N. R. F. 2021. Adaptive Deep Learning-Based Point Cloud Geometry Coding. *IEEE Journal on Selected Topics in Signal Processing*, 15: 415–430.

Quach, M.; Valenzise, G.; and Dufaux, F. 2020. Improved Deep Point Cloud Geometry Compression. *2020 IEEE MMSP Workshop*.

Wang, J.; Ding, D.; Li, Z.; Feng, X.; Cao, C.; and Ma, Z. 2022. Sparse tensor-based multiscale representation for point cloud geometry compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, J.; Ding, D.; Li, Z.; and Ma, Z. 2021. Multiscale point cloud geometry compression. In *2021 Data Compression Conference (DCC)*, 73–82. IEEE.

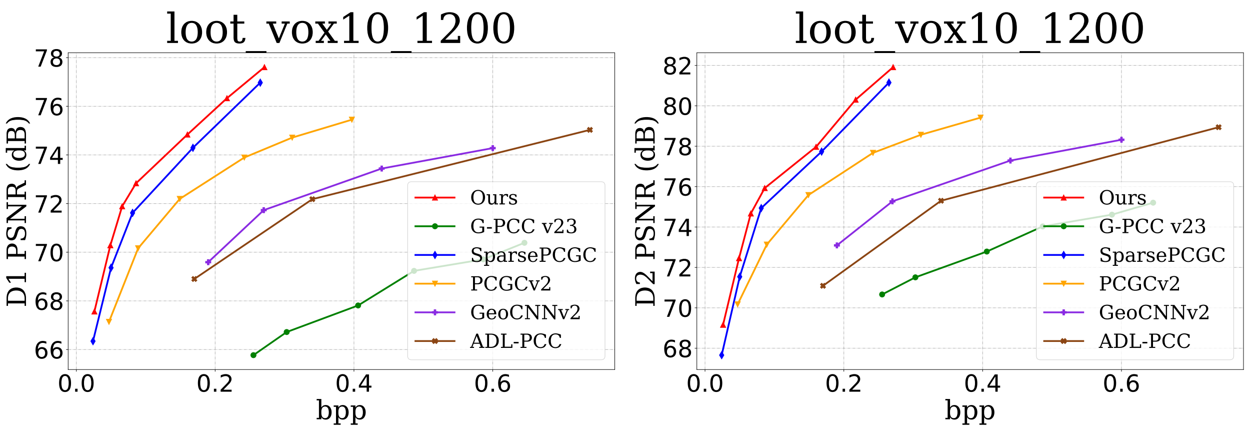

Figure 3: Performance comparison using rate-distortion curves on loot_vox10_1200.

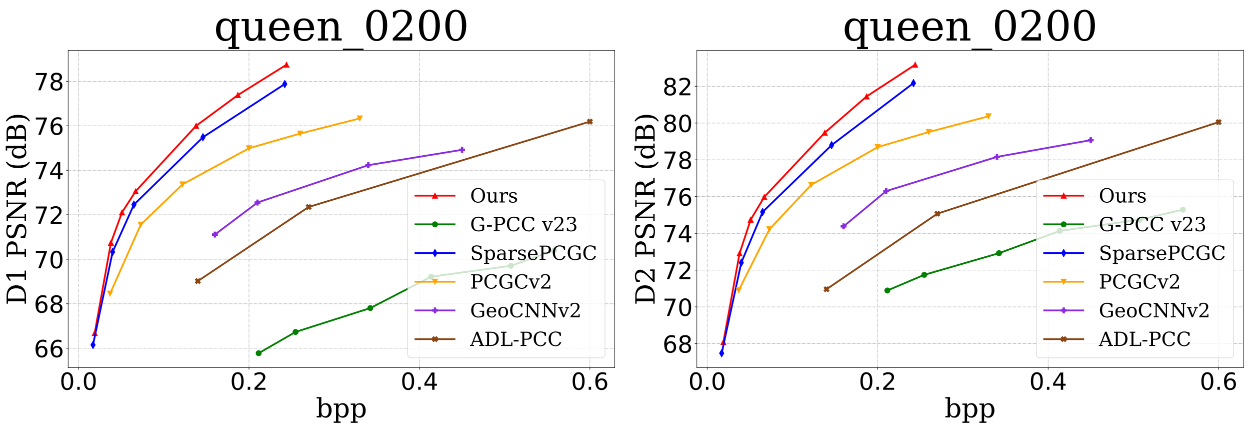

Figure 4: Performance comparison using rate-distortion curves on queen_vox10_0200.

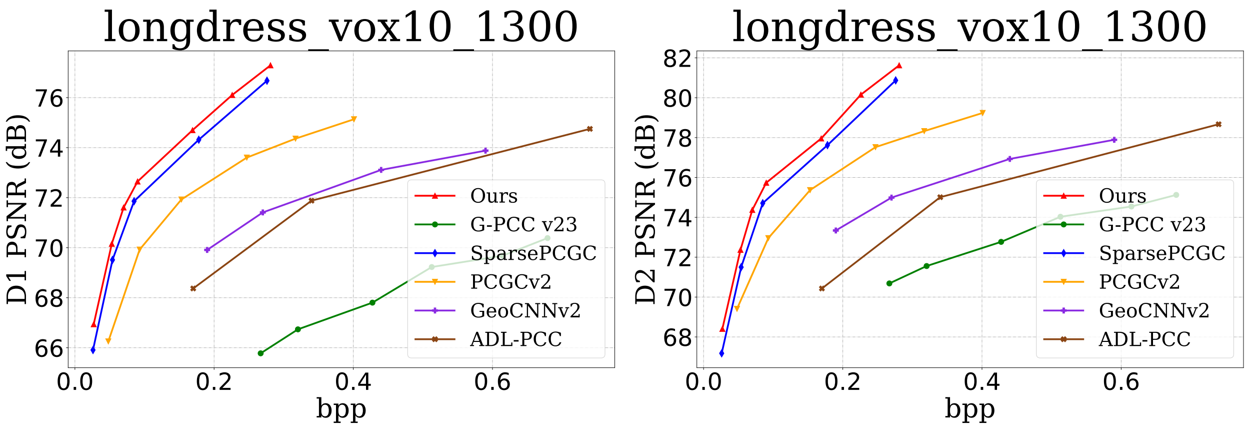

Figure 5: Performance comparison using rate-distortion curves on longdress_vox10_1300.

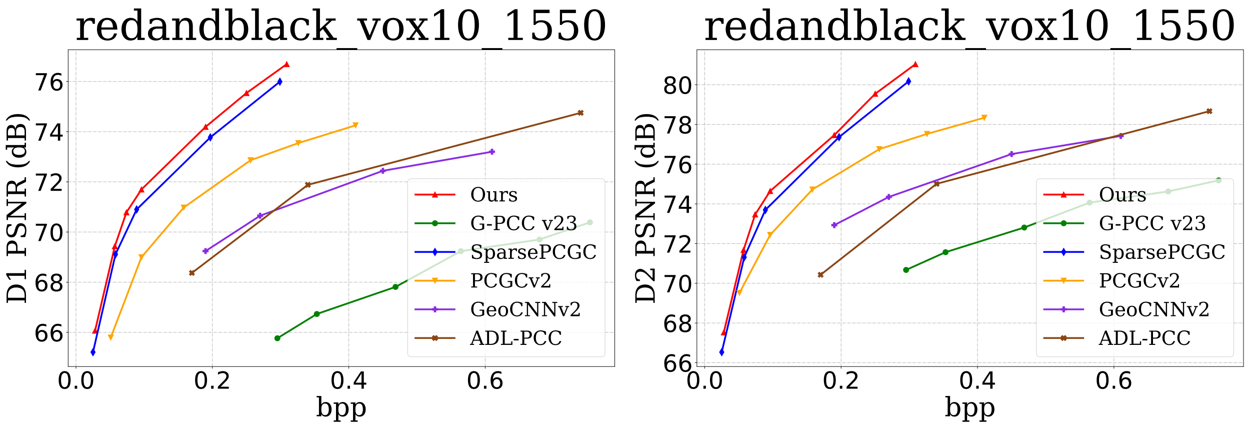

Figure 6: Performance comparison using rate-distortion curves on redandblack_vox10_1550.

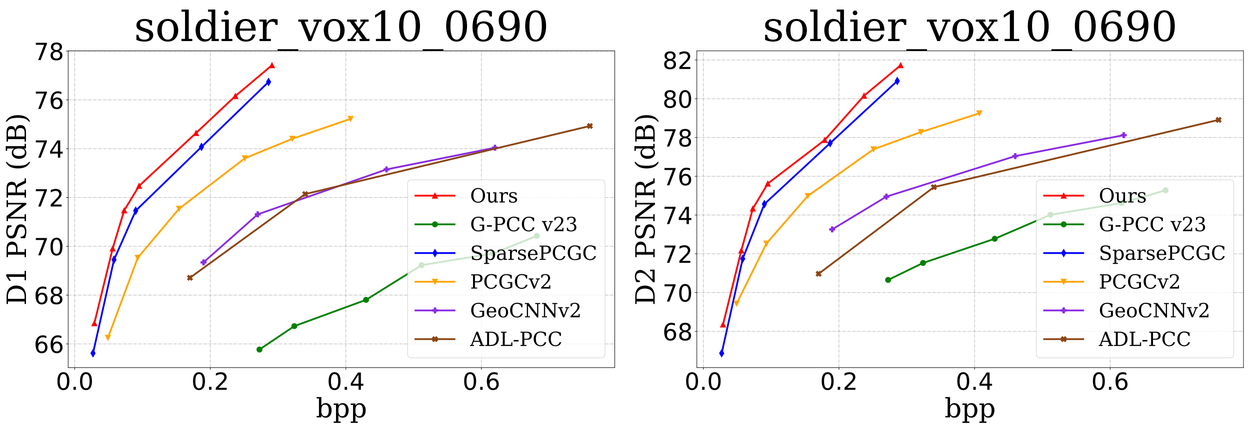

Figure 7: Performance comparison using rate-distortion curves on soldier_vox10_0690.

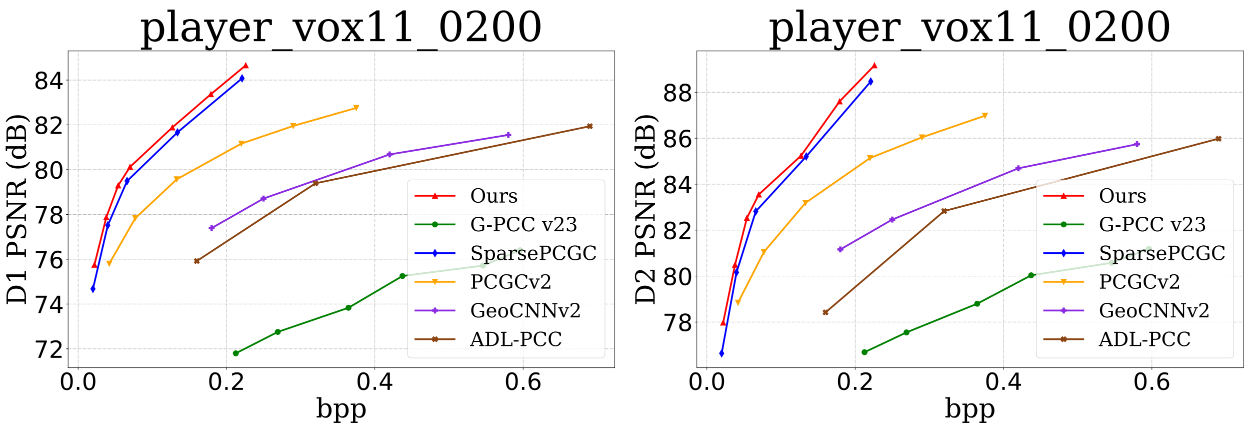

Figure 8: Performance comparison using rate-distortion curves on player_vox11_0200.

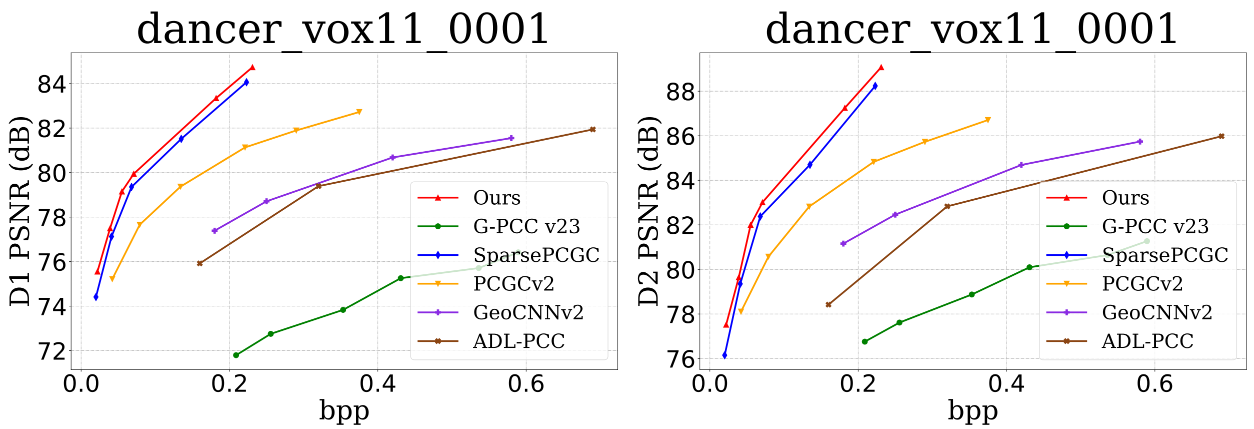

Figure 9: Performance comparison using rate-distortion curves on dancer_vox11_0001.

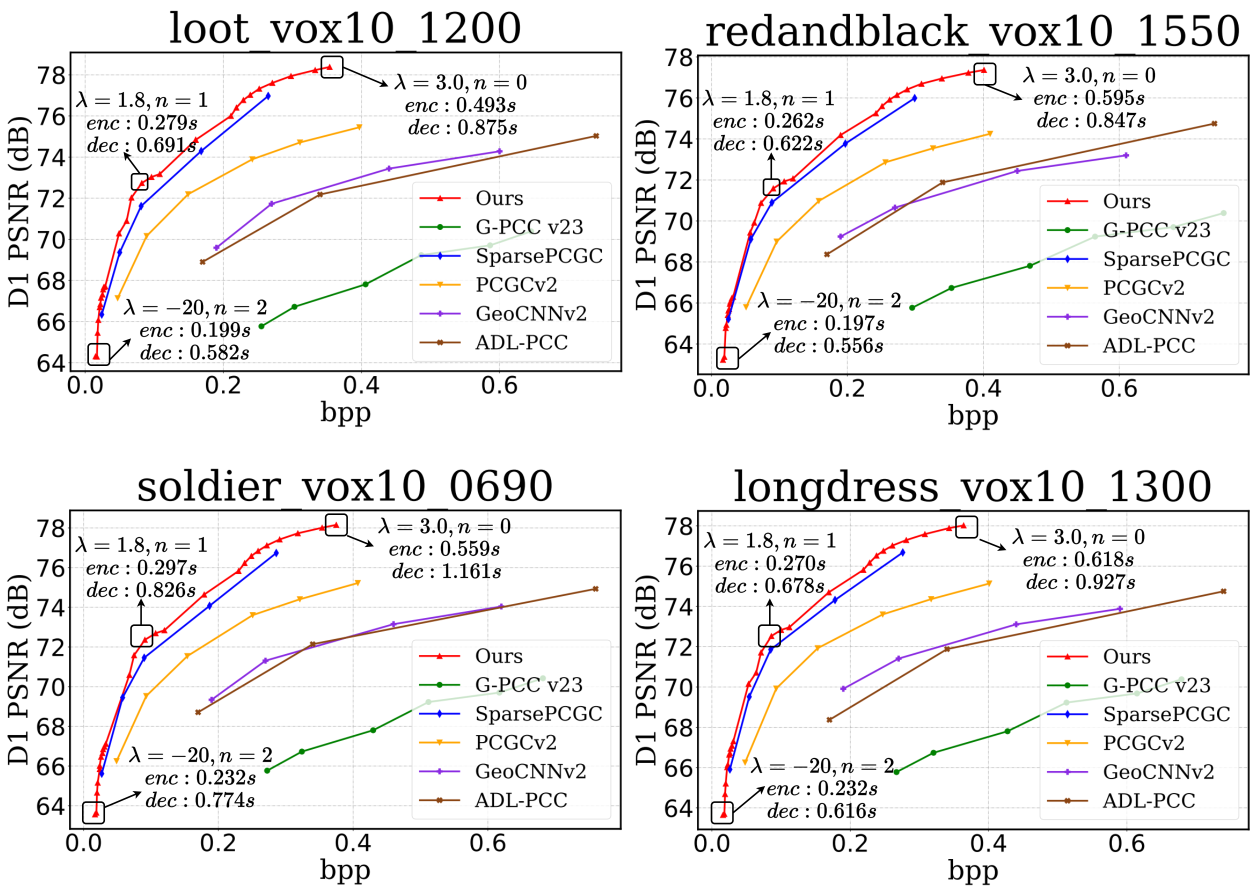

Figure 10: Variable rate and variable complexity support. Take loot_vox_1200, redandblack_vox10_1550, soldier_vox10_0690 and longdress_vox10_1300 as examples.