2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

SPCGC: Scalable Point Cloud Geometry Compression for Machine Vision

Liang Xie[1,2], Wei Gao[1,2], Huiming Zheng[1], and Ge Li[1]

Abstract— **With the proliferation of sensor devices, the extensive utilization of three-dimensional data in multimedia continues to grow. Point clouds are widely adopted within this domain because they are one of the most intuitive representations of three-dimensional data. However, the substantial volume of point cloud data poses significant challenges for storage and transmission. Moreover, a considerable portion of the data loses its semantic information during transmission. Consequently, how can we ensure both the perceptual quality for the human and the performance of downstream tasks during the transmission? To address this issue, we propose a scalable point cloud geometry compression framework (SPCGC) for machine perception. This framework tackles the fidelity issues associated with point cloud compression and preserves more semantic information, enhancing the performance of machine vision tasks. Our solution consists of a base layer bitstream and an enhancement layer bitstream. The base layer bitstream contains geometry data, while the enhancement layer bitstream utilizes semantic-guided residual data. Additionally, we introduce two modules for extracting and coding residual features. And incorporate classification and segmentation losses from downstream tasks into the Rate-Distortion (RD) optimization. Our approach outperforms existing learning-based lossy point cloud coding methods through empirical validation in downstream tasks without sacrificing point cloud compression performance.**

I. INTRODUCTION

Point clouds are a mathematical model employed to represent the surfaces of objects in three-dimensional space. They consist of a multitude of discrete points, each containing spatial information described as (X, Y, Z), along with potential additional attributes such as color, normal direction, reflectance intensity, among others. Point clouds are commonly acquired through 3D scanning devices, laser scanners, or other sensors. The applications of point cloud data are diverse, such as 3D modeling, environment perception, and autonomous driving. Point cloud datasets are frequently characterized by their substantial size, demanding significant computational resources and storage space for processing. Consequently, the point cloud needs compression before transmission.

Video Coding for Machine is a process aimed at encoding image or video data with a primary focus on fulfilling the requirements of machine perception and computer vision tasks rather than exclusively catering to human vision fidelity. This coding approach systematically considers the characteristics of machine vision applications, such as image classification and object detection, while also considering the demands of visual quality. For example, the methods [1]–[3] delve into machine perception-oriented image encoding by optimizing networks and introducing additional loss functions tailored to machine vision tasks. Bai *et al.* [4] introduces an end-to-end image compression and analysis model utilizing Vision Transformer [5] structure to handle image classification directly from compressed features. Similarly, Chen *et al.* [6] presents a novel image codec paradigm suitable for human and machine vision purposes. Furthermore, Liu *et al.* [7] introduce a scalable image compression solution in information requirements between human and machine vision.

With the advancement of deep learning, many point cloud analysis and compression methods based on deep learning have emerged. However, existing solutions for point cloud compression and analysis tend to focus on their respective compression or analysis performance, often overlooking the primary objective of point cloud compression, facilitating downstream data analysis or applications after transmission. For instance, works [8]–[10], involve converting point cloud data into depth maps and employing traditional compression tools for the compression process. These approaches encounter issues with precision loss due to the conversion step and exhibit limited coding performance. In contrast, methods [11], [12] leverage PointNet [13] or PointNet++ [14] as the coding network, but directly compressing point clouds using these methods results in a more significant loss of semantic information. This outcome stems from the feature disparity between the low-level compression and higher-level semantic extraction tasks. Furthermore, OctSqueeze [15] and VoxelContext-Net [16] aim to reduce bandwidth usage by utilizing octrees or voxel representations in dynamically acquired point clouds. However, these approaches often do not take into account vision tasks.

In summary, current point cloud compression frameworks are optimized for human vision fidelity, potentially at the expense of machine vision task performance. We propose a scalable point cloud compression framework for human fidelity and machine perception simultaneously, in which the bitstream consists of a base bitstream and an enhancement bitstream. The base layer primarily handles the coding of geometry coordinates, while the enhancement layer guides the coding of semantic residual features. Additionally, we integrate considerations for both rate optimization loss and semantic loss. Empirical results demonstrate that our approach maintains compression performance and outperforms several learning-based compression algorithms regarding low-rate point cloud classification accuracy and semantic segmentation mean intersection over union (mIoU). The contribution

[1]School of Electronic and Computer Engineering, Peking University Shenzhen Graduate School,[2]Peng Cheng Laboratory. (e-mail: liangxie,hmzheng@stu.pku.edu.cn, gaowei262,geli@pku.edu.cn). *(Corresponding author: Wei Gao)*. This work was supported by Natural Science Foundation of China (62271013, 62031013), Shenzhen Fundamental Research Program (GXWD20201231165807007-20200806163656003), Shenzhen Science and Technology Program (JCYJ20230807120808017), CAAI-MindSpore Open Fund, developed on OpenI Community (CAAIXSJLJJ-2023-MindSpore07).

can be summarized as below:

- We propose a hierarchical scalable point cloud geometry compression architecture consisting of the base and enhancement layers. The original point cloud is compressed by Geometry-based Point Cloud Compression (G-PCC) in the base layer to recover the geometric structure. At the same time, residual compression is performed in the enhancement layer to retain local detail. Sparse Tensor-based Feature Analysis Module (SFAM) and Sparse Tensor-based Feature Synthesis Module (SFSM) are also devised in the enhancement layer to extract and encode the residual information.
- We elaborately design the training strategy by jointly optimizing the whole framework through both human vision fidelity and machine perception. Specifically, latent estimation and downstream task loss are incorporated to improve machine analysis performance.
- The experimental results demonstrate that our scheme surpasses the existing lossy point cloud compression methods on all downstream task metrics at low bitrate. Meanwhile, we achieve comparable RD performance in contrast to other learning-based methods.

II. RELATED WORKS

A. Compression for Human Vision

As deep learning-based approaches advance, many algorithms for high-fidelity point cloud compression emerge. These algorithms can typically be categorized into three main types: octree-based, voxel-based, and sparse tensor-based compression methods. For example, OctSqueeze [15], [17] introduce a hierarchical Multilayer Perceptron (MLP) structure to extract contextual information from parent nodes in octrees, thereby enhancing the accuracy of the entropy estimation during the coding. VoxelContext-Net [16] adopts both octree and voxel structures to handle static and dynamic point clouds. Additionally, OctAttention [18] designs a conditional entropy model with a large receptive field by a Transformer structure. Some other methods transform point clouds into voxel representations and leverage the computational advantages of voxel structures in designing compression structures. For instance, Quach *et al.* [19] introduce pcc_geo_cnn_v2, which employs 3D Convolutional Neural Network (CNN) to compress the voxelized point cloud. On the other hand, Nguyen proposes VoxelDNN_v2 [20], which uses a hybrid coding mode and partitions the point cloud into multiple bitstreams for coding. Furthermore, they present various parallel coding strategies to enhance efficiency [21].

Many researchers incorporate sparse convolutions into point cloud compression models to improve coding efficiency. For instance, Wang *et al.* [22] introduce Learning-PCGC, which employs a variational autoencoder structure for voxelized point clouds. Subsequently, based on sparse convolutions, the multi-scale coding model PCGCv2 [23] emerged and demonstrates the capability to process point clouds with excellent performance. Building upon PCGCv2, several improvement have surfaced, including SparsePCGC [24] and SparsePCGCv2 [25]. These approaches propose various strategies for voxelization point cloud by multi-scales and group-scales, as well as context acquisition methods. They offer lossy and lossless encoding of diverse point cloud data with varying densities. However, most of these approaches aim to optimize human fidelity and often come with a significant loss of semantic information, which is detrimental to downstream visual tasks.

B. Compression for Machine Vision

In image and video compression, many methods aim to address the requirements for both human and machine vision simultaneously. For instance, the methods [26], [27] introduce several scalable image compression methods that employ different strategies to achieve scalability. Literatures [28]–[30] propose machine-perceived coding strategies that involve feature preservation. Additionally, works [31], [32] also present coding mechanisms related to joint perceptual analysis for video compression. In point cloud compression, Xie *et al.* [33] propose a coding network based on sparse convolution, simultaneously addressing the extraction of semantic information for downstream classification tasks. Ulhaq *et al.* [34] reconstruct the point cloud from the compressed bitstream and extract the necessary features for point cloud classification. Finally, approaches [35]–[37] introduce a semantic-guided LiDAR point cloud framework and analysis the machine vision performance. These frameworks typically incorporate semantic information to optimize human and machine vision but do not consider the joint optimization strategy.

III. METHODOLOGY

A. System Overview

We propose a scalable geometry compression approach for point clouds to better address the loss of semantic information during the compression. As shown in Fig. 1. This framework consists of four components: **First,** the input point cloud X_0 is first subjected to lossless encoding using G-PCC to obtain a coarse-grained base layer bitstream. **Second,** the original point cloud undergoes quantization ($Quan$) and de-quantization ($Quan^{-1}$) operations to obtain geometry coordinates X_1, which are then subtracted from the original input X_0 to obtain residual information Res. The geometric residual data is processed through the SFAM to obtain finer-grained local detail features F, encoded into the enhancement layer bitstream using an Entropy Encoder. Simultaneously, the original point cloud is also processed by a Semantic Feature Extraction Network (SFEN), consisting of a classification or segmentation network composed of PointNet++ [14] or DGCNN [38]. The semantic features extracted by SFEN are processed into Jacobian matrices and weighted into the residual Res. **Third,** at the decoding end, the rough point cloud of the base layer bitstream is decoded using a universal G-PCC decoder, and X_1' is obtained through de-quantization ($Quan^{-1}$). The enhancement layer bitstream decodes using an Entropy Decoder to obtain F', which is then used in the SFSM to reconstruct the original residual

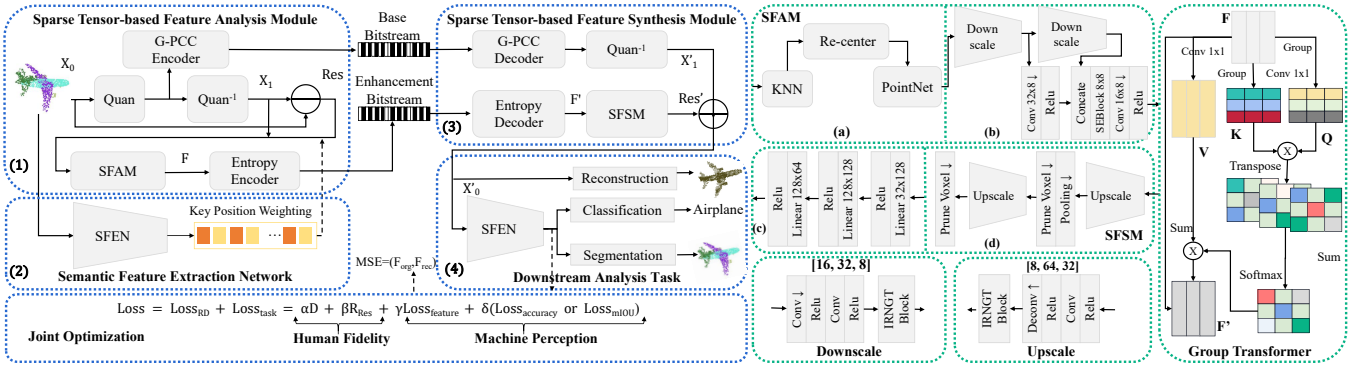


Fig. 1: The structure of the SPCGC framework can be divided into four phases.

features. The final step is to sum Res' and X_1' to obtain the original reconstructed point cloud X_0'. **Finally,** the reconstructed point cloud X_0' is input into the SFEN for point cloud classification or semantic segmentation. Additionally, during the training phase, the classification and segmentation losses of the reconstructed point cloud are incorporated into the RD optimization to guide semantic information extraction during point cloud encoding. The entire coding process draws inspiration from GRASP-Net [39].

B. Formulation

The pipeline of SPCGC can be described in Fig. 2. We denote X as the original point cloud, $\hat{X}$ as the quantization point cloud, and X' as the restored point cloud after de-quantization. In our architecture, the information contained in the feature map decreases as the neural network passes layer by layer. Meanwhile, X_{res} represents the residual derived by the original point cloud, $\overline{X}$ means the reconstructed point cloud, Y_{base} denotes the latent representation in the base layer, Y_{enh} defines the latent representation in the enhancement layer, and T describes the machine vision task. Moreover, the information flow transmitted in the base layer contains more characteristics than the enhancement layer. Y_{base} is derived by $\hat{X}$ while Y_{enh} is derived by X_{res}. Therefore, $\hat{X}$ includes the main geometry feature while X_{res} contains detailed semantic information.

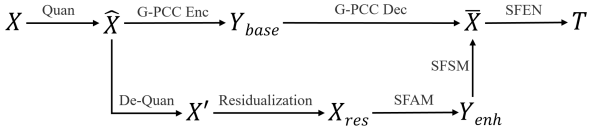


Fig. 2: The data process chain of the whole framework.

As described, the latent space of the reconstructed task v and the downstream task t are contained in the latent space of the cascaded latent representation y (derived by Y_{base} and Y_{enh}). According to Information Bottleneck theory [40], for task V and T:

$$\min_{p(v,t|y),p(y|x),p(x)} I(X;Y_{base},Y_{enh}) - \mu_h I(Y_{base},Y_{enh};\overline{X}) - \mu_m I(Y_{base},Y_{enh};T), \tag{1}$$

where $I(\cdot)$ denotes mutual information function, μ_h denotes the lagrange multiplier of human vision, and μ_m denotes the lagrange multiplier of machine vision. Given the case that y_{base} and y_{enh} are fully conditioned on x and y_{base} and y_{enh} are independent. We use $H(Y_{base}) + H(Y_{enh})$ to replace $I(X;Y_{base},Y_{enh})$. Therefore, the Eq. 1 turns into:

$$\min_{g_a,g_s,g_e} H(Y_{base}) + H(Y_{enh}) + \lambda_h D_h + \lambda_m D_m, \tag{2}$$

where g_a, g_s, g_e denote the transformation of SFAM, SFSM, and SFEN, respectively. λ_h denotes the weight of human vision distortion and λ_m denotes the weight of machine vision distortion. Further, we can obtain the total loss function as:

$$\begin{aligned}\mathcal{L}_{total} &= \mathcal{L}_r + \mathcal{L}_h + \mathcal{L}_m \\ &= R_{base} + \beta R_{enh} + \alpha D_{mse} + \gamma D_f + \delta D_m.\end{aligned} \tag{3}$$

In Eq. 3, $\mathcal{L}_r$ denotes the rate loss, $\mathcal{L}_h$ denotes the distortion loss and $\mathcal{L}_m$ indicates the machine vision loss. α, γ, δ are hyper-parameters of the Mean Squared Error (MSE) loss, feature loss, and machine vision loss. Noted that R_{base} equal zero in the training phase due to G-PCC encoder.

C. Encoder

The first step as shown in Fig. 1 (1). Assuming the input point cloud undergoes quantization, followed by G-PCC octree encoder to obtain base layer bitstream. Simultaneously, the quantized data is subjected to inverse quantization. The original point cloud subtracted quantized data by the $\ominus$ to derive residual information. These residuals are subsequently further encoded by SFAM to capture fine-grained geometry local details.

SFAM. The geometry residuals undergo the SFAM to obtain finer-grained local detail features denoted. The overall structure of SFAM, as illustrated in Fig. 1 (a), comprises the following steps: Firstly, it clusters the input residuals using a K-Nearest Neighbor (KNN) search, forming a local point set originating from the original coordinates. Subsequently, it associates each of these points with the original coordinates. Then, the Re-center module subtracts the coordinates of K neighboring points from each point, subsequently re-centering them within a relative local coordinate system. Finally, the PointNet module extract features from the relocated residual information.

SFEN. As shown in Fig. 1 (2), while generating latent features, the original point cloud also passes through a SFEN. This network can be a classification or segmentation network

composed of PointNet++ or DGCNN. The semantic features extracted by SFEN are transformed into Jacobian matrices and weighted with the residuals. Furthermore, it can be employed to extract features and calculate feature similarity loss as part of the loss calculation.

Key Position Weighting. The process of computing the Jacobian matrix is similar to [41]. We utilize the SFEN to extract semantics information and then integrate it into the residual features to guide the coding. We employ features extracted by the classification network and transform them into gradient information. Assuming a N-point $\mathbf{P} = \left[\mathbf{P}^{(1)}, \mathbf{P}^{(2)}, \ldots, \mathbf{P}^{(N)}\right]$, and classification or segmentation network S, We computer the Jacobian matrix as:

$$\mathbf{J} = \left[\frac{\partial \mathrm{S}\left(\mathbf{P}\right)}{\partial \mathbf{P}^{(1)}}, \frac{\partial \mathrm{S}\left(\mathbf{P}\right)}{\partial \mathbf{P}^{(2)}}, \ldots, \frac{\partial \mathrm{S}\left(\mathbf{P}\right)}{\partial \mathbf{P}^{(N)}}\right]. \tag{4}$$

Subsequently, the Jacobian matrix will be aligned using a 1x1 convolution with the quantized coordinates for fusion with the residuals. The sensitivity of points with large gradients suggests that any alterations will affect the final results. Meanwhile, we introduce an attention mechanism to enhance the spatial positions of crucial points within the residual features before encoding. We weight the aligned Jacobian matrix into the residual as an attention score w_i, and express the process as $\mathbf{r}_w^{(i)} = \mathbf{r}_f^{(i)} + w_i \mathbf{r}_f^{(i)}$. Then, the attention score describing as $\mathbf{R}_f = \left[\mathbf{r}_f^{(1)}, \ldots, \mathbf{r}_f^{(N)}\right]$ work as residual learning. Through the steps mentioned above, enhanced residual can be computed and combined with the de-quantization coordinate to form a sparse tensor, facilitating subsequent coding procedures.

IRNGT and SEBlock. As shown in Fig. 1 (b), the weighted residual representation undergoes two subsequent downsampling steps to explore redundancy between neighboring features within the residual characteristics. Each downsampling module consists of a sparse convolution and an Inception-Residual Network plus Group Transformer (IRNGT) block. Each IRNGR block contains three modules: IRN [23], GT, and IRN module. Simultaneously, we employ group self-attention [42], [43] to learn spatial relationships between different regions of the point.

The GT is depicted on the right of Fig. 1. Given the feature tensor $F \in R^{m \times C}$, we apply group-wise 1×1 convolutions to F to generate the query $Q^{m \times C}$ and the key $K^{m \times C}$. Meanwhile, another 1×1 convolution generates $V^{m \times C}$. We partition the Q and K along the channel into G groups, denoted as $\left\{Q_g \in R^{m \times C/G} \mid g = 1, \ldots, G\right\}$ and $\left\{K_g \in R^{m \times C/G} \mid g = 1, \ldots, G\right\}$, respectively. For each group, we model the attention map as $W_g = Q_g \cdot K_g^{\top}$, where W_g represents the similarity between Q_g and K_g. Finally, we sum up the G to obtain the final attention map $W = \sum_{g=1}^{G} W_g$, which aggregates the similarity information from different regions within the G groups. Consequently, points with similar neighborhood structures are assigned higher weights, allowing us to capture more discriminative local features. By performing matrix multiplication between V and W, followed by applying the Softmax and adding the input F, we obtain $F' = \mathrm{Softmax}\left(\frac{W}{\sqrt{C}}\right) V + F$, where C represents the dimension of the query map, and $\frac{1}{\sqrt{C}}$ serves as the scaling factor. Finally, the F' is combined with the input coordinates of the IRNGT block.

We apply convolution operations to each downsampling result to maintain channel consistency. This allows the results from the first and second downsampling steps to be directly concatenated. This operation aims to enhance the preservation of semantic information lost during the downsampling process. Finally, we utilize SEBlock from [33] to capture spatial correlations between different channels, further enhancing global features.

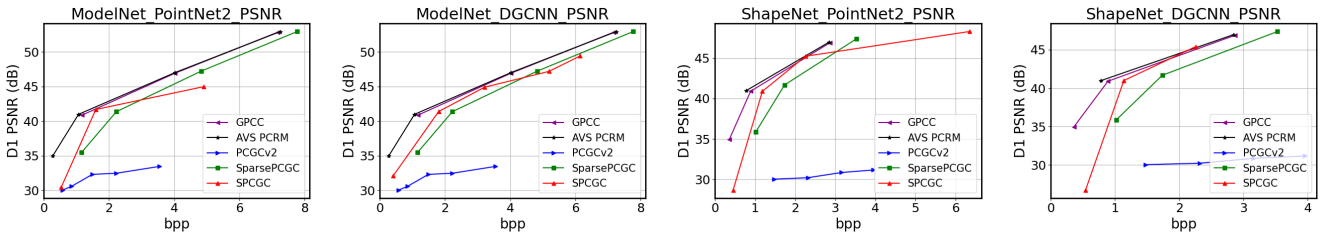
D. Decoder

SFSM. As shown in Fig. 1 (3), upon receiving the bitstreams from the base layer and enhancement layer, the initial step involves decoding the base bitstream using G-PCC. Subsequently, a dequantization process is applied to obtain a coarse point cloud. Within the enhancement layer, the bitstream undergoes the Entropy Decoder to acquire the downsampled feature set. Then, a voxelized coarse point cloud is input into SFSM to upsample. The SFSM consists of two upscale blocks and multiple MLP layers, as shown in Fig. 1 (c) and (d). By passing through MLP layers, it can recover the residual details. Lastly, a summation module ($\oplus$) adds the residual back to the coarse reconstructed point cloud, resulting in a decoded point cloud with fine details. The architecture of the SFSM network is shown in Fig. 1.

Downstream Analysis Task. As shown in Fig. 1 (4), the reconstructed point cloud is input into the SFEN for point cloud classification or semantic segmentation. The SFEN refers to a pre-trained model for either classification or segmentation. Simultaneously, during the training phase, the classification or segmentation loss function, denoted as D_m in Eq. 3, is incorporated into the RD optimization. This inclusion guides the coding process to capture semantic information better. Furthermore, both the reconstructed and original point clouds are input into the same pre-trained network. After passing through the pooling layer of the classification or segmentation network (denoted as $S_(max)$), global features are extracted. These global features are then used to compute MSE between the features of the reconstructed and the original point cloud, which is defined as D_f in Eq. 3 and expressed as:

$$D_f = \frac{1}{k} \sum_{i=1}^{k} \left\| \mathrm{S}_{\max}\left(X_i'\right) - \mathrm{S}_{\max}\left(X_i\right) \right\|_2^2, \tag{5}$$

where X_i' represents the reconstructed point cloud, and X_i represents the original point cloud. Additionally, k denotes the dimension of global features. The classification loss for the reconstructed point cloud is represented as cross-entropy loss expressed as $D_m = -\sum_{i=1}^{m} y_i \log_2 \mathrm{P}_i$, y represents the classification label, P_i represents the class probabilities obtained by passing the reconstructed point cloud through the classification network and the Softmax function, and m means the number of classes.



(a) PointNet2_D1 (b) DGCNN_D1 (c) PointNet2_D1 (d) DGCNN_D1

Fig. 3: Performance comparison in lossy compression using RD curves.

TABLE I: Quantitative performance gains over SPCGC, with BD-PSNR as the metric, where a higher value signifies superior performance. Here, "P" stands for PointNet++ [14], and "D" represents DGCNN [38].

Algorithm	G-PCC [44]		AVS [45]		PCGCv2 [23]		SparsePCGC [24]	
Metric	D1	D2	D1	D2	D1	D2	D1	D2
ModelNet40+P	1.814	3.418	2.381	4.269	-7.287	1.201	0.112	1.992
ModelNet40+D	1.789	3.851	2.471	4.820	-7.164	1.730	-1.225	0.910
ShapeNet+P	2.431	4.934	3.153	5.348	-14.952	-6.460	-0.458	1.832
ShapeNet+D	2.343	4.608	3.006	5.007	-14.981	-6.628	-0.408	1.759
Average	**2.094**	**4.203**	**2.753**	**4.861**	**-11.096**	**-2.539**	**-0.495**	**1.623**

TABLE II: The average encoding and decoding times, measured in seconds, for various point cloud compression algorithms in ModelNet40 and ShapeNet dataset.

Configure	AMD 5950X		Intel 8700K		RTX 3090		RTX 3090		RTX A5000	
Algrithm	G-PCC		AVS		PCGCv2		SparsePCGC		SPCGC	
Time	Enc	Dec	Enc	Dec	Enc	Dec	Enc	Dec	Enc	Dec
ModelNet	0.007	0.008	0.005	0.004	0.143	0.162	3.600	3.776	0.558	0.319
ShapeNet	0.006	0.006	0.004	0.003	0.142	0.162	2.719	2.837	0.466	0.219
Average	**0.007**	**0.005**	**0.005**	**0.004**	**0.143**	**0.162**	**3.159**	**3.307**	**0.512**	**0.269**
Model Size	*		*		3.1M		5.8M		1.8M	

IV. EXPERIMENTS

A. Experimental Setup

Dataset. We utilize two datasets for our experiments: ModelNet40 for classification tasks and ShapeNet_Part_Segmentation for semantic segmentation tasks. We follow the training and testing dataset division established by ModelNet40, and the test data is comprised of 2468 point clouds with class labels. Similarly, during the segmentation training process, we employ ShapeNet_Part_Segmentation as the training dataset and evaluate our model on a test set containing 2874 point clouds, each with segmentation labels. Additionally, we quantize the training and test datasets to 10 bits and conduct the training process using an NVIDIA A5000.

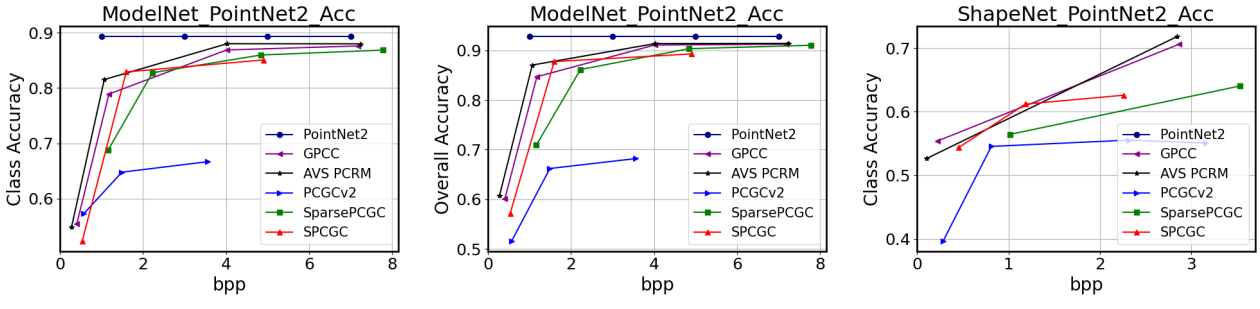
Training. During our training phase, we optimize our model using the following loss function: $\alpha \mathrm{D} + \beta \mathrm{R}_{\mathrm{Res}} + \gamma$ Loss $_{\text{feature}}$ $+\delta$ Loss $_{\text{accuracy}}$, where D represents the geometry distortion, measured using the Chamfer Distance. At the same time, R indicates the bitstream size of the residuals. α and β are used to control the trade-off between bitrate and distortion. We accomplish this by jointly optimizing the quantization step size and RD parameters. Specifically, we set the quantization step from 0.01 to 1. α and β were chosen from 1 to 10. $Loss_(feature)$ denotes the similarity between the original and reconstructed point clouds. Typically, the parameter γ falls within the range of 1 to 100. Additionally, $Loss_(accuracy)$ could be the cross-entropy loss for classification tasks or mIoU for segmentation tasks. We implement classification and segmentation tasks using DGCNN and PointNet++, both pre-trained models. As a result, our joint optimization model offers four distinct combinations.

Benchmark. We employ the following approaches to facilitate comparison: (1) G-PCC octree with version TMC13 v22. (2) AVS PCRM with version v11. (3) PCGCv2 with pre-trained model. (4) SparsePCGC in lossy mode. We utilize the following metrics to evaluate compression task performance: D1 PSNR, D2 PSNR, and bpp. BD-PSNR represented compression performance gain. For classification tasks, we assess classification accuracy using class Accuracy and Overall Accuracy. For segmentation tasks, we use Class mIoU and Overall mIoU as metrics. We also generate RD, RA (Rate-Accuracy), and RM (Rate-mIoU) curves to visualize the results. Furthermore, we consider model complexity, including encoding time and model size.

B. Performance Comparison

Compression. We calculate the BD-PSNR gain over SPCGC with the list benchmark to analyze compression performance. As shown in Tab. I, SPCGC does not outperform G-PCC and AVS because G-PCC and AVS employ octree representation, exhibiting stable performance within a 10-bit quantization range. However, when compared with PCGCv2, SPCGC demonstrates performance improvements of 11.096 and 0.495 in D1 and D2, respectively. Compared to SparsePCGC, our algorithm exhibits a performance gain of 0.495 in D1. Therefore, our approach holds a particular advantage in lossy compression performance compared with recent algorithms.

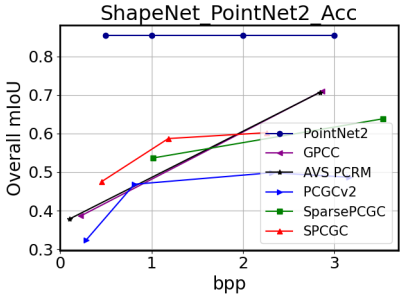
We also present RD curves in Fig. 3 for qualitative analysis. Although our method slightly lags behind G-PCC and AVS within the low bitrate, it still demonstrates favorable results when compared with PCGCv2 and SparsePCGC. One reason for SPCGC's inferior performance compared with traditional methods lies in our adoption of a joint training mode, which necessitates simultaneous consideration of compression and downstream task performance, resulting in a slight overall performance reduction.



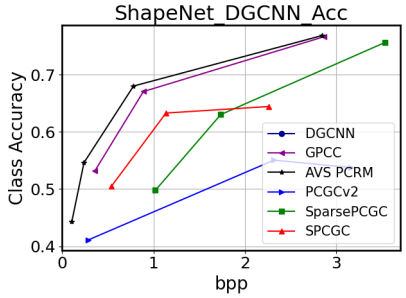
(a) PointNet2_Class_Accuracy

(b) PointNet2_Overall_Accuracy

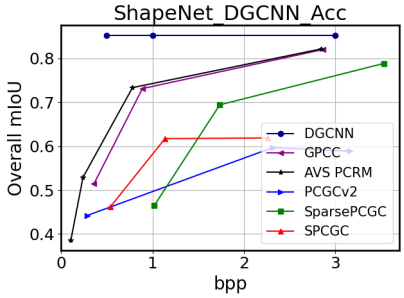
(c) PointNet2_Class_Accuracy



(d) PointNet2_Overall_mIoU



(e) DGCNN_Class_Accuracy



(f) DGCNN_Overall_mIoU

Fig. 4: Performance comparison in lossy compression using RA and RM curves.

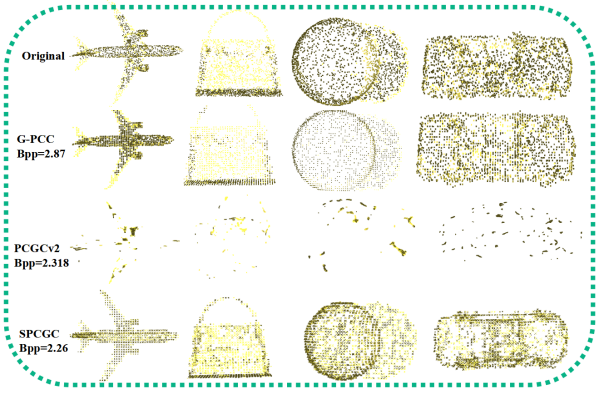


Fig. 5: Visualization of reconstruction outcomes for various algorithms at comparable bit rates in ModelNet40.

Downstream Tasks. For the classification task, we visually present the RA curves of PointNet++ trained jointly on the ModelNet40, including Rate-Class Accuracy and Rate-Overall Accuracy, as shown in Fig. 4 (a) and (b). In ModelNet40 dataset, our approach significantly improves classification accuracy within the low bitrate range. These experiments highlight the substantial advantages of our multi-scale feature extraction network and key positional weighting module. On the other hand, on the ShapeNet dataset, even though we did not perform joint training with classification loss, we also evaluated the classification accuracy. As shown in Fig. 4 (c) and (e), we directly assess the classification results in PointNet++ and DGCNN and observe that our method also achieved some gains.

We display Fig. 4 (d) and (f) as RM curves for the segmentation task. Note that due to the way we compute segmentation labels for each point, which involves searching for the nearest point label by the original test dataset labels (using KDTree search), there exists significant variability in the distribution of reconstructed point clouds among different models. Consequently, this leads to notable differences in segmentation accuracy. As Fig. 4 (d) illustrates, within the low bitrate, our method outperforms algorithms such as G-PCC, AVS, PCGCv2, and SparsePCGC (bpp = 0∼2). Moreover, from Fig. 4 (f), it is evident that our method performs well within the 0∼1.5 bitrate range when employing DGCNN as the segmentation algorithm.

Additionally, we conduct runtime analysis. Due to variations in the configuration environments of each algorithm, we could only provide a rough comparison of encoding and decoding times. As indicated in Tab. II, our model's coding times fall within acceptable ranges, and the model size is relatively compact. We also visualize the reconstruction of selected point clouds under different algorithms on the ShapeNet dataset in Fig. 5.

V. CONCLUSION

We introduce a scalable point cloud geometry compression framework that prioritizes human and machine vision. The bitstream consists of the base layer and the enhancement layer bitstream. The base bitstream is responsible for coarse-grained geometry, while the enhancement bitstream incorporates semantic residual representations. Additionally, we design a network with semantic enhancements and integrate downstream task losses into the RD optimization process. Experimental results demonstrate that our approach significantly improves the machine task performance without compromising coding performance. In the future, we will continue to explore various scalable solutions and strategies in ScanNet and SUN RGB-D.

REFERENCES

[1] S. Yang, Y. Hu, W. Yang, L.-Y. Duan, and J. Liu, "Towards Coding for Human and Machine Vision: Scalable Face Image Coding," *IEEE Transactions on Multimedia*, vol. 23, pp. 2957–2971, 2021.

[2] N. Le, H. Zhang, F. Cricri, R. Ghaznavi-Youvalari, and E. Rahtu, "Image Coding for Machines: an End-to-End Learned Approach," in *IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2021, pp. 1590–1594.

[3] M. Song, J. Choi, and B. Han, "Variable-Rate Deep Image Compression Through Spatially-Adaptive Feature Transform," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2380–2389.

[4] Y. Bai, X. Yang, X. Liu, J. Jiang, Y. Wang, X. Ji, and W. Gao, "Towards End-to-End Image Compression and Analysis with Transformers," in *AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 104–112.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *arXiv preprint arXiv:2010.11929*, 2020.

[6] S. Chen, J. Jin, L. Meng, W. Lin, Z. Chen, T.-S. Chang, Z. Li, and H. Zhang, "A New Image Codec Paradigm for Human and Machine Uses," *arXiv preprint arXiv:2112.10071*, 2021.

[7] K. Liu, D. Liu, L. Li, N. Yan, and H. Li, "Semantics-to-Signal Scalable Image Compression with Learned Revertible Representations," *International Journal of Computer Vision*, pp. 1–17, 2021.

[8] S. Shi, "Deep Learning on Point Clouds for 3D Object Detection," Ph.D. dissertation, The Chinese University of Hong Kong, 2021.

[9] Y. Wu and W. Gao, "End-to-end Lossless Compression of High Precision Depth Maps Guided by Pseudo-residual," *arXiv preprint arXiv:2201.03195*, 2022.

[10] H. Yuan, D. Zhang, W. Wang, and Y. Li, "A Sampling-based 3D Point Cloud Compression Algorithm for Immersive Communication," *Mobile Networks and Applications*, vol. 25, no. 5, pp. 1863–1872, 2020.

[11] T. Huang and Y. Liu, "3D Point Cloud Geometry Compression on Deep Learning," in *ACM International Conference on Multimedia*, 2019, pp. 890–898.

[12] W. Yan, S. Liu, T. H. Li, Z. Li, G. Li *et al.*, "Deep Autoencoder-based Lossy Geometry Compression for Point Clouds," *ArXiv Preprint ArXiv:1905.03691*, 2019.

[13] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017, pp. 652–660.

[14] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[15] L. Huang, S. Wang, K. Wong, J. Liu, and R. Urtasun, "Octsqueeze: Octree-structured Entropy Model for LiDAR Compression," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1313–1323.

[16] Z. Que, G. Lu, and D. Xu, "Voxelcontext-Net: An Octree Based Framework for Point Cloud Compression," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6042–6051.

[17] W. Gao, H. Ye, G. Li, H. Zheng, Y. Wu, and L. Xie, "OpenPointCloud: An Open-source Algorithm Library of Deep Learning Based Point Cloud Compression," in *ACM International Conference on Multimedia*, 2022, pp. 7347–7350.

[18] C. Fu, G. Li, R. Song, W. Gao, and S. Liu, "OctAttention: Octree-based Large-scale Contexts Model for Point Cloud Compression," *ArXiv Preprint ArXiv:2202.06028*, 2022.

[19] M. Quach, G. Valenzise, and F. Dufaux, "Improved Deep Point Cloud Geometry Compression," in *IEEE 22nd International Workshop on Multimedia Signal Processing*. IEEE, 2020, pp. 1–6.

[20] D. T. Nguyen, M. Quach, G. Valenzise, and P. Duhamel, "Lossless Coding of Point Cloud Geometry Using a Deep Generative Model," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 12, pp. 4617–4629, 2021.

[21] S. Biswas, J. Liu, K. Wong, S. Wang, and R. Urtasun, "Muscle: Multi Sweep Compression of LiDAR Using Deep Entropy Models," *Advances in Neural Information Processing Systems*, vol. 33, pp. 22 170–22 181, 2020.

[22] J. Wang, H. Zhu, Z. Ma, T. Chen, H. Liu, and Q. Shen, "Learned Point Cloud Geometry Compression," *ArXiv Preprint ArXiv:1909.12037*, 2019.

[23] J. Wang, D. Ding, Z. Li, and Z. Ma, "Multiscale Point Cloud Geometry Compression," in *Data Compression Conference*. IEEE, 2021, pp. 73–82.

[24] J. Wang, D. Ding, Z. Li, X. Feng, C. Cao, and Z. Ma, "Sparse Tensor-based Multiscale Representation for Point Cloud Geometry Compression," *ArXiv Preprint ArXiv:2111.10633*, 2021.

[25] J. Wang, R. Xue, Z. Ma, H. Wei, Y. Yu, V. Zakharchenko, and D. Wang, "SparsePCGCv2: Improved SparsePCGC with Attention Mechanism," *ISO/IEC JTC1/SC29/WG7 MPEG, M59552, Online*, April, 2022.

[26] H. Choi and I. V. Bajić, "Scalable Image Coding for Humans and Machines," *IEEE Transactions on Image Processing*, vol. 31, pp. 2739–2754, 2022.

[27] Y. Foroutan, A. Harell, A. de Andrade, and I. V. Bajić, "Base Layer Efficiency in Scalable Human-Machine Coding," *arXiv preprint arXiv:2307.02430*, 2023.

[28] B. Azizian and I. V. Bajić, "Privacy-Preserving Feature Coding for Machines," in *Picture Coding Symposium*. IEEE, 2022, pp. 205–209.

[29] A. Harell, Y. Foroutan, N. Ahuja, P. Datta, B. Kanzariya, V. S. Somayaulu, O. Tickoo, A. de Andrade, and I. V. Bajic, "Rate-Distortion Theory in Coding for Machines and its Application," *arXiv preprint arXiv:2305.17295*, 2023.

[30] L. Xie, M. Hu, X. Bai, and W. Huang, "Towards Hardware-Friendly and Robust Facial Landmark Detection Method," in *International Conference on Neural Information Processing*. Springer, 2022, pp. 432–444.

[31] W. Yang, H. Huang, Y. Hu, L.-Y. Duan, and J. Liu, "Video Coding for Machine: Compact Visual Representation Compression for Intelligent Collaborative Analytics," *arXiv preprint arXiv:2110.09241*, 2021.

[32] H. Choi and I. V. Bajić, "Scalable Video Coding for Humans and Machines," in *IEEE 24th International Workshop on Multimedia Signal Processing*. IEEE, 2022, pp. 1–6.

[33] L. Xie, W. Gao, and H. Zheng, "End-to-end Point Cloud Geometry Compression and Analysis with Sparse Tensor," in *International Workshop on Advances in Point Cloud Compression, Processing and Analysis*, 2022, pp. 27–32.

[34] M. Ulhaq and I. V. Bajić, "Learned Point Cloud Compression for Classification," *arXiv preprint arXiv:2308.05959*, 2023.

[35] L. Zhao, K.-K. Ma, Z. Liu, Q. Yin, and J. Chen, "Real-Time Scene-Aware LiDAR Point Cloud Compression Using Semantic Prior Representation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 8, pp. 5623–5637, 2022.

[36] L. Zhao, K.-K. Ma, X. Lin, W. Wang, and J. Chen, "Real-Time LiDAR Point Cloud Compression Using Bi-Directional Prediction and Range-Adaptive Floating-Point Coding," *IEEE Transactions on Broadcasting*, vol. 68, no. 3, pp. 620–635, 2022.

[37] A. Seleem, A. F. Guarda, N. M. Rodrigues, and F. Pereira, "Impact of Conventional and Deep Learning-based Point Cloud Geometry Coding on Deep Learning-based Classification Performance," in *IEEE International Symposium on Multimedia*. IEEE, 2022, pp. 74–81.

[38] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic Graph CNN for Learning on Point Clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[39] J. Pang, M. A. Lodhi, and D. Tian, "GRASP-Net: Geometric Residual Analysis and Synthesis for Point Cloud Compression," in *International Workshop on Advances in Point Cloud Compression, Processing and Analysis*, 2022, pp. 11–19.

[40] N. Tishby, F. C. Pereira, and W. Bialek, "The Information Bottleneck Method," *arXiv preprint physics/0004057*, 2000.

[41] Z. Li, G. Li, T. H. Li, S. Liu, and W. Gao, "Semantic Point Cloud Upsampling," *IEEE Transactions on Multimedia*, 2022.

[42] L. Hui, H. Yang, M. Cheng, J. Xie, and J. Yang, "Pyramid Point Cloud Transformer for Large-Scale Place Recognition," in *IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6098–6107.

[43] L. Xie, M. Hu, and X. Bai, "Online Improved Vehicle Tracking Accuracy via Unsupervised Route Generation," in *IEEE 34th International Conference on Tools with Artificial Intelligence*. IEEE, 2022, pp. 788–792.

[44] K. Mammou, P. A. Chou, D. Flynn, M. Krivokuća, O. Nakagami, and T. Sugio, "G-PCC Codec Description v2," *ISO/IEC TC1/SC29/WG11 MPEG, N18189, Marrakech*, January, 2019.

[45] A. P. C. C. W. Group, "Reference Software Algorithm Description of AVS Point Cloud Coding," *Audio Video Standard, N3445, Online*, September, 2022.