ConPCAC: Conditional Lossless Point Cloud Attribute Compression via Spatial Decomposition

Junzhe Zhang, Tong Chen, *Member, IEEE*, Kang You, Dandan Ding, *Senior Member, IEEE*, and Zhan Ma, *Senior Member, IEEE*

***Abstract*— A conditional lossless point cloud attribute compression method, dubbed ConPCAC, is proposed. The previous work typically codes point attributes in a point cloud in an autoregressive way, incurring unbearable coding time. By contrast, ConPCAC proposes a group-wise conditional entropy model for fast coding while preserving coding performance. Specifically, ConPCAC adopts a "Group Decomposition - Attribute Initialization - Latent Distribution Prediction" framework. First, it flexibly decomposes the original point cloud into multiple groups according to the geometry coordinate distribution. Then, the first group is coded using a base coder, *e.g.*, the standardized G-PCC, and the following groups are progressively coded using a neural coder conditioned on their preceding groups. Two key units, Attribute Initialization (Init) and Latent Distribution Prediction (LDP), are devised in the neural coder. The Init unit employs the nearest neighbor to initialize the attributes of a group, and the LDP unit further predicts the attribute probability distribution for the group. In this way, ConPCAC enables full correlation exploration across groups and parallel processing among points in a group. Finally, the predicted probabilities are fed into the arithmetic engine to code the true attribute values of each group. Extensive experiments demonstrate the performance of ConPCAC. It achieves 14.59%, 10.32%, and 12.26% improvements over the latest G-PCC on the widely used 8iVFB, Owlii, and MVUB datasets, respectively, significantly outperforming state-of-the-art lossless PCAC methods. Moreover, its computational complexity is comparable to G-PCC and much lower than existing learning-based methods. Associated code and models will be released on the website https://github.com/3dpcc/ConPCAC.**

***Index Terms*— Point cloud, attribute compression, lossless coding, probability prediction.**

I. Introduction

With the rapid evolution of metaverse, virtual/augmented reality (VR/AR), autonomous driving, *etc.*, the 3D point cloud has become an increasingly prevalent representation format to vividly render 3D objects and scenes [1], [2], [3], [4], [5], [6], [7], [8]. A 3D point cloud is typically formed by a large number of unordered points, each of which can be characterized by its geometry coordinate (x, y, z) and associated attributes such as RGB color (r, g, b), normal vectors, and/or reflectance. Caching or delivering such a point cloud is resource-intensive. For instance, for a point cloud with 10-bit geometry coordinate and 8-bit RGB color, 54 bits are needed to represent a single point. Consequently, archiving a 1-minute-long 30 Hz point cloud video that contains an average of one million points per frame requires 11 gigabytes of storage, which largely impedes its practical use.

Then, point cloud compression (PCC) techniques have been extensively investigated to tackle the aforementioned issue. Existing PCC studies mostly focus on geometry compression [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29]. However, the attribute information is also indispensable in various applications. For example, vibrantly representing objects in VR/AR for stunning visual perception heavily relies on the color attribute. Thus, this study focuses on lossless point cloud attribute compression (PCAC), targeting the perfect restoration of original attributes. Without loss of generality, losslessly compressed geometry coordinates are assumed for attribute compression [30], [31], [32].

A. Background

Prevalent rules-based solutions for lossless PCAC include G-PCC (geometry-based PCC) [34] and V-PCC (video-based PCC) [35]. G-PCC provides two options: RAHT (Region-Adaptive Hierarchical Transform) [36] and Predlift Transform [37] (interpolation-based hierarchical nearest-neighbor prediction with an update/lifting step). Presently, Predlift Transform offers higher performance. As for V-PCC, it projects 3D point clouds into 2D sequences and uses mature video codecs such as H.265/HEVC [38] or H.266/VVC [39] for compression. Overall, both G-PCC and V-PCC follow handcrafted rules to exploit inter-point[1] correlations for compact representation of point clouds, which is difficult to adapt

Received 1 August 2024; revised 25 November 2024 and 19 January 2025; accepted 5 February 2025. Date of publication 11 February 2025; date of current version 4 July 2025. This work was supported in part by the National Natural Science Foundation of China under Grant 62171174 and in part by the National Key Research and Development Project of China under Grant 2022YFF0902402 and Grant 2023YFC3706605. This article was recommended by Associate Editor C. Herglotz. *(Corresponding author: Dandan Ding.)*

Junzhe Zhang and Dandan Ding are with the School of Information Science and Technology, Hangzhou Normal University, Hangzhou 311121, China (e-mail: DandanDing@hznu.edu.cn).

Tong Chen, Kang You, and Zhan Ma are with the School of Electronic Science and Engineering, Nanjing University, Nanjing 210093, China.

This article has supplementary downloadable material available at https://doi.org/10.1109/TCSVT.2025.3540931, provided by the authors.

Digital Object Identifier 10.1109/TCSVT.2025.3540931

[1]The term "point" in this work indicates a positively occupied voxel.

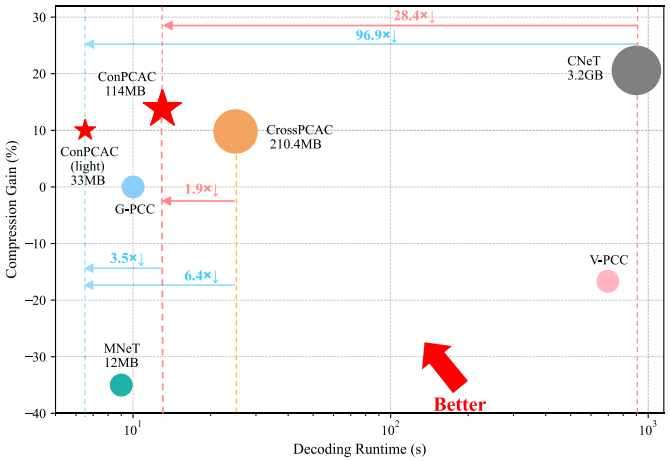


Fig. 1. Compression performance vs. runtime of G-PCC, V-PCC, CNeT [32], MNeT [33], CrossPCAC [31], and our proposed ConPCAC. G-PCC TMC13-v23 (Predlift) serves as the anchor. ConPCAC offers a balanced trade-off between compression performance and computational complexity. All methods are trained using a similar dataset. Results in this figure are tested on four point clouds *loot*, *redandblack*, *queen*, and *thaidancer*, except that the result of MNet is from the former two samples only. This is because MNet is not open-source, we thus copy relevant data from its paper for reference. The top left corner indicates the best result.

to various scenes and content, limiting their compression efficiency.

To this end, learning-based PCAC methods [40], [41], [42], [43], [44], [45], [46], [47], [48], [49] are developed. Among them, CrossPCAC [31], CNeT [32], and MNeT [33] are three typical solutions for lossless PCAC. Specifically, CNeT adopts an autoregressive model to process points in a point cloud sequentially. When processing the current point p_c, all preceding points are used as priors to predict the probability of p_c's attribute. Such an autoregression manner offers high compression performance at the expense of unbearable decoding time, as shown in Fig. 1, which is impractical for real-world applications. Its refinement, MNeT, reduces the decoding runtime, but the performance is severely deteriorated. Later, CrossPCAC replaces the autoregressive probability prediction with a multiscale and multistage structure. It downscales a point cloud into multiple scales and processes it scale by scale. At each scale, eight stages are devised. To remove redundancy across scales and stages, CrossPCAC introduces a series of processing rules, such as initializing the higher scale from the lower one, coding decimal residuals, and handling the last element in a stage. As seen, CrossPCAC requires the sequential decoding of scales and stages, resulting in a long processing time.

B. Approach & Contribution

Essentially, the lossless PCAC aims to drive the compression bitrate of a given point cloud as close to its entropy as possible. Following the chain rule of conditional entropy modeling, we can formulate the entropy modeling of each point or grouped points of the input as:

$$\begin{aligned} H(P) &= H(G^1, G^2, G^3, \cdots, G^L) \\ &= \Sigma_{i=1}^{L} H(G^i | G^1, G^2, \cdots, G^{i-1}), \end{aligned} \tag{1}$$

where $H(\cdot)$ represents the entropy. P indicates the input point cloud attribute and $G^i, i \in [1, L]$ denotes the attribute vector for a specific group of points. Theoretically, entropy modeling gets more accurate when more relevant and preceding groups of elements are used for the probability approximation.

As such, instead of scale-wise processing, we propose to spatially decompose a point cloud into multiple groups in the full resolution and utilize conditional coding to conduct group-wise compression, referred to as ConPCAC. For the very first group, ConPCAC can directly employ an off-the-shelf coder, such as G-PCC (Predlift), providing a baseline representation of the object (see Group 1 in Fig. 2). For the remaining groups, a neural coder is devised for progressive compression. In this way, the following groups can fully leverage information from prior groups through conditional coding.

Concerning the neural coder, ConPCAC adopts a simple yet efficient structure, including two key units: Attribute Initialization and Latent Distribution Prediction. The Attribute Initialization unit utilizes prior groups to initialize attribute values for the current group. Then, the Latent Distribution Prediction unit predicts the probability of each point's attribute according to the initialized current group and preceding groups.

As a result, ConPCAC performs group-wise processing with parallel computation within each group, completely avoiding the point-level autoregression used in CNeT. Furthermore, its neural coder fully exploits spatial correlations across groups in the native spatial resolution, enabling more accurate attribute prediction compared to CrossPCAC which predicts attribute values of the next scale from a lower scale. Meanwhile, CrossPCAC depends more on handcrafted rules to exploit scale and stage correlation, whereas ConPCAC primarily utilizes the adaptive capacity of neural models. These features together drive ConPCAC's improved efficiency over CrossPCAC. Experiments reveal that ConPCAC attains superior performance across various attribute types, such as color in dense point clouds and reflectance in LiDAR point clouds.

The main contributions of this paper are summarized below:

- We propose a lossless PCAC solution, called ConPCAC, which spatially decomposes a point cloud into multiple groups, *e.g.*, eight in this work, for conditional entropy coding. The first group is coded using a base coder, *e.g.*, the standardized G-PCC, and the remaining groups are sequentially coded using a neural coder with its context model conditioned on the preceding groups.
- To fully leverage prior knowledge from previous groups, the neural coder incorporates two units: the Attribute Initialization unit initializes point attributes in the current group, and the Latent Distribution Prediction unit predicts attribute probabilities for all points in the group.
- Extensive experiments demonstrate the performance of ConPCAC, showing 14.59%, 10.32%, and 12.26% gains over G-PCC in 8iVFB, Owlii, and MVUB datasets, respectively. Meanwhile, using sparse convolutions and local computations (*e.g.*, kNN) makes ConPCAC lightweight.

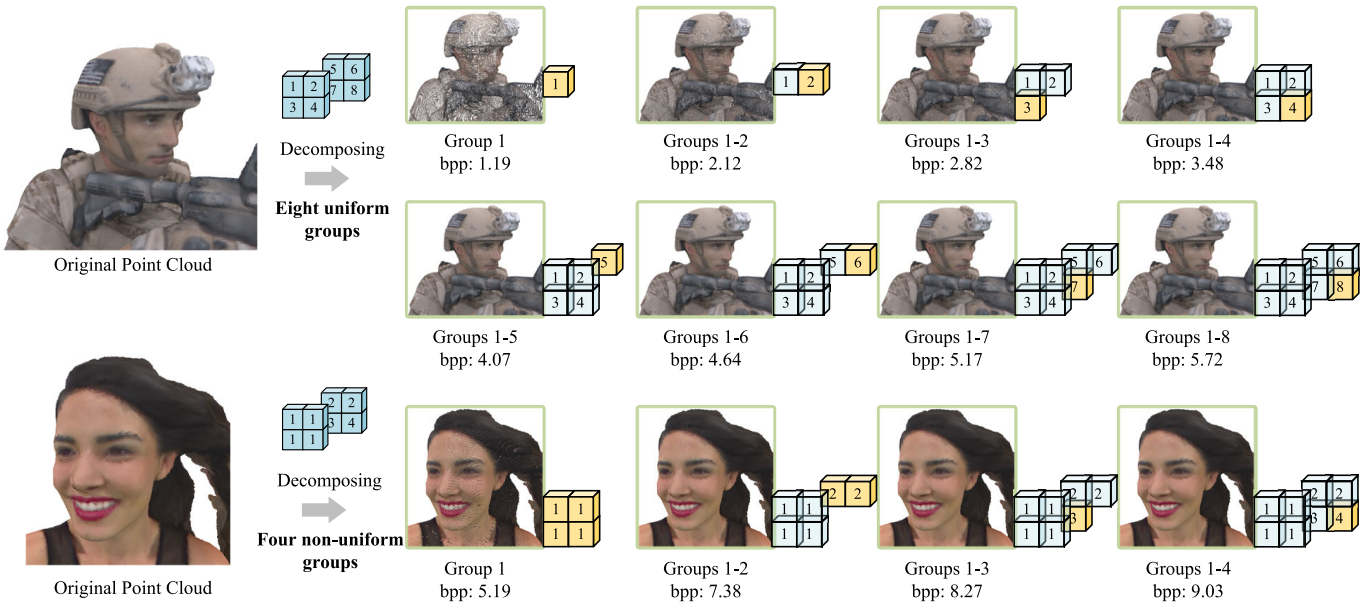


Fig. 2. Visualization of *soldier* (top) and *redandblack* (bottom) coded by the proposed ConPCAC. A point cloud is spatially decomposed into multiple groups and then coded progressively: each group is coded using a neural coder conditioned on its prior groups, except the first group is coded using a base coder. Flexible decomposition is allowed to meet the application requirement (*e.g.,* complexity-performance trade-off). The top line shows *uniform* grouping while the bottom line shows *non-uniform* grouping. The numbers in the squares indicate the group numbers. Blue squares represent groups that have already been coded, while yellow squares denote the group currently being processed.

II. Related Work

This section briefly reviews recent works related to point cloud attribute compression.

A. Rules-Based Lossless PCAC

PCAC has attracted significant attention in recent years. The core idea of PCAC is to transform the input attribute into a latent space, generating a sparse representation [40], [50] for rate-distortion (R-D) optimization. Commonly-used transforms include Graph Fourier Transform [51], [52], [53], [54], RAHT [36], *etc*.

In lossless PCAC, the goal is to minimize the rate cost without introducing any distortion. Typical examples are the Predlift- and RAHT-based lossless PCAC methods in G-PCC. Since the Predlift mode yields higher compression performance than the RAHT mode, we next focus on detailing the Predlift method. First, a Level-of-Detail (LoD) structure is constructed based on the geometry distance between points, by which the entire point cloud can be partitioned into different levels. Next, each level applies the sequential encoding: the point attributes to be encoded are predicted using the nearest-neighbor interpolation based on the previously decoded points, and then the residuals between the original and the predicted attribute values are calculated and encoded. This process continues until all levels are traversed.

The Predlift mode of G-PCC is essentially a kind of conditional coding, but its efficiency still has room for improvement. For example, the level partition overlooks inter-point correlations, and transmitting residuals is somewhat costly.

B. Learning-Based Lossless PCAC

In the past few years, learning-based PCAC methods have been developed with great success. Lossy approaches such as Deep-PCAC [43], SparsePCAC [46], ScalablePCAC [47], and DeepPCC [55] utilize stacked deep neural networks to transform the input attribute into latent features, followed by quantization to achieve compact representations.

In contrast, lossless PCAC aims to preserve the original attribute while minimizing its bitrate. The core is to approximate the distribution of the original attribute as closely as possible. To address this, CNeT [32] developed an autoregressive model for point-by-point PCAC. As such, all (causal) priors are utilized to predict the attribute value of the current point. Although the compression performance is substantially improved, its decoding complexity is unacceptable. For instance, CNeT requires 615 seconds to decode a point cloud frame. Its successor MNeT [33] greatly reduced the decoding time through multiscale processing, but its coding performance dropped sharply, even falling below that of G-PCC (see MNeT in Fig. 1). This may be due to the inefficient context modeling.

Recently, CrossPCAC [31], which used cross-scale, cross-group, and cross-color prediction, was proposed. It downscaled a point cloud into multiple scales for scale-wise processing. The higher scale attributes were used to initialize the lower scale counterparts. Then, within each scale, CrossPCAC designed eight stages to process grouped points one by one. Every time a point was coded, it updated the predicted values for the rest points. Such multiscale and multistage processing eliminates the point-level autoregression used in CNeT, significantly reducing the complexity. However, its performance gain is compromised due to the insufficient neighbor exploration from the downscaled (lower-scale) tensor.

1) Remarks: In summary, while learning-based lossless PCAC methods demonstrate superiority over traditional methods, their high complexity poses challenges for real-world

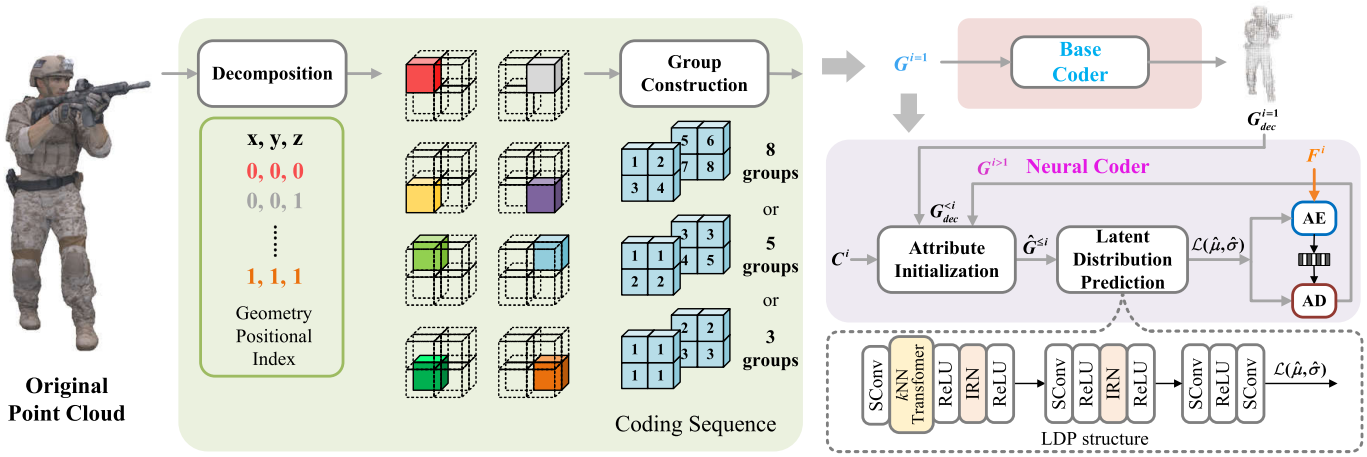


Fig. 3. *ConPCAC* decomposes the original point cloud attribute into eight sub-descriptions based on the geometry coordinate and flexibly assembles a set of sub-descriptions as a group for group-wise coding. Voxels having the same number belong to the same group. The first group is coded using a base coder, such as standardized G-PCC. Conditioned on the prior groups, the Attribute Initialization (Init) unit is first used to initialize the point attribute values in the current group. Then, the Latent Distribution Prediction (LDP) unit is applied to predict the attribute probability distribution for lossless compression. $G^i = (C^i, F^i)$ denotes the i-th group with the geometry coordinate C^i and the attribute information F^i. $\hat{G}^{\leq i} = \{G^{\leq i}_{dec}, \hat{G}^i\}$ is fed into LDP, where $G^{\leq i}_{dec}$ indicates the previously coded groups and $\hat{G}^i = (C^i, \hat{F}^i)$ represents the i-th group's coordinate C^i with initialized attribute $\hat{F}^i$. $\mathcal{L}(\hat{\mu}, \hat{\sigma})$ is Laplace distribution obtained by LDP. AE and AD represent arithmetic encoding and arithmetic decoding, respectively.

applications. Therefore, a high-efficiency and low-complexity solution is desired.

III. Conditional PCAC

Figure 3 illustrates the processing pipeline of ConPCAC. Without loss of generality, the point cloud geometry coordinates are losslessly compressed as prior knowledge when compressing the attribute component. Given a point cloud represented using a sparse tensor $S = (C, F)$ where $C = (x_n, y_n, z_n), n \in [0, N-1]$ is the collection of geometry coordinates, we transform the color attribute F from RGB space to YCoCg [56] space for compression, following the method used in [31] and [32]. Therefore, the color attribute tenor can be represented as $F = (Y_n, Co_n, Cg_n)$ with n denoting the index of n-th point or voxel.

Next, we detail how ConPCAC processes the point cloud attribute F group by group.

A. Spatial Decomposition

First, we decompose the original point cloud into multiple descriptions, for which we directly utilize the geometry coordinate:

$$GPI_n = C\%M = (x_n, y_n, z_n)\%M, n \in [0, N-1]. \quad (2)$$

Here, GPI denotes the geometry positional index. M is a constant value used for the modulo operation. For example, setting $M = 2$ produces $2^3 = 8$ descriptions, and setting $M = 3$ produces $3^3 = 27$ descriptions. This work exemplifies $M = 2$ to align with the octree voxelization principle, where one voxel is subdivided into eight sub-voxels.

For a given geometry coordinate (x, y, z), it is represented using a 3-bit GPI from $(0, 0, 0)$ to $(1, 1, 1)$. Points having the same GPI belong to the same description. We thus have eight descriptions for subsequent group-wise compression. These eight descriptions can be further combined as groups, as illustrated in Fig. 3.

In this way, this work can divide a point cloud into a maximum of eight groups and a minimum of two groups, denoted as $G^i = (C^i, F^i), i \in [1, L]$, with i being the group index and L being the number of groups. Note that the description grouping method can be *flexibly user-defined* according to the requirement of the underlying task or complexity-performance tradeoff. Points belonging to the same group are processed in parallel, and cross-group correlation is leveraged in compression.

As such, ConPCAC decomposes the point cloud attribute into a set of groups for sequential compression. We then compress the first group called $G^{i=1}$ using a base coder, producing the base representation $G^{i=1}_{dec}$.[2] The remaining groups, dubbed $G^{i>1}$, are losslessly compressed using a neural coder from one group to another. Previously-processed groups, *e.g.*, G^j, $j < i$, are priors for conditional context modeling.

B. Base Coder

In our implementation, we offer two options for the base coder to process the first group: one is the lossless configuration of the standardized G-PCC compressor and the other is an end-to-end (E2E) coder.

Directly using G-PCC as the base coder offers a straightforward approach, which is compatible with scenarios already equipped with G-PCC codecs. The low complexity of G-PCC is also friendly to users.

Instead, using the end-to-end coder in the first group $G^{i=1}$ leads the entire ConPCAC solution to a purely learning-based approach, which can be deployed on artificial intelligence platforms. However, directly compressing the attributes of the first group without any prior results in poor coding performance.

[2] In lossless mode $G^{i=1}_{dec} = G^{i=1}$.

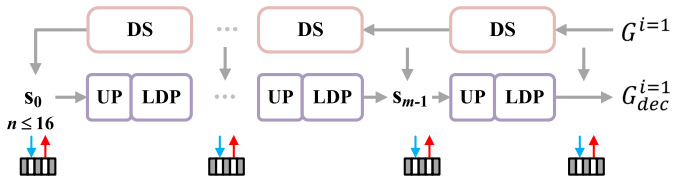


Fig. 4. End-to-end base coder for the first group. The first group is downscaled m times to obtain a root point cloud S_0 having $\leqslant 16$ points for direct coding. Then, conditioned on S_0, we compress the first group scale by scale. The attributes of lower scales are leveraged for context modeling of higher scales. DS denotes average pooling-based downscaling, UP indicates upscaling using MinkowskiPoolingTranspose, and LDP is the Latent Distribution Prediction unit used in our neural coder.

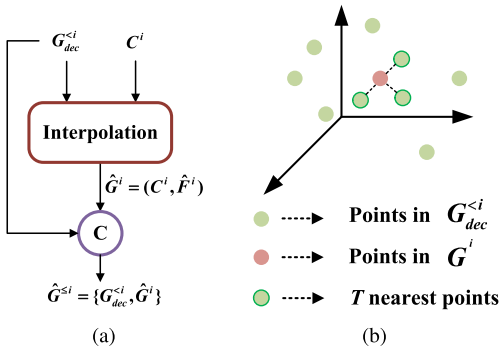


Fig. 5. Attribute initialization process for the current group G^i: (a) the Attribute Initialization unit, (b) the Interpolation unit.

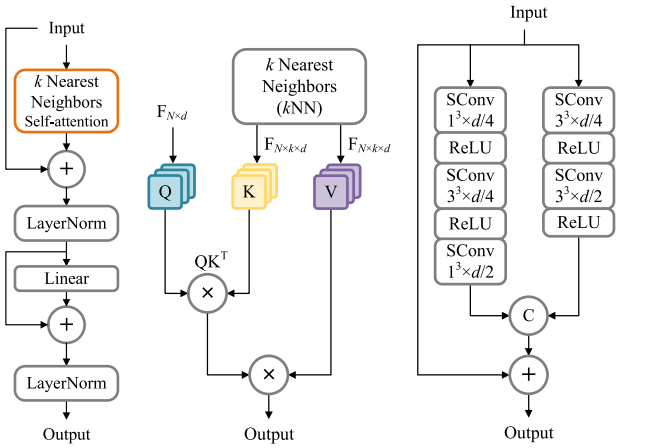


Fig. 6. Modular component. (a) kNN Transformer used in LDP to construct an adaptive local neighborhood by searching for k nearest neighbors of each point. (b) kNN Self-attention for local information aggregation. An attention map is generated for every k points upon the kNN neighborhood. (c) IRN that stacks sparse convolutions (SConv) for processing. N and d represent the number of points and the number of output feature maps, respectively. We set $k = 8$ and $d = 128$ in this work.

To address this, a prior needs to be generated following the conditional coding principle. Thus, the multiscale scheme is leveraged. As illustrated in Fig. 4, the first group is downscaled (through the DS module in Fig. 4) to m scales, with the lowest scale S_0 having $n \leqslant 16$ points. Next, the attributes of scale S_0 are directly coded by arithmetic coding. Then, the subsequent scale S_1 is coded using S_0 as the prior. Specifically, S_0 is upscaled (through the UP module in Fig. 4) to initialize S_1, and the LDP unit (which is detailed in the next subsection) is applied to predict the mean and variance for the attributes in S_1. Other scales are coded similarly. As such, the first group is coded in an end-to-end manner.

C. Neural Coder

After compressing the first group using the base coder, we apply the neural coder to encode the following groups. The neural coder consists of two major units: Attribute Initialization (Init) and Latent Distribution Prediction (LDP). As for the arithmetic coding engine, we refer to the implementation in [57]. Next, we detail the proposed Init and LDP units.

1) Attribute Initialization: Using decoded priors, the Init unit first assigns an initial value to each point in the current group. As presented in Fig. 5(a), we define the current group to be encoded as $G^i = (C^i, F^i)$, where the geometry coordinate C^i is known. For the decoded groups $G^{<i}$, we are able to retrieve their decoded coordinates and attributes $G^{<i}_{dec} = (C^{<i}_{dec}, F^{<i}_{dec})$ as well. Then, the Init unit uses C^i and $G^{<i}_{dec}$ together to predict the attribute value $\hat{F}^i$ of the current group.

More specifically, as depicted in Fig. 5(b), an interpolation operation is applied in the Init unit for prediction. For each point in G^i, we first identify its T nearest points in previously decoded groups, using C^i and $G^{<i}_{dec}$. Then, we collect the attribute values of these T points and calculate their average value as the predicted attribute, finally forming $\hat{F}^i$ of the current group. T is set to 1 according to extensive experiments (more details can refer to the ablation study in Section IV). As such, we initialize the current group as $\hat{G}^i = (C^i, \hat{F}^i)$, which is then concatenated with the previously decoded groups as $\hat{G}^{\leqslant i}$ for latent distribution prediction.

2) Latent Distribution Prediction: The LDP unit predicts the attribute probability for all points in the current group. Given $\hat{G}^{\leqslant i}$ from the Init unit, LDP cascades sparse convolutions (SConvs), kNN-based Transformer, and Inception-Residual Network (IRN) layers to effectively exploit neighborhood correlations for feature embedding (Fig. 3). The network structures of kNN Transformer and IRN are illustrated in Fig. 6.

At the end of LDP, two SConv layers and a ReLU layer are applied to approximate the mean $\hat{\mu}^i_n$ and variance $\hat{\sigma}^i_n$ values of the attribute for point F^i_n, with which the probability of each group can be computed by integrating the Laplacian distribution $\mathcal{L}$:

$$prob(F^i) = \prod_n \left(\mathcal{L}\left(\hat{\mu}^i_n, \hat{\sigma}^i_n\right) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right) \right)(F^i_n), \quad (3)$$

$$\text{with } \hat{\mu}^i_n, \hat{\sigma}^i_n = \text{LDP}(\hat{G}^{\leqslant i}), \quad (4)$$

where $\mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)$ is the uniform distribution between $-\frac{1}{2}$ and $\frac{1}{2}$ and n is the number of points in group i.

3) Cross-Component Prediction: Three components, including Y, Co, and Cg, will be encoded for the color attribute. Previous sections detail the processing stages for the Y component. Next, we extend ConPCAC for Co and Cg processing. It has been discovered that Y, Co, and Cg components are deeply correlated [58], inspiring us

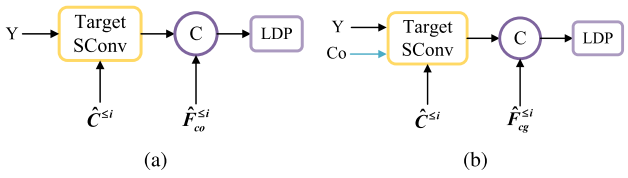


Fig. 7. Cross-component prediction using target sparse convolution (Target SConv). (a) Coding Co using Y. (b) Coding Cg using both Y and Co.

to leverage cross-component correlations for performance improvement.

Specifically, we compress Y, Co, and Cg components sequentially. In this way, the entire Y component is available when compressing the Co component. We apply a target SConv layer (with a kernel size of 3) to leverage Y to estimate the Co component (see Fig. 7(a)). As such, we successfully embed the Y characteristics into Co features as guidance. Next, the output from target SConv is connected with Co features $\hat{F}_{Co}^{\leqslant i}$ from the Init units and sent to LDP for prediction. It is a similar scenario with the Cg component except that Cg uses both Y and Co as input (see Fig. 7(b)).

4) Remarks: The "Group Decomposition - Attribute Initialization - Latent Distribution Prediction" process in ConPCAC is straightforward yet effective for lossless PCAC. Its group-wise mechanism fully leverages inherent correlations across attribute groups in its native resolution for highly efficient conditional coding. Extensive experiments demonstrate that our ConPCAC outperforms state-of-the-art G-PCC compressors and most neural coders (except CNet, which employs an ultra-expensive autoregressive mechanism). The coding algorithm of ConPCAC is presented in our supplementary material.

IV. Experiments and Analysis

A. Training and Testing Conditions

1) Datasets: We conduct experiments on point cloud datasets recommended by the international standardization committee MPEG [66] to evaluate the performance of ConPCAC. These datasets cover a variety of objects and scenes, which can fully examine the compression capability of a PCAC method. Below, we briefly introduce these datasets. Visualizations of the associated point clouds are included in the supplementary material, and detailed information about the testing samples is presented in Table I.

- **RWTT [67] dataset.** The Real World Textured Things (RWTT) dataset, a collection of publicly available textured 3D models, is used for training. We select 548 3D models from RWTT and quantize them to point clouds with 10-bit geometry precision. Due to the limitation of GPU memory, we apply the K-D tree[3] to partition these 3D point clouds into 3D blocks, each of which has 100,000 points. As a result, 10,334 training samples are generated. The corresponding testing datasets include 8iVFB [59], Owlii [62], and MVUB [63].
- **Mixed dataset of 8iVFB [59], Owlii [62], and MVUB [63].** To compare with existing works, we follow the settings in [31] and [32] to build the training and testing datasets. The point clouds *longdress* and *soldier* from 8iVFB, *model* and *exercises* from Owlii, *andrew*, *david*, and *sarah* from MVUB, are partitioned into 100,000 points per patch for training. The point clouds *loot* and *redandblack* from 8iVFB, *basketball_player* and *dancer* from Owlii, and *phil* and *ricardo* from MVUB are used for testing. Other samples widely used in MPEG, including *queen* from CGI [60] and *thaidancer* from 8iVSLF [61], are also used as testing samples.
- **ScanNet [64].** This dataset has 1,603 indoor point cloud scans, with 1,503 scans allocated for training and 100 scans reserved for testing.
- **Ford [65].** Ford is the designated LiDAR dataset in MPEG, comprising three sequences, each with 1,500 frames. Following the MPEG AI-PCC common test condition (CTC) [66], [68], we use the first sequence for training and the other two for testing. Ford has no color but only 8-bit reflectance attribute information for compression.

TABLE I
Detailed Information of Testing Datasets in This Work

Class	Point Cloud	Number of Points
8iVFB [59] (10-bit)	longdress	765,821
	loot	805,285
	redandblack	729,133
	soldier	1,059,810
CGI [60] (10-bit)	queen	1,000,993
8iVSLF [61] (12-bit)	thaidancer	2,942,011
Owlii [62] (11-bit)	basketball_player	2,925,514
	dancer	2,592,758
	exercise	2,391,718
	model	2,458,429
MVUB [63] (10-bit)	andrew	1,276,312
	david	1,492,780
	phil	1,660,959
	ricardo	960,703
	sarah	1,355,867
ScanNet [64] (q2cm)	ScanNet from #0707 to #0806	126,761
Ford [65] (q1mm)	Ford_02	83,834
	Ford_03	84,062

2) Training Settings: We implement ConPCAC using PyTorch and MinkowskiEngine [69] and run it on a computer with an Intel® i9-12900K CPU, 32 GB memory, and Nvidia 3090Ti RTX GPU. We optimize the network model using Adam with parameters β_1 and β_2 set to 0.9 and 0.999, respectively. The learning rate decays by half every two epochs and starts from 1e-4 to 1e-5. The model is randomly initialized and trained on the dataset for up to 50 epochs. Convergence takes approximately 50 to 60 hours on our platform.

3) Testing Setting: The testing conditions strictly conform to the MPEG AI-PCC CTC. Quantitative analysis is measured in bits per point (bpp) and compression ratio gain. The latest G-PCC version TMC13v23,[4] which provides state-of-the-art performance of lossless PCAC through Predlift attribute transform, is compared. For a fair comparison, we disable the

[3]https://en.wikipedia.org/wiki/K-d_tree

[4]https://github.com/MPEGGroup/mpeg-pcc-tmc13

TABLE II
QUANTITATIVE COMPRESSION GAINS AGAINST EXISTING METHODS USING THE RWTT TRAINING DATASET

Class	Point Cloud	bits per point (bpp)						Compression Gain		
		G-PCC (Predlift)	V-PCC (VTM)	CrossPCAC	Proposed ConPCAC Group 1	Group 2-8	Overall	vs. G-PCC	vs. V-PCC	vs. CrossPCAC
8iVFB (10-bit)	longdress	11.55	14.46	10.19	1.92	8.01	9.93	-14.04%	-31.33%	-2.56%
	loot	6.06	8.22	5.52	1.03	4.18	5.22	-13.93%	-36.51%	-5.53%
	redandblack	9.20	11.90	8.19	1.58	6.43	8.01	-12.97%	-32.66%	-2.24%
	soldier	6.93	9.36	5.97	1.19	4.53	5.72	-17.40%	-38.85%	-4.04%
	Average	**8.44**	**10.98**	**7.47**	-	-	**7.22**	**-14.59%**	**-34.84%**	**-3.59%**
Owlii (11-bit)	basketball	7.67	8.71	7.26	1.19	5.68	6.86	-10.55%	-21.15%	-5.42%
	dancer	7.74	8.78	7.27	1.20	5.74	6.94	-10.42%	-21.00%	-4.58%
	excercise	8.58	9.64	8.18	1.27	6.57	7.84	-8.58%	-18.64%	-4.06%
	model	7.64	8.63	6.99	1.21	5.54	6.74	-11.75%	-30.06%	-3.52%
	Average	**7.91**	**8.94**	**7.42**	-	-	**7.10**	**-10.32%**	**-20.66%**	**-4.40%**
MVUB (10-bit)	andrew	11.03	13.50	10.11	1.67	8.44	10.11	-8.34%	-25.10%	-0.02%
	david	7.25	9.05	6.47	1.16	5.17	6.33	-12.70%	-30.10%	-2.16%
	phil	10.24	12.54	9.16	1.58	7.54	9.12	-10.95%	-27.29%	-0.40%
	ricardo	5.88	7.11	5.24	0.95	3.98	4.93	-16.06%	-30.64%	-5.94%
	sarah	4.67	5.71	4.22	0.76	3.29	4.05	-13.23%	-29.13%	-4.00%
	Average	**7.81**	**9.58**	**7.04**	-	-	**6.91**	**-12.26%**	**-28.45%**	**-2.51%**

TABLE III
QUANTITATIVE COMPRESSION GAINS AGAINST EXISTING METHODS USING THE MIXED TRAINING DATASET

Class	Point Cloud	bits per point (bpp)								Compression Gain				
		G-PCC (Predlift)	V-PCC (VTM)	CNeT	MNeT	CrossPCAC	Proposed ConPCAC Group 1	Group 2-8	Overall	vs. G-PCC	vs. V-PCC	vs. CNeT	vs. MNeT	vs. CrossPCAC
8iVFB (10-bit)	loot	6.06	8.22	4.65	8.38	5.18	1.03	4.10	5.13	-15.36%	-37.57%	9.36%	-38.78%	-0.96%
	redandblack	9.20	11.90	7.21	12.23	8.07	1.58	6.45	8.03	-12.80%	-32.53%	10.17%	-34.37%	-0.55%
	Average	**7.63**	**10.06**	**5.93**	**10.31**	**6.63**	-	-	**6.58**	**-14.08%**	**-35.05%**	**9.76%**	**-36.58%**	**-0.75%**
Owlii (11-bit)	basketball	7.67	8.71	-	-	6.78	1.19	5.22	6.41	-16.52%	-26.42%	-	-	-5.52%
	dancer	7.74	8.78	-	-	6.80	1.20	5.29	6.49	-16.18%	-26.07%	-	-	-4.54%
	Average	**7.71**	**8.74**	-	-	**6.79**	-	-	**6.45**	**-16.35%**	**-26.24%**	-	-	**-5.03%**
MVUB (10-bit)	phil	10.24	12.54	4.47	10.49	6.78	1.58	4.68	6.26	-38.91%	-50.12%	28.54%	-40.37%	-7.73%
	ricardo	5.88	7.11	2.61	7.31	3.59	0.95	2.62	3.57	-39.20%	-49.76%	26.94%	-51.13%	-0.50%
	Average	**8.06**	**9.83**	**3.54**	**8.90**	**5.19**	-	-	**4.91**	**-39.06%**	**-49.94%**	**27.74%**	**-45.75%**	**-4.12%**
Other	queen	7.66	8.76	7.96	-	7.43	1.23	6.01	7.24	-5.48%	-17.33%	-9.07%	-	-2.61%
	thaidancer	7.65	9.02	7.02	-	6.82	1.36	5.22	6.57	-14.01%	-27.14%	-6.32%	-	-3.61%
	Average	**7.65**	**8.89**	**7.49**	-	**7.13**	-	-	**6.91**	**-9.74%**	**-22.24%**	**-7.69%**	-	**-3.11%**

[1]Results of MNet are directly copied from its paper as the open-source checkpoint has expired.

angular coding mode of G-PCC when compressing LiDAR point clouds, based on the assumption that no prior knowledge from the point cloud acquisition stage is utilized. We also compare with V-PCC version TMC2v22.1,[5] which uses VTM version 13.0[6] as the lossless attribute coder. Additionally, we compare with existing learning-based methods, including CNeT [32], MNeT [33], and CrossPCAC [31]. We try our best to follow their training datasets to retrain ConPCAC for fair comparisons.

Extensive experiments show that using G-PCC or the end-to-end coder as the base coder leads to comparable performance (referring to the ablation study in Section IV). In the next, we exemplify our ConPCAC using eight groups ($L = 8$) with G-PCC as the base coder.

4) Loss Function: The neural coder is trained to minimize all groups' total bitrate except the first group using G-PCC. To this end, we set the loss function as:

$$Loss = \sum_{i}^{L} -\log_2\left(prob\left(F^i\right)\right), \tag{5}$$

where F^i is the attribute of each group and $prob(\cdot)$ represents the corresponding probability.

[5]https://github.com/MPEGGroup/mpeg-pcc-tmc2
[6]https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM

B. Performance Evaluation

Table II presents the overall bitrate and the compression ratio gain of ConPCAC against G-PCC, V-PCC, and CrossPCAC. Both ConPCAC and CrossPCAC are trained using the RWTT training set. We list the specific bpp for group one and groups two to eight in ConPCAC for observation. As seen, ConPCAC uniformly outperforms G-PCC in three testing datasets, gaining 14.59%, 10.32%, and 12.26%. Compared to CrossPCAC, ConPCAC also has 3.59%, 4.40%, and 2.51% gains on 8iVFB, Owlii, and MVUB datasets, respectively. As our ConPCAC progressively encodes from the first group to the last one, the bpp consumption of each group is gradually decreased, as depicted in Fig. 8. These findings validate the effectiveness of our conditional coding: as the prior knowledge increases, ConPCAC can effectively leverage priors across groups for efficient compression.

1) Compare With CrossPCAC: Although CrossPCAC also adopts a group-wise solution, it is substantially different from our ConPCAC. Built on a multiscale framework, CrossPCAC performs PCAC from the lowest scale to the highest scale, with each scale producing an individual bitstream. The decoded lower scale is used to initialize the higher scale. Within each scale, CrossPCAC applies stage-wise processing to exploit stage correlations: after coding points in a stage, a rules-based

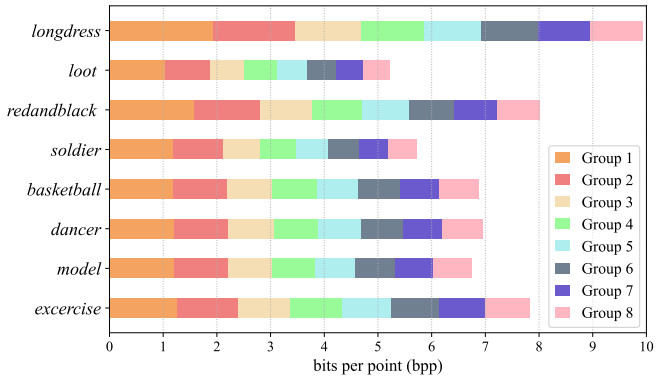


Fig. 8. Bitrate consumption of each group in ConPCAC. Eight groups are exemplified.

updating stage is executed to update all values of the remaining stages. Overall, CrossPCAC follows a "initialization - updating" paradigm, in which the initialization of a scale relies solely on its lower scale, compromising the accuracy; the rules-based updating stage yields gains but is time-consuming due to its cascading process. As a result, a long decoding time is required.

By contrast, our proposed ConPCAC exploits group correlations in the native resolution, which better utilizes priors than the multiscale CrossPCAC. Moreover, ConPCAC merely involves the initialization stage (using decoded groups), which significantly saves decoding time. Experimental results in Fig. 1 sufficiently prove that our ConPCAC not only achieves higher coding gains but also less computational complexity than CrossPCAC.

2) Compare With CNet and MNet: We further compare ConPCAC with other state-of-the-art learning-based methods like CNeT and MNeT using a similar mixed training dataset as theirs, as shown in Table III. It is worth pointing out that the open-source website of CNet[7] only provides the encoding source code, without checkpoint and decoding source code. We fix and retrain its network to obtain models for comparison.

As seen, ConPCAC achieves an average of 36.58% and 45.75% gain over MNeT in 8iVFB and Owlii, respectively. In comparison to CNeT, ConPCAC attains 9.76% and 27.74% loss. Note that CNeT improves performance by employing an autoregressive coding mechanism, which requires a much larger model size (>1GB for each color component, 28× of ours) and intolerable coding time (33× of ours). Fig. 1 comprehensively illustrates the compression gain and computational complexity across various methods. Our method achieves the optimal trade-off: it exhibits compression performance secondary to the autoregressive model while maintaining complexity comparable to the rules-based G-PCC.

Considering that CNet uses quite large models, we suspect that overfitting may occur. To this end, we test different methods trained on the same mixed dataset using two additional point clouds (*queen* and *thaidancer*). These point clouds are widely used in MPEG PCC for performance evaluation. They have similar densities to our previous testing samples but are excluded from the training dataset. As shown in Table III, CNet loses its leading position and even performs worse than the anchor G-PCC on *queen*. This may be due to CNet's overfitting to the specific dataset. In contrast, our method achieves the best performance, surpassing CNet by 7.69% and CrossPCAC by 3.11%. This demonstrates that our ConPCAC is more generalizable and robust across different scenarios.

3) Compare on the Reflectance Attribute: In addition to the color attribute, we also evaluate ConPCAC on the reflectance attribute on ScanNet and Ford datasets. As V-PCC and other learning-based methods don't provide results on these two datasets, we compare ConPCAC with G-PCC only. As presented in Table IV, ConPCAC consistently outperforms G-PCC, *e.g.*, 6.86% on ScanNet and 3.58% on Ford. These results also confirm the generalization of ConPCAC: it is effective on various point cloud types (solid, sparse, and LiDAR point clouds) and attributes (color and reflectance attributes).

[7]https://github.com/Weafre/CNeT

TABLE IV
COMPRESSION PERFORMANCE OVER G-PCC ON SCANNET (Q2CM) AND FORD (Q1MM) DATASETS

Class	bits per point (bpp)				Gain
	G-PCC (Predlift)	Proposed ConPCAC			vs. G-PCC
		Group 1	Group 2-8	Overall	
ScanNet	13.12	1.82	10.40	12.22	-6.86%
Ford	5.32	2.72	2.41	5.12	-3.58%

TABLE V
RUNTIME (SECONDS/FRAME) AND MODEL SIZE COMPARISON WITH STATE-OF-THE-ART METHODS

	G-PCC	V-PCC	CNeT	MNeT	CrossPCAC	ConPCAC (light)	ConPCAC
Enc.	10.4	-	25	3	17.6	1.8	5.2
Dec.	10.2	-	590	-	19.7	6.5	13.3
Total	20.6	697.1	615	-	37.3	8.4	18.5
Model Size	-	-	3.2 GB	12 MB	210 MB	33 MB	114 MB

[1]Runtime of CNet and MNet are directly copied from their papers.

C. Complexity

Table V reports the computational complexity of compared methods measured using runtime and model size. For a fair comparison with existing works CNet and MNet, we test our runtime complexity using the same point clouds as theirs, including four 10-bit objective point clouds (*loot* and *redandblack* from 8iVFB, *phil* and *ricardo* from MVUB). Notice that the runtime comparison is only intended as an intuitive reference to provide a glance at the computational complexity, as G-PCC and V-PCC run on the CPU while the other methods mainly run on the GPU. We can accelerate encoding by processing multiple groups in parallel. Such parallel processing is also used in CNet encoding but is challenging to implement in CrossPCAC due to its successive multiscale mechanism.

As shown, the encoding time of ConPCAC is 5.2 seconds per frame, and the decoding time is 13.3 seconds, which is in the same order of magnitude as that of G-PCC. In comparison, CNeT requires the longest decoding time (590 seconds) due to the autoregressive mechanism, about 33× of ours. Due to the multiscale coding and complex attribute updating mechanism,

TABLE VI
RUNTIME (SECONDS/FRAME) COMPARISON WITH G-PCC ON SCANNET AND FORD DATASETS

Class	G-PCC		ConPCAC	
	Enc.	Dec.	Enc.	Dec.
ScanNet	1.27	0.86	1.59	1.74
Ford	1.04	1.70	0.84	1.32

TABLE VII
QUANTITATIVE PERFORMANCE OF CONPCAC (LIGHT)

Class	bits per point (bpp)		Compression Gain
	ConPCAC (light)	ConPCAC	
8iVFB	7.47	7.22	-3.36%
Owlii	7.42	7.10	-4.37%
MVUB	7.22	6.91	-4.31%

TABLE VIII
ABLATION ON THE CODING STRATEGY OF THE FIRST GROUP

Dataset	First Group				Compression Gain	
	G-PCC Coder		E2E Coder		Using	Using
	bpp	time (s)	bpp	time (s)	G-PCC Coder	E2E Coder
8iVFB	1.43	2.27	1.37	9.27	-14.59%	-15.11%
Owlii	1.22	10.56	1.21	16.77	-10.32%	-10.43%
MVUB	1.23	4.50	1.17	11.40	-12.26%	-12.37%

the overall runtime of CrossPCAC is as long as 37.3 seconds, which is 2× of ours. The average runtime of ConPCAC on sparse point cloud datasets, including ScanNet and Ford, is reported in Table VI. The runtime of ConPCAC on LiDAR point clouds is also observed to be of the same order of magnitude as that of G-PCC.

Table V also details the model size of various methods. The model size here is the sum of Y, Co, and Cg models. As observed, CNet requires as large as 3.2GB (in our reproduction), making it impractical for real-world use. CrossPCAC needs about 210MB model size. Instead, the proposed ConPCAC, including Y, Co, and Cg models, only costs 114MB model size, which is 3.5% of CNet and 57% of CrossPCAC. Although MNeT only requires 12MB, its performance is largely inferior to ours (see Table III).

D. Ablation Studies

Ablation studies are conducted to validate ConPCAC.

1) ConPCAC (light): A light ConPCAC is implemented by reducing the number of feature maps in ConPCAC from 128 to 64. Table VII shows that ConPCAC (light) receives 3.36% to 4.37% loss compared to the full version. However, its computational complexity is largely reduced, with only 29% (33MB vs. 114MB) model size and 49% (6.5 s vs. 13.3) decoding time (see Table V). Figure 1 presents that ConPCAC (light) achieves comparable compression gains to CrossPCAC with much less computational complexity. Overall, ConPCAC (light) is more friendly to resource-constrained platforms. Practitioners can choose flexibly between the light and full versions based on the specific requirements of their applications.

2) End-to-End Coding: As shown in Fig. 4, we implement an end-to-end coder in ConPCAC as an alternative to encode the first group. Table VIII reveals a slight compression gain

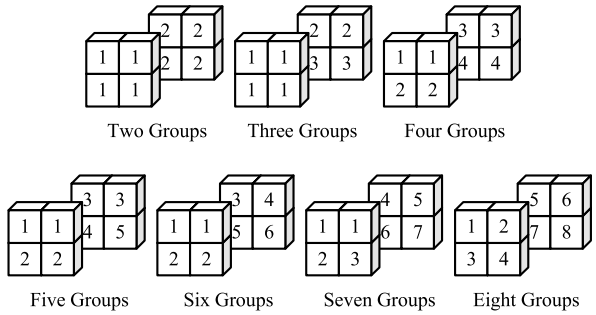


Fig. 9. Grouping strategy used to form different numbers of groups.

TABLE IX
ABLATION ON THE NUMBER OF GROUPS L

Class	Compression Gain vs. G-PCC						
	L=2	L=3	L=4	L=5	L=6	L=7	L=8
8iVFB	-2.73%	-5.33%	-9.36%	-9.48%	-9.60%	-10.19%	-14.59%
Owlii	-0.64%	-2.53%	-3.80%	-3.92%	-5.06%	-5.19%	-10.32%
MVUB	-2.08%	-2.09%	-5.28%	-5.66%	-6.82%	-7.07%	-12.26%

TABLE X
ABLATION STUDY ON THE INTERPOLATION OPERATION BASED ON T NEAREST NEIGHBORS

Class	bits per point (bpp)			Compression Gain	
	T=1	T=2	T=3	T=1 vs. T=2	T=1 vs. T=3
8iVFB	7.22	7.38	7.42	-2.10%	-2.73%
Owlii	7.10	7.23	7.25	-1.78%	-2.08%
MVUB	6.91	7.14	7.16	-3.25%	-3.51%

TABLE XI
ABLATION STUDY ON THE kNN TRANSFORMER

Class	bits per point (bpp)		Compression Gain
	w/o kNN Trans.	w/ kNN Trans.	w/ vs. w/o kNN Trans.
8iVFB	7.53	7.22	-4.31%
Owlii	7.51	7.10	-5.79%
MVUB	7.29	6.91	-5.58%

(< 1%) but > 1.5× longer inference time in comparison to the use of G-PCC. The longer time is owing to the scale-by-scale processing. The negligible gain is mainly due to the much sparser distribution of downscaled point clouds, upon which G-PCC and the end-to-end coder perform comparably.

3) Multiple Groups: To investigate the influence of group size on compression performance, we partition a point cloud into multiple groups for separate compression, as illustrated in Fig. 9. Results in Table IX show that the gains improve as the number of groups increases. This phenomenon occurs because using more groups leads to more available context information during encoding, enhancing the entropy coding performance. Consequently, employing more groups results in a linear increase in runtime. In practical applications, we can flexibly construct the group size to adapt to different time complexity requirements.

4) Interpolation: The Interpolation operation is a crucial component in the Attribute Initialization unit, providing initial attribute values for subsequent groups. As the value of T plays a pivotal role in this process, we conducted ablation experiments to validate its impact. We separately set $T = 1$,

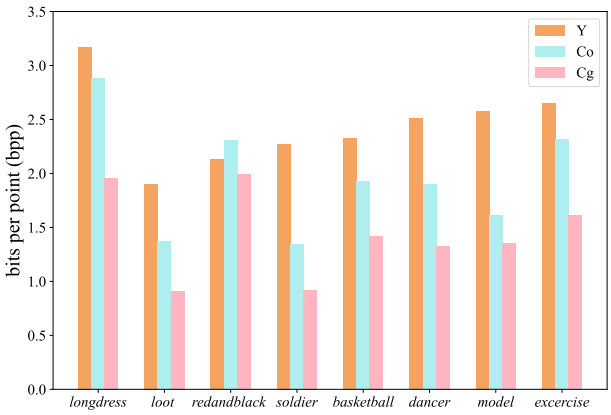


Fig. 10. Bitrate consumption of each color component on different point clouds.

TABLE XII
ABLATION STUDY ON THE CROSS-COMPONENT PREDICTION

Class	bits per point (bpp)		Compression Gain
	w/o Cross-comp.	w/ Cross-comp.	vs. w/o Cross-comp.
8iVFB	7.38	7.22	-2.21%
Owlii	7.37	7.10	-3.73%
MVUB	7.14	6.91	-3.29%

$T = 2$, and $T = 3$. Table X shows that $T = 1$ performs the best. This demonstrates that the closest points can provide better prior information. The mean of multiple-point attributes might interfere with its prediction.

5) kNN Transformer: In ConPCAC, we set $k = 8$ for the kNN Transformer in the LDP unit. We further conduct an ablation experiment on LDP, shown in Table XI, where the kNN Transformer is disabled to compare with the anchor ConPCAC. It is observed that without kNN Transformer, the compression results of ConPCAC increase from 7.22 bpp to 7.53 bpp on the 8iVFB dataset, introducing 4.31% loss. The loss is even larger on the other two datasets: 5.79% on Owlii and 5.58% on MVUB. This confirms the advantage of using kNN Transformer in LDP.

6) Cross-Component Prediction: As aforementioned, we use the cross-component prediction to process Co and Cg. Fig. 10 illustrates the bitrate distribution of Y, Co, and Cg on different point clouds when using this strategy. It is observed that with the guidance of Y, the bitrate of Co is significantly decreased. It is the same case with Cg. For comparison, we disable the cross-component prediction by feeding only Co or Cg to the neural coder. Results in Table XII show a 2.21% to 3.73% decrease in compression gain.

V. CONCLUSION

This paper proposed a lossless point cloud attribute compression method, dubbed ConPCAC. ConPCAC offers a simple yet efficient solution by fully utilizing group-wise conditional entropy coding. It first decomposes the original point cloud attribute into multiple groups by spatial decomposition. Then, conditioned on their previously decoded counterparts, the current and succeeding groups are progressively coded. Specifically, the first group is compressed using a base coder such as the standard-compliant G-PCC coder, providing a base representation of the input point cloud. The remaining groups are compressed using a neural coder, where the attribute initialization and latent distribution probability units are devised to predict the attribute probability of each point in a group. Experimental results confirm the high compression performance of the proposed ConPCAC. Tested under the common test condition defined in MPEG AI-PCC, ConPCAC significantly surpasses rules-based G-PCC and most learning-based methods. Additionally, its low complexity is friendly to real-life applications.

REFERENCES

[1] C. Cao, M. Preda, and T. B. Zaharia, "3D point cloud compression: A survey," in *Proc. 24th Int. Conf. 3D Web Technol.*, 2019, pp. 1–9.
[2] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2020.
[3] C. Cao, M. Preda, V. Zakharchenko, E. S. Jang, and T. Zaharia, "Compression of sparse and dense dynamic point clouds—Methods and standards," *Proc. IEEE*, vol. 109, no. 9, pp. 1537–1558, Sep. 2021.
[4] S. Schwarz et al., "Emerging MPEG standards for point cloud compression," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 133–148, Mar. 2019.
[5] H. Liu, H. Yuan, Q. Liu, J. Hou, and J. Liu, "A comprehensive study and comparison of core technologies for MPEG 3-D point cloud compression," *IEEE Trans. Broadcast.*, vol. 66, no. 3, pp. 701–717, Sep. 2020.
[6] M. Quach, J. Pang, D. Tian, G. Valenzise, and F. Dufaux, "Survey on deep learning-based point cloud compression," *Frontiers Signal Process.*, vol. 2, Feb. 2022, Art. no. 846972.
[7] Q. Zhang, J. Hou, Y. Qian, Y. Zeng, J. Zhang, and Y. He, "Flattening-net: Deep regular 2D representation for 3D point cloud analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 8, pp. 9726–9742, Aug. 2023.
[8] Y. Zeng, J. Hou, Q. Zhang, S. Ren, and W. Wang, "Dynamic 3D point cloud sequences as 2D videos," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 9371–9386, Dec. 2024.
[9] C. Fu, G. Li, R. Song, W. Gao, and S. Liu, "Octattention: Octree-based large-scale contexts model for point cloud compression," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 625–633.
[10] A. Akhtar, W. Gao, L. Li, Z. Li, W. Jia, and S. Liu, "Video-based point cloud compression artifact removal," *IEEE Trans. Multimedia*, vol. 24, pp. 2866–2876, 2021.
[11] F. Song, Y. Shao, W. Gao, H. Wang, and T. Li, "Layer-wise geometry aggregation framework for lossless LiDAR point cloud compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4603–4616, Dec. 2021.
[12] P. Gao, S. Luo, and M. Paul, "Rate-distortion modeling for bit rate constrained point cloud compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 5, pp. 2424–2438, May 2023.
[13] 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 Trans. Broadcast.*, vol. 68, no. 3, pp. 620–635, Sep. 2022.
[14] T. Fan, L. Gao, Y. Xu, Z. Li, and D. Wang, "D-DPCC: Deep dynamic point cloud compression via 3D motion prediction," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 898–904.
[15] S. Xia, T. Fan, Y. Xu, J.-N. Hwang, and Z. Li, "Learning dynamic point cloud compression via hierarchical inter-frame block matching," in *Proc. 31st ACM Int. Conf. Multimedia*, Oct. 2023, pp. 7993–8003.
[16] Q. Zhang, J. Hou, Y. Qian, A. B. Chan, J. Zhang, and Y. He, "RegGeoNet: Learning regular representations for large-scale 3D point clouds," *Int. J. Comput. Vis.*, vol. 130, no. 12, pp. 3100–3122, Dec. 2022.
[17] T. Fan, L. Gao, Y. Xu, D. Wang, and Z. Li, "Multiscale latent-guided entropy model for LiDAR point cloud compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 12, pp. 7857–7869, May 2023.
[18] H. Liu, H. Yuan, J. Hou, R. Hamzaoui, and W. Gao, "PUFA-GAN: A frequency-aware generative adversarial network for 3D point cloud upsampling," *IEEE Trans. Image Process.*, vol. 31, pp. 7389–7402, 2022.

[19] T. Huang et al., "3QNet: 3D point cloud geometry quantization compression network," *ACM Trans. Graph.*, vol. 41, no. 6, pp. 1–13, Dec. 2022.

[20] M. A. Lodhi, J. Pang, and D. Tian, "Sparse convolution based octree feature propagation for LiDAR point cloud compression," in *Proc. ICASSP - IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[21] R. Song, C. Fu, S. Liu, and G. Li, "Efficient hierarchical entropy model for learned point cloud compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 14368–14377.

[22] J. Deng, Y. An, T. H. Li, S. Liu, and G. Li, "ScanPCGC: Learning-based lossless point cloud geometry compression using sequential slice representation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2024, pp. 8386–8390.

[23] R. Xue, J. Li, T. Chen, D. Ding, X. Cao, and Z. Ma, "NeRI: Implicit neural representation of LiDAR point cloud using range image sequence," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2024, pp. 8020–8024.

[24] J. Wang, D. Ding, Z. Li, X. Feng, C. Cao, and Z. Ma, "Sparse tensor-based multiscale representation for point cloud geometry compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 7, pp. 9055–9071, Jul. 2022.

[25] J. Wang, H. Zhu, H. Liu, and Z. Ma, "Lossy point cloud geometry compression via end-to-end learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4909–4923, Dec. 2021.

[26] J. Wang, D. Ding, Z. Li, and Z. Ma, "Multiscale point cloud geometry compression," in *Proc. Data Compress. Conf. (DCC)*, Mar. 2021, pp. 73–82.

[27] J. Pang, M. A. Lodhi, and D. Tian, "GRASP-Net: Geometric residual analysis and synthesis for point cloud compression," in *Proc. 1st Int. Workshop Adv. Point Cloud Compress., Process. Anal.*, Oct. 2022, pp. 11–19.

[28] X. Wu, P. Zhang, M. Wang, P. Chen, S. Wang, and S. Kwong, "Geometric prior based deep human point cloud geometry compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 9, pp. 8794–8807, Sep. 2024.

[29] C. Sun, H. Yuan, S. Li, X. Lu, and R. Hamzaoui, "Enhancing context models for point cloud geometry compression with context feature residuals and multi-loss," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 14, no. 2, pp. 224–234, Jun. 2024.

[30] D. T. Nguyen, M. Quach, G. Valenzise, and P. Duhamel, "Lossless coding of point cloud geometry using a deep generative model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4617–4629, Dec. 2021.

[31] J. Wang, D. Ding, and Z. Ma, "Lossless point cloud attribute compression using cross-scale, cross-group, and cross-color prediction," in *Proc. Data Compress. Conf. (DCC)*, Mar. 2023, pp. 228–237.

[32] D. T. Nguyen and A. Kaup, "Lossless point cloud geometry and attribute compression using a learned conditional probability model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 8, pp. 4337–4348, Jan. 2023.

[33] D. T. Nguyen, K. G. Nambiar, and A. Kaup, "Deep probabilistic model for lossless scalable point cloud attribute compression," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[34] *Information Technology—Coded Representation of Immersive Media Part 9: Geometry-based Point Cloud Compression*, Standard 23090-9:2023, Mar. 2023, p. 9.

[35] *Information Technology—Coded Representation of Immersive Media—Part 5: Visual Volumetric Video-based Coding (V3C) and Video-Based Point Cloud Compression (V-PCC)*, Standard 23090-5:2023, 2023, p. 5.

[36] R. L. de Queiroz and P. A. Chou, "Compression of 3D point clouds using a region-adaptive hierarchical transform," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3947–3956, Aug. 2016.

[37] K. Mammou, A. Tourapis, J. Kim, F. Robinet, V. Valentin, and Y. Su, *Lifting Scheme for Lossy Attribute Encoding in TMC1*, document ISO/IEC JTC1/SC29/WG11 m42640, San Diego, CA, USA, 2018.

[38] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649–1668, Dec. 2012.

[39] B. Bross, J. Chen, J. Ohm, G. J. Sullivan, and Y. Wang, "Developments in international video coding standardization after AVC, with an overview of versatile video coding (VVC)," *Proc. IEEE*, vol. 109, no. 9, pp. 1463–1493, Sep. 2021.

[40] S. Gu, J. Hou, H. Zeng, H. Yuan, and K.-K. Ma, "3D point cloud attribute compression using geometry-guided sparse representation," *IEEE Trans. Image Process.*, vol. 29, pp. 796–808, 2019.

[41] Y. Shen, W. Dai, C. Li, J. Zou, and H. Xiong, "Multi-scale structured dictionary learning for 3-D point cloud attribute compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 7, pp. 2792–2807, Jul. 2021.

[42] W. Zhu, Y. Xu, D. Ding, Z. Ma, and M. Nilsson, "Lossy point cloud geometry compression via region-wise processing," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4575–4589, Dec. 2021.

[43] X. Sheng, L. Li, D. Liu, Z. Xiong, Z. Li, and F. Wu, "Deep-PCAC: An end-to-end deep lossy compression framework for point cloud attributes," *IEEE Trans. Multimedia*, vol. 24, pp. 2617–2632, 2022.

[44] H. Liu, H. Yuan, Q. Liu, J. Hou, H. Zeng, and S. Kwong, "A hybrid compression framework for color attributes of static 3D point clouds," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 3, pp. 1564–1577, Apr. 2021.

[45] G. Fang, Q. Hu, H. Wang, Y. Xu, and Y. Guo, "3DAC: Learning attribute compression for point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 14799–14808.

[46] J. Wang and Z. Ma, "Sparse tensor-based point cloud attribute compression," in *Proc. IEEE 5th Int. Conf. Multimedia Inf. Process. Retr. (MIPR)*, Aug. 2022, pp. 59–64.

[47] J. Zhang, J. Wang, D. Ding, and Z. Ma, "Scalable point cloud attribute compression," *IEEE Trans. Multimedia*, early access, Nov. 9, 2023, doi: 10.1109/TMM.2023.3331584.

[48] J. Zhang, T. Chen, D. Ding, and Z. Ma, "YOGA: Yet another geometry-based point cloud compressor," in *Proc. 31st ACM Int. Conf. Multimedia*, Oct. 2023, pp. 9070–9081.

[49] R. Huang, P. Yu, S. Liao, and F. Liang, "Efficient point cloud attribute compression using rich parallelizable context model," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2024, pp. 8436–8440.

[50] F. Song, G. Li, X. Yang, W. Gao, and S. Liu, "Block-adaptive point cloud attribute coding with region-aware optimized transform," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 8, pp. 4294–4308, Aug. 2023.

[51] C. Zhang, D. Florencio, and C. Loop, "Point cloud attribute compression with graph transform," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 2066–2070.

[52] R. A. Cohen, D. Tian, and A. Vetro, "Attribute compression for sparse point clouds using graph transforms," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 1374–1378.

[53] S. Gu et al., "3D point cloud attribute compression via graph prediction," *IEEE Signal Process. Lett.*, vol. 27, pp. 176–180, Jan. 2020.

[54] J. Zhang, Y. Chen, G. Liu, W. Gao, and G. Li, "Efficient point cloud attribute compression framework using attribute-guided graph Fourier transform," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2024, pp. 8426–8430.

[55] J. Zhang, G. Liu, J. Zhang, D. Ding, and Z. Ma, "DeepPCC: Learned lossy point cloud compression," *IEEE Trans. Emerg. Topics Comput. Intell.*, early access, Oct. 11, 2024, doi: 10.1109/TETCI.2024.3467192.

[56] H. Malvar and G. Sullivan, *YCoCg-R: A Color Space With RGB Reversibility and Low Dynamic Range*, document JVT-I014r3, ISO/IEC JTC1/SC29/WG11, ITU-T SG16 Q.6, Jul. 2003.

[57] J. Rissanen and G. G. Langdon, "Arithmetic coding," *IBM J. Res. Develop.*, vol. 23, no. 2, pp. 149–162, Mar. 1979.

[58] K. Zhang, J. Chen, L. Zhang, X. Li, and M. Karczewicz, "Enhanced cross-component linear model for chroma intra-prediction in video coding," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3983–3997, Aug. 2018.

[59] E. d'Eon, T. M. B. Harrison, and P. A. Chou, *8i Voxelized Full Bodies—A Voxelized Point Cloud Dataset*, document m40059, SO/IEC JTC1/SC29, WG11/WG1, 117th MPEG Meeting, Geneva, Switzerland, Jan. 2017.

[60] J. Ricard, C. Guede, R. Dore, and S. Lasserre, *CGI-based Dynamic Point Cloud Test Content*, document m40050, ISO/IEC JTC1/SC29/WG11, 117th MPEG Meeting, Geneva, Switzerland, Jan. 2017.

[61] M. Krivokuca, P. A. Chou, and P. Savill, *8i Voxelized Surface Light Field (8iVSLF) Dataset*, document m42914, ISO/IEC JTC1/SC29/WG11, 123th MPEG Meeting, Ljubljana, Slovenia, Jul. 2018.

[62] X. Yi, L. Yao, and W. Ziyu, *Owlii Dynamic Human Mesh Sequence Dataset*, document m41658, ISO/IEC JTC1/SC29/WG11, 120th MPEG Meeting, Macau, China, Oct. 2017.

[63] C. Loop, S. O. E. Q. Cai, and P. A. Chou, *Microsoft Voxelized Upper Bodies—A Voxelized Point Cloud Dataset*, document m38673, ISO/IEC JTC1/SC29 Joint WG11/WG1, 115th MPEG Meeting, Geneva, Switzerland, May 2016.

[64] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2432–2443.

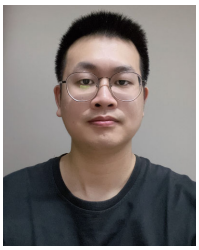
[65] S. Agarwal, A. Vora, G. Pandey, W. Williams, H. Kourous, and J. McBride, "Ford multi-AV seasonal dataset," *Int. J. Robot. Res.*, vol. 39, no. 12, pp. 1367–1376, Sep. 2020.

[66] WG 07 MPEG 3D graphics coding, *CTC on AI-based Point Cloud Coding*, document N01058, ISO/IEC JTC1/SC29/WG7, 148th MPEG Meeting, Kemer, Turkey, Nov. 2024.

[67] A. Maggiordomo, F. Ponchio, P. Cignoni, and M. Tarini, "Real-world textured things: A repository of textured models generated with modern photo-reconstruction tools," *Comput. Aided Geometric Design*, vol. 83, Nov. 2020, Art. no. 101943.

[68] A. Zaghetto, D. Graziosi, and A. Tabatabai, *Dataset Split for AI-PCC*, document m58754, ISO/IEC JTC1/SC29/WG7, 137th MPEG Meeting, Jan. 2022.

[69] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal ConvNets: Minkowski convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3070–3079.

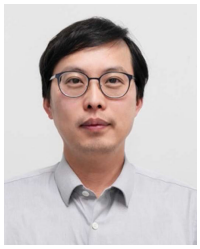


Junzhe Zhang received the B.S. degree in electronic information engineering from Zhejiang University of Science and Technology, Hangzhou, China, in 2021. He is currently pursuing the M.S. degree in electronic information with Hangzhou Normal University. His research interests include point cloud compression and processing.

Tong Chen (Member, IEEE) received the B.E., M.E., and Ph.D. degrees from Nanjing University, in 2015, 2018, and 2022, respectively. He is an Associate Researcher with the School of Electronic Science and Engineering, Nanjing University, Jiangsu, China. His current research focuses on image, video, and 3D data processing and compression, including deep learning-based coding frameworks. He was a co-recipient of the 2018 PCM Best Paper Finalist Award and the 2020 IEEE MMSP Image Compression Grand Challenge Best Performing Solution Award.

Kang You received the B.S. degree in computer science and technology from Wenzhou University, Wenzhou, China, in 2019, and the M.S. degree in computer science and technology from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2023. He is currently pursuing the Ph.D. degree with Nanjing University, Nanjing. His current research interests include deep learning and point cloud compression.

Dandan Ding (Senior Member, IEEE) received the B.Eng. degree (Hons.) in communication engineering and the Ph.D. degree from Zhejiang University, China, in 2006 and June 2011, respectively. From 2007 to 2008, she was an Exchange Student at the Microelectronic Systems Laboratory (GR-LSM), EPFL, Switzerland. She was a Post-Doctoral Researcher from July 2011 to June 2013 and then a Research Associate at Zhejiang University till 2015. Since 2016, she has been with the School of Information Science and Technology, Hangzhou Normal University, as a tenured Faculty Member. Her research interests include artificial intelligence-based point cloud processing and image/video coding.



Zhan Ma (Senior Member, IEEE) received the B.S. and M.S. degrees from the Huazhong University of Science and Technology, Wuhan, China, in 2004 and 2006, respectively, and the Ph.D. degree from New York University, New York, in 2011. From 2011 to 2014, he was with Samsung Research America, Dallas, TX, and Futurewei Technologies, Inc., Santa Clara, CA, respectively. He is a Full Professor with the School of Electronic Science and Engineering, Nanjing University, Jiangsu, China. His research focuses include learned image/video coding and computational imaging. He was a co-recipient of multiple awards including the 2019 IEEE Broadcast Technology Society Best Paper Award, the 2020 IEEE MMSP Grand Challenge Best Image Coding Solution, and the 2023 IEEE WACV Best Algorithms Paper Award.