Spatio-Temporal Context-Guided Algorithm for Lossless Point Cloud Geometry Compression

ZHANG Huiran[1, 2], DONG Zhen[3], WANG Mingsheng[1, 2]

(1. Guangzhou Urban Planning and Design Survey Research Institute, Guangzhou 510060, China;
2. Guangdong Enterprise Key Laboratory for Urban Sensing, Monitoring and Early Warning, Guangzhou 510060, China;
3. State Key Laboratory of Information Engineering in Surveying Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China)

DOI: 10.12142/ZTECOM.202304003

https://kns.cnki.net/kcms/detail/34.1294.TN.20231108.1004.002.html, published online November 8, 2023

Manuscript received: 2023-09-11

Abstract: Point cloud compression is critical to deploy 3D representation of the physical world such as 3D immersive telepresence, autonomous driving, and cultural heritage preservation. However, point cloud data are distributed irregularly and discontinuously in spatial and temporal domains, where redundant unoccupied voxels and weak correlations in 3D space make achieving efficient compression a challenging problem. In this paper, we propose a spatio-temporal context-guided algorithm for lossless point cloud geometry compression. The proposed scheme starts with dividing the point cloud into sliced layers of unit thickness along the longest axis. Then, it introduces a prediction method where both intra-frame and inter-frame point clouds are available, by determining correspondences between adjacent layers and estimating the shortest path using the travelling salesman algorithm. Finally, the few prediction residual is efficiently compressed with optimal context-guided and adaptive fast-mode arithmetic coding techniques. Experiments prove that the proposed method can effectively achieve low bit rate lossless compression of point cloud geometric information, and is suitable for 3D point cloud compression applicable to various types of scenes.

Keywords: point cloud geometry compression; single-frame point clouds; multi-frame point clouds; predictive coding; arithmetic coding

Citation (Format 1): ZHANG H R, DONG Z, WANG M S. Spatio-temporal context-guided algorithm for lossless point cloud geometry compression [J]. ZTE Communications, 2023, 21(4): 17 - 28. DOI: 10.12142/ZTECOM.202304003
Citation (Format 2): H. R. Zhang, Z. Dong, and M. S. Wang, "Spatio-temporal context-guided algorithm for lossless point cloud geometry compression," *ZTE Communications*, vol. 21, no. 4, pp. 17 - 28, Dec. 2023. doi: 10.12142/ZTECOM.202304003.

1 Introduction

With the improvement of multi-platform and multi-resolution acquisition equipment performance, light detection and ranging (LiDAR) technology can efficiently simulate 3D objects or scenes with massive point sets. Compared with traditional multimedia data, point cloud data contain more physical measurement information which represents objects from free viewpoints, even scenes with complex topological structures. This results in strong interactive and immersive effects that provide users with a vivid and realistic visualization experience. Additionally, point cloud data have stronger anti-noise ability and parallel processing capability, which seems to have gained attraction from the industry and academia, notably for application domains such as cultural heritage preservation, 3D immersive telepresence and automatic driving[1 - 2].

However, point cloud data usually contain millions to billions of points in spatial domains, bringing burdens and challenges to the storage space capacity and network transmission bandwidth. For instance, a common dynamic point cloud utilized for entertainment usually comprises roughly one million points per frame, which, at 30 frames per second, amounts to a total bandwidth of 3.6 Gbit/s if left uncompressed[3]. Therefore, the research on high efficiency geometry compression algorithms for point clouds has important theoretical and practical value.

Prior work tackled this problem by directly building grids or on-demand down-sampling, due to limitations in computer computing power and point cloud collection efficiency, which resulted in low spatio-temporal compression performance and loss of geometric attribute feature information. Recent studies were mainly based on computer graphics and digital signal processing techniques to implement block operations on point cloud data[4 - 5] or combined video coding technology[6 - 7] for optimization. In 2017, the Moving Picture Experts Group (MPEG) solicited proposals for point cloud compression and conducted subsequent discussions on how to compress this

type of data. With increasing approaches to point cloud compression available and presented, two-point cloud data compression frameworks—TMC13 and TMC2 were issued in 2018. The research above shows remarkable progress has been made in the compression technology of point cloud. However, prior work mostly dealt with the spatial and temporal correlation of point clouds separately but had not yet been exploited to their full potential in point cloud compression.

To address the aforementioned challenges, we introduce a spatio-temporal context-guided method for lossless point cloud geometry compression. We first divide point clouds into unit layers along the main axis. We then design a prediction mode via a travelling salesman algorithm, by adopting spatio-temporal correlation. Finally, the residuals are written into bitstreams with a utilized context-adaptive arithmetic encoder. Our main contributions are as follows.

1) We design a prediction mode applicable to both intra-frame and inter-frame point cloud, via the extended travelling salesman problem (TSP). By leveraging both the spatial and temporal redundancies of point clouds, the geometry prediction can make better use of spatial correlation and therefore enable various types of scenarios.

2) We present an adaptive arithmetic encoder with fast context update, which selects the optimal 3D context from the context dictionary, and suppresses the increase of entropy estimation. As a result, it enhances the probability calculation efficiency of entropy encoders and yields significant compression results.

The rest of this paper is structured as follows. Section 2 gives an outline of related work on point cloud geometry compression. Section 3 firstly presents an overview of the proposed framework. Then, the proposed method is descibed in detail. Experimental results and conclusions are presented in Sections 4 and 5, respectively.

2 Related Work

There have been many point cloud geometry compression algorithms proposed in the literature. CAO et al. [8] and GRAZIOSI et al. [9] conduct an investigation and summary of current point cloud compression methods, focusing on spatial dimension compression technology and MPEG standardization frameworks respectively. We provide a brief review of recent developments in two categories: single-frame point cloud compression and multi-frame point cloud compression.

2.1 Single-Frame Point Cloud Compression

Single-frame point clouds are widely used in engineering surveys, cultural heritage preservation, geographic information systems, and other scenarios. The octree is a widely used data structure to efficiently represent point clouds, which can be compressed by recording information through the occupied nodes. HUANG et al.[10] propose an octree-based method that recursively subdivides the point cloud into nodes with their positions represented by the geometric center of each unit. FAN et al.[11] further improve this method by introducing cluster analysis to generate a level of detail (LOD) hierarchy and encoding it in a breadth-first order. However, these methods can cause distortion due to the approximation of the original model during the iterative process.

To address these limitations, scholars have introduced geometric structure features, such as the triangular surface model[12], the planar surface model[13–14], and the clustering algorithm[15], for inter-layer prediction and residual calculation. RENTE et al.[16] propose a concept of progressive layered compression that first uses the octree structure for coarse-grained encoding and then uses the graph Fourier transform for compression and reconstruction of cloud details. In 2019, MPEG released the geometry-based point cloud compression (G-PCC) technology for both static and dynamic point clouds, which is implemented through coordinate transformation, voxelization, geometric structure analysis, and arithmetic coding step by step[17].

Since certain octants within an octree may be sparsely populated or even empty, some methods have been proposed to optimize the tree structure by pruning sub-nodes and therefore conserve memory allocation. For example, DRICOT et al. [18] propose an inferred direct coding mode (IDCM) for terminating the octree partition based on predefined conditions of sparsity analysis, which involves pruning the octree structure to save bits allocated to child nodes. ZHANG et al. [19] suggest subdividing the point cloud space along principal components and adapting the partition method from the binary tree, quadtree and octree. Compared with the traditional octree partitioning, the hybrid models mentioned above can effectively reduce the number of bits used to represent sparse points, therefore saving nodes that need to be encoded. However, complex hyperparameter conditions and mode determination are required in the process, making it difficult to meet the requirements of self-adaptation and low complexity.

With deep neural networks making significant strides in image and video compression, researchers have explored ways to further reduce bit rates by leveraging super prior guidance and the redundancy of latent space expression during the compression process. QUACH et al.[20] and HUANG et al.[21] propose methods that incorporate these concepts. GUARDA et al. combine convolutional neural networks and autoencoders to exploit redundancy between adjacent points and enhance coding adaptability in Ref. [22]. Recently, WANG et al.[23] propose a point cloud compression method based on the variational auto-encoder, which improves the compression ratio by learning the hyperprior and reducing the memory consumption of arithmetic coding. The aforementioned methods use neural network encoders to capture the high-order hidden vector of the point cloud, the entropy model probabilities, and the edge probabilities of which fit better, thus reducing the memory consumption of arithmetic coding.

Generally speaking, the research on single-frame point cloud geometric compression is relatively mature, but there are two challenges that remain yet. Spatial correlation has not been utilized effectively, and most methods do not code the correlation of point cloud data thoroughly and efficiently. Besides, the calculation of the probability model for entropy coding appears long and arduous due to the massive number of contexts.

2.2 Multi-Frame Point Cloud Compression

Multi-frame point clouds are commonly used in scenarios such as real-time 3D immersive telepresence, interactive VR, 3D free viewpoint broadcasting and automatic driving. Unlike single-frame point cloud compression, multi-frame point cloud compression prioritizes the use of time correlation, as well as motion estimation and compensation. The existing methods for multi-frame point cloud compression can be divided into two categories: 2D projection and 3D decorrelation.

The field of image and video compression is extensive and has been well-explored over the past few decades. Various algorithms convert point clouds into images and then compress them straightforwardly by FFmpeg and H.265 encoders, etc. AINALA et al[24] introduce a planar projection approximate encoding mode that encodes both geometry and color attributes through raster scanning on the plane. However, this method causes changes in the target shape during the mapping process, making accurate inter-prediction difficult. Therefore, SCHWARZ et al.[25] and SEVOM et al.[26] suggest rotated planar projection, cube projection, and patch-based projection methods to convert point clouds into 2D videos, respectively. By placing similar projections in adjacent frames at the same location in adjacent images, the video compressor can fully remove temporal correlation. In Ref. [27], inter-geometry prediction is conducted via TSP, which computes the one-to-one correspondence of adjacent intra-blocks by searching for the block with the closest average value. MPEG released the video-based point cloud compression (V-PCC) technology for dynamic point clouds in 2019[28]. This framework divides the input point cloud into small blocks with similar normal vectors and continuous space, then converts them to the planar surface through cubes to record the occupancy image and auxiliary information. All resulting images are compressed by mature video codecs, and all bitstreams are assembled into a single output file. Other attempts have been made to improve the effectiveness of these methods. COSTA et al.[29] exploit several new patch packaging strategies from the perspective of optimization for the packaging algorithm, data packaging links, related sorting, and positioning indicators. Furthermore, PARK et al.[30] design a data-adaptive packing method that adaptively groups adjacent frames into the same group according to the structural similarity without affecting the performance of the V-PCC stream.

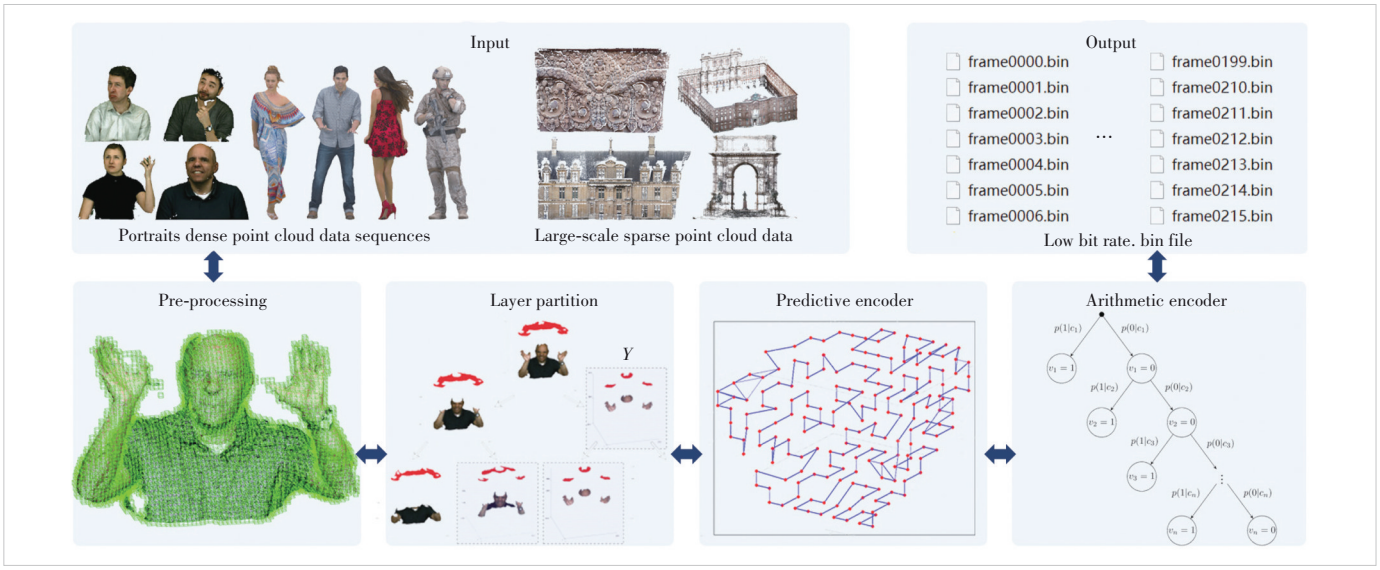
Due to the inevitable information loss caused by point cloud projection, scholars have developed effective techniques to compress the point cloud sequence of consecutive frames using motion compensation technology based on 3D space. KAMMERL et al.[31] propose an octree-based geometric encoding method, which achieves high compression efficiency by performing the exclusive OR (XOR) differences between adjacent frames. This method has been not only adopted in the popular Point Cloud Library (PCL)[32] but also widely used for further algorithm research. Other interframe approaches convert the 3D motion estimation problem into a feature matching problem[33] or use reconstructed geometric information[34] to predict motion vectors and identify the corresponding relationship between adjacent frames accurately. Recent explosive studies[35-36] have shown that the learned video compression offers better rate-distortion performance over traditional ones, bringing significant reference significance to point cloud compression. ZHAO et al.[37] introduce a bi-directional inter-frame prediction network to perform inter-frame prediction and bring effective utilization of relevant information in spatial and temporal dimensions. KAYA et al.[38] design a new paradigm for encoding geometric features of dense point cloud sequences, optimizing the CNN for estimating the encoding distribution to realize lossless compression of dense point clouds.

Despite progress in the compression coding technology of multi-frame point cloud models, two problems persist. The existing multi-frame point cloud compression approaches mainly rely on video coding and motion compensation, which inevitably involves information loss or distortion caused by mapping and block edge discontinuity. In addition, predictive coding exhibits low applicability due to the inconsistency of inter-frame point cloud geometry. The apparent offset of points between frames and the unavoidable noise increases the difficulty of effectively using predictive coding in inter-frame compression.

3 Proposed Spatio-Temporal Context-Guided Lossless Geometry Point Cloud Compression Method

3.1 Overview

The overall pipeline of our spatio-temporal context-guided algorithm is shown in Fig. 1. First, we preprocess the input point cloud by applying voxelization and scale transformation. Then, the point cloud is divided into unit thickness sliced layers along the main axis. Next, we design a prediction mode that makes full use of the temporal and spatial correlation information within both intra-frame and inter-frame. We calculate the shortest path of points of reference layers (R-layers) via travelling salesman algorithms, and the results of the R-layers are then used to predict spatio-temporally and encode the rest of the point clouds, namely predicted layers (P-layers). Finally, the improved entropy coding algorithms are adopted to obtain the compressed binary file.



▲Figure 1. Proposed framework for spatio-temporal context-guided lossless point cloud geometry compression

3.2 Image Sliced-Based Hierarchical Division

1) Pre-processing

The pre-processing module includes voxelization and scale transformation, for better indexing of each certain point. In voxelization, we divide the space into cubes of size N, which corresponds to the actual resolution of the point cloud. Each point is assigned a unique voxel based on its position. A voxel is recorded as 1; if it is positively occupied, it is 0 otherwise.

Scale transformation can reduce the sparsity for better compression by zooming out the point cloud, where the distance between points gets smaller. We aggregate the point cloud coordinates (x, y, z) using a scaling factor s, i.e.,

$$\hat{P}_n = P_n \times s = (x_n \times s, y_n \times s, z_n \times s), s \leqslant 1 . \quad (1)$$

To ensure lossless compression, we need to ensure that the scaling factor s cannot cause geometry loss and needs to be recorded in the header file.

2) Sliced-layer division

This module works by dividing the 3D point cloud along one of its axes, creating several unit-sliced layers with occupied and non-occupied information only that can be further compressed using a predictive encoder and an arithmetic coder. The function is defined as:

$$S(a,b) = \text{slice}(G, \text{axis}) = \begin{cases} G(x, a, b), & \text{if axis} = X \\ G(a, y, b), & \text{if axis} = Y \\ G(a, b, z), & \text{if axis} = Z \end{cases}, \quad (2)$$

where G refers to the input point cloud coordinate matrix, axis refers to the selected dimension, and $S(a, b)$ is the 2D slice extracted by each layer.

In general, we conduct experiments on a large number of test sequences, and results suggest that division along the longest axis of point cloud spatial variation yields the lowest bitrate, i.e.

$$\text{axis} = \begin{cases} X, \text{if} (x_{\max} - x_{\min}) \geqslant (y_{\max} - y_{\min}), (x_{\max} - x_{\min}) \geqslant (z_{\max} - z_{\min}) \\ Y, \text{if} (y_{\max} - y_{\min}) > (x_{\max} - x_{\min}), (y_{\max} - y_{\min}) \geqslant (z_{\max} - z_{\min}) \\ Z, \text{if else} \end{cases} . \quad (3)$$

3) Minimum bounding box extraction

In most cases, on-occupied voxels are typically unavoidable and greatly outnumber occupied voxels. As a result, processing and encoding both types of voxels simultaneously burdens the computational complexity and encoding speeds of the compression algorithm. Therefore, we adopt the oriented bounding box (OBB) [39] to calculate the minimum bounding box for each sliced layer, ensuring that the directions of the bounding boxes are consistent across layers. In subsequent processing, only the voxels located within the restricted rectangle are compressed.

3.3 Spatial Context-Guided Predictive Encoding

The goal of spatial context-guided predictive encoding is to encode all the points layer by layer. Inspired by the TSP, we design a prediction mode to explore the potential orders and correlation within each sliced layer. This module consists of partition and the shortest path calculation.

At first, we partition the sliced layers and determine the R-layer and R-layers for each group. We traverse the point cloud layer by layer along the selected axis. When the length of the main direction of the minimum bounding box between adjacent layers differs by a specified unit length, it is recorded as the same group. Otherwise, it is used as the reference layer of

the next group, and each point cloud in the following group uses the same shortest path. In this paper, we set the first layer of each group as the R-layer, and the others as P-layers. We also carry out experiments on a large number of test sequences and recommend setting this specified parameter as 3 units to obtain the best compression.

Afterwards, we conduct the shortest path calculation on the R-layers and record the residuals of P-layers. According to the distribution regulation of the point cloud of each slice layer, we optimally arrange the irregular point clouds for each slice layer based on the TSP algorithm. This allows us to efficiently compute the shortest path to the point cloud of the R-layers, and then record the residuals of the corresponding prediction layers. Algorithm 1 shows the pseudo-code of the prediction procedure.

Algorithm 1. Spatial context-guided predictive encoding

Input: point cloud sliced-layers
Output: the shortest path $\min \sum_{i,j=1}^{n-1} \mathrm{dist}(\mathrm{pc}_i, \mathrm{pc}_j)$, the shortest path record tables of R-layers, and predictive residuals
Definition: $\mathrm{dist}\left(\mathrm{pc}_i, \mathrm{pc}_j\right) = \mathrm{norm}(\mathrm{pc}_i, \mathrm{pc}_j)$
Initialization: randomly selected point pc_1
while add a new point pc_i **do :**
    $\mathrm{path}(P, \mathrm{init}) = min\{\mathrm{path}(P-i, i) + \mathrm{dist}[i][\mathrm{init}]\}, \forall t \in P$
end while
    return $\min \sum_{i,j=1}^{n-1} \mathrm{dist}(\mathrm{pc}_i, \mathrm{pc}_j)$ and shortest path record tables of R-layers
for P-layers under-process **do :**
        R-frame $\mathrm{distPC}_i = \min \sum_{i,j=1}^{n-1} \mathrm{dist}(\mathrm{pc}_i, \mathrm{pc}_j)$
        calculate $\mathrm{residuals}_i = \mathrm{diff}(\mathrm{PC}_i(P,:))$
end for
    return $\mathrm{residuals}_i$

Firstly, we define the distance calculation rule between points in the local area and initialize the path state with a randomly selected point pc_1. In each iteration, whenever a new point pc_i is added, the permutation is dynamically updated through the state transition equation $\mathrm{path}(P-i, i)$ until all added points are recorded in P in the order of the shortest path. This process is modified gradually based on the minimal distance criterion. After all iterations are completed in the total shortest path, we calculate the $\min \sum_{i,j=1}^{n-1} \mathrm{dist}(\mathrm{pc}_i, \mathrm{pc}_j)$ in each of the R-layers, and return the shortest path record table of point clouds in each of the R-layers. For further compression, we calculate the deviation of the P-layers from the shortest path of the R-layer within the same group and record them as predictive residuals. Finally, the shortest path of the R-layer and the residuals of each group are output and passed to the entropy encoder to compress prediction residuals further.

3.4 Spatio-Temporal Context-Guided Predictive Encoding

The spatial context-guided prediction mode encodes single-frame point clouds individually. However, applying spatial encoding to each single-frame point cloud separately can miss out on opportunities exposed by the temporal correlations across multi-frame point cloud. Considering that multi-frame point cloud shares large chunks of overlaps, we focus on using temporal redundancy to further enhance the compression efficiency. Hence, based on the proposed spatial context-guided prediction mode, we can compress multi-frame point cloud by identifying a correspondence between adjacent layers across frames.

1) Inter-frame partition

To enhance the effectiveness of inter-frame prediction mode, it is crucial to ensure adequate similarity between adjacent layers of frames. As a result, we need to partition the groups between adjacent frames and determine the R-layers and P-layers across frames. By estimating the shortest path of the P-layers based on the shortest path of the R-layers, we record the prediction residuals and further compress them through the entropy encoder. Algorithm 2 shows the pseudo-code of the inter-frame partition.

Algorithm 2. Inter-frame partition

Input: point cloud sliced-layers $S_1, S_2, \cdots, S_n$, and principal axis lengths h_i of S_i inter-frame point cloud sliced layers $SS_1, SS_2, \cdots, SS_n$, and principal axis lengths hh_i of SS_i
Output: correspondence and partition of the adjacent layers' relationship
Initialization: set S_1 and SS_1 as corresponding layers
for new S_i and SS_i **do :**
    coarse partition: set S_i and SS_i as corresponding layers
    if $\left|h_i - hh_i\right| \leqslant 3$ **:**
    fine partition: set S_i and SS_i as corresponding layers
    else if
    compare $\left|h_i - hh_i\right|$, $\left|h_i - hh_{i-1}\right|$, and $\left|h_i - hh_{i+1}\right|$, and pick the minimum
    set the slice layer corresponding to the minimum and SS_i as corresponding layers
    else
    set as a single layer
end for

Based on sliced-layers orientation alignment, we realize coarse partition and fine partition successively. For coarse partition, we sort the sliced layers of each frame based on the coordinates corresponding to the division axes, from small to large. As a result, each slice layer of each frame has a unique layer number, allowing us to coarsely partition the slice layers with the same number between adjacent frames. Afterward, we compute the difference between the principal axis lengths of the minimum bounding boxes of adjacent layers with the same number. If this value is less than or equal to a specified length

unit, the layers will be partitioned into the same group. Otherwise, we compare the difference in the length of the main direction axis of the minimum bounding box in the corresponding layer of the adjacent frame with the specified layer before and after the number in the adjacent frame. The layer with the smallest difference is then partitioned into the same group. This ensures a fine partition between adjacent layers, and so as to realize the fine partition of the adjacent relationship.

2) Spatio-temporal context-guided prediction mode

Based on the partition, we apply and expand the prediction mode mentioned in Section 3.3. We incorporate inter-frame context in the process, meaning that the first layer of each group, which serves as the R-layer, may not necessarily yield the best prediction result. To fully explore the potential correlation between adjacent layers, we need to expose the optimal prediction mode.

Firstly, we calculate the prediction residuals for each sliced-layer in the current group when used as the R-layer. By comparing the prediction residuals in all cases, we select the R-layer with the smallest absolute residual value as the best prediction mode. For R-layer shortest path calculation, we use the travelling salesman algorithm to compute the shortest path of the R-layers under the best prediction mode. Moreover, we calculate the prediction residuals for each group under their respective best prediction modes. We also record the occupancy length and R-layer information of each group for further compression in subsequent processing.

In the follow-up operation, we use arithmetic coding based on the best context selection for the above information to complete the entire process of the multi-frame point cloud geometry compression algorithm.

3.5 Arithmetic Coding Based on Context Dictionary

The massive amount of context in point cloud significantly burdens the overall compression scheme in terms of arithmetic coding computational complexity. We improve the arithmetic coding from the following two modules. 1) We set up a context dictionary, and select and update the global optimal value according to the entropy estimate, and then 2) we adopt adaptive encoders to efficiently calculate the upper and lower bounds of probabilities.

1) Context dictionary construction

We construct a context dictionary that represents a triple queue, consisting of coordinates of the point cloud at each sliced-layer and the integer representation of its corresponding non-empty context. Thus, we associate the voxels contained in the point cloud with the minimum bounding box of each layer with its non-empty context. To illustrate the construction of the triple queue array of the context dictionary clearly, we give an intuitive explanation in Fig. 2.

For the shaded two squares in Fig. 2, only the context map positions pc_1 and pc_2 are considered. The context contribution along the x-axis and the y-axis is recorded to the two queues $\mathcal{Q}^{X-}$ and $\mathcal{Q}^{Y-}$ respectively. Thus the context dictionary consists of $\mathcal{Q}^{X-}$ and $\mathcal{Q}^{Y-}$. Queue elements with the same coordinates are integrated into a triplet, the context integer representation of which is computed as the sum of the context contributions of the merged triplet.

Therefore, the context of each voxel can be computed as the sum of the independent contributions of occupied voxels in its context dictionary. This structure helps determine whether a voxel should be added to the context dictionary without tedious matrix lookups, resulting in a significant reduction in computational complexity and runtime.

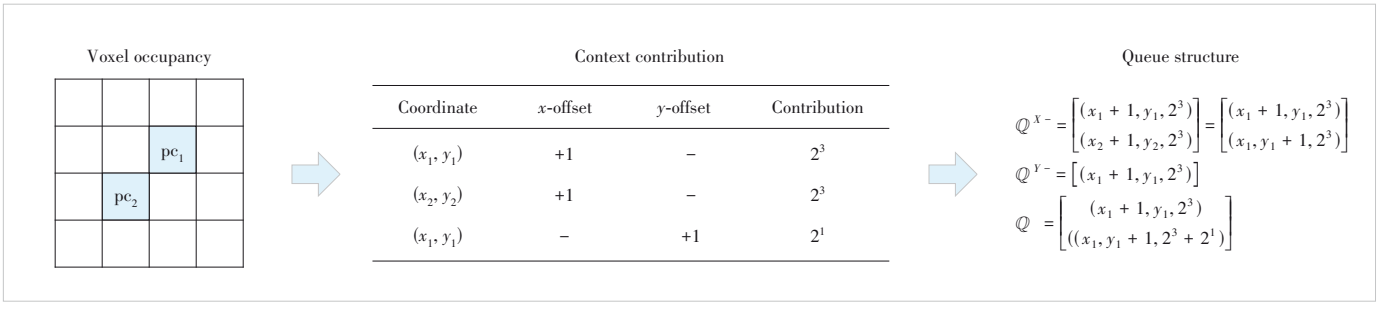
2) Probability calculation

To calculate entropy probability, both the length of the sequence and the context of its constituent voxels must be taken into account. In this module, we design an adaptive encoder that first estimates the upper and lower cumulative probability bounds for each group from the context dictionary, and then encodes it subsequently.

First of all, we construct a binary tree based on the Markov chain model. By traversing the occupancy of voxels, we assign values of 1 and 0 to occupied and empty voxels, respectively, and calculate the probability based on the tree structure. Starting from the root node, when a voxel is occupied, we record the left child node as 1. Otherwise, we mark the right child node as 0 and proceed to the next step of judgment and division. The calculation formula for the run probability of occupied voxels can be found in Eq. (4).

$$P(l) = p\left(1|c_i\right)\cdot\prod_{i-1}^{l-1} p(0|c_i)\,, \tag{4}$$

where l is the length of the run ending at the occupied voxel.



▲Figure 2. Construction of the context dictionary

For run lengths less than or equal to n, there may be $2n$ of tree nodes representing the occupancy states of voxels. Therefore, the probability of any occupied voxel is represented by the independent joint probability of traversing all states starting at the root and ending at any childless node of the tree.

Based on Eq. (4), to perform arithmetic encoding on the occupancy of the voxel sequence, we need the cumulative upper and lower probabilities of the sequence, as shown in Eq. (5).

$$\begin{cases} \mathrm{Low}(l) = \sum_{r=1}^{l-1} P(r) = \sum_{r=1}^{l-1} p\left(1|c_r\right) \cdot \prod_{i=1}^{r} p(0|c_i) \\ \mathrm{High}(l) = \sum_{r=1}^{l} P(r) = \sum_{r=1}^{l} p\left(1|c_r\right) \cdot \prod_{i=1}^{r} p(0|c_i) \end{cases}. \quad (5)$$

Employing this approach, we can utilize the adaptive properties of arithmetic coding to adjust the probability estimation value of each symbol based on the optimized probability estimation model and the frequency of each symbol in the current symbol sequence. This allows us to calculate the upper and lower bounds of the cumulative probability of occupied voxels and complete the encoding process.

4 Experiment

4.1 Implementation Details

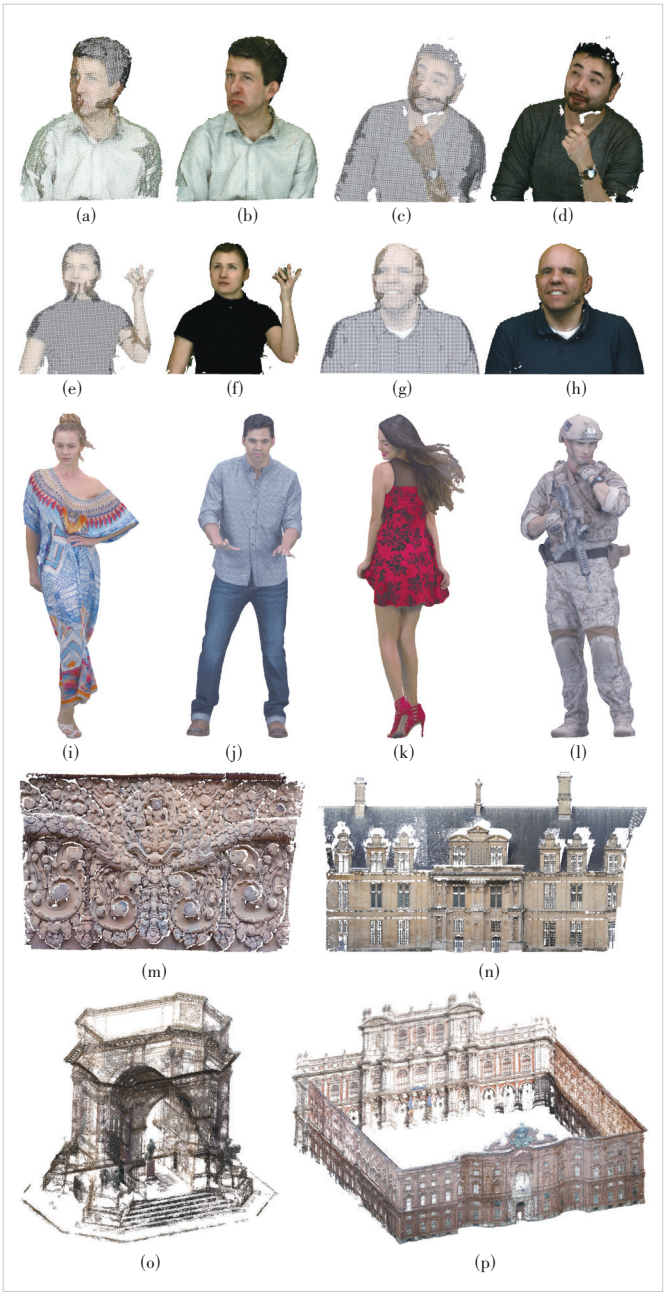
1) Dataset. To verify the performance of our proposed method, extensive experiments were conducted over 16 point cloud datasets that can be downloaded from Ref. [40], as shown in Fig. 3, in which Figs. 3(a) – 3(l) are portraits with dense points, and Figs. 3(m) – 3(p) are architecture with sparse points. Figs. 3(a) – 3(h) are voxelized upper bodies point cloud data sequences of two spatial resolutions obtained from Microsoft. Figs. 3(i) – 3(l) are chosen from 8i voxelized full bodies point cloud data sequences. Remaining large-scale sparse point clouds in Figs. 3(k) – 3(p) are static facade and architecture datasets.

2) Evaluation metrics. The performance of the proposed method is evaluated in terms of bit per point (BPP). The BPP refers to the sum of bits occupied by the coordinate information attached to the point. The lower the value, the better the performance.

$$\mathrm{BPP} = \frac{\mathrm{Size}_{\mathrm{dig}}}{k}, \quad (6)$$

where $\mathrm{Size}_{\mathrm{dig}}$ represents the number of bits occupied by the coordinate information of point cloud data, and k refers to the number of points in the original point cloud.

3) Benchmarks. We mainly compare our method with other baseline algorithms, including: PCL-PCC: octree-based compression in PCL; G-PCC (MPEG intra-coders test model) and interEM (MPEG inter-coders test model) target single-frame and multi-frame point cloud compression respectively; The Silhouette 3D (S3D)[41] and Silhouette 4D (S4D)[42] target single-frame and multi-frame point cloud compression, respectively. For PCL, we use the octree point cloud compression approach in PCL-v1.8.1 for geometry compression only. We set octree resolution parameters from point precision and voxel resolution. For G-PCC (TM13-v11.0), we choose a lossless geometry—lossless attributes condition in an octree-predictive mode, leaving parameters as default. For interEM (tmc3v3.0), we use the experimental results under lossless geometry and



▲ **Figure 3. Point cloud sequences used in experiments: (a) Andrew_vox09, (b) Andrew_vox10, (c) David_vox09, (d) David_vox10, (e) Ricardo_vox09, (f) Ricardo_vox10, (g) Sarah_vox09, (h) Sarah_vox10, (i) Longdress_vox10, (j) Loot_vox10, (k) Redandblack_vox10, (l) Soldier_vox10, (m) Facade_00009_vox12, (n) Facade_00015_vox14, (o) Arco_Valentino_Dense_vox12, and (p) Palazzo_Carignano_Dense_vox14**

lossless attributes conditions as a comparison[43]. For S3D and S4D, we follow the default conditions and parameters.

4) Hardware. The proposed algorithm is implemented in Matlab and C++ using some functions of the PCL-v1.8.1. All experiments have been tested on a laptop with Intel Core i7-8750 CPU @2.20 GHz with 8 GB memory.

4.2 Results of Single-Frame Point Cloud Compression

1) Compression results of portraits of dense point cloud data sequences

Table 1 shows the performance of our spatial context-guided lossless point cloud geometry compression algorithms compared with PCL-PCC, G-PCC and S3D methods on portraits of dense point cloud data sequences.

It can be seen from Table 1 that for all the point cloud of the same sequences, the proposed method achieves the lowest compression BPP compared with other methods. Our algorithm offers averaged gains from −1.56% to −0.02% against S3D, and gains from −10.62% to −1.45% against G-PCC. It shows a more obvious advantage, that is, the compression performance gains of the proposed algorithm range from −10.62% to −1.45%; For PCL-PCC, the proposed algorithm shows a nearly doubled gain on all sequences, ranging from −154.43% to −85.39%.

2) Compression results of large-scale sparse point cloud data

Because the S3D cannot work in this case, we only compare our spatial context-guided lossless geometry point cloud compression algorithm with PCL-PCC and G-PCC methods on large-scale sparse point cloud data.

Again, our algorithm achieves considerable performance with G-PCC and PCL-PCC, as shown in Table 1. Results have shown that averaged BPP gains ranging from −8.84% to −4.35% are captured compared with G-PCC. For PCL- PCC, our proposed algorithm shows more obvious advantages, with gains ranging from −34.69% to −23.94%.

3) Summary

To provide a more comprehensible comparison of the single-frame point cloud compression results, Table 2 presents the average results between our spatial context-guided compression method and other state-of-the-art benchmark methods. Compared with S3D, our proposed method shows average gains ranging from −0.58% to −3.43%. As for G-PCC and PCL-PCC, the average gains achieve at least −3.43% and −95.03% respectively.

Experimental analysis reveals that our spatial context-guided compression method exceeds current S3D, G-PCC and PCL-PCC by a significant margin. Thus, it can satisfy the lossless compression requirements of point cloud geometry for various scene types, e.g., dense or sparse distributions, and the effectiveness of our method consistently remains.

4.3 Results of Multi-frame Point Cloud Compression

We evaluate our proposed spatial-temporal context-guided point cloud geometry compression algorithm against existing compression algorithms such as S4D, PCL-PCC, G-PCC and interEM. Only portraits of dense point cloud data sequences are used in this experiment. The results are illustrated in

▼**Table 1. BPP comparisons of our spatial context-guided compression algorithm and the baseline methods**

Point Cloud Data	BPP/bit				Gains		
	Single ↓	G-PCC ↓	PCL-PCC ↓	S3D ↓	G-PCC/%	PCL-PCC/%	S3D/%
Andrew_vox09	1.118 83	1.135 068	2.074 226	1.12	−1.45	−85.39	−0.10
Andrew_vox10	1.010 745	1.104 986	1.952 745	-	−9.32	−93.20	-
David_vox09	1.058 42	1.114 673	2.105 917	1.06	−5.31	−98.97	−0.15
David_vox10	1.028 09	1.090 388	1.974 752	-	−6.06	−92.08	-
Ricardo_vox09	1.037 76	1.081 282	2.046 144	1.04	−4.19	−97.17	−0.22
Ricardo_vox10	0.965 985	1.068 567	1.944 874	-	−10.62	−101.34	-
Sarah_vox09	1.063 19	1.107 865	2.101 666	1.07	−4.20	−97.68	−0.64
Sarah_vox10	1.012 36	1.065 947	1.978 308	-	−5.29	−95.42	-
Longdress_vox10	0.945 35	1.025 244	2.347 862	0.95	−8.45	−148.36	−0.49
Loot_vox10	0.909 825	0.945 36	2.314 874	0.91	−3.91	−154.43	−0.02
Redandblack_vox10	1.014 15	1.082 107	2.400 688	1.03	−6.70	−136.72	−1.56
Soldier_vox10	0.958 515	1.032 572	2.423 025	0.96	−7.73	−152.79	−0.15
Facade 00009 vox12	6.941 5	7.243 8	9.349 4	-	−4.35	−34.69	-
Facade_00015_vox14	7.937 2	8.638 5	10.030 5	-	−8.84	−26.37	-
Arco_Valentino_Dense_vox12	9.077 9	9.826 4	11.251 4	-	−8.25	−23.94	-
Palazzo_Carignano_Dense_vox14	7.647 5	8.164 4	9.943 4	-	−6.76	−30.02	-

BPP: bit per point
G-PCC: geometry-based point cloud compression
PCC: point cloud compression
PCL: Point Cloud Library
S3D: Silhouette 3D

▼Table 2. BPP comparison with state-of-the-art algorithms on single-frame point cloud data

Point Cloud Data	Average BPP/bit				Average Gains		
	Single ↓	G-PCC ↓	PCL-PCC ↓	S3D ↓	G-PCC	PCL-PCC	S3D
Microsoft voxelized upper bodies	1.036 923	1.096 097	2.022 329	1.072 5	−5.71%	−95.03%	−3.43%
8i voxelized full bodies	0.956 96	1.021 321	2.371 612	0.962 5	−6.73%	−147.83%	−0.58%
MPEG Facade and architecture	1.158 62	1.198 392	2.336 034	-	−3.43%	−101.62%	-

BPP: bit per point
G-PCC: geometry-based point cloud compression
MPEG: Moving Picture Experts Group
PCC: point cloud compression
PCL: Point Cloud Library
S3D: Silhouette 3D

Table 3. As we can see, after optimizations in prediction mode and arithmetic encoder, the proposed algorithm shows superiority on all test sequences. Specifically, compared with interEM and G-PCC, the proposed algorithm shows significant gains ranging from −51.94% to −17.13% and −46.62% to −5.7%, respectively. Compared with S4D, the proposed algorithm shows robust improvement ranging from −12.18% to −0.33%. As for PCL-PCC, our proposed algorithm has nearly halved over all test sequences.

Furthermore, we summarize the compression results and gains of the proposed method on the portraits dense point cloud data sequences, listed in Table 4. On average, it delivers gains between −11.5% and −2.59% compared with the spatial context-guided point cloud geometry compression algorithm proposed previously. Moreover, it shows superior average gains of −19% compared with G-PCC, and has achieved an average coding gain of −24.55% compared with interEM. Additionally, compared with S3D and S4D, it gains more than −6.11% and −3.64% on average respectively.

The overall experimental analysis shows that the spatio-temporal context-guided point cloud compression method can make full use of both the spatial and temporal correlation of adjacent layers within intra-frames and inter-frames. We also improve the global context selection and probability model of the arithmetic encoder to obtain a lower bit rate. The proposed method surpasses the performance of state-of-

▼Table 3. Bit per point comparisons of our spatio-temporal context-guided compression algorithm and the baseline methods

Point Cloud Sequences	BPP/bit					Gains			
	Multiple ↓	G-PCC ↓	InterEM ↓	PCL-PCC ↓	S4D ↓	G-PCC/%	InterEM/%	PCL-PCC/%	S4D/%
Andrew_vox09	1.072 25	1.135 068	-	2.074 226	1.08	−5.86	-	−93.45	−0.72
Andrew_vox10	0.972 24	1.104 986	-	1.952 745	-	−13.65	-	−100.85	-
David_vox09	1.046 565	1.114 673	-	2.105 917	1.05	−6.51	-	−101.22	−0.33
David_vox10	1.020 547	1.090 388	-	1.974 752	-	−6.84	-	−93.50	-
Ricardo_vox09	0.982 66	1.081 282	-	2.046 144	1.02	−10.04	-	−108.23	−3.80
Ricardo_vox10	0.954 235	1.068 567	-	1.944 874	-	−11.98	-	−103.81	-
Sarah_vox09	1.028 745	1.107 865	-	2.101 666	1.04	−7.69	-	−104.29	−1.09
Sarah_vox10	1.008 465	1.065 947	-	1.978 308	-	−5.70	-	−96.17	-
Longdress_vox10	0.896 585	1.025 244	1.056 275	2.347 862	0.95	−14.35	−17.81	−161.87	−5.96
Loot_vox10	0.861 815	0.945 36	1.009 412	2.314 874	0.89	−9.69	−17.13	−168.60	−3.27
Redandblack_vox10	0.970 43	1.082 107	1.140 317	2.400 688	1.01	−11.51	−17.51	−147.38	−4.08
Soldier_vox10	0.704 24	1.032 572	1.070 037	2.423 025	0.79	−46.62	−51.94	−244.06	−12.18

G-PCC: geometry-based point cloud compression
PCC: point cloud compression
PCL: Point Cloud Library
S4D: Silhouette 4D

▼Table 4. Bit per point comparison with state-of-the-art algorithms on multi-frame point cloud data

Average BPP/bit							
Point cloud data	Multiple ↓	Single ↓	G-PCC ↓	InterEM ↓	PCL-PCC ↓	S4D ↓	S3D ↓
Microsoft voxelized upper bodies	1.010 713	1.036 923	1.096 097	-	2.022 329	1.047 5	1.072 5
8i voxelized full bodies	0.858 268	0.956 96	1.021 321	1.069 01	2.371 612	0.91	0.962 5
Average Gains							
Point cloud data		Single	G-PCC	interEM	PCL-PCC	S4D	S3D
Microsoft voxelized upper bodies		−2.59%	−8.45%	-	−100.09%	−3.64%	−6.11%
8i voxelized full bodies		−11.50%	−19.00%	−24.55%	−176.33%	−6.03%	−12.14%

G-PCC: geometry-based point cloud compression
PCC: point cloud compression
PCL: Point Cloud Library
S3D: Silhouette 3D
S4D: Silhouette 4D

the-art algorithms, so as to meet the requirements of point cloud geometry lossless compression in multimedia application scenarios such as dynamic portraits.

4.4 Ablation Study

We perform ablation studies on predictive encoding over 8i voxelized full-body point cloud data sequences to demonstrate the effectiveness of the partition. It can be seen from Table 5 that the improvement shows a stable gain of −70% on multi-frame point cloud compression and −60% on single-frame point cloud compression against the non-partition predictive coding.

Next, we perform an ablation experiment on arithmetic coding to demonstrate the effectiveness of the context dictionary. As shown in Table 6, a robust improvement of −33% on multi-frame point cloud compression and that of −41% on single-frame point cloud compression against the arithmetic coding without context dictionary are observed in our method.

▼Table 5. Ablation study on predictive encoding

Point Cloud Data	Partition		Non-Partition		Gains/%	
	Multiple ↓	Single ↓	Multipl ↓	Single ↓	Multiple ↓	Single ↓
Longdress_vox10	0.896 585	0.945 35	1.501 45	1.514 88	−67.46	−60.25
Loot_vox10	0.861 815	0.909 825	1.477 48	1.493 59	−71.44	−64.16
Redandblack_vox10	0.970 43	1.014 15	1.620 92	1.548 96	−67.03	−52.73
Soldier_vox10	0.704 24	0.958 515	1.521 01	1.563 37	−115.98	−63.10

▼Table 6. Ablation study on arithmetic coding

Point Cloud Data	With Context Dictionary		Without Context Dictionary		Gains/%	
	Multiple ↓	Single ↓	Multiple ↓	Single ↓	Multiple ↓	Single ↓
Longdress_vox10	0.896 585	0.945 35	1.279 66	1.489 1	−42.73	−57.52
Loot_vox10	0.861 815	0.909 825	1.272 72	1.364 27	−47.68	−49.95
Redandblack_vox10	0.970 43	1.014 15	1.294 69	1.435 11	−33.41	−41.51
Soldier_vox10	0.704 24	0.958 515	1.112 31	1.374 98	−57.94	−43.45

4.5 Time Consumption

We test the time consumption to evaluate the algorithm complexity and compare the proposed methods with others. The algorithm complexity is analyzed by encoders and decoders independently, listed in Table 7. As we can see, G-PCC, interEM and PCL-PCC can achieve an encoding time of less than 10 s and a decoding time of less than 5 s for portrait dense point cloud data. They also perform well in large-scale sparse point cloud data compared with others. Our proposed algorithms take around 60 s and 15 s to encode and decode portrait sequences, even more on facade and architecture point cloud data. There is a trade-off between bitrates and compression speed. Compared with S3D and S4D, which take hundreds of seconds to encode, our time-consuming method can show superiority.

In summary, the time consumption of our proposed methods is medium among all the compared algorithms but still necessary to be further improved.

5 Conclusions

In this paper, we propose a spatio-temporal context-guided method for lossless point cloud geometry compression. We consider sliced point cloud of unit thickness as the input unit and adopt the geometry predictive coding mode based on the travelling salesman algorithm, which applies to both intra-frame and inter-frame. Moreover, we make full use of the global context information and adaptive arithmetic encoder based on context fast update to achieve lossless compression and decompression results of point clouds. Experimental results demonstrate the effectiveness of our methods and their superiority over previous studies. For future work, we plan to further study the overall complexity of the algorithm, by reducing algorithm complexity to achieve a high-speed compression rate and low bit rate compression results. A low bit rate and real-time/low-delay supported method is highly desired in various types of scenes.

▼Table 7. Time consumption comparison with state-of-the-art algorithms in encoding and decoding

Encoding Time/s							
Point cloud data	Multiple	Single	S4D	S3D	G-PCC	InterEM	PCL-PCC
Microsoft voxelized upper bodies	52.1	64.2	806.03	489.72	3.813	-	2.235
8i voxelized full bodies	56.7	66.9	904.67	640.85	7.105	4.708	3.549
MPEG facade and architecture	-	111.2	-	-	15.37	-	22.4
Decoding Time/s							
Point cloud data	Multiple	Single	S4D	S3D	G-PCC	InterEM	PCL-PCC
Microsoft voxelized upper bodies	13.7	14.4	117.4	74.03	1.031	-	0.809
8i voxelized full bodies	16.3	17.1	194.25	113.95	1.376	4.10	0.922
MPEG facade and architecture	-	22.4	-	-	2.703	-	7.74

G-PCC: geometry-based point cloud compression
MPEG: Moving Picture Experts Group
PCC: point cloud compression
PCL: Point Cloud Library
S3D: Silhouette 3D
S4D: Silhouette 4D

References

[1] MI X X, YANG B S, DONG Z, et al. Automated 3D road boundary extraction and vectorization using MLS point clouds [J]. IEEE transactions on intelligent transportation systems, 2022, 23(6): 5287 – 5297. DOI: 10.1109/TITS.2021.3052882
[2] DONG Z, LIANG F X, YANG B S, et al. Registration of large-scale terrestrial laser scanner point clouds: a review and benchmark [J]. ISPRS journal of photogrammetry and remote sensing, 2020, 163: 327 – 342. DOI: 10.1016/j.isprsjprs.2020.03.013
[3] GRAZIOSI D, NAKAGAMI O, KUMA S, et al. An overview of ongoing point cloud compression standardization activities: video-based (V-PCC) and geometry-based (G-PCC) [J]. APSIPA transactions on signal and information processing, 2020, 9: e13
[4] DE QUEIROZ R L, CHOU P A. Compression of 3D point clouds using a region-adaptive hierarchical transform [J]. IEEE transactions on image processing, 2016, 25(8): 3947 – 3956. DOI: 10.1109/TIP.2016.2575005
[5] BLETTERER A, PAYAN F, ANTONINI M, et al. Point cloud compression using depth maps [J]. Electronic imaging, 2016, 2016(21):1 – 6
[6] MEKURIA R, BLOM K, CESAR P. Design, implementation, and evaluation of a point cloud codec for tele-immersive video [J]. IEEE transactions on circuits and systems for video technology, 2017, 27(4): 828 – 842. DOI: 10.1109/TCSVT.2016.2543039
[7] DE QUEIROZ R L, CHOU P A. Motion-compensated compression of dynamic voxelized point clouds [J]. IEEE transactions on image processing, 2017, 26(8): 3886 – 3895. DOI: 10.1109/TIP.2017.2707807
[8] CAO C, PREDA M, ZAHARIA T. 3D point cloud compression: a survey [C]//The 24th International Conference on 3D Web Technology. ACM, 2019: 1 – 9. DOI: 10.1145/3329714.3338130
[9] GRAZIOSI D, NAKAGAMI O, KUMA S, et al. An overview of ongoing point cloud compression standardization activities: video-based (V-PCC) and geometry-based (G-PCC) [J]. APSIPA transactions on signal and information processing, 2020, 9(1): e13. DOI: 10.1017/atsip.2020.12
[10] HUANG Y, PENG J L, KUO C J, et al. Octree-based progressive geometry coding of point clouds [C]//The 3rd Eurographics/IEEE VGTC Conference on Point-Based Graphics. IEEE, 2016: 103 – 110
[11] FAN Y X, HUANG Y, PENG J L. Point cloud compression based on hierarchical point clustering [C]//Asia-Pacific Signal and Information Processing Association Annual Summit and Conference. IEEE, 2014: 1 – 7. DOI: 10.1109/APSIPA.2013.6694334
[12] DRICOT A, ASCENSO J. Adaptive multi-level triangle soup for geometry-based point cloud coding [C]//The 21st International Workshop on Multimedia Signal Processing (MMSP). IEEE, 2019: 1 – 6. DOI: 10.1109/MMSP.2019.8901791
[13] HE C, RAN L Q, WANG L, et al. Point set surface compression based on shape pattern analysis [J]. Multimedia tools and applications, 2017, 76(20): 20545 – 20565. DOI: 10.1007/s11042-016-3991-0
[14] IMDAD U, ASIF M, AHMAD M, et al. Three dimensional point cloud compression and decompression using polynomials of degree one [J]. Symmetry, 2019, 11(2): 209. DOI: 10.3390/sym11020209
[15] SUN X B, MA H, SUN Y X, et al. A novel point cloud compression algorithm based on clustering [J]. IEEE robotics and automation letters, 2019, 4(2): 2132 – 2139. DOI: 10.1109/LRA.2019.2900747
[16] DE OLIVEIRA RENTE P, BRITES C, ASCENSO J, et al. Graph-based static 3D point clouds geometry coding [J]. IEEE transactions on multimedia, 2019, 21(2): 284 – 299. DOI: 10.1109/TMM.2018.2859591
[17] ISO. Geometry-based point cloud compression (G-PCC): ISO/IEC 23090-9 [S]. 2021
[18] DRICOT A, ASCENSO J. Hybrid octree-plane point cloud geometry coding [C]//The 27th European Signal Processing Conference (EUSIPCO). IEEE, 2019: 1 – 5
[19] ZHANG X, GAO W, LIU S. Implicit geometry partition for point cloud compression [C]//Proceedings of 2020 Data Compression Conference (DCC). IEEE, 2020: 73 – 82. DOI: 10.1109/DCC47342.2020.00015
[20] QUACH M, VALENZISE G, DUFAUX F. Learning convolutional transforms for lossy point cloud geometry compression [C]//The 2019 IEEE International Conference on Image Processing (ICIP). IEEE, 2019: 4320 – 4324. DOI: 10.1109/ICIP.2019.8803413
[21] HUANG T X, LIU Y. 3D point cloud geometry compression on deep learning [C]//The 27th ACM International Conference on Multimedia. ACM, 2019: 890 – 898. DOI: 10.1145/3343031.3351061
[22] GUARDA A F R, RODRIGUES N M M, PEREIRA F. Point cloud coding: Adopting a deep learning-based approach [C]//Picture Coding Symposium (PCS). IEEE, 2020: 1 – 5. DOI: 10.1109/PCS48520.2019.8954537
[23] WANG J Q, ZHU H, MA Z, et al. Learned point cloud geometry compression [EB/OL]. [2023-09-01]. https://arxiv.org/abs/1909.12037.pdf
[24] AINALA K, MEKURIA R N, KHATHARIYA B, et al. An improved enhancement layer for octree based point cloud compression with plane projection approximation [C]//SPIE Optical Engineering+Applications. SPIE, 2016: 223 – 231. DOI: 10.1117/12.2237753
[25] SCHWARZ S, HANNUKSELA M M, FAKOUR-SEVOM V, et al. 2D video coding of volumetric video data [C]//Picture Coding Symposium (PCS). IEEE, 2018: 61 – 65. DOI: 10.1109/PCS.2018.8456265
[26] FAKOUR SEVOM V, SCHWARZ S, GABBOUJ M. Geometry-guided 3D data interpolation for projection-based dynamic point cloud coding [C]//The 7th European Workshop on Visual Information Processing (EUVIP). IEEE, 2019: 1 – 6. DOI: 10.1109/EUVIP.2018.8611760
[27] KATHARIYA B, LI L, LI Z, et al. Lossless dynamic point cloud geometry compression with inter compensation and traveling salesman prediction [C]//Data Compression Conference. IEEE, 2018: 414. DOI: 10.1109/DCC.2018.00067
[28] ISO. Visual volumetric video-based coding (V3C) and video-based point cloud compression: ISO/IEC 23090-5 [S]. 2021
[29] PARK J, LEE J, PARK S, et al. Projection-based occupancy map coding for 3D point cloud compression [J]. IEIE transactions on smart processing & computing, 2020, 9(4): 293 – 297. DOI: 10.5573/ieiespc.2020.9.4.293
[30] COSTA A, DRICOT A, BRITES C, et al. Improved patch packing for the MPEG V-PCC standard [C]//IEEE 21st International Workshop on Multimedia Signal Processing (MMSP). IEEE, 2019: 1 – 6. DOI: 10.1109/MMSP.2019.8901690
[31] KAMMERL J, BLODOW N, RUSU R B, et al. Real-time compression of point cloud streams [C]//Proceedings of 2012 IEEE International Conference on Robotics and Automation. IEEE, 2012: 778 – 785. DOI: 10.1109/ICRA.2012.6224647
[32] PCL. Point cloud library. [EB/OL]. [2023-09-01]. http://pointclouds.org/
[33] THANOU D, CHOU P A, FROSSARD P. Graph-based compression of dynamic 3D point cloud sequences [J]. IEEE transactions on image processing, 2016, 25(4): 1765 – 1778. DOI: 10.1109/TIP.2016.2529506
[34] LI L, LI Z, ZAKHARCHENKO V, et al. Advanced 3D motion prediction for video based point cloud attributes compression [C]//Data Compression Conference (DCC). IEEE, 2019: 498 – 507. DOI: 10.1109/DCC.2019.00058
[35] ZHAO L L, MA K K, LIN X H, et al. Real-time LiDAR point cloud compression using Bi-directional prediction and range-adaptive floating-point coding [J]. IEEE transactions on broadcasting, 2022, 68(3): 620 – 635. DOI: 10.1109/TBC.2022.3162406
[36] LIN J P, LIU D, LI H Q, et al. M-LVC: Multiple frames prediction for learned video compression [C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, 2020: 3543 – 3551. DOI: 10.1109/CVPR42600.2020.00360
[37] YANG R, MENTZER F, VAN GOOL L, et al. Learning for video compression with hierarchical quality and recurrent enhancement [C]//IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, 2020: 6627 – 6636. DOI: 10.1109/CVPR42600.2020.00666
[38] KAYA E C, TABUS I. Lossless compression of point cloud sequences using sequence optimized CNN models [J]. IEEE access, 2022, 10: 83678 – 83691. DOI: 10.1109/ACCESS.2022.3197295
[39] DING S, MANNAN M A, POO A N. Oriented bounding box and octree based global interference detection in 5-axis machining of free-form surfaces [J]. Computer-aided design, 2004, 36(13): 1281-1294
[40] ALEXIOU E, VIOLA I, BORGES T M, et al. A comprehensive study of the rate-distortion performance in MPEG point cloud compression [J]. APSIPA transactions on signal and information processing, 2019, 8: e27. doi:10.1017/ATSIP.2019.20
[41] PEIXOTO E. Intra-frame compression of point cloud geometry using dyadic

decomposition [J]. IEEE signal processing letters, 2020, 27: 246 – 250. DOI: 10.1109/LSP.2020.2965322
[42] RAMALHO E, PEIXOTO E, MEDEIROS E. Silhouette 4D with context selection: lossless geometry compression of dynamic point clouds [J]. IEEE signal processing letters, 2021, 28: 1660 – 1664. DOI: 10.1109/lsp.2021.3102525
[43] ISO. Common test conditions for G-PCC document N00106: ISO/IEC JTC 1/SC 29/WG 7 MPEG [S]. 2021

Biographies

ZHANG Huiran received her BE and ME degrees in School of Geodesy and Geomatics and State Key Laboratory of Information Engineering in Surveying Mapping and Remote Sensing, both from Wuhan University, China in 2020 and 2023, respectively. She is currently the surveyor of Guangzhou Urban Planning and Design Survey Research Institute, China. Her research interests include point cloud data processing and compression. She participated in several projects related to the field of remote sensing and published one paper in Geomatics and Information Science of Wuhan University.

DONG Zhen (dongzhenwhu@whu.edu.cn) received his BE and PhD degrees in remote sensing and photogrammetry from Wuhan University, China in 2011 and 2018, respectively. He is a professor with the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing (LIESMARS), Wuhan University. His research interests include 3D reconstruction, scene understanding, point cloud processing as well as their applications in intelligent transportation system, digital twin cities, urban sustainable development and robotics. He received over 10 honors from various national and international competitions and published around 60 papers in various journals and conferences.

WANG Mingsheng received his BE degree in College of Computer Science and Technology from Jilin University, China in 2001, and ME degree in School of Computer Science and Engineering from South China University of Technology, China in 2004. He is currently a senior engineer with Guangzhou Urban Planning & Design Survey Research Institute, China. His research interests include computer applications and software, physiography, and surveying. He received over 20 honors from various national competitions and published around 50 papers in various journals and conferences.