Article

Point Cloud Geometry Compression Based on Multi-Layer Residual Structure

Jiawen Yu [1], Jin Wang [1], Longhua Sun [1], Mu-En Wu [2] and Qing Zhu [1,*]

[1] Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China
[2] Department of Information and Finance Managment, National Taipei University of Technology, Taipei 10608, Taiwan
* Correspondence: ccgszq@bjut.edu.cn

Abstract: Point cloud data are extensively used in various applications, such as autonomous driving and augmented reality since it can provide both detailed and realistic depictions of 3D scenes or objects. Meanwhile, 3D point clouds generally occupy a large amount of storage space that is a big burden for efficient communication. However, it is difficult to efficiently compress such sparse, disordered, non-uniform and high dimensional data. Therefore, this work proposes a novel deep-learning framework for point cloud geometric compression based on an autoencoder architecture. Specifically, a multi-layer residual module is designed on a sparse convolution-based autoencoders that progressively down-samples the input point clouds and reconstructs the point clouds in a hierarchically way. It effectively constrains the accuracy of the sampling process at the encoder side, which significantly preserves the feature information with a decrease in the data volume. Compared with the state-of-the-art geometry-based point cloud compression (G-PCC) schemes, our approach obtains more than 70–90% BD-Rate gain on an object point cloud dataset and achieves a better point cloud reconstruction quality. Additionally, compared to the state-of-the-art PCGCv2, we achieve an average gain of about 10% in BD-Rate.

Keywords: point cloud geometry compression; multi-layer residual module; progressive sampling

Citation: Yu, J.; Wang, J.; Sun, L.; Wu, M.-E.; Zhu, Q. Point Cloud Geometry Compression Based on Multi-Layer Residual Structure. *Entropy* **2022**, *24*, 1677. https://doi.org/10.3390/e24111677

Academic Editors: Jose Santamaria and Francisco Roca

Received: 19 September 2022
Accepted: 8 November 2022
Published: 17 November 2022

1. Introduction

Owing to a rapid innovation of visual capture technology, point clouds have been regarded as vital data to describe both 3D objects and scenes [1]. Therefore, point clouds have a broad range of applications in areas, such as autonomous driving and augmented reality. Point clouds collections are points in space, including the coordinates of geometry information. In addition, every point can have attribute information attached, including colors, normals and reflectances. Since point cloud data are usually very large in quantity and is disordered and irregular [2] in structure, when compared with image and video compression [3–7], how to efficiently compress point cloud data becomes a challenging task.

Point cloud compression can be divided into geometric compression and attribute compression according to the compression object. The geometric information is coded independently, and the attribute information needs to be coded based on the known geometric structure information. The irregular spatial distribution of point clouds makes it difficult to compress. Therefore, it is necessary to convert point clouds into normalized and organized data structures, such as volume model, tree structure, grid model and a multi-view image. Hence using the traditional methods, the encoding techniques based on tree structure [8–13], surface approximation [13,14] and mapping [15–19], are several representative encoding schemes of point cloud geometry. The existing standard compression methods for point clouds are released by MPEG [1] consisting of geometry-based PCC (G-PCC) applied to static point clouds and video-based PCC (V-PCC) applied to dynamic point clouds, which are both typical methods. Moreover, with the development of artificial intelligence technology, recent works [20–23] have applied deep learning to Point

Cloud Compression (PCC), e.g., PointNet [24], which greatly improves the compression performance of point clouds, including point-based point cloud compression methods and voxel-based compression methods.

In this work, we propose a multi-layer residual architecture for Point Cloud Geometry Compression (PCGC) based on voxel using sparse convolution [25]. The main contributions of this work are as follows:

- A multi-layer residual module is introduced to take advantage of the distortion in entropy coding by geometric subtraction, to constrain the accuracy of the sampling process at the encoder side;
- We employ sparse convolution to design a multi-layer residual module and progressive up-sampling reconstruction for efficient processing of sparse tensors at low complexity;
- We adopt a novel joint loss distortion by designing a multi-layer residual loss obtained by multi-layer residual operation to improve the quality of the reconstructed point cloud.

Extensive experimental results demonstrate that our method can lessen the feature information loss caused during the quantization process and improve the accuracy of point cloud information extraction. Our method also performs well in subjective visual quality. In addition, our method outperforms G-PCC [1] and V-PCC [1] substantially at BD-Rate gain and achieves a 10% BD-Rate gain over the state-of-the-art method PCGCv2 [20].

The remainder of this paper is as follows: Section 2 provides a summary overview of the related work; Section 3 describes our proposed multi-layer residual architecture; Section 4 gives the experimental details and presents the experimental results; The Section 5 briefly concludes the paper.

2. Related Work

2.1. Conventional PCGC

Conventional non-deep learning based PCC methods include MPEG G-PCC [1], V-PCC [1], etc. According to the data structure into which the point cloud is converted, it can be divided into tree structure based, surface approximation based and mapping based encoding methods.

2.1.1. Tree Structure Based PCGC

The encoding method based on tree structure is suitable for sparse point cloud, which is simple, direct and effective, and was the earliest to be developed. Many works store point cloud data in such octrees and use specific entropy models, such as adaptive histogram, parent-child node context [8], estimation based on plane approximation [9] or nearest neighbor [10]. To remove temporal redundancy in point cloud sequences, Kammerl et al. [11] encoded the differences between consecutive octrees, while Mekuria et al. [12] use encoded rigid-body transformed blocks, with both methods using empirical histograms for range coding when coding. The main disadvantage of octree-based compression methods is that the number of bits required increases dramatically with tree depth. This scheme is the basic scheme of point-cloud coding based on geometric information adopted by the MPEG G-PCC [1] standard group.

2.1.2. Surface Approximation Based PCGC

Coding methods based on surface approximation are suitable for densely sampled and smooth surface point clouds. Compared with tree structure based coding methods, they can bring significant performance improvements at low bit rates. The point cloud itself is a sampling of the surface of the object, so the surface mesh model can be used to fit the distribution of the origin point cloud in space, and then the point cloud can be recovered by sampling on the surface. Surface approximation-based methods are often combined with octree-based space division. First, the point cloud is divided to obtain local point-cloud blocks, and then the plane is used to fit the local point cloud to achieve dimensionality reduction. In this way, only the edge corners of the plane need to be encoded, so it has higher encoding efficiency. This scheme is also one of the geometric coding schemes of MPEG G-PCC [1]. The disadvantage is that the plane-based coding method cannot achieve lossless coding due to the constant error of the plane approximation.

2.1.3. Mapping-Based PCGC

Mapping-based encoding methods are mainly applied to dense dynamic point clouds. This type of method [15–19] maps the point cloud into a two-dimensional image, and then directly use a mature image and a video encoder for compression, which can achieve high encoding efficiency. The research mainly focuses on the mapping method, that is, how to realize the 3D to 2D mapping and its reconstruction as much as possible without loss; additionally, to help the mapped images develop a better spatial and temporal correlation in order to make better use of the efficient video coding methods and forecasting techniques. A more advanced mapping technique currently projects a patch of points with similar normal vectors onto the surface of a cube surrounding the point cloud, and then converts it into a depth map for compression. This patch-based projection avoids massive loss of occluded contiguous points. This scheme was adopted by the MPEG V-PCC [1] standard as a video-based point cloud coding standard.

2.2. Deep Learning Based PCGC

In recent years, with the rise of artificial intelligence technology, the methods using deep learning outperform them. In terms of point-cloud geometry compression, deep-learning-based approaches can be simply classified as voxel-based and point-based.

2.2.1. Voxel-Based PCGC

This method extends the 2D Convolutional Neural Network (CNN) based image compression framework to 3D CNN-based volume model compression. Many such works [22,24] use 3D CNN to design codec networks. For the convolutional features, a learning-based entropy model is used for rate estimation. On the decoding side, the distortion is optimized by using a classification loss function with the reconstructed voxels being classified by a binary classification. There are also some studies, [26] representing volume models, based on Truncated Signed Distance Fields (TSDF), which can achieve a better geometry compression performance. Recently Wang et al. [23] proposed a multiscale geometric compression method, which achieves a good rate-distortion (R-D) tradeoff when compared to other approaches, while the distortion caused by quantization is ignored in the entropy coding stage. This will cause some feature information of the encoded point cloud to be lost, which leads to the reconstructed point cloud to ignore more details. Therefore, we further process the distortion through the proposed multi-layer residual module.

On the basis of voxelized point clouds, there are some works that use octree representation for PCC. For example, Huang et al. [27] and Que et al. [28] first divide the point cloud data into the octree structure, and the entropy model is constructed by exploiting the relationship between the nodes of the octree compression including ancestor nodes or neighbor nodes or their combination. Their works use 3D convolution to design a multilayer perceptron (MLP), which finally obtains the symbol prediction through the softmax layer.

2.2.2. Point-Based PCGC

Voxel-based compression algorithms are limited by fixed precision constraints, which are difficult to be effectively applied to unevenly distributed and sparse point clouds, while neural networks that directly process point sets, such as PointNet have the ability to deal with them. Wen et al. [29] proposed a deep learning-based framework for lossy compression of geometric structures via hybrid representations of point clouds. The method firstly decomposes the input original point cloud into non-overlapping local sub-blocks adaptively through the decomposition and clustering of the adaptive octree. Then, a point cloud autoencoder network framework with quantization layers is proposed to learn compact latent feature representations from each sub-block. Subsequently, Zhu et al. [30] exploited regional similarity to achieve efficient lossy point cloud geometry compression. To this end, the input point cloud is divided into multiple local regions, and they are grouped into distinct clusters according to the region surface vectors, ensuring that the inter-cluster similarity is minimized and the intra-cluster similarity is maximized. Alignment transformation is performed on each cluster to predict non-reference regions of similar features from selected reference regions to achieve a considerable data reduction. In addition, Gao et al. [31] developed a more efficient point-based method for sparse point cloud compression, employing an end-to-end variational autoencoder structure to extract latent key points from point clouds using multi-scale neural graph sampling (NGS), taking neighboring structures as latent features. The decoder directly uses hierarchical convolution to gradually refine point reconstructions with aggregated features.

This kind of network [29–31] goes straight to taking the point cloud coordinates as an input and solves the disordered arrangement of point clouds through symmetric functions to learn latent features. The decoder directly generates the coordinates of the point cloud through a network such as a fully connected layer.

3. Method

The proposed multi-layer residual architecture is designed on the basis of the popular convolutional autoencoders via sparse convolution [25]. In order to reduce the loss of feature information caused by the quantization process, we introduced a multi-layer residual module at the encoder side to solve the distortion problem caused by the entropy encoding stage. We down-sample the input point clouds with the multi-layer residual module and reconstruct the points cloud in a hierarchical way. The holistic network structure is shown in Figure 1.

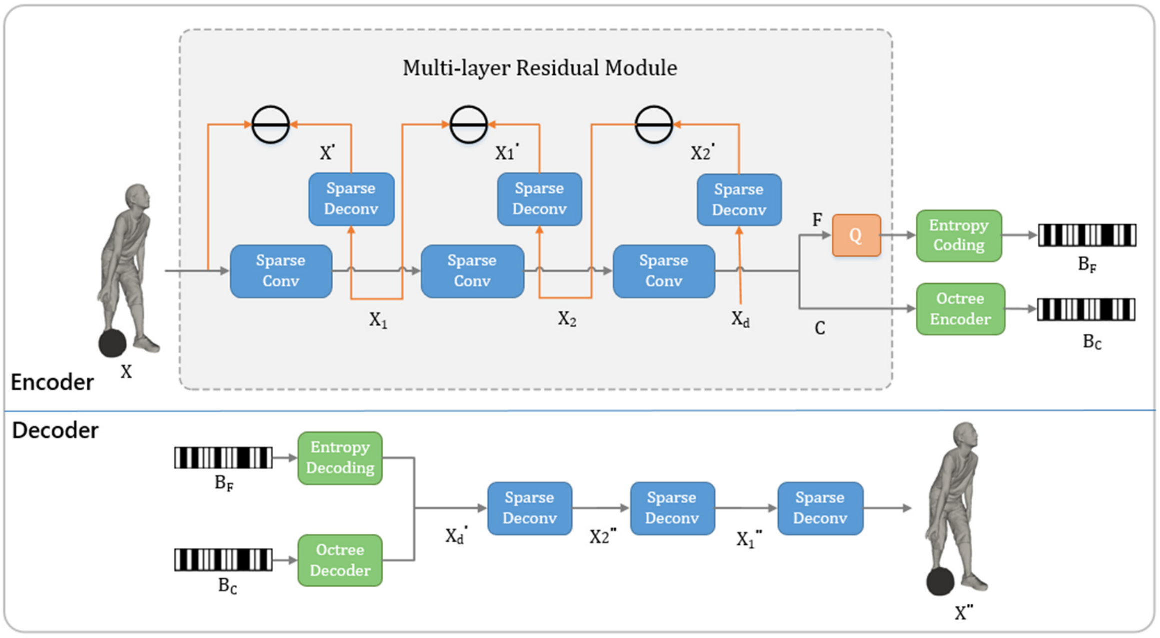

Figure 1. The architecture of the proposed method. Our multi-layer module is on the encoding side. The minus stands for geometric subtraction. C and F are the coordinate tensor and feature tensor, respectively. Q is quantization. B_C and B_F are the encoded bit-streams. Other symbols are described and explained in method.

3.1. Sparse Convolution

We adopt a convolutional neural network based on sparse convolution [25] to exploit the sparsity of point clouds. Point cloud data are expressed as a set of sparse tensors, including their coordinates $C = \{(x_i, y_i, z_i)\}_i$ and their associated features $F = \{f_i\}_i$ in sparse convolution. Moreover, the convolution only extracts features of occupied coordinates. Its process is defined in [25] as follows:

$$x_t^{out} = \sum_{i \in K^3(t, C^{in})} W_i x_{t+i}^{in}, \text{ for } t \in C^{out} \tag{1}$$

where C^{in} and C^{out} are coordinates of input and output point clouds. x_t^{in} and x_t^{out} are feature vectors at coordinate u of input and output point clouds. $K^3\left(t, C^{in}\right) = \left\{i \middle| t + i \in C^{in}, i \in K^3\right\}$ is the definition of a 3D convolutional kernel centered on t with an offset of i. W_i is the value of the convolution kernel at offset i.

Using sparse convolutions can help to reduce the complexity and efficiently extract features from point clouds. In this work, sparse convolution is utilized to efficiently down-sample to aggregate features and reconstruct the point clouds while up-sampling. Specifically, we utilize sparse convolutions on the encoder side for triple down-sampling and use in the up-sampling process of the multi-layer residual module as shown in Figure 2. In addition, in sparse convolution, we need to use geometric coding and octree coding for the down-sampled geometric coordinates C and feature information F, respectively.

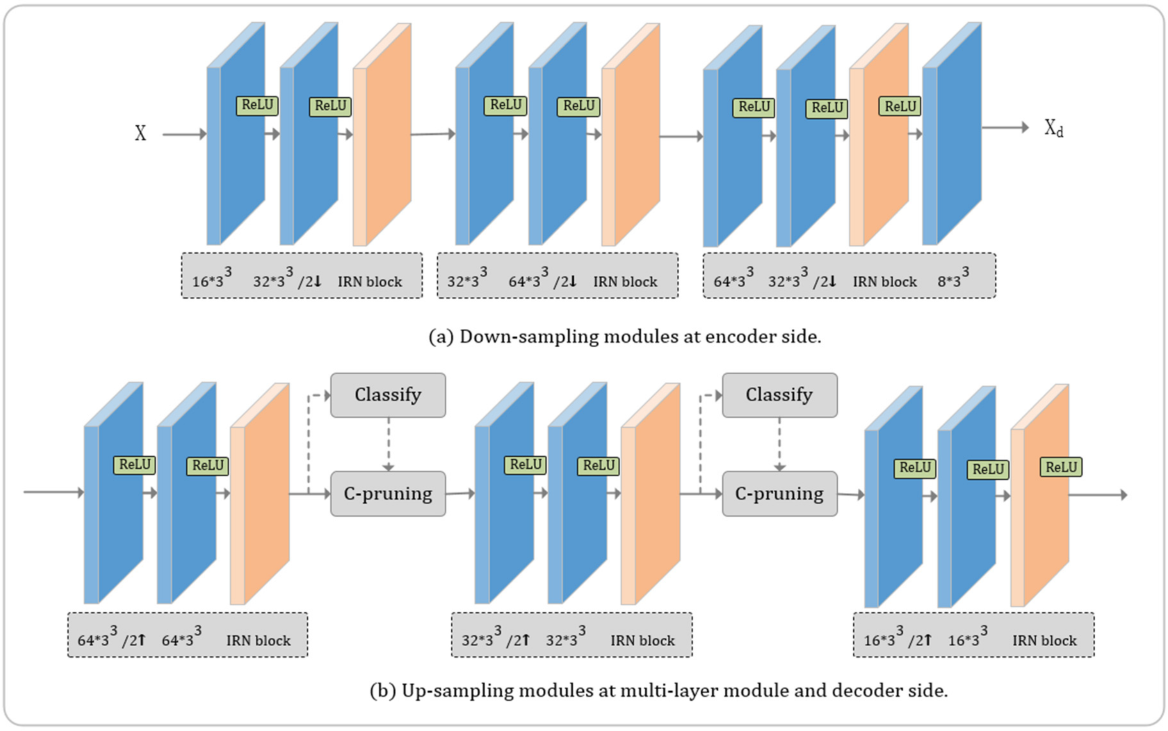

Figure 2. Sparse convolution based up-sampling and down-sampling modules. The upper picture is a down-sampling module based on sparse convolution, and the lower picture is an up-sampling module based on sparse convolution. "$o \times k^3$" represent a sparse convolutional layer with output channel o and convolution kernel $3 \times 3 \times 3$. "2↑" and "2↓" represent up-sampling and down-sampling with a stride of 2, respectively.

3.2. Multi-Layer Residual Architecture

Encoder: At the encoder side of our whole point cloud compression architecture, the input point cloud X is down-sampled progressively, and the feature information F and coordinate information C of the X_d point cloud obtained are coded, respectively. Due to an additional quantization process in the entropy coding stage for F, this process makes our feature F compression lossy. Considering the distortion caused by quantization or the introduction of noise, we design a multi-layer residual architecture, as shown in Figure 1. A sparse conv module includes a sparse convolutional layer with stride 2 for down-sampling by a factor of 2, followed by an Inception ResNet block (IRN) [31] for a more efficient feature extraction. Each of the IRN blocks contains three consecutive IRN units. The above module reduces the amount of compressed data while extracting point cloud features.

For the multi-layer residual module, we perform specially up-sampling after each down-sampling. Specifically, the input point cloud X is first down-sampled to obtain X_1 by aggregating features, and then X_1 is up-sampled to obtain X' with voxel pruning, which imposes coarse-level constraints on the accuracy of the point cloud down-sampling process. Then geometrically subtract the geometric coordinates of X and X' to obtain the geometric residual. The process from X_1 to X_2 and from X_2 to X_d which imposes finer level constraints on the accuracy of the point cloud down-sampling process is similar. For a given point cloud $M(x_i, y_i, z_i) \in X$, the corresponding point $M'(x_i', y_i', z_i')$ is obtained after up-sampling, then their difference $\left(x_i^{res}, y_i^{res}, z_i^{res}\right)$ is calculated by the geometric subtraction. We minimize the distortion of the difference to constrain the down-sampling accuracy. This way we obtain two bitstreams B_C and B_F through the encoder.

Decoder: At the decoder side, a sparse deconv module includes a sparse convolutional layer with stride 2 for up-sampling by a factor of 2, followed by an IRN [31] to better recover point cloud. Each of the IRN blocks contains three consecutive IRN units, as shown in Figure 2b. Then voxel pruning is performed according to the probability that it may be occupied. During the voxel pruning stage with layered reconstruction, we sort the resulting probabilities and consider the top k voxels as the most likely occupied voxels after a binary classification by a sparse convolution with output channel 1. We set k to be the number of ground truth labels to obtain the lowest distortion. We can obtain X_2'', X_1'' and X'' successively according to the decoded X_d'' in this way. In particular, the pruning of voxels, in the multi-layer residual module, does not depend on the probability different with the reconstruction when up-sampling.

3.3. Quantization and Entropy Coding Model

In this work, we use octree coding and entropy coding for the geometric information C and feature information F obtained by down-sampling, respectively. Before entropy coding, we need to quantize the feature information F. In order to ensure the differentiability of backpropagation, we add uniform noise:

$$F^{(\eta)} = F + \eta,\ F^{(\eta)} \sim \eta(F + \frac{1}{2}, F - \frac{1}{2}) \quad (2)$$

where η is random noise, $F^{(\eta)}$ and F are the original and quantized feature representations, respectively. $F^{(\eta)}$ follows a uniform distribution η centered on. For the entropy model stage, we use the probability density model to encode the quantized feature information, i.e., the full factorization model. Then under the hyperpriors [32,33] condition c, we use the Laplacian distribution $\mathcal{L}$ to approximate the probability density $P(F^{(\eta)}\big|c)$, which is defined as follows:

$$P_{F^{(\eta)}|c}(F^{(\eta)}|c) = \prod_i c_i(\mathcal{L}(\mu_i, \sigma_i) \times \eta) \quad (3)$$

3.4. Joint Optimization Distortion

In our deep learning framework, we adopt joint optimization distortion for better compression performance and reconstruction quality. In detail, the loss of the multi-layer residual module is introduced while following the conventional rate-distortion. It is defined as follows:

$$J_{loss} = R + \lambda D + \alpha L_{res} \quad (4)$$

where λ is used to control the bitrate. D measures the reconstruction loss using cross-entropy in Equation (5), and R is the bitrate of probability obtained by the prior model. L_{res}

denotes the residual loss obtained by geometric subtraction at different precision levels and parameter α is the weight of L_{res} as in Equation (6):

$$D = \frac{1}{M}\sum\nolimits_{j}^{M}\frac{1}{N}\sum_{i} -(x_i \log(p_i) + (1 - x_i)\log(1 - p_i)) \tag{5}$$

where M is the number of sampling layers, j is the sampling layer index. x indicates whether the voxel is occupied, occupied is 1 and empty is 0. p represents the probability that it may be occupied, $p \in (0,\ 1)$ is activated by the sigmoid method:

$$L_{res} = \text{sigmoid}\frac{1}{n}\sum_{i}^{n}||x_i - x'_i||_2 + ||y_i - y'_i||_2 + ||z_i - z'_i||_2 \tag{6}$$

where n is the number of point clouds in a batch. (x_i, y_i, z_i) is the point cloud before down-sampling, and (x'_i, y'_i, z'_i) is the point cloud after up-sampling. We can obtain $(x_i^{res}, y_i^{res}, z_i^{res})$ by geometric subtraction and calculate the L2 norm sum. Therefore, L_{res} can be obtained by activating its mean through the sigmoid method.

4. Experiments

4.1. Experimental Setup

In training, we select 20,000 3D mesh models from ShapeNet [34] randomly. We generate points from the surfaces of the mesh. In addition, we adopt joint optimization distortion in Equation (4) and set the weight α for the residual loss to 10 for a better trade-off between the compression cost and the performance. To obtain models at different bit rates, we set λ from 0.15 to 5. We adjust the batch size to eight when training the model and optimize our proposed network with the help of the Adam [35] optimizer.

In the tests, we use point cloud test data from 8i Voxelized Full Bodies (8iVFB) [36], Owlii dynamic human mesh and Microsoft Voxelized Upper Bodies (MVUB) [37]. These point cloud test data cover different types and scales, MPEG Common Test Condition(CTC) [38] and JPEG CTC [39] and use them in the compression work. The preview and details of the test point cloud are shown in Figure 3 and Table 1. Specifically, Class A (full bodies) exhibit a smooth surface and a complete shape, while Class B (upper bodies) present a noisy and incomplete surface (even having visible holes and missing parts).

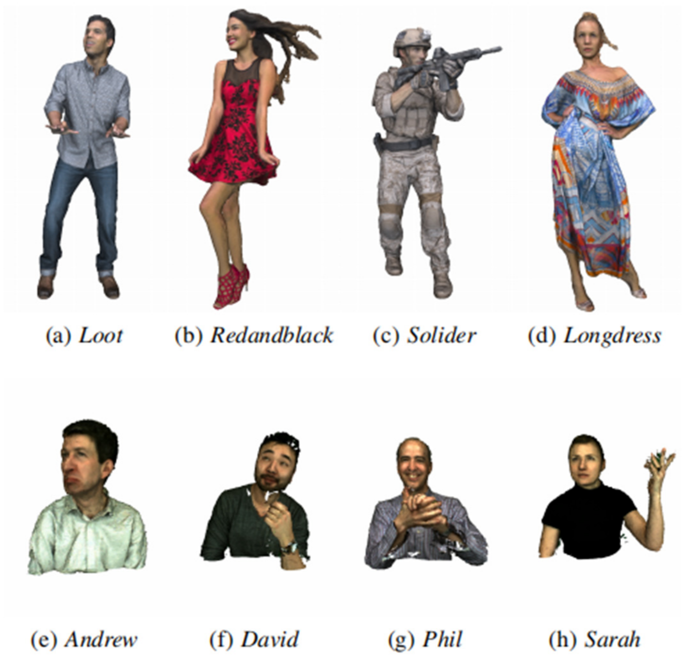

Figure 3. A preview of Testing Datasets.

Table 1. Details of Testing Datasets.

Point Cloud		Points	Precision	Frame
Class A	Longdress	857,966	10	1300
	Soldier	1,089,091	10	690
	Loot	805,285	10	1200
	RedandBlack	757,691	10	1550
Class B	Andrew	279,664	9	1
	Phil	370,798	9	1
	David	330,791	9	1
	Sarah	302,437	9	1

4.2. Experiment Results

Objective Quality Comparison: To validate the performance of our proposed method, we compared our multi-layer residual architecture with the state-of-the-art MPEG G-PCC [1] and V-PCC [1] schemes and followed the CTC [38] suggestions using TM13-v6.0 [40] to encode. In addition, we also compared our multi-layer residual architecture with the state-of-the-art PCGCv2 [20] work. Following the common objective quality measures, we use the bit rate as measurement which measures the bitsper input point (bpp). In our framework, the compression of geometric information occupies a small rate, while the feature information consumes most of the rate. Additionally, we adopt point-to-point distance (D1) [41,42] and point-to-plane distance (D2) as the distortion evaluation metrics. The rate-distortion curves including D1 and D2 are displayed in Figure 4. The BD-Rate and BD-PSNR gains are also displayed in Tables 2 and 3, respectively.

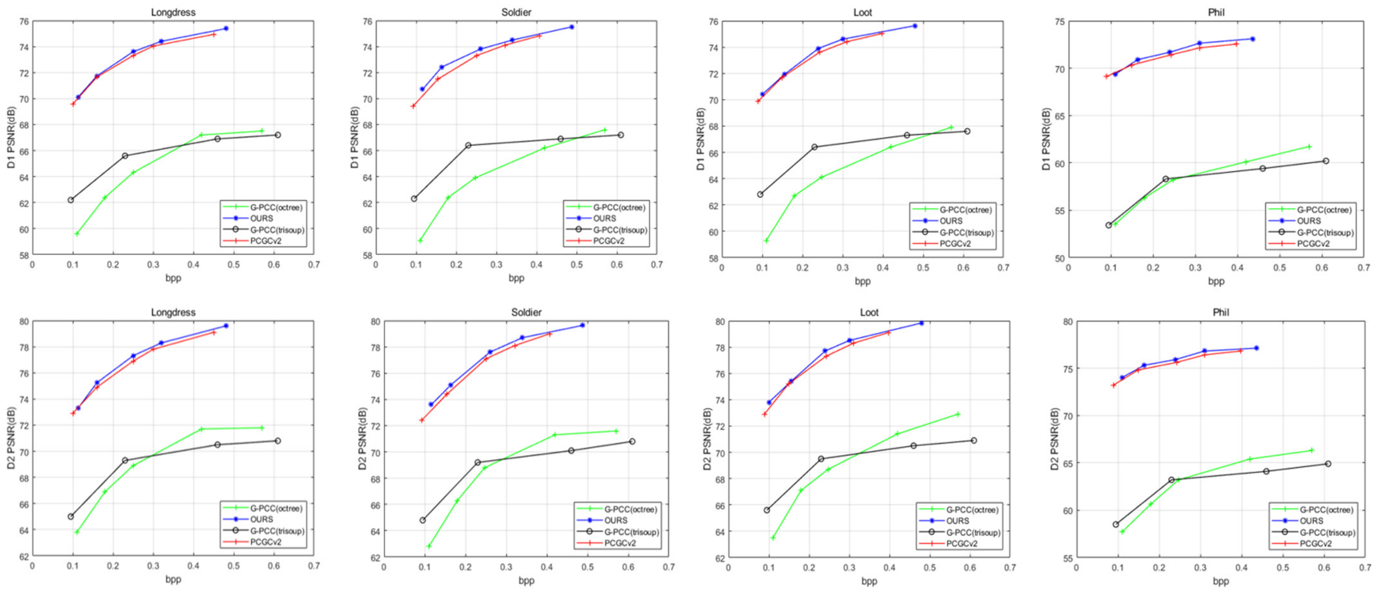

Figure 4. R-D performance including D1, D2 for G-PCC (octree), G-PCC (trisoup), PCGCv2 and our method.

As shown in Tables 2 and 3, compared with point cloud compression schemes G-PCC (octree) [1] and G-PCC (trisoup) [1], our method successfully achieves on average more than 84% of BD-Rate gains and more than 72% of BD-Rate gains, respectively. Meanwhile, we obtain more than 38% of BD-Rate gains against V-PCC [1]. Moreover, compared to the state-of-the-art PCGCv2 [20], we achieve an average gain of about 10% in BD-Rate and show an improvement in terms of the performance on BD-PSNR.

Table 2. BD-Rate gains of some test data against G-PCC (octree), G-PCC (trisoup), V-PCC and PCGCv2 using D1 and D2 distortion measurement.

Point Cloud	**D1 (p2point)**				**D2 (p2plane)**			
	G-PCC (Octree)	**G-PCC (Trisoup)**	**V-PCC**	**PCGCv2**	**G-PCC (Octree)**	**G-PCC (Trisoup)**	**V-PCC**	**PCGCv2**
Longdress	−91.35%	−77.30%	−39.93%	−8.65%	−84.76%	−73.37%	−42.06%	−6.96%
Soldier	−90.37%	−76.88%	−39.50%	−13.25%	−84.74%	−72.92%	−41.97%	−10.04%
Loot	−91.22%	−81.65%	−39.67%	−13.34%	−85.19%	−73.44%	−42.13%	−10.23%
RedandBlack	−90.36%	−81.21%	−39.61%	−12.75%	−85.03%	−73.28%	−42.04%	−9.37%
Andrew	−92.14%	−87.38%	−60.94%	−14.80%	−83.17%	−87.79%	−53.61%	−11.30%
Phil	−92.35%	−87.97%	−61.06%	−14.32%	−83.79%	−80.06%	−53.88%	−10.25%
David	−92.41%	−86.73%	−61.55%	−13.79%	−82.93%	−82.45%	−54.17%	−10.17%
Sarah	−93.16%	−86.66%	−60.39%	−14.64%	−83.61%	−86.36%	−53.65%	−11.40%
Average	−91.67%	−83.22%	−50.33%	−13.19%	−84.40%	−76.71%	−45.94%	−9.97%

Table 3. BD-PSNR gains using D1 distortion of some test data against G-PCC (octree), G-PCC (trisoup), V-PCC and PCGCv2.

Point Cloud	**BD-PSNR**			
	G-PCC (Octree)	**G-PCC (Trisoup)**	**V-PCC**	**PCGCv2**
Longdress	8.89	7.91	3.11	0.24
Soldier	9.29	7.43	3.59	0.41
Loot	9.66	7.31	3.52	0.40
RedandBlack	8.41	6.94	3.22	0.35
Andrew	9.74	11.15	3.95	0.33
Phil	10.43	12.10	4.36	0.51
David	9.54	10.57	3.14	0.40
Sarah	8.94	9.78	3.36	0.37
Average	9.36	9.13	3.53	0.38

Subjective Quality Comparison: We compare the subjective quality of reconstructed point clouds of the proposed method with G-PCC (octree) and the state-of-the-art method PCGCv2. We visualize the reconstructed point clouds of different methods when ours are at a lower bit rate or a similar bit rate, as shown in Figures 5 and 6. It can be seen that we still have a higher reconstruction quality despite the low bit rate. Our reconstruction point cloud retains more detail and is smoother and more distributed in these details. For example, the hair part of Longdress, reconstructed by our method, is denser and finer in detail. Our method has reconstructed the contour of the backpack in the Soldier with clearer and more accurate edges. In addition, the jaw of Andrew, reconstructed by our method, is much finer and smoother.

4.3. Ablation Study

Weight of multi-layer residual distortion: In the training process, in order to make the model have a better compression performance, we set different weights for the multi-layer residual loss. To estimate the effect of multi-layer residual loss on model performance, we set the parameter α as $\alpha = 5$, $\alpha = 10$, $\alpha = 15$ and $\alpha = 20$. Experiments show that when the parameter $\alpha = 10$, the quality of the reconstructed point cloud of the model is the best, as shown in Tables 4 and 5. Therefore, while training, we set the parameter α to 10 to obtain the best R-D tradeoff.

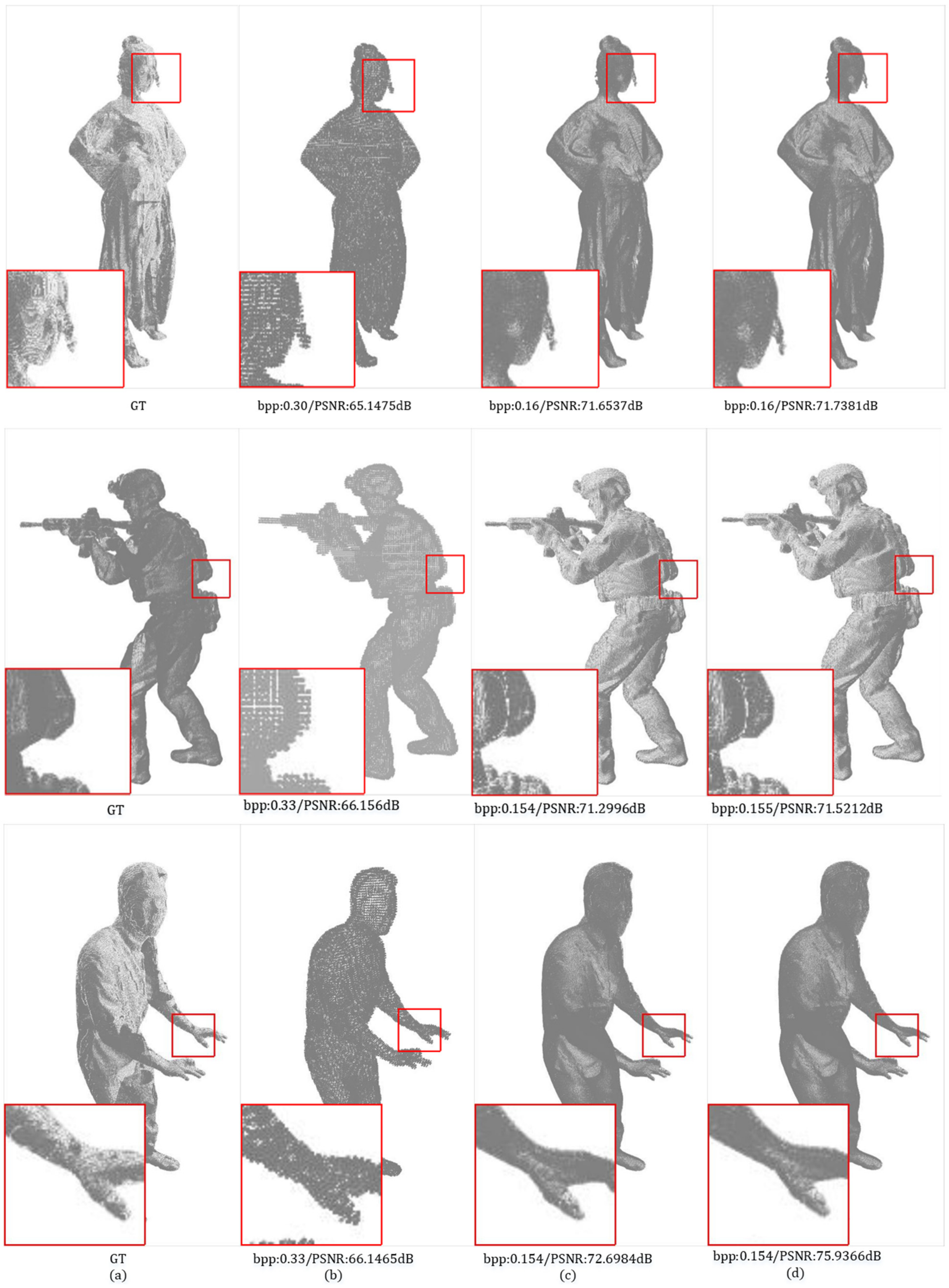

Figure 5. Subjective visualization of 8iVFB for comparison methods, our method and ground truth. (**a**) Ground Truth. (**b**) G-PCC (octree). (**c**) PCGCv2. (**d**) Ours.

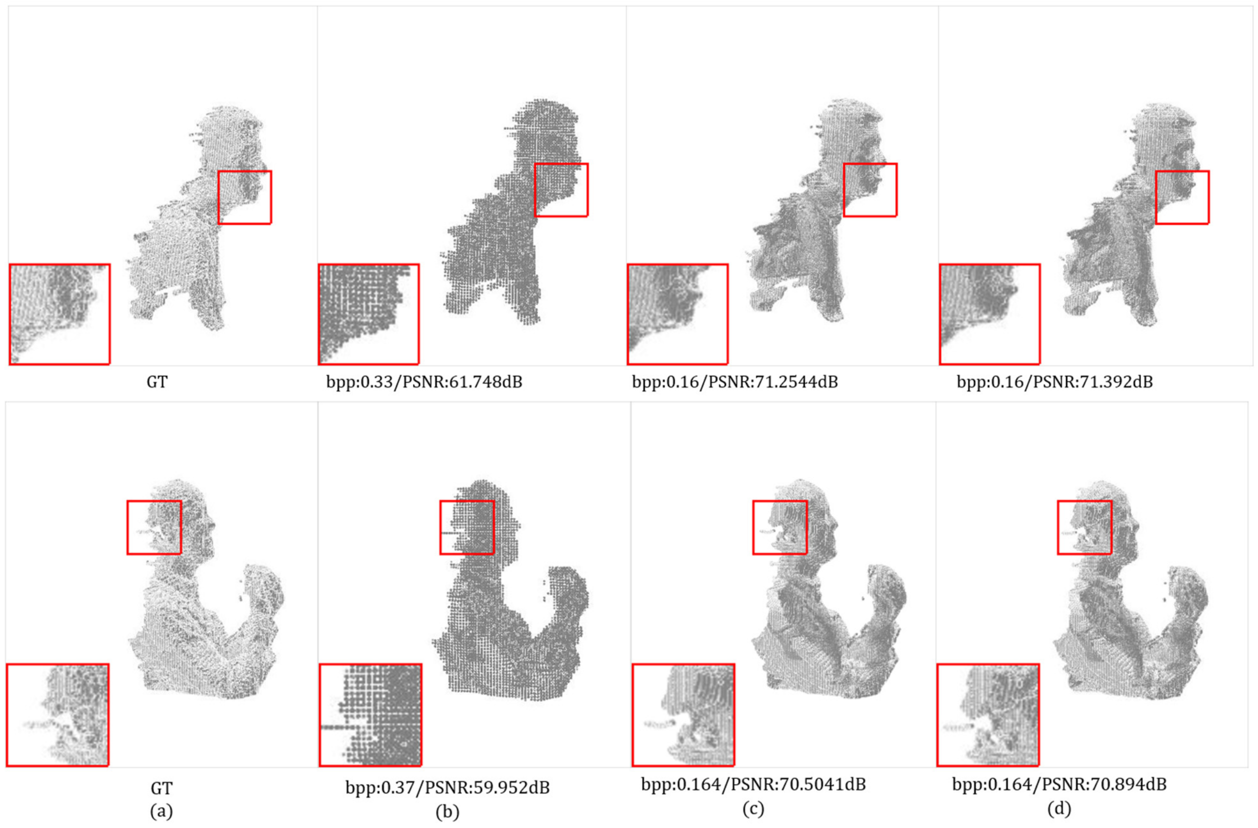

Figure 6. Subjective visualization of MVUB for comparison methods, our method and ground truth. (**a**) Ground Truth. (**b**) G-PCC (octree). (**c**) PCGCv2. (**d**) Ours.

Table 4. BD-Rate gains with different weights of multi-layer residual module.

Point Cloud	**BD-Rate**			
	α = 5	**α = 10**	**α = 15**	**α**
Longdress	−4.32%	−6.96%	−6.76%	−4.55%
Soldier	−7.41%	−10.04%	−8.69%	−7.28%
Loot	−9.56%	−10.23%	−9.15%	−10.13%
RedandBlack	−5.51%	−9.37%	−7.28%	−7.73%
Average	−6.70%	−9.15%	−7.94%	−7.42%

Table 5. BD-PSNR gains with different weights of multi-layer residual module.

Point Cloud	**BD-PSNR**			
	α	**α = 10**	**α = 15**	**α = 20**
Longdress	0.15	0.24	0.21	0.19
Soldier	0.23	0.41	0.37	0.36
Loot	0.28	0.40	0.32	0.33
RedandBlack	0.21	0.36	0.30	0.31
Average	0.22	0.35	0.30	0.28

4.4. Complexity Discussion

In terms of complexity, our experiments are conducted on a workstation equipped with Intel Core i7-9700K CPU and an Nvidia GeForce GTX 1080 GPU. Our prototype, on average, takes 1.59 s for encoding and 5.44 s for decoding on an 8iVFB dataset at the highest bitrate, while the encoding and decoding time for G-PCC (octree) is about 1.6 s and 0.6 s, respectively, as shown in Table 6. The reason for this is that the point cloud reconstruction structure based on a binary classification needs to deal with more voxels. Additionally, we need to point out that our method is implemented in a prototype, which can be further optimized and improved in our future work. The number of parameters of our model is about 894 kb, while the parameter size of PCGCv2 is 778 kb, which shows our method has a relatively lightweight network when compared to other popular algorithms.

Table 6. Average running time (s) of different methods.

	G-PCC (Octree)	G-PCC (Trisoup)	V-PCC	PCGCv2	Ours
Encoding	1.60	8.16	103.41	1.56	1.59
Decoding	0.60	6.58	0.67	5.42	5.44

5. Conclusions

In this paper, we proposed a multi-layer residual module for point-cloud geometry compression, which imposes different level constraints on the accuracy of the point cloud sampling process to deal with the distortion caused by quantization or the introduction of noise. We take advantage of the sparsity of point clouds, with sparse convolutions for the complexity reduction in the space and time. For point cloud applications, such as autonomous driving, a mainstream 64-line LiDAR can produce 1 TB raw point cloud data, which brings a great burden to its transmission and processing. Additionally, our method can greatly reduce it to about 1/130 of its original size. Experimental results validate that our method improves the reconstruction quality of point clouds and preserves more detail. Meanwhile, our method outperforms MPEG G-PCC and V-PCC substantially at a BD-Rate gain. Additionally, compared to the state-of-the-art PCGCv2, our method can achieve an average gain of about 13% and 10% in BD-Rate with D1 and D2 distortion criteria. In our future work, we can further improve the entropy coding module by constructing a more accurate context probabilistic model utilizing 3D neighboring points. One future research direction is to generalize our multi-layer residual architecture to other point cloud processing tasks, for example, point cloud denoising, point cloud segmentation and classification.

Author Contributions: Conceptualization, J.W. and J.Y.; methodology, J.Y.; investigation: L.S.; formal analysis: L.S.; resources, M.-E.W.; data curation, M.-E.W.; writing—review and editing, J.W. and J.Y.; project administration, Q.Z.; funding acquisition, Q.Z. and J.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant 62272016, 61906008 and 61976011, and the Common Program of Beijing Municipal Commission of Education under Grant KM202010005013.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This work was also supported by the Opening Project of Beijing Key Laboratory of Internet Culture and Digital Dissemination Research, and by the Special Academic Collaborative Research Projects (NTUT-BJUT-111-02) between BJUT and NTUT.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Schwarz, S.; Preda, M.; Baroncini, V.; Budagavi, M.; Cesar, P.; Chou, P.A.; Zakharchenko, V. Emerging mpeg standards for point cloud compression. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2019**, *9*, 133–148. [CrossRef]
2. Bello, S.A.; Yu, S.; Wang, C.; Adam, J.M.; Li, J. Review: Deep Learning on 3D Point Clouds. *Remote Sens.* **2020**, *12*, 1729. [CrossRef]
3. Minnen, D.; Ballé, J.; Toderici, G.D. Joint autoregressive and hierarchical priors for learned image compression. In Proceedings of the 2018 Conference on Neural Information Processing Systems, Montréal, QC, Canada, 3–8 December 2018; pp. 10771–10780.

4. Lu, G.; Ouyang, W.; Xu, D.; Zhang, X.; Cai, C.; Gao, Z. DVC: An end-to-end deep video compression framework. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR, Long Beach, CA, USA, 15–20 June 2019; pp. 11006–11015.
5. Wu, C.-Y.; Singhal, N.; Krähenbühl, P. Video compression through image interpolation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 425–440. [CrossRef]
6. Lu, G.; Zhang, X.; Ouyang, W.; Chen, L.; Gao, Z.; Xu, D. An end-to-end learning framework for video compression. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *43*, 3292–3308. [CrossRef] [PubMed]
7. Wiegand, T.; Sullivan, G.; Bjontegaard, G.; Luthra, A. Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 560–576. [CrossRef]
8. Garcia, D.C.; de Queiroz, R.L. Intra-frame context-based octree coding for pointcloud geometry. In Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018; pp. 1807–1811.
9. Schnabel, R.; Klein, R. Octree-based point cloud compression. In *Symposium on Point-Based Graphics*; The Eurographics Association: Eindhoven, The Netherlands, 2006.
10. Huang, Y.; Peng, J.; Kuo, C.C.J.; Gopi, M. A generic scheme for progressive point cloud coding. *IEEE Trans. Vis. Comput. Graph.* **2008**, *14*, 440–453. [CrossRef] [PubMed]
11. Kammerl, J.; Blodow, N.; Rusu, R.B.; Gedikli, S.; Beetz, M.; Steinbach, E. Real-time compression of point cloud streams. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, Chengdu, China, 5–8 August 2012; pp. 778–785.
12. Mekuria, R.; Blom, K.; Cesar, P. Design, Implementation, and Evaluation of a Point Cloud Codec for Tele-Immersive Video. *IEEE Trans. Circuits Syst. Video Technol.* **2016**, *27*, 828–842. [CrossRef]
13. Kathariya, B.; Li, L.; Li, Z.; Alvarez, J.; Chen, J. Scalable point cloud geometry coding with binary tree embedded quadtree. In Proceedings of the 2018 IEEE International Conference on Multimedia and Expo (ICME), San Diego, CA, USA, 23–27 July 2018; pp. 1–6.
14. Xu, Y.; Zhu, W.; Xu, Y.; Li, Z. Dynamic point cloud geometry compression via patch-wise polynomial fitting. In Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brighton, UK, 12–17 May 2019; pp. 2287–2291.
15. Budagavi, M.; Faramarzi, E.; Ho, T.; Najaf-Zadeh, H.; Sinharoy, I. *Samsung's Response to CfP for Point Cloud Compression (Category 2)*; Document ISO/IEC: Macau, China, 2017.
16. Merkle, P.; Smolic, A.; Muller, K.; Wiegand, T. Multi-view video plus depth representation and coding. In Proceedings of the 2007 IEEE International Conference on Image Processing(ICIP), San Antonio, TX, USA, 16 September–19 October 2007; Volume 1, pp. 201–204.
17. Lien, J.-M.; Kurillo, G.; Bajcsy, R. Multi-camera tele-immersion system with real-time model driven data compression. *Vis. Comput.* **2009**, *26*, 3–15. [CrossRef]
18. Ainala, K.; Mekuria, R.N.; Khathariya, B.; Li, Z.; Wang, Y.-K.; Joshi, R. An improved enhancement layer for octree based point cloud compression with plane projection approximation. In Proceedings of the Applications of Digital Image Processing XXXIX, San Diego, CA, USA, 29 August–1 September 2016; pp. 223–231.
19. He, L.; Zhu, W.; Xu, Y. Best-effort projection based attribute compression for 3d point cloud. In Proceedings of the 2017 23rd Asia-Pacific Conference on Communications (APCC), Perth, Australia, 11–13 December 2017; pp. 1–6.
20. Wang, J.; Ding, D.; Li, Z.; Ma, Z. Multiscale point cloud geometry compression. In Proceedings of the Data Compression Conference, Snowbird, UT, USA, 23–26 March 2021; pp. 73–82.
21. Quach, M.; Valenzise, G.; Dufaux, F. Learning convolutional transforms for lossy point cloud geometry compression. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 4320–4324.
22. Huang, T.; Liu, Y. 3d point cloud geometry compression on deep learning. In Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 21–25 October 2019.
23. Wang, J.; Zhu, H.; Liu, H.; Ma, Z. Lossy Point Cloud Geometry Compression via End-to-End Learning. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 4909–4923. [CrossRef]
24. Charles, R.Q.; Su, H.; Kaichun, M.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 77–85. [CrossRef]
25. Choy, C.; Gwak, J.; Savarese, S. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 3075–3084.
26. Tang, D.; Singh, S.; Chou, P.A.; Hane, C.; Dou, M.; Fanello, S.; Keskin, C. Deep Implicit Volume Compression. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 1290–1300.
27. Huang, L.; Wang, S.; Wong, K.; Liu, J.; Urtasun, R. Octsqueeze: Octreestructured entropy model for lidar compression. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 1310–1320.

28. Que, Z.; Lu, G.; Xu, D. Voxelcontext-net: An octree based framework for point cloud compression. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 6038–6047.
29. Wen, X.; Wang, X.; Hou, J.; Ma, L.; Zhou, Y.; Jiang, J. Lossy geometry compression of 3d point cloud data via an adaptive octree-guided network. In Proceedings of the 2020 IEEE International Conference on Multimedia and Expo (ICME), London, UK, 6–10 July 2020; pp. 1–6.
30. Zhu, W.; Xu, Y.; Ding, D.; Ma, Z.; Nilsson, M. Lossy Point Cloud Geometry Compression via Region-Wise Processing. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 4575–4589. [CrossRef]
31. Gao, L.; Fan, T.; Wan, J.; Xu, Y.; Sun, J.; Ma, Z. Point cloud geometry compression via neural graph sampling. In Proceedings of the 2021 IEEE International Conference on Image Processing (ICIP), Anchorage, AK, USA, 19–22 September 2021; pp. 3373–3377.
32. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, Inception-ResNet and the impact of residual connections on learning. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; pp. 4278–4284.
33. Balle, J.; Minnen, D.; Singh, S.; Hwang, S.J.; Johnston, N. Variational image compression with a scale hyperprior. *arXiv* **2018**, arXiv:1802.01436.
34. Chang, A.X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Yu, F. Shapenet: An information-rich 3d model repository. *arXiv* **2015**, arXiv:1512.03012.
35. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
36. Eugene d'Eon, E.; Harrison, B.; Myers, T.; Chou, P.A. 8i Voxelized Full Bodies—A Voxelized Point Cloud Dataset. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) Input Document m38673/M72012. May 2016. Available online: http://plenodb.jpeg.org/pc/8ilabs (accessed on 10 September 2022).
37. Charles, L.; Cai, Q.; Sergio, O.E.; Philip, A.C. Microsoft Voxelized Upper Bodies—A voxelized Point Cloud Dataset, ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document m38673/M72012. May 2016. Available online: https://plenodb.jpeg.org/pc/microsoft (accessed on 10 September 2022).
38. Schwarz, S.; Chou, P.A.; Sinharoy, I. Common Test Conditions for Point Cloud Compression. ISO/IEC JTC1/SC29/WG11 N18474. 2018. Available online: https://ds.jpeg.org/documents/jpegpleno/wg1n88044-CTQ-JPEG_Pleno_PCC_Common_Test_Conditions_3_3.pdf (accessed on 10 September 2022).
39. JPEG Pleno PCC, Jpeg Pleno Point Cloud Coding Common Test Conditions. JPEG (ISO/IEC JTC 1/SC 29/WG1). 2019. Available online: https://ds.jpeg.org/documents/jpegpleno/wg1n88044-CTQ-JPEG_Pleno_PCC_Common_Test_Conditions_3_4.pdf (accessed on 10 September 2022).
40. MPEG, "Mpeg-pcc-tmc2". Available online: https://github.com/MPEGGroup/mpeg-pcc-tmc2 (accessed on 10 September 2022).
41. Tian, D.; Ochimizu, H.; Feng, C.; Cohen, R.; Vetro, A. Geometric distortion metrics for point cloud compression. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 3460–3464.
42. Mekuria, R.; Laserre, S.; Tulvan, C. Performance assessment of point cloud compression. In Proceedings of the 2017 IEEE Visual Communications and Image Processing (VCIP), St. Petersburg, FL, USA, 10–13 December 2017; pp. 1–4.