Voxel-based Deep Point Cloud Geometry Compression

Giuseppe Valenzise, Maurice Quach, Dat-Than Nguyen, Frédéric Dufaux

Voxel-based Deep Point Cloud Geometry Compression

Giuseppe Valenzise, Maurice Quach, Dat-Than Nguyen, Frédéric Dufaux
Université Paris-Saclay, CNRS, CentraleSupelec, Laboratoire des Signaux et Systèmes (L2S)

Abstract : *We present two learning-based methods for coding point clouds geometry. The two methods target lossy and lossless compression, respectively, and have in common the fact to use a voxel-based representation of geometry. This representation enables us to extend well-known architectures used for 2D image generation and compression to 3D. We show that, when the point cloud density is sufficiently high, the voxel-based approach achieves state-of-the-art performance compared to conventional octree-based methods such as MPEG G-PCC.*

Keywords: point cloud, geometry coding, voxels, auto-encoder, generative models.

1 Introduction

Due to recent advances in visual capture technology, point clouds have been recognized as a crucial data structure for 3D content. In particular, point clouds are essential for numerous applications such as virtual and mixed reality, sensing for autonomous vehicle navigation, architecture and cultural heritage, etc. Point clouds are sets of 3D points identified by their coordinates, which constitute the geometry of the point cloud. In addition, each point can be associated with attributes like colors, normals and reflectance. Point clouds can have a massive number of points, especially in high precision or large scale captures. This entails a huge storage and transmission cost. As a result, Point Cloud Compression (PCC) is fundamental in practice. The Moving Picture Experts Group (MPEG) has recently released two PCC standards [1]: Geometry-based PCC (G-PCC) and Video-based PCC (V-PCC). G-PCC approaches PCC from a 3D perspective and compresses point clouds in their native form using 3D data structures such as octrees. On the other hand, V-PCC approaches PCC from a 2D perspective, projects 3D data onto a 2D plane and makes use of video compression technology. Recently, deep point cloud compression (DPCC) methods have been proposed and shown to provide significant coding gains compared to traditional methodologies [2, 3, 4, 5].

In this paper, we focus on the compression of point cloud geometry, and we review two recently proposed learning-based methods for lossy and lossless coding. We consider the case of voxelized point clouds. Voxelization is the process that quantizes the coordinates of a point cloud to integer precision prior to the coding process and is typically applied in most codecs to discretize the geometry. We also make the implicit hypothesis that the point cloud is dense enough to exhibit local correlations among neighboring points on the voxel grid – in other terms, we assume there is not too much "empty space" between points. This enables us to employ deep neural networks with voxel-based 3D convolutions (see, e.g., [2]), which have been shown to be particularly effective in point cloud compression. On the other hand, point-based convolutions [6, 7] are also possible [8], but their performance in PCC is still lagging behind traditional hand-crafted methods such as those used in MPEG G-PCC.

When a point cloud is voxelized, its geometry can be expressed as a binary signal over the voxel grid. In particular, a voxel is consider occupied if it contains at least one point, and is non occupied otherwise. Based on this observation, learning-based methods for geometry coding typically cast decoding as a binary classification problem, see Section 2. Instead, in lossless compression an explicit, accurate estimation of the likelihood of voxel occupancy is necessary: in this case, the decoding can be interpreted as a voxel generation process, as we will see in Section 3.

2 Lossy compression

Our lossy compression scheme is inspired by the success of variational auto-encoder (VAE) methods for image compression [9, 10]. The general architecture of a VAE-based codec is illustrated in Figure 1. An input signal x (pixels for the case of 2D images, binary voxel occupancies for 3D PCs) is transformed by an *analysis* network f_a into a latent representation y and quantized into $\tilde{y}$. This is later used as input to a *synthesis* network, producing an approximated reconstruction $\tilde{x}$ of the original signal. The quantizer Q represents the main difference with respect to a conventional VAE, in which the latent space is typically continuous. Since quantization is not differentiable, several approximations have been proposed; in this work, we replace quantization noise by uniform noise during training, as initially suggested in [9]. Quantization, along with the fact that y has smaller dimensionality compared to x, contribute both to achieve compression. The quantized latent code $\tilde{y}$ is entropy coded and transmitted as bitstream. A basic version of the VAE-based codec assumes that the components of $\tilde{y}$ are i.i.d. and computes symbol probabilities for entropy coding accordingly. However, this has been shown to be suboptimal, and later versions introduce a *hyperprior* model [10], where the probability of the quantized latent variables is also modeled through a VAE. This allows the codec to capture the residual spatial dependencies among voxels.

The model is trained end-to-end using a $D + \lambda R$ loss function, for a given value of λ. The rate term includes both the bits for the latent variables $\tilde{y}$ and $\tilde{z}$, which are approximated by their differential entropy at training time. The D part is computed using the *focal loss*, a variant of the binary cross-entropy loss [11] used in classification, which has been shown to be more effective when the class distribution is strongly unbalanced (in the case of PC, most of the voxels are empty). The output of the VAE is

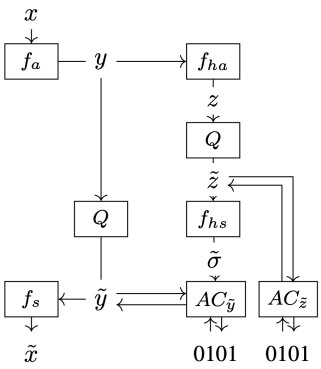

Figure 1: Scheme of a VAE-based codec with hyperprior.

Point cloud	Metric	BD-PSNR
loot	D1	5.91 / 6.99
	D2	6.87 / 6.13
redandblack	D1	5.01 / 6.48
	D2	5.93 / 5.63
longdress	D1	5.55 / 6.94
	D2	6.60 / 6.01
soldier	D1	5.57 / 6.93
	D2	6.57 / 6.04
Average	D1	5.51 / 6.83
	D2	6.50 / 5.95

Table 1: RD performance of the proposed VAE-based codec compared to G-PCC (version 10.00). We specify BD-PSNR values (dB) compared to G-PCC trisoup and G-PCC octree in each cell (trisoup BD-PSNR / octree BD-PSNR).

a set of per-voxel probabilities of occupancy, which need to be thresholded in order to provide the final binary occupancy values.

The choice of the threshold is of paramount importance for coding performance. We choose to optimize this threshold at the encoder side and transmit it as a side information into the bitstream. Our codec operates on a block-by-block basis, in such a way that all decisions are taken locally and adapted to the spatially varying density in the point cloud. Further details about the architecture of the codec are reported in [3], and the code is publicly available as a toolbox [12].

2.1 Performance evaluation

We report RD performance on a subset of four dense point clouds. Details about the training dataset, as well as the training hyperparameters, are given in [3]. We evaluate the different conditions using G-PCC trisoup and octree as baselines. The octree is the basic coding structure of G-PCC: the point cloud is recursively subdivided into octants, and only those nodes that contain at least a point are further split. On top of the basic vanilla octree, G-PCC adds a number of additional modes and optimizations, including direct coding for isolated points, planar modes and sophisticated contexts for entropy coding (see [1] for a survey). The triangle soup (trisoup) mode, in particular, adds local triangular approximations at the octree leaves, and is typically included in PC compression benchmarks (although it is not included in the released standard). The distortion metrics D1 and D2 are obtained, respectively, from symmetrized point-to-point and point-to-distance mean squared errors, which are converted to PSNR using the original PC bit depth as peak error [13].

Table 1 reports Bjontegaard Delta PSNR of the proposed scheme compared to G-PCC trisoup and octree: the coding gains are significant for all the considered point clouds, demonstrating the potential of voxel-based convolutions in the compression of dense point clouds.

3 Lossless compression

Voxel-based convolutional architectures can be successfully used also for lossless geometry coding of dense point clouds. Compared to lossy compression, the goal here is to estimate accurately the voxel occupancy probabilities for entropy coding, rather than handling class imbalance to favor a precise binary reconstruction. In the following, we present briefly the *VoxelDNN* codec we proposed in [14], whose general architecture is illustrated in Figure 2.

The basic element of the codec is the context model, which is based on an auto-regressive generative model inspired by PixelCNN [15]. Specifically, let v_i denote the binary occupancy of a voxel i. We factorize the joint distribution $p(v)$ of a block of voxels v as a product of conditional distributions $p(v_i|v_{i-1},\dots,v_1)$ over the voxel volume: $p(v) = \prod_{i=1}^{N} p(v_i|v_{i-1},v_{i-2},\dots,v_1)$, with N the number of voxels in a block. Each term $p(v_i|v_{i-1},\dots,v_1)$ is the probability of the voxel v_i being occupied given the occupancy of all *previous* voxels. An illustration is given in Figure 2(c). We approximate $\hat{p}(v_i|v_{i-1},\dots,v_1)$ using a convolutional neural network, which we train by minimizing the binary cross-entropy $\mathbb{E}_{v\sim p(v)}\left[\sum_{i=1}^{N} -\log \hat{p}(v_i)\right]$. This is equivalent to minimize the distance between the estimated conditional distributions and the real data distribution, yielding accurate context distributions for arithmetic coding. This process can be carried out on blocks of different sizes (typically, N ranges between 8^3 and 64^3), using a rate-optimization algorithm. Since empty blocks in a point cloud do not bring any useful context, we apply an octree-based partitioning to pre-process the PC and remove the non-occupied space. Further details about VoxelDNN are available in [14].

3.1 Performance evaluation

A comparison of the bitrates of VoxelDNN and G-PCC (v. 12) for various PC categories is reported in Table 2. We observe that VoxelDNN achieves significant gains of up to 37% on dense point clouds (MVUB and 8i). On sparser point clouds the gains are smaller, but still competitive compared to G-PCC. The only exception is the PC "Arco Valentino", which has large density variations and very sparse regions where context modeling is ineffective.

Notice that a drawback of VoxelDNN is the sequential decoding of voxels, which is equivalent to a sequential sampling from the voxel occupancy distribution. As a result, the decoding times are significantly higher than G-PCC. In a follow-up work [16], we propose a partial solution consisting in breaking some dependencies to parallelize decoding, achieving execution times withing one order of magnitude from G-PCC.

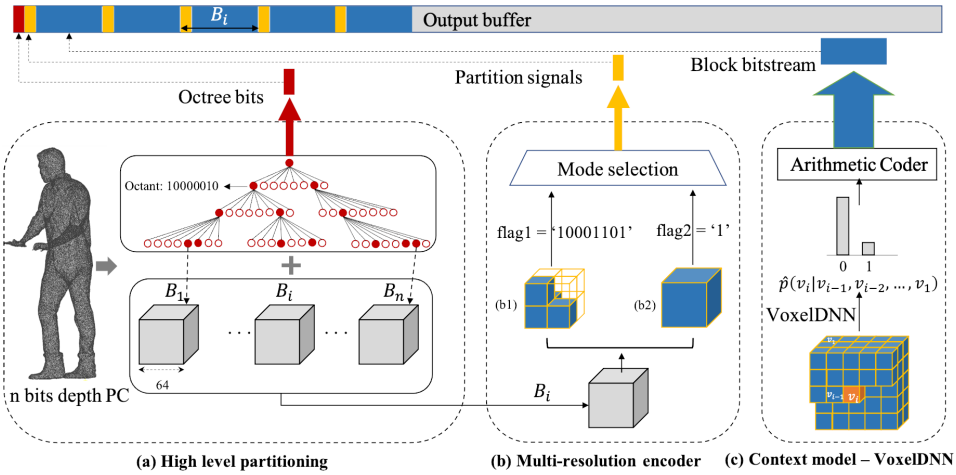

Figure 2: General architecture of the VoxelDNN codec, composed by a high-level octree partitioning part; a multi-resolution encoder; and the basic context model unit.

		G-PCC	VoxelDNN	
Dataset	Point Cloud	bpov	bpov	Gain over G-PCC
MVUB	Phil	1.1599	0.8252	-28.86%
	Ricardo	1.0673	0.7572	-29.05%
	Average	**1.1136**	**0.7912**	**-28.95%**
8i	Redandblack	1.0893	0.7003	-35.71%
	Loot	0.9524	0.6084	-36.12%
	Thaidancer	0.9990	0.6627	-33.66%
	Boxer	0.9492	0.5906	-37.78%
	Average	**0.9975**	**0.6405**	**-35.79%**
CAT1	Frog	1.8990	1.7071	-10.11%
	Arco Valentino	4.8531	4.9900	+2.82%
	Shiva	3.6716	3.5135	-4.31%
	Average	**3.4746**	**3.4035**	**-3.86%**
USP	BumbaMeuBoi	5.4068	5.066	-6.29%
	RomanOiLight	1.8604	1.6231	-12.76%
	Average	**3.6336**	**3.4855**	**-9.52%**

Table 2: Average rate in bits per occupied voxel (bpov) of proposed method and percentage reductions compared with MPEG G-PCC.

4 Conclusion and perspectives

We have described two deep learning-based architectures for point cloud geometry compression (lossy and lossless). In both cases, the use of voxel-based convolutions provides significant gains over the reference G-PCC solution for dense point clouds. We believe this advantage is given by the ability to represent the underlying geometric structure (local surfaces, objects, etc.), which is not captured by simple octree-based approaches. On the other hand, when the point cloud is sparser or scant (as for LiDAR data), voxel-based techniques break down, and other kinds of approaches are more suitable, such as point-based and graph convolutions [17]. Compression of very sparse PC is still an open challenge.

References

[1] C. Cao, M. Preda, V. Zakharchenko, E. S. Jang, and T. Zaharia, "Compression of sparse and dense dynamic point clouds—methods and standards," *Proceedings of the IEEE*, vol. 109, no. 9, pp. 1537–1558, 2021.

[2] M. Quach, G. Valenzise, and F. Dufaux, "Learning convolutional transforms for lossy point cloud geometry compression," in *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 4320–4324, ISSN: 1522-4880.

[3] ——, "Improved deep point cloud geometry compression," in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, 2020, pp. 1–6.

[4] A. F. R. Guarda, N. M. M. Rodrigues, and F. Pereira, "Deep learning-based point cloud coding: A behavior and performance study," in *2019 8th European Workshop on Visual Information Processing (EUVIP)*, pp. 34–39, ISSN: 2164-974X.

[5] J. Wang, H. Zhu, H. Liu, and Z. Ma, "Lossy point cloud geometry compression via end-to-end learning," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2021.

[6] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[8] W. Yan, S. Liu, T. H. Li, Z. Li, G. Li *et al.*, "Deep autoencoder-based lossy geometry compression for point clouds," *arXiv preprint arXiv:1905.03691*, 2019.

[9] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *2017 5th International Conference on Learning Representations (ICLR)*.

[10] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *2018 6th International Conference on Learning Representations (ICLR)*.

[11] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, ISSN: 2380-7504.

[12] M. Quach, G. Valenzise, and F. Dufaux, "A Deep Point Cloud Geometry Coding Toolbox," in *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, Shenzhen, China, Jul. 2021.

[13] D. Tian, H. Ochimizu, C. Feng, R. Cohen, and A. Vetro, "Geometric distortion metrics for point cloud compression," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, pp. 3460–3464. [Online]. Available: http://ieeexplore.ieee.org/document/8296925/

[14] D. T. Nguyen, M. Quach, G. Valenzise, and P. Duhamel, "Lossless Coding of Point Cloud Geometry using a Deep Generative Model," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[15] A. van den Oord and N. Kalchbrenner, "Pixel RNN," in *ICML*, 2016.

[16] D. T. Nguyen, M. Quach, G. Valenzise, and P. Duhamel, "Multiscale deep context modeling for lossless point cloud geometry compression," in *IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, Shenzhen (virtual), China, Jul. 2021.

[17] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Transactions On Graphics*, vol. 38, no. 5, pp. 1–12, 2019.