Deep Learning-based Point Cloud Geometry Coding with Resolution Scalability

André F. R. Guarda
Instituto Superior Técnico
Instituto de Telecomunicações
Lisbon, Portugal
andre.guarda@lx.it.pt

Nuno M. M. Rodrigues
ESTG, Instituto Politécnico de Leiria
Instituto de Telecomunicações
Leiria, Portugal
nuno.rodrigues@co.it.pt

Fernando Pereira
Instituto Superior Técnico
Instituto de Telecomunicações
Lisbon, Portugal
fp@lx.it.pt

***Abstract*—Point clouds are a 3D visual representation format that has recently become fundamentally important for immersive and interactive multimedia applications. Considering the high number of points of practically relevant point clouds, and their increasing market demand, efficient point cloud coding has become a vital research topic. In addition, scalability is an important feature for point cloud coding, especially for real-time applications, where the fast and rate efficient access to a decoded point cloud is important; however, this issue is still rather unexplored in the literature. In this context, this paper proposes a novel deep learning-based point cloud geometry coding solution with resolution scalability via interlaced sub-sampling. As additional layers are decoded, the number of points in the reconstructed point cloud increases as well as the overall quality. Experimental results show that the proposed scalable point cloud geometry coding solution outperforms the recent MPEG Geometry-based Point Cloud Compression standard which is much less scalable.**

Keywords—point cloud coding, deep learning, scalable coding, resolution scalability, interlaced sampling

I. Introduction

With the rising popularity of virtual and augmented reality applications, 3D visual representation formats such as point clouds (PCs) have become a hot research topic. Since PCs are essentially a set of points in the 3D space with associated features, they are naturally suitable to facilitate user interactivity and offer a high level of immersion. The point coordinates constitute the so-called *geometry*, while properties related to color, normals, etc., are considered as *attributes*. However, providing realistic, interactive and immersive experiences typically requires PCs with a rather high number of points. In this context, efficient coding is critical as recognized by standardization groups such as MPEG [1], [2], [3] and JPEG [4] which have been developing appropriate PC coding standards. Scalability is often a requirement for several PC applications where the speed of access to a PC is relevant, even if at lower quality or resolution, usually by partially decoding a bitstream structured in multiple layers. Although it generally comes at the cost of a reduced compression efficiency, scalable PC coding is nonetheless a feature that has been less explored in the literature. For PC geometry, scalability can be achieved in multiple ways, notably: i) resolution scalability, where the number of points of the reconstructed PC is increased as additional layers are decoded for a fixed precision; ii) quality scalability, where the PC quality is improved along the layers, for a fixed precision; and iii) precision scalability, where the precision (bit depth) of the point coordinates is increased as more layers are decoded.

This work was funded by Fundação para a Ciência e Tecnologia (FCT), Portugal, Ph.D. Grant SFRH/BD/118218/2016, by FCT/MEC through national funds and when applicable co-funded by EU funds under the project UIDB/EEA/50008/2020.

The popularity of deep learning in multimedia processing tasks has largely increased in recent years due to its impressive performance. In terms of coding, recent deep learning-based image coding solutions offer very promising results, even outperforming state-of-the-art image codecs [5]. Part of this success may be attributed to convolutional neural networks, which take advantage of the spatial redundancy by hierarchically detecting patterns to obtain a meaningful latent representation. In this context, it is natural to extend the deep learning-based coding approach to PCs, for example, coding 3D blocks of voxels instead of 2D blocks of pixels as for image and video coding.

In this context, this paper proposes a novel resolution scalable deep learning-based PC geometry coding solution (RS-DLPCC), a first in the literature. This coding solution divides the PC into interlaced 3D blocks, which are progressively coded with a 3D convolutional neural network-based deep model. At the decoder, the first scalable layer provides a low-resolution PC reconstruction, in practice a sub-sampling of the original PC, while subsequent layers successively add more points until reaching full resolution. Experimental results show that the proposed RS-DLPCC solution is able to reconstruct PCs with a high subjective quality from the very first, low-rate layer, clearly outperforming in terms of rate-distortion (RD) performance the MPEG Geometry-based Point Cloud Compression (G-PCC) standard, which is much less scalable. The remaining sections of this paper are organized as follows: Section II briefly reviews the relevant state-of-the-art on PC geometry coding. Section III describes the proposed RS-DLPCC solution. Experimental results and their analysis are presented in Section IV. Finally, Section V concludes the paper.

II. Background Work

PC geometry data is sparse, unordered, unstructured and unconnected. These unique characteristics present new challenges for PC coding methods [6]. Octree-based methods have become the first PC geometry representation approach, with the Point Cloud Library [7] codec becoming the first reference solution. An octree data structure is built by dividing the 3D space into eight child nodes; occupied nodes are then recursively divided, until the desired level of detail depth is reached. The state-of-the-art octree-based coding approach is the MPEG G-PCC standard [1], which first uses an octree to represent the 'skeleton' of the PC, and then complements it with a triangle-based surface estimation, the so-called *trisoup*. Despite octree-based coding methods being inherently scalable, allowing different levels of detail, the G-PCC scalability potential has not been much exercised. On the other hand, the MPEG Video-based Point Cloud Compression (V-PCC) standard [3] does not code the 3D PC geometry directly, and instead projects it into 2D images, which can then be coded with very efficient video codecs such as HEVC [8].

978-1-7281-9320-5/20/$31.00 ©2020 IEEE

More recently, deep learning-based methods were proposed for PC coding. In [9], a point-based autoencoder is used to directly encode the PC geometry coordinates. However, due to the PCs unstructured nature, it is only able to extract global PC features and does not exploit local point relations. Unlike images, PCs are not regularly structured, making it challenging to process them with regular convolutional neural networks (CNN). To circumvent this issue, in [10], [11], [12], the PC geometry is converted into a 3D binary, voxel-based representation, where voxels may be occupied or not; in practice, a '1' signals a filled voxel while a '0' signals an empty voxel. This voxel-based representation defines a regular structure that allows the use of CNNs, similarly to image and video data. These solutions apply a CNN as an autoencoder, mimicking the transform-based coding architecture, typical in image coding. However, none of the published deep learning-based coding solutions offers scalability features as targeted in this paper.

III. Resolution Scalable Point Cloud Coding

This section describes the proposed RS-DLPCC solution.

A. Overall Codec Architecture

As in [10], [11], [12], RS-DLPCC adopts a voxel-based representation. However, this type of representation may be rather heavy since it does not only explicitly represent the 'filled voxels' but also the remaining empty space, for a given precision, thus requiring a significant amount of storage and processing memory. Since this restricts the size of data that can be processed, the target PC has to be divided into regular 3D blocks of smaller size, which are then handled separately.

To provide resolution scalability, a key target for the proposed RS-DLPCC solution, the PC is divided into coarser, interlaced blocks, as described in the next section. Each block is then coded separately in different scalable layers using a single CNN-based deep model. To improve the RD performance, a dedicated module optimizes the order by which the interlaced blocks are assigned to the successive resolution scalable layers. At the decoder side, the PC is reconstructed by progressively decoding each layer of blocks, thus increasing the total number of decoded points and reconstructing an increasingly denser and richer PC. In fact, since blocks, and therefore layers, are coded independently and each offers a meaningful PC (what does not necessarily happen in scalable coding solutions), RS-DLPCC can also be regarded as a multiple description coding solution. The overall RS-DLPCC architecture is presented in Fig. 1 and each main module is described in the following.

B. Interlaced Blocks Creation

A straightforward way to organize a PC into blocks is to divide it directly into disjoint blocks of a specific size. However, given the resolution scalability goal in this paper, it is proposed to divide the PC into interlaced blocks, as represented in Fig. 2. This interlaced approach allows to successively increase the number of decoded points (i.e., the PC density) with each new decoded layer, which is very effective from a subjective quality point of view. The interlaced blocks are obtained as follows:

- **Disjoint super-blocks PC division** – The PC is first divided into disjoint blocks, referred as *super-blocks*.
- **Super-blocks interlaced sub-sampling** – Using interlaced sub-sampling with a sampling factor *sf*, each super-block is divided into smaller blocks with the target size for deep learning-based coding. An example is shown in Fig. 3 where, for $sf = 2$, the coding blocks are half the size of the super-blocks in each spatial dimension. This novel PC representation approach arranges the full PC into several super-blocks, each with up to $nib = sf^3$ interlaced blocks, thus enabling resolution scalability with up to *nib* scalable layers. Each interlaced block is independently coded with a deep learning-based solution described in the next section.

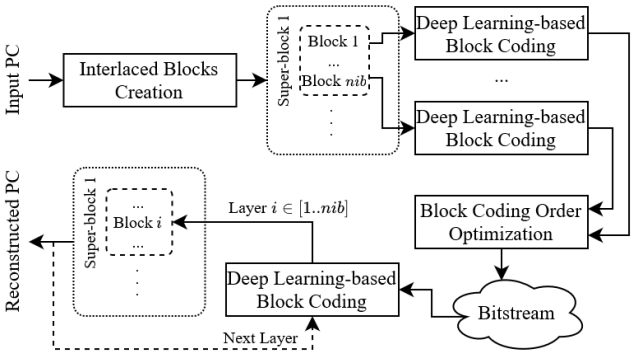

Fig. 1. Overall architecture of the proposed RS-DLPCC solution.

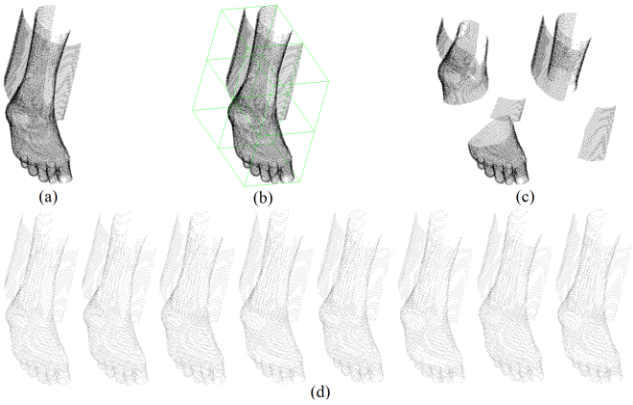

Fig. 2. Example of disjoint and interlaced blocks for the same super-block: (a) 128×128×128 super-block; (b) disjoint division into eight 64×64×64 blocks, four of which are empty; (c) the four disjoint occupied blocks obtained from (b); (d) the eight 64×64×64 interlaced blocks obtained from (a).

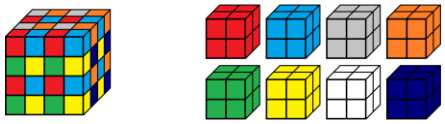

Fig. 3. Example of interlaced sampling with $sf = 2$. A super-block of 4×4×4 samples results into 8 blocks, each with 2×2×2 samples.

C. Deep Learning-based Block Coding

This section presents the adopted deep learning-based PC geometry block coding solution. Based on successful CNN architectures for image coding, the adopted end-to-end coding model is presented in Fig. 4. This architecture can be divided into four main coding stages as follows:

- **Autoencoder** – The convolutional autoencoder transforms the input block into a latent representation with lower dimensionality, in a way comparable to the transform coding stage in traditional image coding.

- **Conditional entropy bottleneck** – A conditional entropy bottleneck layer from the Tensorflow compression library [13] is used to quantize, as a simple rounding operation, and then entropy code the block latent representation. This bottleneck uses a Gaussian scale mixture conditioned on a

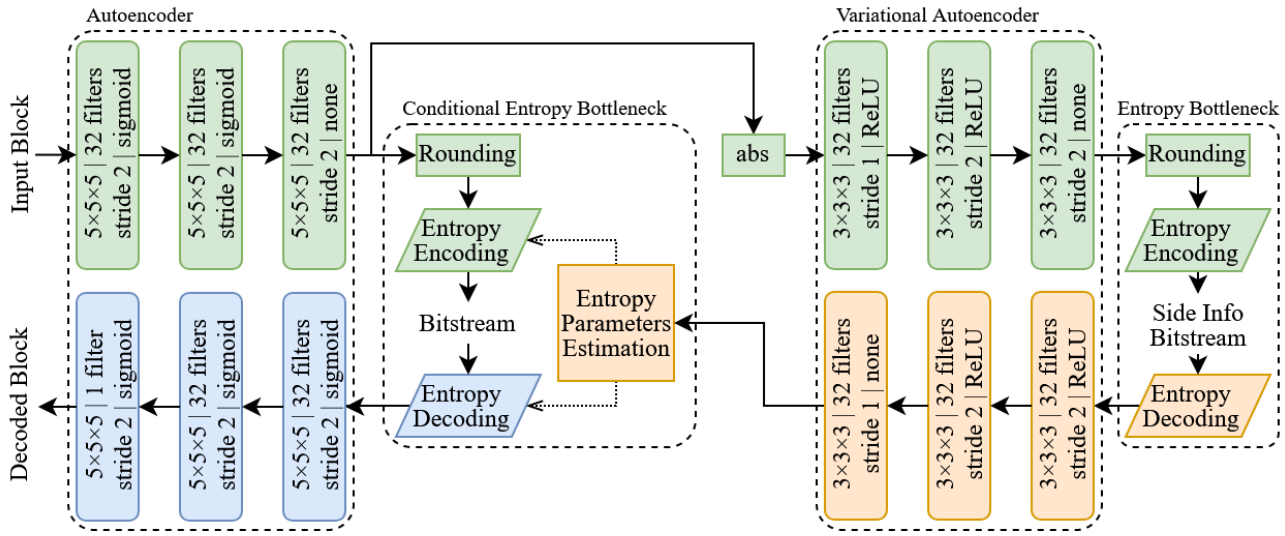

Fig. 4. Deep learning-based block coding architecture corresponding to the end-to-end coding model. Green blocks are encoder-only, blue blocks are decoder-only, and orange blocks are both encoder and decoder.

hyperprior as the entropy coding model. During training, this layer estimates the entropy of the latent representation according to the entropy coding model, which is used for the RD optimization process. At coding time, a range encoder is used to create the bitstream.

- **Variational autoencoder** – A variational autoencoder (VAE) is used to capture possible structure information still present in the block latent representation, which is then used as a hyperprior for the conditional entropy bottleneck. This way, the entropy coding model parameters can be more accurately estimated and adapted for each coded block. In this process, the VAE generates its own latent representation, which also has to be coded and transmitted in the bitstream as additional side information to the decoder, so that the entropy coding model parameters can be replicated at the decoder.

- **Entropy bottleneck** – Similar to the conditional entropy bottleneck, this entropy bottleneck quantizes and entropy codes the VAE latent representation. However, it uses a fixed entropy coding model instead of an adaptive one. As all the components of the end-to-end coding model are jointly trained, the additional side information rate is compensated by the latents rate reduction, thus optimizing the overall RD performance. Experiments show that this side information represents around 7% of the total block coding rate.

D. Block Coding Order Optimization

At the decoder side, all super-blocks must be reconstructed to generate a full PC although not necessarily always with all the interlaced blocks. In this resolution scalable context, at each layer, only one block in each super-block is decoded and added to the reconstructed PC, independently of the other blocks. Although the coding order of the blocks within a super-block does not impact the reconstruction quality at the last scalable layer, experiments have shown that the block coding order has an impact on the quality of the intermediate decoded layers. With this in mind, after encoding all blocks with the end-to-end coding model, the best coding order for the *nib* interlaced blocks constituting each super-block is determined by sequentially adding the block that reduces the accumulated distortion the most at each layer. This block coding order optimization only considers the accumulated distortion at each layer since it has been observed that the rate for all interlaced blocks within a super-block tends to be very similar; this approach allows reducing the overall coding complexity since no rate has to be computed for this optimization. Since the coding order for each super-block will vary depending on its specific content, this order has to be signaled in the coding bitstream at a rather minimal rate cost.

E. Training Process

Prior to actually coding the PCs, the deep learning-based end-to-end coding model has to be trained. To optimize the RD performance, the training loss function considers both the distortion and the rate, with a Lagrangian multiplier λ to balance both terms. The block distortion is measured at voxel level as a binary classification error using the so-called Focal Loss [14], defined as follows:

$$FL(v,u) = \begin{cases} -\alpha(1-v)^{\gamma}\log v\,, & u=1 \\ -(1-\alpha)v^{\gamma}\log(1-v)\,, & u=0 \end{cases}, \qquad (1)$$

where u is the original binary voxel value and v is the corresponding reconstructed voxel value. A weight parameter, α, is used to control the class imbalance effect since the number of '0' valued voxels in a block is vastly superior to the number of '1' valued voxels. The parameter γ allows increasing the importance of correcting misclassified voxels in relation to improving the classification score of already correct voxels; $\gamma = 2$ was found to be an appropriate value. In summary, the RD training loss is defined as:

$$Loss = \sum_{v,u \in block} (FL(v,u)) + \lambda \times rate, \qquad (2)$$

where *rate* is the estimated rate per block, including also the VAE latents side information rate. The end-to-end coding model was trained for several values of the α weight, namely 0.5, 0.6, 0.7, 0.8 and 0.9, and for λ=500.

IV. Performance Assessment

In this section, the performance of the proposed RS-DLPCC solution is evaluated and discussed.

A. Experimental Setup

The proposed RS-DLPCC solution has been evaluated for *Statue Klimt*, *Queen*, *Longdress* and *House without Roof* PCs selected from the MPEG PCC dataset [15], down-sampled to match the precision specified in the MPEG PCC test

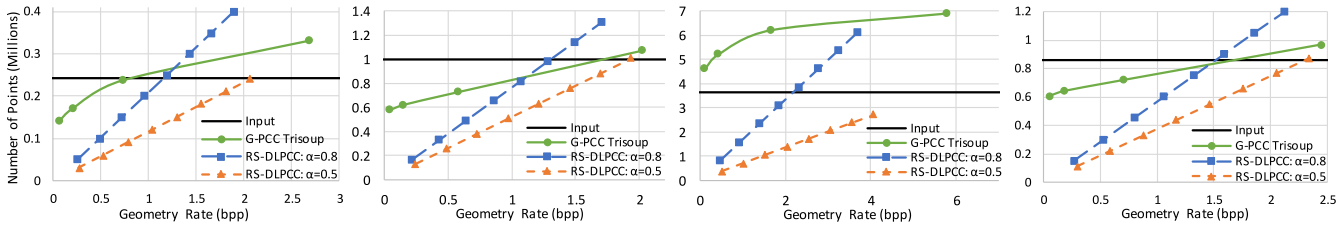

Fig. 5. Number of points in the reconstructed PCs. From left to right: *Statue Klimt*, *Queen*, *House without Roof* and *Longdress*.

conditions [15], which serves as reference for the used objective quality metrics. PCs were divided into interlaced blocks with the target coding size of 64×64×64 voxels, using an interlaced sampling factor *sf* = 2, allowing to obtain eight scalable layers for 128×128×128 super-blocks. The training PCs were selected from the remaining MPEG PCC dataset and divided into interlaced blocks as described above; interlaced blocks with less than 500 occupied voxels were discarded, generating a total of 6000 blocks for training.

The coding benchmark for comparison with the proposed RS-DLPCC solution is naturally the MPEG G-PCC standard [1], namely the reference software version 7.0, using the octree+trisoup coding mode. The RD performance is assessed using three PC geometry objective quality metrics, the first two adopted by MPEG for the development of its standards:

- **D1 (point-to-point distance) PSNR** [15], based on the distance between a point in the reconstructed PC and its closest neighbor in the reference PC and vice-versa;
- **D2 (point-to-plane distance) PSNR** [15], based on the distance between a point in the reconstructed PC and a plane orthogonal to the normal at its closest neighbor in the reference PC and vice-versa;
- **Plane-to-plane angular similarity (AS)** [16], based on the angle between the normal at a point in the reconstructed PC and the normal at its closest neighbor in the reference PC and vice-versa.

B. Blocks Statistics

Before assessing the RD performance, it is useful to highlight two important characteristics of the interlaced blocks when compared with the corresponding disjoint blocks. First, the interlaced blocks PC division typically produces more non-empty blocks than the simpler disjoint blocks PC division, for the same coding size, as shown in Table I. Since the (sub-sampled) interlaced blocks span over the same larger PC area of the super-block, the probability of obtaining empty interlaced blocks is smaller than for disjoint blocks, as shown in Fig. 2. As a consequence, interlaced blocks tend to contain a smaller number of points, as shown in Table I. Second, the *nib* interlaced blocks resulting from a super-block tend to be more similar than the disjoint blocks, as the interlaced blocks are just slightly different sub-samplings of the same PC area (the super-block), as shown in Fig. 2. As a result, even with a single interlaced block (corresponding to the first scalable layer), it is possible to have a rather good idea of the full super-block geometry, thus achieving a rather good subjective quality at a rather low rate cost. This is a relevant feature of the proposed interlaced blocks representation, with important implications in terms of RD performance.

C. Number of Decoded Points

Since the focus of this paper is on resolution scalability, the number of points in the reconstructed PCs for each of the reconstruction layers is shown in Fig. 5. As expected, due to the interlaced sampling, the proposed RS-DLPCC solution shows a linear increase of the number of decoded points along the layers. As previously mentioned, various deep learning end-to-end coding models were trained with different α weight values, affecting the balance between full and empty voxels in a block as described in Section IV-A, notably more points are reconstructed when α is increased. While α=0.5 presented the best results for most PCs, achieving a number of reconstructed points at the full resolution layer very similar to the original number of points, this was not the case for the *House without Roof* PC. As per Table I, this PC presents a significantly smaller number of points per block, meaning that small α values lead to many reconstructed interlaced blocks with very few to no points at all, thus creating holes in the PC. A larger α value may thus be required to adequately balance full and empty voxels in this case, with 0.8 being the best.

TABLE I. NUMBER OF NON-EMPTY BLOCKS AND AVERAGE NUMBER OF POINTS PER BLOCK IN EACH POINT CLOUD

	Super-blocks: 128^3	**Disjoint: 64^3**		**Interlaced: 64^3**	
		Blocks	***Points***	***Blocks***	***Points***
Statue Klimt	10	46	5307	80	3052
Queen	51	208	4813	408	2453
House w/o Roof	539	2000	1819	4226	844
Longdress	45	190	4516	360	2383

D. RD Performance Assessment

Besides objective quality metrics, visual subjective assessment is vital to evaluate the RD performance, especially for resolution scalable coding, since PCs with significantly lower number of points are reconstructed; in fact, some objective quality metrics may not deal with this situation properly. Fig. 6 compares some examples of original and reconstructed PC regions obtained with the MPEG G-PCC benchmark and RS-DLPCC, at various rates, showing their respective objective quality score. For MPEG G-PCC, results are shown for the two highest quality RD points. For RS-DLPCC, results are shown for the first, third and the last scalable layers. In addition, Fig. 7 shows the RD performance comparison between G-PCC and RS-DLPCC using the three selected objective quality metrics, for the two best end-to-end coding models, notably trained using α=0.5 and α=0.8.

Subjective quality assessment – The examples in Fig.6 show that the MPEG G-PCC triangle-based surface estimation often leads to geometry artifacts, notably for the lower rates. This effect is especially noticeable in the hands (Fig. 6 b) and feet (Fig. 6 e), where the reconstructed surface seems wrinkled, and occasionally partly incomplete in smaller detailed regions like the fingers. In contrast, RS-DLPCC consistently presents reconstructed PCs with smoother surfaces. Even for a reduced number of scalable layers (thus low rates), the complete PC shape can be observed with a good level of detail and quality.

Objective quality assessment – Despite the good subjective results for the proposed RS-DLPCC solution, notably at lower rates, it is important to also assess the RD performance charts, shown in Fig. 7.

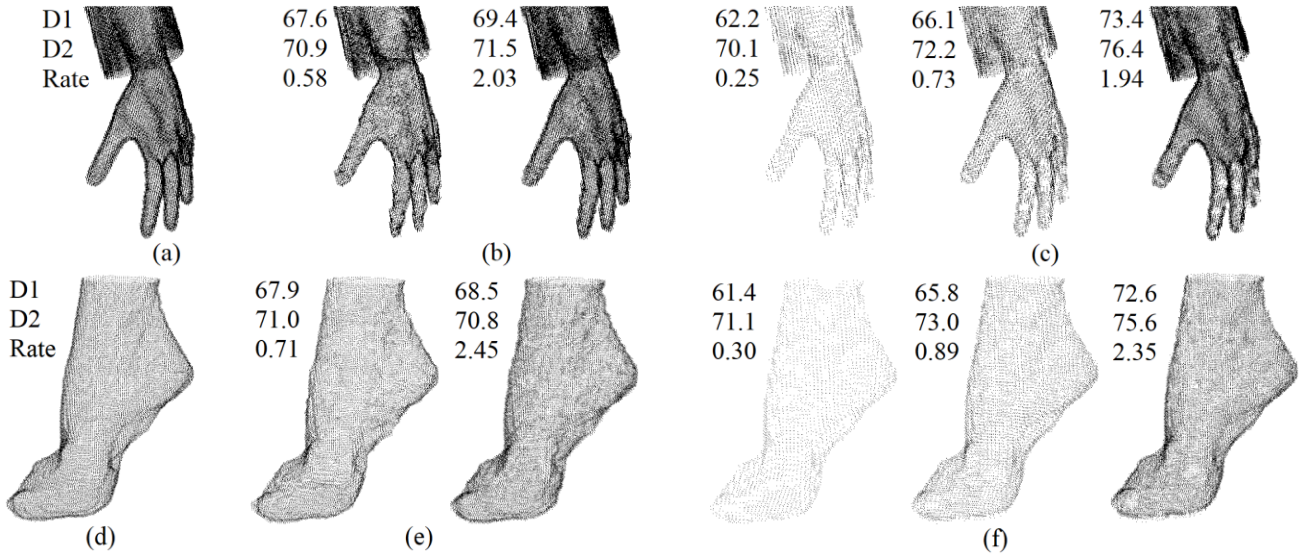

Fig. 6. Detail comparison of test PCs excerpts. *Queen* PC: (a) input, (b) coded with MPEG G-PCC, (c) coded with RS-DLPCC; *Longdress* PC: (d) input, (e) coded with MPEG G-PCC, (f) coded with RS-DLPCC.

- **D1 PSNR metric** – Fig. 7 (left column) shows that RS-DLPCC generally achieves a good D1 PSNR quality at the final layers (higher rate), clearly surpassing G-PCC; however, the D1 PSNR quality drops significantly for the earlier layers. This is due to the significant mismatch between the number of points in the reconstructed and reference PCs for the early layers, which results in a large error distance when directly measuring the distance between neighboring points, even though the subjective quality is rather good. This is corroborated by the fact that α=0.8 achieves better results than α=0.5, for all PCs in the earlier layers, since for α=0.8 more points are reconstructed. Considering the previously discussed subjective results, one may conclude that D1 PSNR does not seem to correlate very well with the subjective quality, especially for resolution scalable PC coding solutions, at lower rates. It is, therefore, essential to assess the RD performance using also other objective quality metrics.

- **D2 PSNR metric** – Fig. 7 middle column shows that the proposed RS-DLPCC solution consistently outperforms MPEG G-PCC for D2 PSNR, including for the earlier layers, contrary to D1 PSNR; the best reconstruction quality happens for α=0.5, again with the exception of *House without Roof*. Another exception is the *Statue Klimt* PC for the lower layers; this is due to the low number of points of this PC, making the first scalable layers very sparse, thus leading to a larger quality drop in the first few layers than for the other PCs. These results are more in line with the previous subjective quality assessment, showing that D2 metric has a better correlation with the subjective quality, notably for the lower resolutions.

- **AS metric** – Fig. 7 right column shows that, for the AS metric, RS-DLPCC achieves very good results already from the first layer, with the quality remaining rather constant throughout the rate, once again outperforming MPEG G-PCC. Although a bit surprising, this conclusion matches the subjective quality assessment from Fig. 6 since the overall PC shape is rather well represented for all layers. AS is more robust to the point count differences as it compares mostly the PC surfaces; thus better correlating to the subjective performance, especially for much sub-sampled PCs.

E. Ablation Study: Block Coding Order Optimization

To demonstrate the advantage of the block coding order optimization module, this section presents an experiment evaluating the RD performance with and without this module. In the absence of this module, the block coding order within each super-block is always the same, notably a raster scan order. Fig. 8 shows that the RD performance significantly improves for the intermediate scalable layers when optimizing the block coding order. This stems from the fact that earlier layers contain a very small number of points, making it relevant for the points to be more uniformly distributed to reduce the error distance as assessed by objective quality metrics such as D2 PNSR. Since point uniformity may vary depending on the specific sampling positions (see Fig. 3) of the various interlaced blocks, the coding order is optimized to facilitate this even distribution. While this positively affects the objective quality metrics, and thus the RD charts, it has almost no visual impact.

V. Conclusions and Future Work

Scalability is an important requirement in PC coding, notably allowing users to quickly preview a lower resolution/rate version of the PC. This paper proposes a novel resolution scalable PC geometry coding solution, which divides the PC into interlaced blocks that are independently coded with a single deep learning model. The proposed solution allows to progressively decode layers that successively add points to the reconstructed PC, thus increasing its point resolution and quality. Experimental results demonstrate that the proposed RS-DLPCC solution outperforms the less scalable MPEG G-PCC standard in terms of RD performance for relevant quality metrics. In addition, it is shown that subjectively good PC quality is achieved even for very low rates. Future work will target developing post-processing methods such as super-resolution to further improve the RD performance of the proposed coding solution.

References

[1] S. Schwarz *et al*., "Emerging MPEG Standards for Point Cloud Compression," *IEEE Trans. Emerg. Sel. Topics Circuits Syst.*, vol. 9, no. 1, pp. 133–148, Mar. 2019.

[2] L. Cui *et al*., "Point-Cloud Compression: Moving Picture Experts Group's New Standard in 2020," *IEEE Consum. Electron. Mag.*, vol. 8, no. 4, pp. 17-21, July 2019.

[3] E. S. Jang et al., "Video-Based Point-Cloud-Compression Standard in MPEG: From Evidence Collection to Committee Draft [Standards in a Nutshell]," *IEEE Signal Process. Mag.*, vol. 36, no. 3, pp. 118-123, May 2019.

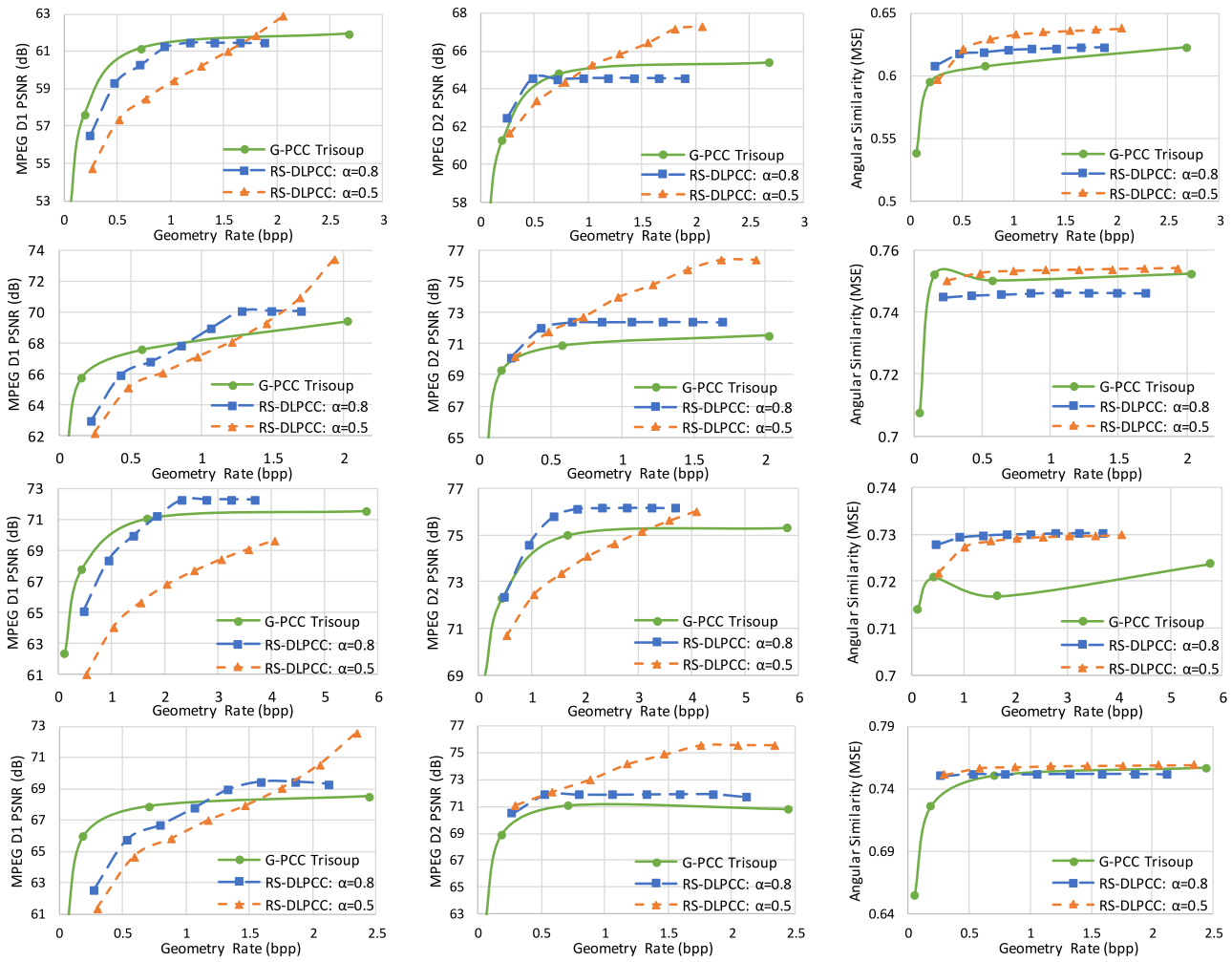

Fig. 7. RD performance comparison between MPEG G-PCC and the proposed RS-DLPCC solution. From top to bottom: *Statue Klimt*, *Queen*, *House without Roof* and *Longdress*. From left to right; D1 PSNR, D2 PSNR and AS.

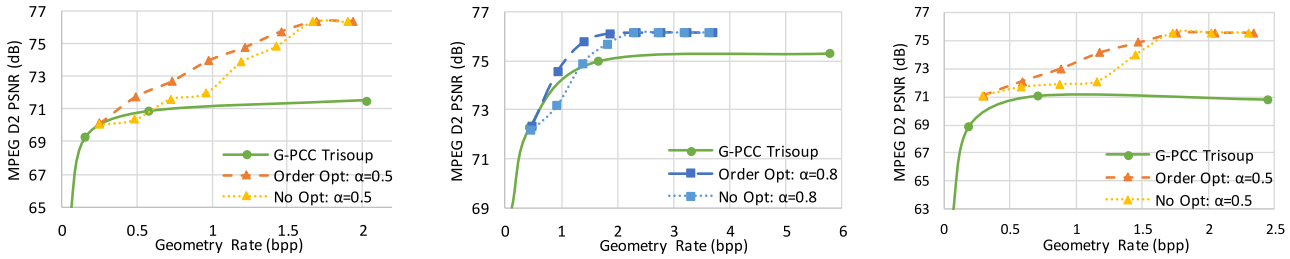

Fig. 8. D2 PSNR RD performance for the proposed RS-DLPCC solution, with and without coding order optimization. From left to right: *Queen*, *House without Roof*, *Longdress*.

[4] ISO/IEC JTC 1/SC29/WG1 N86013, "First Call for Evidence on JPEG Pleno Point Cloud Coding," Sydney, Australia, Jan. 2020.

[5] D. Minnen *et al.*, "Joint Autoregressive and Hierarchical Priors for Learned Image Compression," *Advances in Neural Inf. Process. Syst.*, Montreal, Canada, Dec. 2018.

[6] C. Cao *et al.*, "3D Point Cloud Compression: A Survey," *Int. Conf. 3D Web Technol.*, Los Angeles, CA, USA, July 2019.

[7] S. Cousins *et al.*, "3D is Here: Point Cloud Library (PCL)," *IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011.

[8] G. J. Sullivan *et al.*, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[9] W. Yan *et al.*, "Deep AutoEncoder-based Lossy Geometry Compression for Point Clouds," *arXiv:1905.03691v1 [cs.CV]*, Apr. 2019.

[10] M. Quach *et al.*, "Learning Convolutional Transforms for Lossy Point Cloud Geometry Compression," *IEEE Int. Conf. Image Process.*, Taipei, Taiwan, Sept. 2019.

[11] A. Guarda *et al.*, "Point Cloud Coding: Adopting a Deep Learning-based Approach," *Picture Coding Symp.*, Ningbo, China, Nov. 2019.

[12] A. Guarda *et al.*, "Deep Learning-Based Point Cloud Coding: A Behavior and Performance Study," *Eur. Workshop Vis. Inf. Process.*, Roma, Italy, Oct. 2019.

[13] J. Ballé *et al.*, "Variational Image Compression with a Scale Hyperprior," *Int. Conf. Learn. Representations*, Vancouver, Canada, Apr. 2018.

[14] T. Lin *et al.*, "Focal Loss for Dense Object Detection," *IEEE Int. Conf. Comput. Vision*, Venice, Italy, Oct. 2017.

[15] ISO/IEC JTC1/SC29/WG1 N19084, "Common Test Conditions for Point Cloud Compression," Brussels, Belgium, Jan. 2020.

[16] E. Alexiou *et al.*, "Point Cloud Quality Assessment Metric Based on Angular Similarity," *IEEE Int. Conf. Multimedia and Expo*, San Diego, CA, USA, July 2018.