Double-Deep Learning-Based Point Cloud Geometry Coding with Adaptive Super-Resolution

Manuel Ruivo, André F. R. Guarda, and Fernando Pereira, *Fellow, IEEE*

***Abstract*—Point clouds represent 3D visual data in a very immersive and realistic way, offering to the users a large degree of navigation and interaction. For some key use cases, point clouds are potentially lighter and easier to acquire than other 3D representation models, thus offering an alternative with lower computational cost. To offer visual realistic and immersive experiences, notably the illusion of well-formed surfaces, point clouds typically require a large number of points. To make its storage and transmission feasible, efficient point cloud coding is essential. Recently, deep learning-based point cloud coding solutions have proven to be competitive in compression performance, excelling in distinct scenarios, although struggling to achieve similar results for sparser point clouds and lower coding rates. To tackle these limitations, this paper proposes a double-deep learning-based approach for point cloud coding by integrating a super-resolution tool. The main idea consists on converting sparser point clouds into denser ones via a down-sampling step performed before coding. Since this is a lossy process, a super-resolution step is included after decoding to mitigate the point losses and bringing the point cloud to the initial resolution. Furthermore, the sampling factor can be adaptively selected, thus offering additional flexibility to the point cloud characteristics. The proposed double-deep coding and super-resolution solution outperforms both the G-PCC Octree and V-PCC Intra point cloud coding standards achieving, respectively, 81.9% and 22.3% rate reduction measured as BD-Rate for the PSNR D1 metric.**

***Index Terms*—Deep learning, point cloud coding, point cloud super-resolution**

I. INTRODUCTION

IT is well recognized that visual data-based applications are spreading over all human activity domains, with realism and immersion becoming key requirements for these visual experiences. Point Clouds (PCs) are recognized as one of the most versatile 3D visual representation models. While providing a large degree of realism, immersion, interaction, and navigation freedom to the user, PCs standout from other representation models for being lighter (e.g., compared to meshes) as they only contain information regarding the point positions and not their connectivity, and easier to capture (e.g., compared to light fields). A PC can be defined as an unstructured set of 3D points defining the surface of a 3D object or scene, thus providing information regarding its shape, the so-called *PC geometry*. However, since this data might not be enough to offer realistic and immersive experiences, PCs often include additional information, notably color/texture, and surface normals. These are the so-called *PC attributes*, which sit on top of the PC geometry. PCs may show rather different characteristics, namely regarding their point density. A PC can be characterized as dense if, for a fixed precision, the average distance between points is small, or as sparse, if otherwise. Independently of the PC characteristics, the large number of points required to create the illusion of well-formed and dense surfaces critically asks for compression efficient coding solutions, notably to make transmission and storage of PCs feasible in practice.

Point Cloud Coding (PCC) has been an area of intense research in recent years. Acknowledging the need for interoperability, the MPEG standardization group has recently issued two PCC standards [1]: the Geometry-based PCC (G-PCC) and the Video-based PCC (V-PCC) standards. More recently, several Deep Learning (DL) based PC coding solutions have been proposed, achieving competitive performance. This led the JPEG standardization group to launch a Call for Proposals on JPEG Pleno Point Cloud Coding with the goal to develop a learning-based PC coding standard [2]. Despite the promising results, DL-based PCC solutions for geometry fail to achieve the same level of compression performance for all types of PCs, commonly offering poorer performance for the sparser ones. To overcome this problem, PC grid/precision down-sampling prior to coding became an interesting approach targeting to offer to the codec denser PCs. However, this naturally requires performing the corresponding grid up-sampling step after decoding, to recover the original PC precision.

In the literature, PC sampling comes in many flavors. In this paper, PC sampling is broadly defined as the set of operations able to change the PC resolution/precision or/and point density. In this context, two types of sampling operations deserve to be more precisely defined: *grid sampling* refers to the operations where the PC's resolution/precision is modified by changing the voxel size; and *set sampling* refers to the operations over the number of points, i.e. point set cardinality. Despite being distinct, these operations are often performed together, notably when up-sampling both the precision and the number of points, the so-called *super-resolution*. Naturally, these sampling operations may be more or less sophisticated, allowing to reach better quality at a complexity cost.

In this context, this paper proposes a double DL-based PC geometry coding solution with adaptive super-resolution (2DL-PCC-ASR), capable of exploiting the PC characteristics, to achieve competitive rate-distortion (RD) performance, notably for PCs with different densities. The proposed super-resolution post-processing tool allows to

This work has been financially supported by the Fundação para a Ciência (FCT, Portugal) through the research project PTDC/EEI-COM/1125/2021, entitled "Deep Learning-based Point Cloud Representation".

M. Ruivo, and F. Pereira are with Instituto Superior Técnico, Universidade de Lisboa and Instituto de Telecomunicações, Lisbon, Portugal (e-mail: manuel.ruivo@tecnico.ulisboa.pt, fp@lx.it.pt).

A. F. R. Guarda is with Instituto de Telecomunicações, Lisbon, Portugal (e-mail: andre.guarda@lx.it.pt).

increase the reconstructed PC quality at virtually no additional rate cost and may be used with any type of geometry coding solution. The 'double-deep' attribute in 2DL-PCC-ASR refers to the approach considered for both the PC geometry codec itself and the advanced super-resolution tool. This is the first solution of this type in the literature and outperforms both the G-PCC Octree and V-PCC Intra PCC standards for a large variety of PCs coded at different resolutions and rates.

This paper is organized as follows: Section II reviews some related work, namely in PC coding and PC super-resolution. Section III provides an overview of the proposed architecture and DL-based coding model, whereas Section IV focuses on the review of the DL-based super-resolution approach and respective training process. Section V presents the testing conditions and discusses obtained results. Finally, Section VI presents some final remarks.

II. Related Work

This section briefly reviews the most relevant background work for the proposed solution, notably regarding both key PC geometry coding and PC super-resolution solutions.

A. Point Cloud Coding

As mentioned before, two MPEG PCC standards have been recently launched [1], G-PCC and V-PCC. G-PCC leverages the use of a multi-level-of-detail tree to structure the PC geometry, creating a so-called *octree*. The octree can be fully coded to provide lossless compression or pruned for lossy compression. As for V-PCC, it targets dynamic PCs and relies on available video codecs, e.g., HEVC [3], to code selected 2D projections of the 3D geometry and color. This approach greatly benefits from the very high compression efficiency achieved with conventional video codecs, following decades of research and improvements. Despite being a PCC standard oriented towards coding dynamic PCs, the so-called *V-PCC Intra* mode allows to code static PCs.

Given the considerable success of DL-based solutions in areas such as computer vision [4] and image coding [5], this technology was also brought to PCC. Among the most relevant DL-based PC geometry coding solutions in the literature, a few deserve to be highlighted. In 2020, Guarda *et al.* proposed the so-called Adaptive Deep Learning-based PCC (ADL-PCC) solution [6]. This is a block-based PC geometry codec that relies on a double Auto-Encoder (AE) architecture. The first AE is used to learn a compact latent representation of the PC geometry, thus performing compression through dimensionality reduction, whereas the second AE - a Variational AE (VAE) - is responsible for exploiting the statistical redundancy in the latents for optimal rate reduction with arithmetic coding [7]. The ADL-PCC's adaptive behavior is achieved through the parallel assessment of multiple DL models for each PC's block at encoding time, where each DL model was trained to fit different block densities. The model reaching the smaller RD cost is selected for coding and signaled to the decoder in the bitstream. This allows ADL-PCC to dynamically adapt to blocks/PCs with distinct characteristics while keeping competitive RD performance. However, despite showing good compression results for rather dense PCs, the performance drops when handling sparser PCs, and low rates are often difficult or impossible to reach.

Another relevant PC geometry solution in the literature is the so-called Multiscale PC Geometry Coding (M-PCGC) proposed by Wang *et al.* in 2021 [8]. This solution relies on two modules – the sampling module and the Inception Residual Network [9]; while the first is responsible for changing the PC scale to offer multiscale capabilities, the second is responsible for extracting meaningful features. Unlike ADL-PCC, M-PCGC relies on sparse tensors and convolutions to represent and process the PC features, respectively, allowing to greatly reduce the memory and computation complexity. After three steps of down-sampling and feature extraction, an octree encoder is used for final coding using an arithmetic encoder. At the decoder side, the architecture is mirrored with appropriate up-sampling instead of down-sampling modules. Competitive RD performance is obtained for a varied range of bitrates, thus showing the benefits of using appropriate sampling tools in a coding context to adapt to varying PC characteristics.

B. Point Cloud Super-Resolution

Most PC up-sampling solutions in the literature can be organized in two main categories: optimization-based and learning-based. In this paper, the focus is naturally on the learning-based solutions as they tend to perform better under more demanding circumstances [10]. One of the most relevant learning-based super-resolution solutions has been proposed by Akhtar *et al.* in 2020 [11]. This solution is based on a two-step process where the PC resolution is first increased by simply multiplying the points' coordinates by the up-sampling factor, thus performing a basic grid up-sampling, followed by a PC geometry densification process using an advanced, learning-based set up-sampling solution. For this latter stage, the proposed architecture adopts a 3D variation of a 2D convolutional model, called *U-net* [12], used to extract and expand PC features to predict the optimal location of the new, super-resolved points in the original PC resolution/precision. The results obtained show a considerable quality improvement over the basic grid up-sampling solution. In 2022, this solution has been extended into a more powerful super-resolution solution capable of dealing with much larger and sparser PCs, notably LiDAR PCs [13].

A promising meta-learning-based advanced set up-sampling solution, named Meta-PU, has been proposed by Ye *et al.* in 2021 [10]. Meta-learning is the ability of learning to learn which, in practice, translates into the use of two networks in parallel: the main network, a residual graph convolutional network is used for feature extraction and processing, while a second fully connected network, the so-called *meta-network*, is responsible for adjusting the main network's behavior in real time, by refining some of its weights. This approach offers adaptability to varying PC densities, allowing to use a single trained DL model for any

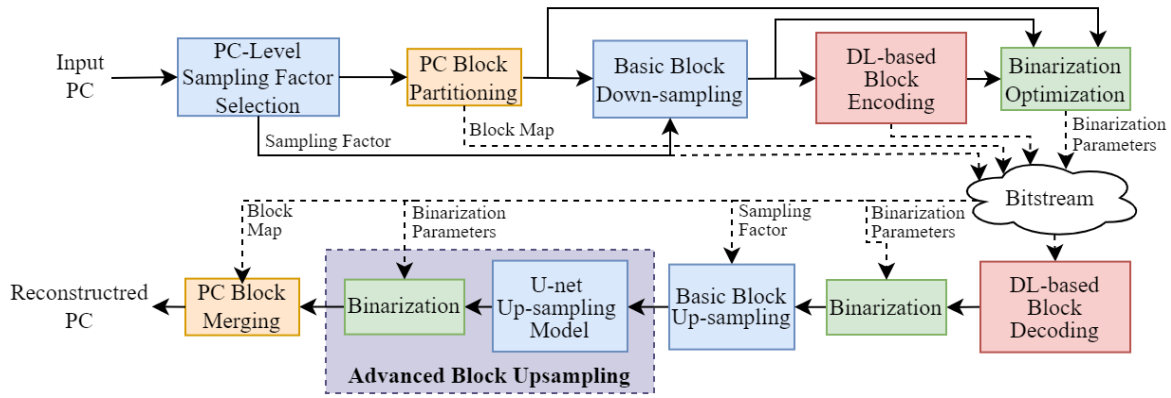

Fig. 1. 2DL-PCC-ASR architecture including the sampling related (blue), partitioning/merging (yellow), coding (red), binarization (green) and advanced up-sampling (purple) modules.

sampling factor while outperforming the state-of-the-art set up-sampling solutions, namely PU-GAN [14].

III. OVERALL 2DL-PCC-ASR ARCHITECTURE AND DL-BASED CODING MODEL

This section will offer an overview of the proposed 2DL-PCC-ASR solution by presenting the overall PC geometry coding and super-resolution pipeline and walkthrough; finally, the DL coding model will be also addressed.

A. Architecture and Walkthrough

The overall 2DL-PCC-ASR architecture is presented in Fig. 1. The pipeline's walkthrough proceeds as follows:

- **PC-Level Sampling Factor Selection –** This module is responsible for estimating the optimal down-sampling factor for PCC, e.g. using some point distance-based algorithm.
- **PC Block Partitioning –** This module breaks down the PC into binary 3D blocks of fixed size where '1' and '0' correspond to full and empty voxels, respectively. This is a vital step as all the following operations are performed at the block level.
- **Basic Block Down-sampling –** This module reduces the input block resolution/precision, thus increasing the voxel size by the previously selected sampling factor. This is a basic operation as, for each point/voxel, a simple coordinate scaling is performed, followed by a rounding operation. This is an irreversible and lossy process as some points may be lost, especially when dealing with dense blocks, as multiple points in the original set may collapse into a single point/voxel in the down-sampled set.
- **DL-based Block Encoding** – The encoder is responsible for representing the binary input block in the most compact way for the required trade-off between quality and rate. More details will be provided in the next sub-section.
- **Binarization Optimization** – This module is responsible for selecting the optimal number of filled voxels to be reconstructed at the decoder, which is determined by the product between the number of points in the input block and an optimization factor. This factor is selected from within a given range, by optimizing the reconstruction quality using a selected distortion metric. In practice, only the number of points to be reconstructed for each block needs to be transmitted to the decoder. Since the proposed architecture includes two binarization modules, this optimization process can either generate a single factor for both binarization modules or two distinct factors, one for each binarization step. In this paper, the later approach was adopted.
- **DL-based Block Decoding** – The decoder is responsible for recovering from the coding bitstream a 3D block with the probabilities of each voxel being filled. More details on this module will be provided in the next sub-section.
- **Binarization** – This module is responsible for converting the voxels occupancy probabilities into binary values. In this case, the so-called *optimized Top-K* approach is considered where the K voxels with largest decoding probabilities are filled; in this context, K is the number of points determined by the previously mentioned binary optimization module. This approach establishes a strong correlation and fine control between the number of points in the original and reconstructed blocks.
- **Basic Block Up-sampling –** This module restores the original block resolution/precision by simply multiplying the voxels' coordinates by the selected sampling factor, thus reducing the voxel size. This basic operation inevitably produces a sparser block than the original since it does not recover the points that were lost during the voxel merging process in the down-sampling module.
- **Advanced Block Up-sampling (ABU) –** ABU is an optional post-processing module performing advanced set up-sampling, thus increasing the number of points in the block to eventually recover the down-sampling losses. This module targets at increasing the PC quality at no rate cost.
- **PC Block Merging –** Reverts the PC block partitioning operation, thus merging all the decoded and super-resolved blocks into a single reconstructed PC.

B. Deep Learning-based Coding Model

The DL-based coding model is central in the proposed PC geometry coding solution. However, it is important to

highlight that this overall is compatible with any coding approach, DL-based or not; for example, one of the MPEG PCC codecs may also be used. In this paper, a variation of the ADL-PCC codec [6] will be considered for performance assessment purposes.

As ADL-PCC, the adopted DL-based coding model adopts an AE for feature extraction and processing, and a VAE to characterize the latents statistical properties to be used for efficient entropy coding. However, the DL-based coding model adopted in this paper follows a more complex AE architecture with the convolutional layers for feature extraction being complemented with a modified version of the well-known Inception Residual Network [9]; moreover, the number of filters increases as the network deepens. Due to these additions, the number of trainable parameters grows to 6.6 million, a 2-times increase over the baseline version. Additionally, the ADL-PCC approach for PC adaptability based on the usage of multiple trained DL coding models in parallel has been dropped in favor of the previously described optimized binarization solution. It is worth noting that the DL-based coding models were trained in the same conditions as ADL-PCC in [6], namely using the same RD loss function.

IV. ADVANCED BLOCK UP-SAMPLING SOLUTION

This section provides an in-depth description of the Advanced Block Up-sampling (ABU) module. This module consists of two complementary processes: an advanced set up-sampling performed by the DL-based up-sampling model followed by a final binarization process in which the super-resolved voxels' filling probabilities are converted into binary occupancy values, as described in Section III.A.

A. DL-based Up-Sampling Architecture

The DL-based up-sampling module, i.e. the ABU model, is based on the up-sampling solution proposed by Akhtar *et al.* in 2021 [11], a variation of the 3D Convolutional U-net [15]. The main differences between the proposed DL-based ABU model and the up-sampling model in [11] can be summarized as follows: firstly, the adopted U-net model considers dense 3D convolutions instead of the sparse ones used in [11], since the sparse ones are not available in TensorFlow; secondly, the number of channels in the input layer has been decreased to half, and thus C=16, since this allowed achieving the same performance at a lower complexity cost. The proposed architecture, presented in Fig. 2, is invariant to the sampling factor used since basic grid up-sampling is performed before. In total, each ABU model includes 7 288 893 weights.

The ABU model gets as input a basic-up-sampled binary block of fixed size, e.g. $64 \times 64 \times 64 \times 1$, where the first three dimensions correspond to the three spatial block dimensions and the fourth to the geometry channel denoting the occupancy value assigned to each voxel. The ABU model includes the following layers:

- **Input Convolutional Layer (orange)** – Responsible for converting the input binary block into the latent representation space with 16 channels, C = 16.

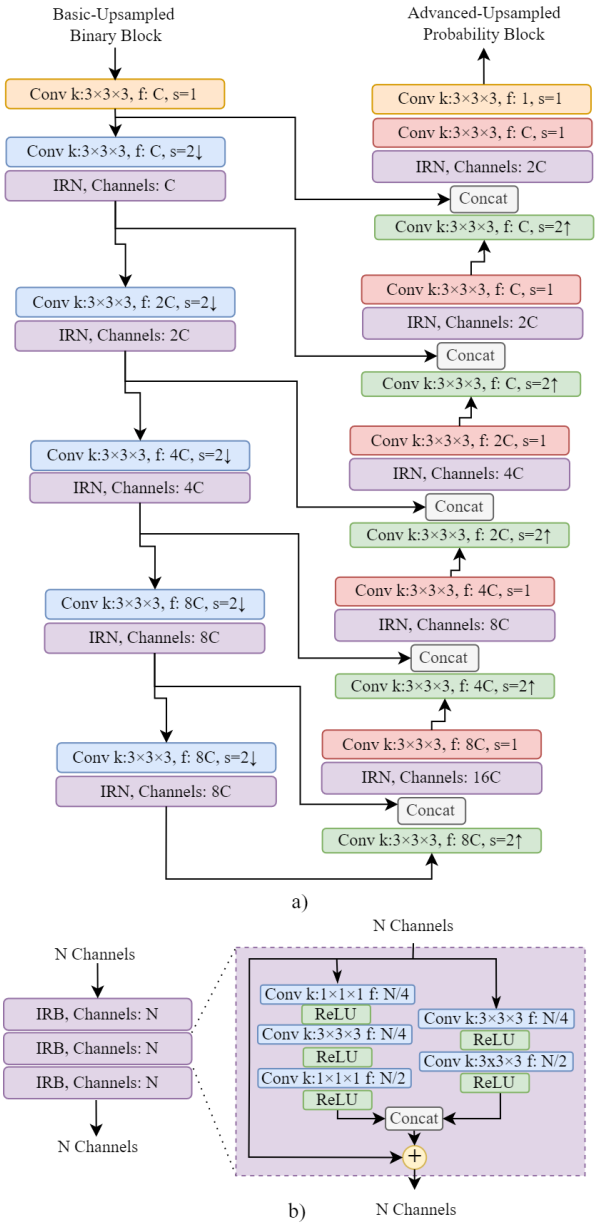

Fig. 2. (a) DL-based ABU model architecture; (b) IRN layer architecture with three Inception Residual Blocks (IRBs) based on [11]. For every convolution layer (conv), *k* represents the kernel size, *f* the number of filters and *s* the stride. Note that each convolutional layer is followed by a ReLU activation function, except the last conv layer in (a) which uses a sigmoid activation.

- **Down-Sampling Convolutional Layer (blue)** – Since a stride of 2 is considered, this layer is responsible for reducing the feature map size to half while doubling the number of filters; this layer is applied five times on the contracting path, the left branch of the U-net.
- **Inception Residual Network (IRN, purple)** – This module is responsible for feature extraction; see its architecture in Fig. 2 a). The Inception Residual Blocks (IRB) offer a great complexity-performance trade-off since although deep they are lightweight, allowing to extract meaningful features based on several local neighboring contexts with reduced impact on training

time. Despite using this module several times, the complexity growth is limited as the IRBs rely on small size kernels, as shown Fig. 2 b). Furthermore, the use of these small filters allows to highlight the most relevant features on a local scope, hence capable of dealing with fine details as desired. Moreover, the use of a skip connection allows to pass forward the features extracted in previous layers to preserve the global context. This module is used between sampling convolutional layers and applied five times in each path.

- **Up-Sampling Deconvolutional Layer (green)** – Symmetrically to the down-sampling layers, these layers are responsible for feature up-sampling with a stride of two; this layer is applied 5 times on the expanding path, the right branch of the U-net.
- **Merging Convolutional Layer (red)** – After processing the up-sampled features, a merging process occurs to combine the information obtained with the up-sampled features and the skip-connection; this layer is applied five times in the expanding path.
- **Output Convolutional Layer (orange)** – This layer uses expanded features to predict the probability of each voxel being occupied; thus, the output block includes filling probabilities and not binary values as the input block.

Since the output is a probabilities block, the same binarization process as described in Section III.A is here required to predict the final voxel occupancy. It is important to recall that this is an *optimized Top-K* approach, and the optimization factor used may be the same as in the codec itself for a faster coding process or distinct for optimal performance.

B. Training Data, Loss Function and Hyperparameters

The training process is a fundamental procedure for every DL model, hence it should be carefully examined.

Sampling Factors: For simplicity, the sampling factors were narrowed down to powers of two. However, due to the complexity of the ABU models, and consequent memory restrains verified during the training process, only two models were trained, notably for sampling factors 2 and 4.

Training and Validation Material: 28 PCs were selected from the JPEG Common Training and Test Conditions (CTTC) PC dataset [16] to form the training data and four as validation. Each of these PCs was divided into blocks of fixed size, 64×64×64 and 128×128×128 when training for sampling factor of 2 and 4, respectively. To simulate the super-resolution process, the resolution of each block is reduced using a basic grid down-sampling operation followed by the symmetric basic grid up-sampling. Given that the down-sampling process is lossy, the basic up-sampled blocks are not the same as the original blocks. This is desirable as the ABU model's task is to learn how to mitigate the down-sampling losses. The basic down- and up-sampled blocks are used as input training data, and the original blocks are set as reference for the training loss. Given the coding context in which ABU is here considered, it is only natural to have the coding process involved in the training dataset. However, the best final RD performance was obtained without coding during training, as described above.

Loss Function: A distortion-only loss function is required since there is no coding involved. Therefore, the so-called *Focal Loss* (FL) [6] was used. Regarding the parameters γ, expressing the relevance of voxels hard to classify, and α expressing the imbalance between empty and occupied voxels, through experimental testing, $\gamma = 2$ and $\alpha = 0.7$ were considered appropriate.

Hyperparameters: The ABU models have been trained using ADAM optimizer [17] with learning rate of 10^{-4}, and a batch size of 8 and 1 for sampling factors of 2 and 4, respectively. Early stopping was used to prevent overfitting, assuring that the model generalizes well and is not biased towards the training data. A patience of 5 epochs was defined, meaning that the training stopped when the validation loss did not decrease for 5 consecutive epochs. The models were trained for a total of 16 and 44 epochs for sampling factors 2 and 4, respectively.

V. PERFORMANCE ASSESSMENT

This section reports the 2DL-PCC-ASR RD performance, namely when considering the proposed ABU super-resolution model. All results presented have been obtained under the conditions defined in the JPEG CTTC [16].

A. Test Dataset, Benchmarks and Performance Metrics

To obtain meaningful results, the test conditions must be carefully defined.

Test Dataset: In DL-based coding, it is vital that the test data is not used for training. Moreover to fully assess the 2DL-PCC-ASR performance, the test dataset must include PCs with distinct characteristics, namely in density. Having this in mind, four PCs were selected, notably: *Longdress, Romanoillamp, RWT130* and *Housewithoutroof*; the first three test PCs are rather dense, the last one is much sparser.

Benchmarks: As in JPEG CTTC, the benchmarks are the MPEG PCC standards – G-PCC Lossless, G-PCC Octree and V-PCC Intra, all using their respective reference software version 14. Moreover since 2DL-PCC-ASR is an evolution of ADL-PCC [6], its RD performance is also considered.

Performance Metrics: Given the 2DL-PCC-ASR compression goals, the RD performance is assessed using the point-to-point PSNR D1 and point-to-plane PSNR D2 geometry quality metrics, as recommended in the JPEG CTTC [16], bits per input point (bpp) to measure the rate.

B. 2DL-PCC-ASR Configurations RD Performance

This sub-section reports the 2DL-PCC-ASR RD performance, notably with and without ABU to clearly show its impact on the final RD performance. Fig. 3 shows RD performance results for multiple 2DL-PCC-ASR configurations as well as the convex hull curve corresponding to the best set of configurations along the rate. The results show that the lower coding rates can only be achieved when performing down-sampling, thus showing how critical this step is to cover a large range of rates. However, given the lossy nature of the basic down-sampling

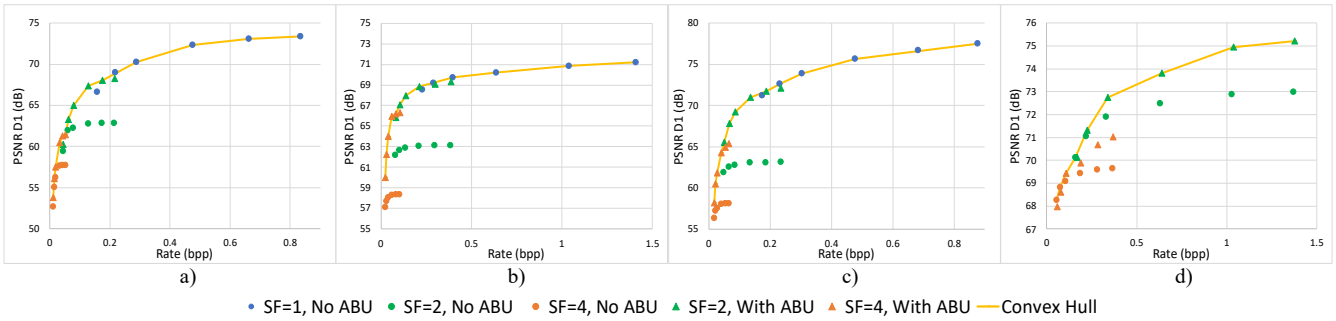

Fig. 3. PSNR D1 RD performance for 2DL-PCC-ASR in multiple configurations, namely for distinct sampling factors (SF) and with or without ABU, for all four test PCs: a) *Longdress*, b) *Romanoilamp*, c) *RWT130*, d) *Housewithoutroof*.

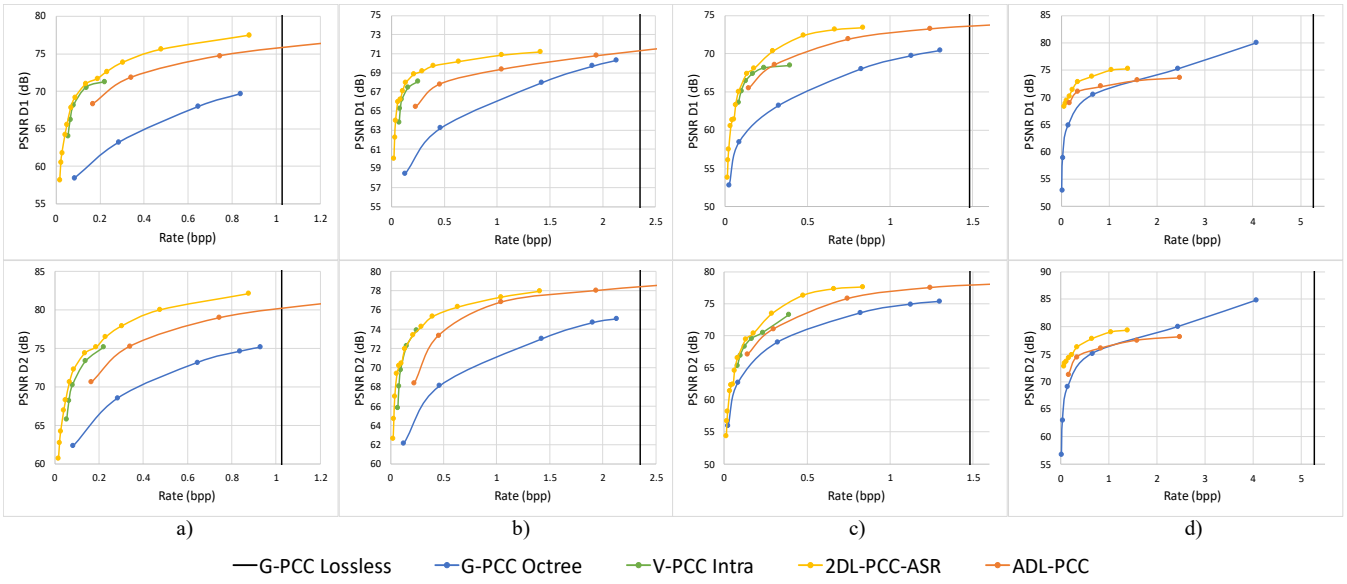

Fig. 4. PSNR D1 and PSNR D2 RD performance comparison between 2DL-PCC-ASR and MPEG standards – G-PCC Octree, V-PCC Intra – and ADL-PCC, for all four test PCs: a) *Longdress*, b) *Romanoilamp*, c) *RWT130*, d) *Housewithoutroof*.

operation, the super-resolution proves to be indispensable to mitigate the quality drops and achieve competitive RD performance. The gains obtained with ABU decrease with the rate, notably due to the appearance of strong coding artifacts for the lower rate points.

When considering denser PCs, Fig. 3 a) to c), ABU gains go from 3 to 10 dB over the basic up-sampling configuration, with virtually no impact on rate (only a couple of binarization parameters). In opposition, for the sparser PC, Fig. 3 d) shows that ABU has a similar behavior although with smaller gains, improving up to 2 dB over the basic up-sampling configuration. For this PC, no results are included for sampling factor 1 since competitive results are only obtained when considering down-sampling, i.e. for sampling factors larger than 1.

Finally, the convex hull curve shows the set of best 2DL-PCC-ASR performing configurations, thus selected for comparison with the benchmarks in the next sub-section.

C. Benchmarking RD Performance

This sub-section aims at comparing the 2DL-PCC-ASR and MPEG PCC standards RD performance. In the charts, the black vertical line indicates the rate at which G-PCC achieves lossless quality; so only the RD points to the left of that line are relevant for lossy coding. The results in Fig. 4 demonstrate that the adopted improvements allow the proposed 2DL-PCC-ASR to achieve much better RD performance than the previous ADL-PCC solution (with 53.2% rate reduction measured as BD-Rate for PSNR D1). The proposed solution largely outperforms G-PCC Octree (with 81.9% rate reduction measured as BD-Rate for PSNR D1), which typically performs better for sparse PCs, and outperforms/equals V-PCC Intra which is based on very powerful video coding standards (achieving a 22.3% rate reduction measured as BD-Rate for PSNR D1). This occurs for PCs with rather different characteristics, namely density.

The fact that 2DL-PCC-ASR outperforms or is equivalent to V-PCC Intra in RD performance is a great achievement for 2DL-PCC-ASR since V-PCC Intra represents the best of conventional PC geometry coding as it relies on powerful video coding standards, improved along the last three decades. It is relevant to mention that it was not possible to code *Housewithoutroof* with the V-PCC Intra reference software since this coding solution is not appropriate for sparse PCs.

Overall, the RD performance show that the proposed

2DL-PCC-ASR solution achieves top-notch RD performance independently of the PC characteristics.

VI. FINAL REMARKS AND FUTURE WORK

This paper proposes a double DL-based approach for PC geometry coding and super-resolution in a single pipeline, capable of outperforming the state-of-the-art MPEG PCC standards.

The gains obtained with the ABU super-resolution model are very significant and can be as high as 10 dB when coding dense PCs with high rates. Future work will target the extension of this coding pipeline to jointly code geometry and color. Additionally, a single ABU model will be developed to accommodate all sampling factors.

REFERENCES

[1] D. Graziosi *et al.*, "An Overview of Ongoing Point Cloud Compression Standardization Activities: Video-based (V-PCC) and Geometry-based (G-PCC)," *APSIPA Trans. Signal Inf. Process.*, vol. 9, pp. 1–17, Mar. 2020.

[2] "Final Call for Proposals on JPEG Pleno Point Cloud Coding." ISO/IEC JTC 1/SC29/WG1 N100097, Online Meeting, Jan. 2022.

[3] G. J. Sullivan *et al.*, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Trans. Circuits Sys. Video Technol.*, vol. 22, pp. 1649–1668, Dec. 2012.

[4] A. Voulodimos *et al.*, "Deep Learning for Computer Vision: A Brief Review," *Comput. Intell. and Neuroscience*, vol. 2018, Feb. 2018.

[5] M. I. Patel *et al.*, "Survey on Image Compression using Machine Learning and Deep Learning," *Int. Conf. on Intell. Computing and Control Sys. (ICCS)*, Faro, Portugal, May 2019.

[6] A. F. R. Guarda *et al.*, "Adaptive Deep Learning-Based Point Cloud Geometry Coding," *IEEE J. Sel. Topics Signal Process.*, vol. 15, no. 2, pp. 415–430, Feb. 2021.

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

[8] J. Wang *et al.*, "Multiscale Point Cloud Geometry Compression," *Data Compression Conf. (DCC)*, Cliff Lodge, UT, USA, Mar. 2021.

[9] C. Szegedy *et al.*, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," *AAAI Conf. on AI*, San Francisco, CA, USA, Feb. 2017.

[10] S. Ye *et al.*, "Meta-PU: An Arbitrary-Scale Upsampling Network for Point Cloud," *IEEE Trans. on Vis. and Comput. Graphics (TVCG)*, accepted, Feb. 2021.

[11] A. Akhtar *et al.*, "Point Cloud Geometry Prediction Across Spatial Scale using Deep Learning," *IEEE Int. Conf. on Vis. Commun. and Image Process (VCIP)*, Hong Kong, China, Dec. 2020.

[12] O. Ronneberger *et al.*, "U-Net: Convolutional Networks for Biomedical Image Segmentation," *Int. Conf. on Med. Image Comput. and Computer-Assisted Intervention (MICCAI)*, Munich, Germany, Oct. 2015.

[13] A. Akhtar, "PU-Dense: Sparse Tensor-based Point Cloud Geometry Upsampling," *IEEE Trans. on Image Process.*, accepted, vol. 31, Apr. 2022.

[14] R. Li *et al.*, "PU-GAN: a Point Cloud Upsampling Adversarial Network," *Int. Conf. on Comput. Vision (ICCV)*, Seoul, South Korea, Jul. 2019.

[15] Ö. Çiçek *et al.*, "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation," *Int. Conf. on Med. Image Comput. and Computer-Assisted Intervention (MICCAI)*, Athens, Greece, Jun. 2016.

[16] "Common Training and Test Conditions for Point Cloud Compression." ISO/IEC JTC 1/SC29/WG1 N100112, Online Meeting, Jan. 2022.

[17] D. P. Kingma *et al.*, "Adam: a Method for Stochastic Optimization," *Int. Conf. Learning Representations*, San Diego, CA, USA, May 2015.