IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 5 May 2025, accepted 12 June 2025, date of publication 17 June 2025, date of current version 27 June 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3580680

RESEARCH ARTICLE

Point Cloud Geometry Scalable Coding Using a Resolution and Quality-Conditioned Latents Probability Estimator

DANIELE MARI[1], ANDRÉ F. R. GUARDA[2], (Member, IEEE), NUNO M. M. RODRIGUES[2,3], (Senior Member, IEEE), SIMONE MILANI[1], (Member, IEEE), AND FERNANDO PEREIRA[2,4], (Fellow, IEEE)

[1]Department of Information Engineering, University of Padova, 35131 Padua, Italy
[2]Instituto de Telecomunicações, 1049-001 Lisbon, Portugal
[3]ESTG, Politécnico de Leiria, 2411-901 Leiria, Portugal
[4]Instituto Superior Técnico, Universidade de Lisboa, 1049-001 Lisbon, Portugal

Corresponding author: Daniele Mari (daniele.mari@dei.unipd.it)

This work was funded in part by the European Union (EU) through the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, with a partnership on "Telecommunications of the Future" Program "RESearch and innovation on future Telecommunications systems and networks (Restart)" under Grant PE00000001; in part by the Fundação para a Ciência e a Tecnologia (FCT), Portugal, entitled "Deep Learning-Based Point Cloud Representation," under Project PTDC/EEI-COM/1125/2021; and in part by FCT/Ministério da Educação, Ciência e Inovação (MECI) through National Funds and when applicable co-funded EU Funds: Instituto de Telecomunicações under Grant UID/50008. The work of Daniele Mari was supported by Fondazione CaRiPaRo under Grant "Dottorati di Ricerca" 2021/2022.

ABSTRACT In the current age, users consume multimedia content in very heterogeneous scenarios in terms of network, hardware, and display capabilities. A naive solution to this problem is to encode multiple independent streams, each covering a different possible requirement for the clients, with an obvious negative impact in both storage and computational requirements. These drawbacks can be avoided by using codecs that enable scalability, i.e., the ability to generate a progressive bitstream, containing a base layer followed by multiple enhancement layers, that allow decoding the same bitstream serving multiple reconstructions and visualization specifications. While scalable coding is a well-known and addressed feature in conventional image and video codecs, this paper focuses on a new and very different problem, notably the development of scalable coding solutions for deep learning-based Point Cloud (PC) coding. The peculiarities of this 3D representation make it hard to implement flexible solutions that do not compromise the other functionalities of the codec. This paper proposes a joint quality and resolution scalability scheme, named Scalable Resolution and Quality Hyperprior (SRQH), that, contrary to previous solutions, can model the relationship between latents obtained with models trained for different RD tradeoffs and/or at different resolutions. Experimental results obtained by integrating SRQH in the emerging JPEG Pleno learning-based PC coding standard show that SRQH allows decoding the PC at different qualities and resolutions with a single bitstream while incurring only in a limited RD penalty and increment in complexity w.r.t. non-scalable JPEG PCC that would require one bitstream per coding configuration.

INDEX TERMS Point cloud geometry coding, JPEG Pleno PCC, deep learning-based codec, scalable coding.

I. INTRODUCTION

PCs have emerged as a fundamental representation for spatial data, comprising collections of points sampled from object surfaces in 3D space. Each point is characterized by its spatial coordinates and may include additional attributes such as color components and surface normals. The prominence of 3D representations, particularly PCs, has grown significantly due to three key factors: their ability to create immersive environments, their support for six-degrees-of-freedom

The associate editor coordinating the review of this manuscript and approving it for publication was Cesar Vargas-Rosales.

navigation, and their capacity for accurate environmental representation. These characteristics have established PCs as the de facto standard in various applications, including virtual and augmented reality, autonomous navigation, and cultural heritage preservation [1]. However, achieving high-fidelity scene representation often requires PCs with millions of points, resulting in substantial storage and bandwidth demands.

The growing importance of PCs, coupled with their considerable raw data size, has made the development of efficient Point Cloud Coding (PCC) algorithms crucial for practical applications. PCC presents unique challenges due to two inherent characteristics of PCs: their unstructured nature and the non-uniform point density resulting from acquisition processes. To address these challenges, standardization efforts have emerged: MPEG has developed two distinct standards [2] i.e. Geometry-based Point Cloud Compression (G-PCC), originally targeting static or dynamically acquired PCs (e.g. LiDAR PCs), and Video codec-based Point Cloud Compression (V-PCC), originally targeting dynamic PCs. On the other hand, JPEG has finalized the development of JPEG Pleno Learning-based Point Cloud Coding Standard (JPEG PCC) [3], the first learning-based standard for static PCC.

JPEG's effort in particular, being JPEG PCC learning-based, aims to create a standard that delivers competitive Rate-Distortion (RD) performance while providing effective compressed domain representations for both human visualization and machine processing [4].

While RD performance is crucial, real-world applications demand additional features, including stream scalability. Scalability enables serving content at various qualities, resolutions, or framerates through a single bitstream, rather than maintaining multiple independent streams. This capability becomes particularly valuable given the heterogeneity of receiving devices, which often operate under different network conditions and hardware constraints. Implementing this feature can thus allow a codec to properly adapt to a wide variety of receiving conditions. A scalable bitstream is generally organized into a base layer providing minimal rate decoding capabilities, and multiple enhancement layers allowing progressive refinement of the visual fidelity.

The growing importance of scalability in point cloud coding has prompted standardization organizations to incorporate this feature into their codec requirements [5] and to propose extensions to existing solutions [6]. In particular, according to these requirements, some of the most relevant forms of scalability for static PCs are: *Quality Scalability* where enhancement layers improve visual fidelity without affecting resolution, and *Resolution Scalability* where enhancement layers increase content resolution, with quality improvement being a secondary effect.

Recent advancements in this direction can be found in [7] where the authors presented a method for geometry quality scalability in JPEG PCC by working in the latent domain. Even if only quality scalability is considered, the approach demonstrates how visual fidelity can be progressively enhanced with minimal additional information by correctly exploiting the previously encoded base and enhancement layers.

Some PC codecs (e.g. JPEG PCC) reduce the resolution of the input PC as a form of point domain quantization, controlled by a scaling factor sf. Downscaling is thus an effective RD tuning strategy for PCs since it helps obtaining lower bitrates through the reduction of the information in the content. Additionally, it can help increasing the spatial redundancy in sparser PCs (making them more efficient to compress). Therefore, reducing the resolution of the input is not only useful for addressing the needs of devices that don't support higher resolutions, but it is also an effective way to improve RD performance and increase the number of covered rate points. While JPEG PCC includes a learned super-resolution module to reverse this process, it does not provide true resolution scalability as it operates without enhancement layers, potentially resulting in lower visual fidelity compared to non-super-resolved reconstructions.

Very few learning-based codecs offer resolution and quality scalability. Notable exceptions include SparsePCGC [8] and Unicorn [9], [10]. However, their scalability functionality is inherently provided by their use of a multiscale architecture which cannot be easily integrated into Variational Auto-Encoder (VAE)-based architectures, which are the prevalent solution in multimedia coding. This reflects broader challenges described in the scalable coding literature: many methods either lack certain scalability features [7], leading to suboptimal performance compared to the non-scalable versions of the underlying codecs, or are difficult to integrate into existing frameworks [8], [9], [10]. Nevertheless, the importance of scalability in PCC is widely recognized. This has led both JPEG and MPEG to include scalability as a key objective in the specifications for their DL-based PCC standards [11], [12]. However, neither group has yet published a fully-fledged scalable version of their PCC codecs.

To address these challenges, this paper proposes Scalable Resolution and Quality Hyperprior (SRQH), a novel approach providing joint resolution and quality scalability for PC geometry coding. Integrated into JPEG PCC, SRQH enables scalability while maintaining compatibility with the base codec's non-scalable operation mode. This integration is achieved by replacing the hyper-analysis and hyper-synthesis transforms with the Resolution and Quality-conditioned Latents Probability Estimator (RQuLPE) model during enhancement layer coding.

The main key innovations and contributions brought by SRQH are:

1) Joint scalability of point cloud resolution and quality: SRQH enables JPEG PCC to serve diverse devices through a single scalable bitstream.
2) Minimal RD performance impact: the proposed approach pays a small price for scalability compared to non-scalable JPEG PCC where, for one RD point, the

PC can be decoded only at one specific resolution and quality.
3) Reduced model size: SRQH requires smaller neural networks compared to state-of-the-art solutions [7].
4) Minimal computational overhead: when using SRQH encoding time increases by less than <10% and decoding time by less than <20% per decoded enhancement layer w.r.t. JPEG PCC.
5) Correlated latent spaces: this work shows that the correlation between the latent spaces in the different JPEG PCC models arises from sequential training. This property helps providing effective scalability algorithms and it can be easily introduced in other autoencoder based codecs such as [13], [14], and [15]

SRQH was integrated in JPEG PCC without requiring model retraining or a complete redesign of the VAE architecture. In the context of the integration with the JPEG PCC standard, for which the standardization process has already been concluded, SRQH offers scalability as a simple plug and play module, that is able to use a standard JPEG PCC compliant bit stream at its base-layer, thus enhancing its potential for future adoption. Furthermore, by using a plug-and-play approach, the current SRQH design facilitates the integration within the two stages of any Hierarchical VAE architecture, which is prevalent in Deep Learning (DL)-based multimedia coding, e.g. JPEG AI, among others.

The rest of the paper is organized as follows. Section II overviews the current state-of-the-art in PC coding, while Section III describes the JPEG PCC codec, serving as reference and base layer codec for the proposed approach and the Scalable Quality Hyperprior (SQH) algorithm proposed in [7] which is improved upon in this work. Section IV proposes the novel SRQH algorithm for joint resolution and quality scalability across a wide variety of coding configurations, and the newly proposed RQuLPE which is the main neural network used in SRQH. Finally, Section V reports and analyses the most relevant experimental results and Section VI discusses future work directions.

II. RELATED WORKS

State-of-the-art PCC encompasses various approaches, ranging from traditional signal processing techniques to modern learning-based solutions. Among the conventional approaches, the most relevant are G-PCC and V-PCC [2], the two MPEG standards for PCC. G-PCC, or Geometry-based Point Cloud Compression, leverages octree representations for efficient geometry coding, and uses predictive or hierarchical transforms for attribute coding. On the other hand, V-PCC, or Video-based Point Cloud Compression, projects 3D PC data into a 2D domain representing the geometry and texture information by means of images that are then coded using state-of-the-art video codecs like HEVC and VVC [16] in their Intra coding mode. While G-PCC inherently supports resolution scalability for both geometry and attributes, achieving scalability in V-PCC presents challenges due to its reliance on video coding frameworks. Although MPEG has initiated investigations into various scalability techniques for V-PCC [6], these features remain to be specified in the current version of the standard.

More recently, the advent of DL has revolutionized PC compression, yielding numerous high-performing solutions [3], [8], [15], [17], [18]. Nevertheless, among the many DL-based PCC solutions, few approaches address any form of scalability. DL-PCSC [19] implements quality scalability by channelwise partitioning of latent representations, enabling progressive quality enhancement through incremental transmission. However, this approach faces limitations: the requirement for zero-padding untransmitted latents constrains the latent space design, and the reduced latent space dimensionality at lower rates leads to reduced modeling capabilities [13]. These constraints significantly impact the rate-distortion performance making scalability less appealing.

GRASP-Net [20] offers an alternative approach, by implementing a DL-based enhancement layer atop a G-PCC base layer. However, scalability is limited to this two-layer structure, and the extremely low-resolution base layer may prove impractical for real-world applications.

The work by Ulhaq et al. [21] implements scalability by adapting content for human and machine consumption. In particular, the base layer can be used to solve computer vision tasks (e.g. PC classification) while the enhancement layer allows reconstructing the PC for human visualization. This approach thus addresses scalability requirements that are different from those explored in this work.

SparsePCGC [8] and its successor, Unicorn [9], [10], represent significant advances in resolution scalability for PCC, due to their inherently multiscale nature. At the encoder side, Unicorn employs a hierarchical downscaling approach, encoding the information necessary for losslessly upscaling it at the decoder. When this enhancement information is unavailable, the decoder employs a lossy thresholding strategy for upscaling. By dividing the bitstream in different enhancement layers, required to upscale the PC, Unicorn achieves resolution scalability. Furthermore, Unicorn has a very competitive coding performance when compared with other PC codecs. The intrinsic scalability mechanism of Unicorn, which results from its architecture, contrasts with SRQH, which offers a modular, plug-and-play solution applicable to various codecs.

Additionally, it is also important to mention that recently the MPEG group issued a Call for Proposals for AI-based PCC [12] whose test model is currently under development.

III. JPEG PLENO POINT CLOUD GEOMETRY CODEC AND PREVIOUS EXTENSION FOR QUALITY SCALABILITY

The SRQH method proposed in this paper is implemented on top of the verification model software for the JPEG PCC standard [22], which will be presented next in this section. After that, SQH [7], a previously proposed solution for implementing quality scalability in JPEG PCC for geometry

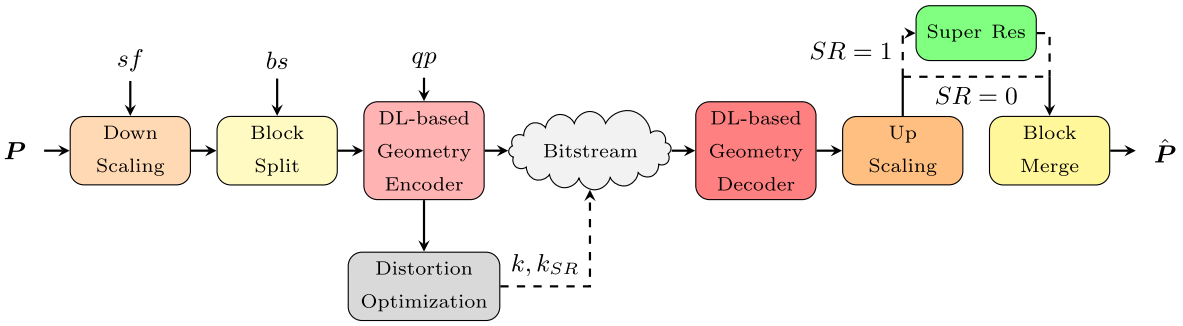


FIGURE 1. High-level scheme of the coding procedure for PC geometry in JPEG PCC.

coding, which served as the basis for the SRQH method, will be described.

A. JPEG PLENO POINT CLOUD CODEC

JPEG PCC is the JPEG standard for PC coding, which uses a learning-based approach for coding both PC geometry and color attributes [23]. The geometry coding in JPEG PCC utilizes a deep learning model structured as an autoencoder, complemented with a variational autoencoder model that determines a mean and scale hyperprior that improves the performance for entropy coding the compressed domain latent representation [14]. To enhance compression performance, particularly for sparse PCs and lower-rate coding scenarios, JPEG PCC incorporates two additional tools:

1) A down-scaling module using a scaling factor parameter, sf.
2) A deep learning-based super-resolution (SR) module to improve reconstruction quality when down-scaling is applied.

JPEG PCC adopts a sparse tensor representation [24] for geometry coding, offering advantages in both computational complexity and rate-distortion performance. In this representation, PCs are described as a tuple $\boldsymbol{x} = (\boldsymbol{x}_C, \boldsymbol{x}_F)$, where $\boldsymbol{x}_C$ represents the coordinates of non-empty voxels, and $\boldsymbol{x}_F$ denotes the corresponding features (initially set to "1" to indicate occupied voxels).

For encoding the color attributes, JPEG PCC projects texture patches onto an image (similarly to V-PCC [25]) which is then coded using the emerging JPEG AI codec [26]. Since the focus of this paper lies in geometry coding, the remaining of this section will focus only on this component.

A high-level description of the full geometry coding and decoding procedures is shown in Fig. 1. Specifically, to encode the geometry of a point cloud $\boldsymbol{P} \in \mathbb{R}^3$, the encoder performs the following operations:

E1. *Downscaling*: The input PC is downscaled by a factor sf through the operation $\boldsymbol{P}' = \lceil \boldsymbol{P}/sf \rfloor$.
E2. *Block Split*: The downscaled points are divided into non-overlapping blocks $\boldsymbol{x}_{l,C} \in \mathbb{R}^3, l \in 1, \ldots, N$ of size bs, such that $\boldsymbol{P}' = \bigcup_{l=1}^{N} \boldsymbol{x}_{l,C}$.
E3. *Sparse Tensor Construction*: For each block, a sparse tensor representation $\boldsymbol{x}_l = (\boldsymbol{x}_{l,C}, \boldsymbol{x}_{l,F})$ is created, where $\boldsymbol{x}_{l,F}$ contains ones to indicate occupied voxels.
E4. *DL-Based Encoding*: The blocks are processed through the deep learning-based coding procedure to generate the bitstream
E5. *Distortion Optimization*: Two parameters per block, k_l and $k_{SR,l}$, are computed and added to the bitstream. These parameters represent the optimal number of points to be retained in the decoded block (with and without super-resolution) to minimize a chosen distortion metric.

At the decoder side, the PC reconstruction process consists of the following operations:

D1. *DL-Based Decoding*: The decoder reconstructs the blocks $\hat{\boldsymbol{x}}_l$ by inputting the compressed domain latent representation, extracted from the bitstream, in the DL-based decoder.
D2. *Top-K Points Selection*: For each decoded block $\hat{\boldsymbol{x}}_l$, only the k_l points with the highest occupancy probabilities are retained, ensuring optimal point selection.
D3. *Upscaling*: The blocks are upscaled according to scaling factor sf (included in the bitstream) to restore the original spatial resolution.
D4. *Super-Resolution*: When super-resolution is enabled ($SR = 1$), the upscaled blocks are processed through the SR network to obtain enhanced blocks $\hat{\boldsymbol{x}}_{SR,l}$.
D5. *Post-SR Top-K Point Selection*: From each super-resolved block, the $k_{SR,l}$ points with the highest probability values are selected, ensuring optimal point selection.
D6. *Block Merge*: Finally, all processed blocks are merged to reconstruct the complete point cloud geometry $\hat{\boldsymbol{P}}$.

The deep learning-based encoding (and decoding) process for each block x_l, is illustrated in Fig. 2. It consists of the following sequence of operations:

E1. *Latents Generation*: Generate latents $\boldsymbol{y}_l = (\boldsymbol{y}_{l,C}, \boldsymbol{y}_{l,F})$ through the analysis transform $\mathcal{G}_a$, expressed as $\boldsymbol{y}_l = \mathcal{G}_a(\boldsymbol{x}_l)$.
E2. *Coordinates Encoding*: code the latent coordinates $\boldsymbol{y}_{l,C}$ using an octree encoder to generate the coordinates bitstream.

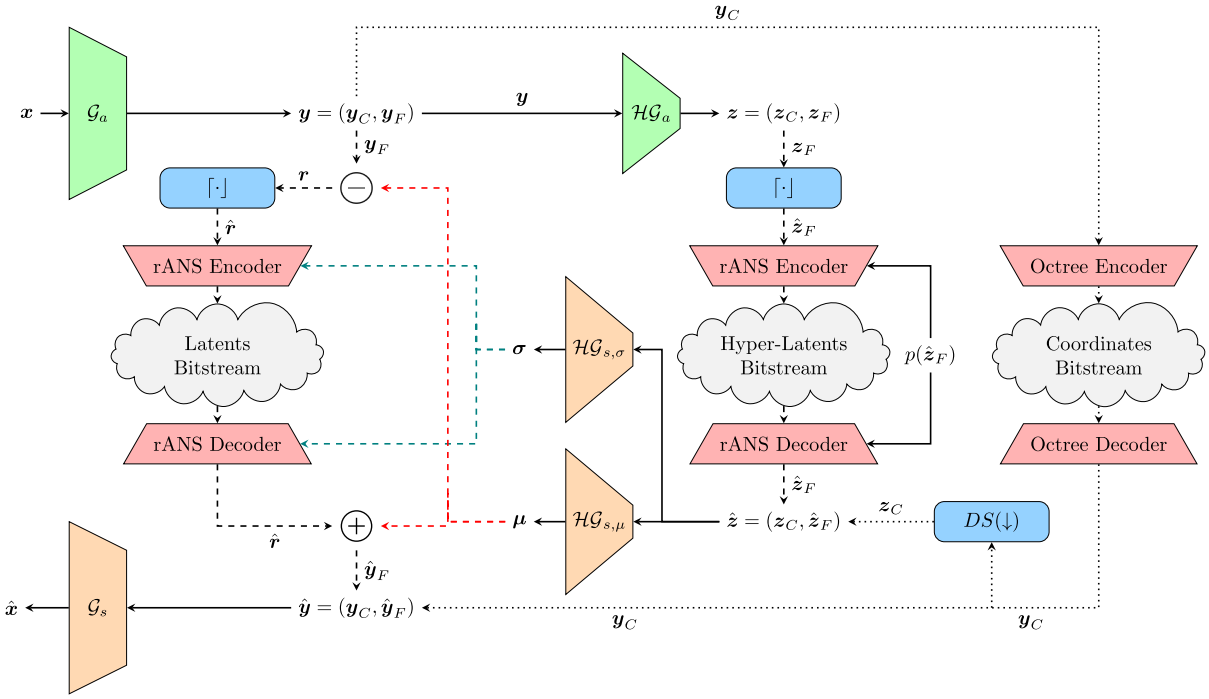


FIGURE 2. Model architecture of the deep learning based codec in JPEG PCC (DL-based Geometry Encoder and DL-based Geometry Decoder in Fig. 1).

E3. *Hyper-Latents Generation*: Generate hyper-latents z_l using the hyper-analysis transform $\mathcal{HG}_a$, where $z_l = \mathcal{HG}_a(y_l)$.

E4. *Hyper-Latents Quantization*: Quantize the hyper-latent features to obtain $\hat{z}_{l,F} = \lceil z_{l,F} \rfloor$.

E5. *Entropy Coding*: Apply rANS entropy coding to the hyper-latents according to a fully factorized prior $p(\hat{z}_{l,F})$ to generate the hyper-latents bitstream.

E6. *Sparse Tensor Construction*: Reconstruct the quantized hyper-latents' sparse representation as $\hat{z}_l = (z_{l,C}, \hat{z}_{l,F})$.

E7. *Latents Distribution Estimation*: Process $\hat{z}_l$ through hyper-synthesis transforms $\mathcal{HG}_{s,\mu}$ and $\mathcal{HG}_{s,\sigma}$ to estimate Gaussian parameters $\mu_l = \mathcal{HG}_{s,\mu}(\hat{z}_l)$, $\sigma_l = \mathcal{HG}_{s,\sigma}(\hat{z}_l)$.

E8. *Residual Encoding*: Calculate and encode quantized residuals $r_l = \lceil y_{l,F} - \mu_l \rfloor$ using $\mathcal{N}(\mathbf{0}, \sigma_l)$ to produce the final latents bitstream.

Conversely, a receiver that needs to decode the blocks from the bitstream will have to:

D1. *Coordinates Decoding*: Losslessly decode $y_{l,C}$ from the coordinates bitstream.

D2. *Hyper-latents Decoding*: Entropy decode $\hat{z}_{l,F}$ from the hyper-latents bitstream using the probability distribution $p(\hat{z}_{l,F})$.

D3. *Coordinates Down-scaling*: Down-scale $y_{l,C}$ by a factor of 4 (as determined by the stride parameters in $\mathcal{HG}_a$'s convolutional layers) to obtain $z_{l,C}$.

D4. *Hyper-Latents Sparse Tensor Construction*: Build the sparse representation of hyper-latents as $\hat{z}_l = (z_{l,C}, \hat{z}_{l,F})$.

D5. *Latents Distribution Estimation*: Compute Gaussian parameters using hyper-synthesis transforms as $\mu_l = \mathcal{HG}_{s,\mu}(\hat{z}_l)$, $\sigma_l = \mathcal{HG}_{s,\sigma}(\hat{z}_l)$.

D6. *Residuals Decoding*: Entropy decode r_l from the latents' bitstream using $\mathcal{N}(\mathbf{0}, \sigma_l)$.

D7. *Latent Features Reconstruction*: Recover the latent features $\hat{y}_{l,F} = \mu_l + r_l$.

D8. *Latents Sparse Tensor Construction*: Reconstruct the sparse representation of latents as $\hat{y}_l = (y_{l,C}, \hat{y}_{l,F})$.

D9. *Block Reconstruction*: Apply the synthesis transform $\mathcal{G}_s$ to the decoded latents to determine the probability for the occupancy state of each voxel in the reconstruct the block: $\hat{x}_l = \mathcal{G}_s(\hat{y}_l)$.

The model training follows an end-to-end approach incorporating all previously described operations except for two differences: quantization is replaced by a differentiable approximation and entropy coding is removed, to ensure full model differentiability. The training utilizes a rate-distortion optimization framework defined by the loss function:

$$\mathcal{L}(x, \hat{x}, y, z) = \mathcal{D}(x, \hat{x}) + \lambda \mathcal{H}(y, z), \tag{1}$$

where $\mathcal{D}(\cdot, \cdot)$ is the distortion, measured as the focal loss [27], $\mathcal{H}(\cdot, \cdot)$ denotes the entropy of the bitstream components under the probability distributions $p(\hat{z})$ and $p(y|\hat{z})$, and λ controls the rate-distortion trade-off. Generally, one model is trained for each RD point corresponding to one value of λ. In JPEG PCC, five different coding models are trained to support the defined range of tradeoffs. The training procedure is carried out by sequentially spanning the chosen values of $\lambda \in \{0.0025, 0.005, 0.01, 0.025, 0.05\}$, using the

checkpoint for the previous λ as a starting point, progressively moving from the lowest value (highest rate/quality) to the highest one (lowest rate/quality). These five models naturally define a quality parameter $qp \in \{1, \ldots, 5\}$, with $qp = 1$ corresponding to $\lambda = 0.05$ (lowest rate/quality) and $qp = 5$ to $\lambda = 0.0025$ (highest rate/quality).

B. SCALABLE QUALITY HYPERPRIOR

The SRQH method proposed in this paper follows a previous work [7] that introduced a quality scalability algorithm, known as SQH. SQH constructs a quality scalable bitstream by leveraging information from latents $\boldsymbol{y}_i$ obtained at a lower quality parameter (QP) ($qp = i$) to predict probability distributions for latents $\boldsymbol{y}_j$ at a higher QP ($qp = j$).

Starting from a low-quality base layer of latents $\boldsymbol{y}_i$, which have already been encoded, the encoder must execute the following sequence of steps to generate a new enhancement layer:

E1. *Higher Quality Latents Generation*: Generate new latents $\boldsymbol{y}_j$ using the JPEG PCC coding model with $qp = j > i$.

E2. *Latents Distribution Estimation*: Predict the means and standard deviations of the latents $\boldsymbol{y}_j$ based on the previous latents $\boldsymbol{y}_i$, using the DL-based Quality-conditioned Latent Probability Estimator (QuLPE) model (detailed in [7]) as $\mu_j, \sigma_j = QuLPE(\hat{\boldsymbol{y}}_i, i, j)$, under the assumption of independently distributed Gaussian latents, $P(\boldsymbol{y}_j|\hat{\boldsymbol{y}}_i)$.

E3. *Entropy Coding*: Generate the SQH bitstream by entropy encoding $\boldsymbol{y}_j$ using μ_j, σ_j.

SQH employs a Mean and Scale Hyperprior entropy model, analogous to JPEG PCC. The key distinction lies in SQH's utilization of previously decoded latents $\hat{\boldsymbol{y}}_i$ as side information, rather than hyper-latents $\hat{\boldsymbol{z}}_j$.

To reconstruct the higher rate/quality PC, the decoder, which can access the base layer information $\hat{\boldsymbol{y}}_i$, performs the following decoding procedure:

D1. *Latents Distribution Estimation*: Derive $\mu_j, \sigma_j = QuLPE(\hat{\boldsymbol{y}}_i, i, j)$ from the base layer information $\hat{\boldsymbol{y}}_i$ using the QuLPE model.

D2. *Higher Quality Latents Decoding*: Decode the higher quality latents $\hat{\boldsymbol{y}}_j$ by applying a rANS decoder to the SQH bitstream using the estimated distribution.

D3. *Higher Quality PC Reconstruction*: Reconstruct the higher quality PC by processing the decoded latents through the JPEG PCC synthesis transform as $\hat{\boldsymbol{x}}_j = \mathcal{G}_{s,j}(\hat{\boldsymbol{y}}_j)$.

D4. *Super Resolution*: If specified in the coding parameters the Super Resolution model is used to enhance the decoded blocks

While SQH effectively handles quality scalability through latents refinement, it faces limitations when dealing with JPEG PCC's downscaling strategy. The challenge arises because varying sf produces latents at different resolutions, a scenario not supported by SQH's U-Net-based QuLPE model, which requires consistent input-output dimensions. This architectural constraint, coupled with the absence of a multi-resolution handling strategy, restricts SQH's practical applicability.

The next sections introduce and evaluate SRQH, an enhanced framework that addresses these limitations by enabling joint quality and resolution scalability in the latent domain. These functional advantages are relevant in the framework of the JPEG PCC codec, but also for the generalization of the SRQH method for other autoencoder-based codecs.

IV. SCALABLE RESOLUTION AND QUALITY HYPERPRIOR

Previous research [7] revealed a significant correlation between latents encoded with varying qp parameters. This fundamental property is crucial to guarantee SQH's effectiveness as it enables a single model to manage diverse coding parameter combinations. However, this raises an important question: does this correlation persist when latents exhibit different resolutions due to varying scaling factors, potentially alongside quality differences? To rigorously investigate this relationship, an analysis was conducted using cosine similarity measurements between latents encoded across different combinations of quantization parameters qp and scaling factors sf. For clarity in subsequent discussions, a simplified notation convention is adopted: lower-rate latents and their associated parameters are designated as ''source'' elements, denoted with the suffix s (e.g., $\boldsymbol{y}_s$), while higher-rate latents and their corresponding parameters are termed ''target'' elements, indicated by the suffix t (e.g., $\boldsymbol{y}_t$).

A. CORRELATION BETWEEN THE LATENTS

Given blocks $\boldsymbol{x}$ in the validation dataset, latent vectors $\boldsymbol{y}_i = Enc(\boldsymbol{x}, sf_s, i)$ and $\boldsymbol{y}_j = Enc(\boldsymbol{x}, sf_t, j)$ are generated, where sf_s, sf_t denote the scaling factors, i, j represent the quality parameters, and $Enc(\cdot, \cdot, \cdot)$ represents the encoding function that performs down-scaling and applies the analysis transform to the input block. Following the methodology established for SQH, cosine similarities between different compressed representations of identical blocks were evaluated across varying quality parameters and resolutions.

The analysis considers source latents encoded with parameters $qp_s = i, sf_s$ and target latents with $qp_t = j, sf_t$. With $\boldsymbol{y}_s = Enc(\boldsymbol{x}, sf_s, i)$ and $\boldsymbol{y}_t = Enc(\boldsymbol{x}, sf_t, j)$, the matrix coefficients at position i, j were derived as the average cosine similarity between latents across all the blocks of the validation dataset. For cases where $sf_s \neq sf_t$, the source and target coordinates do not match due to the resolution difference. In such instances, the lower resolution coordinates were upscaled and the cosine similarities were computed relative to nearest neighbors within a radius of 2, selected based on the ratio between sf_s and sf_t.

This analysis generated three distinct similarity matrices. The first two matrices (shown in Figs. 3a, 3b) were computed using latents from JPEG PCCv4.0, featuring sequentially trained models. Fig. 3c corresponds to $sf_s = 2, sf_t = 1$, while Fig. 3b represents $sf_s = sf_t = 1$. The third matrix (shown in Fig. 3c) utilizes latents from independently

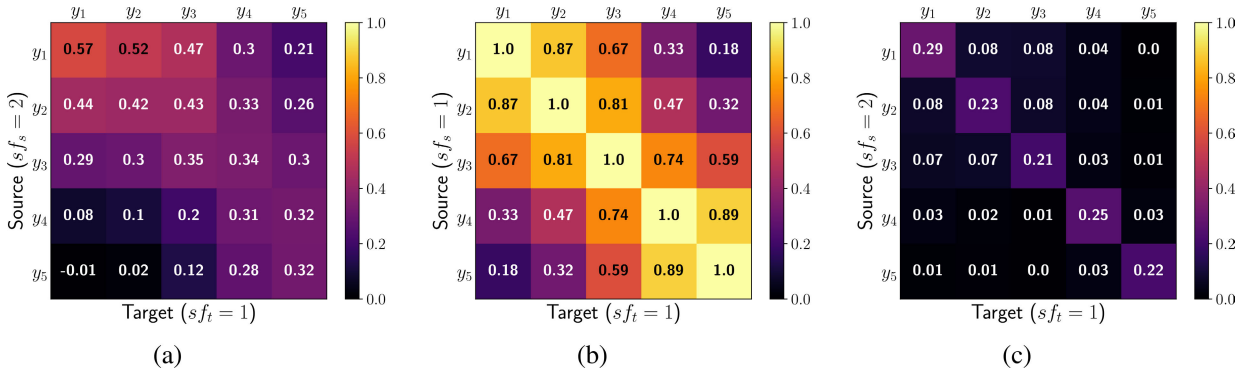


FIGURE 3. Average cosine similarity between corresponding latents produced by the five different JPEG PCC coding models. The left figure (a) corresponds to the sequentially trained models using $sf_s \neq sf_t$, the central figure (b) corresponds to the sequentially trained models using $sf_s = sf_t$, and the figure on the right (c) corresponds to independently trained models with $sf_s \neq sf_t$.

trained models, maintaining identical architecture, training data, and parameters as the sequential case. This comparison serves to determine whether latent alignment emerges as a consequence of sequential training.

The results in Fig. 3a, representing the sequential case with $sf_s \neq sf_t$, demonstrate positive cosine similarity between latents, although lower than cases with identical scaling factors (Fig. 3b). This effect becomes particularly pronounced with increasing disparities between qp_s and qp_t. Notably, configurations where $qp_t < qp_s$ exhibit lower cosine similarity compared to cases where $qp_t > qp_s$, indicating suboptimal conditions for SRQH operation under such parameters configurations.

The analysis of Fig. 3c reveals that independent model training results in completely unaligned latent spaces, thereby demonstrating that sequential training is the key mechanism enabling latent space alignment. This alignment property facilitates the mapping between latent domains generated under different coding configurations, which is an important factor for the effectiveness of the proposed SRQH algorithm, showing the advantage of using sequential training.

B. DESIGN OF THE SCALABLE RESOLUTION AND QUALITY HYPERPRIOR

Designing SRQH to handle latents at different resolutions requires effectively encoding $\boldsymbol{y}_t$ given known source latents $\hat{\boldsymbol{y}}_s$. Since these latents are sparse tensors such that $\boldsymbol{y}_{s,F} \neq \boldsymbol{y}_{t,F}$, $\boldsymbol{y}_{s,C} \neq \boldsymbol{y}_{t,C}$ then SRQH requires two main modules.

1) A coordinates coding module (referred to as RQuLPE-C) capable of encoding target coordinates $\boldsymbol{y}_{t,C}$ at higher resolutions w.r.t. the source coordinates $\boldsymbol{y}_{s,C}$.
2) A features coding module (referred to as RQuLPE-F) that can estimate probability distributions for target latent representations $\boldsymbol{y}_{t,F}$ based on the source latents $\boldsymbol{y}_s$.

The coordinates coding module becomes essential when $sf_s > sf_t$, as this condition results in a higher resolution for the target block compared to the source block. This resolution disparity is mirrored in the resolution of the latent representations, necessitating the encoding of supplementary information to enable the decoder to reconstruct the higher-resolution coordinates accurately. For this purpose, a lossless geometry coding solution was adopted. In particular, a set of plausible high-resolution coordinates $\tilde{\boldsymbol{y}}_{t,C}$ are obtained from $\boldsymbol{y}_{s,C}$ and then the ground truth target occupancy $GT(\tilde{\boldsymbol{y}}_{t,C})$ is losslessly encoded using a probability distribution predicted by the RQuLPE-C model.

On the other hand, RQuLPE-F is needed because the values of the source and target latents are different in both resolution and quality scalability scenarios. This means that a network that can model the target values given the source latents is required.

The described components form the foundation of SRQH, a generalization of SQH, that extends its functionality to handle varying latents' resolutions. This enhanced coding scheme replaces the original QuLPE model with RQuLPE, which comprises the two specialized components: RQuLPE-C and RQuLPE-F.

As illustrated in Fig. 4, the algorithm begins with a base layer generated using a low-rate JPEG PCC bitstream, containing latents, hyper-latents, and coordinates bitstreams (detailed in Section III). Enhancement layers are then constructed by stacking SRQH bitstreams, enabling progressive decoding at higher resolutions and/or qualities.

When $sf_s = sf_t$, the SRQH bitstream is equivalent to the standard SQH latents bitstream. However, when $sf_s \neq sf_t$, an additional bitstream for upsampling the latents' coordinates is required. Additionally, the receiver does not need a side bitstream for decoding hyper-latents $\hat{\boldsymbol{z}}$ since $\hat{\boldsymbol{y}}_s$ serves as the new side information. More specifically, given a base layer $\hat{\boldsymbol{y}}_s$ encoded with parameters qp_s, sf_s using the JPEG PCC coding procedure, the encoder executes the following sequence to generate higher rate/quality layer bitstreams:

E1. *Higher Rate Latents Generation*: Encode the PC $\boldsymbol{x}$ using JPEG PCC with target parameters sf_t and qp_t to obtain $\boldsymbol{y}_t$.
E2. *Upsampled Coordinates Probability Estimation*: For cases where $sf_s \neq sf_t$, compute candidate coordinates $\tilde{\boldsymbol{y}}_{t,C}$ from $\hat{\boldsymbol{y}}_{s,C}$ and estimate $P(\tilde{\boldsymbol{y}}_{t,C}|\hat{\boldsymbol{y}}_s, qp_s)$ using RQuLPE-C.

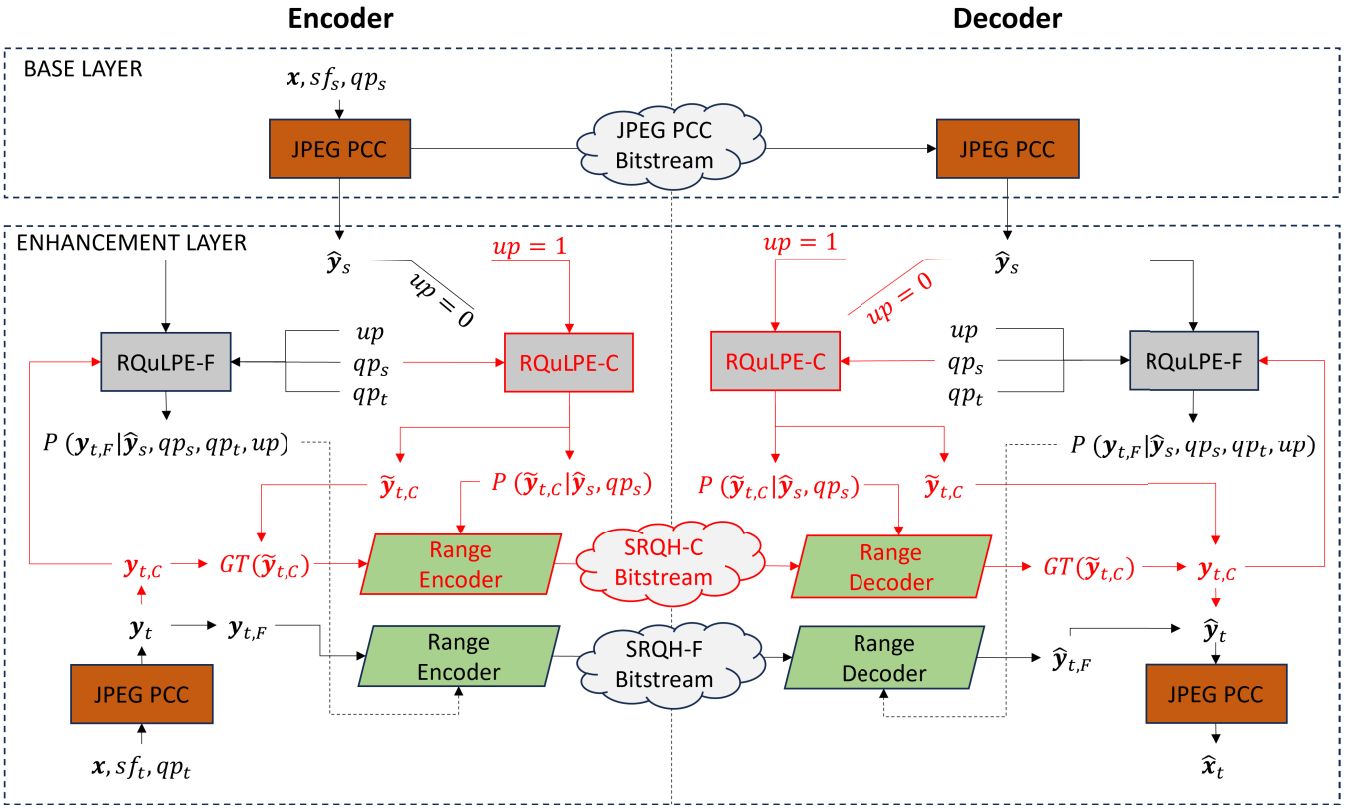


FIGURE 4. Scalable Resolution and Quality Hyperprior coding scheme. In red are the blocks that are relative to the encoding of the coordinates.

E3. *Coordinates Entropy Encoding*: Generate the SRQH-C bitstream by entropy encoding the ground truth occupancy $GT(\tilde{\mathbf{y}}_{t,C})$ using the estimated probability distribution $P(\tilde{\mathbf{y}}_{t,C}|\hat{\mathbf{y}}_s, qp_s)$.

E4. *Latents Features Probability Estimation*: Use RQuLPE-F to estimate μ_t, σ_t, assuming independently distributed Gaussian latents: $P(\mathbf{y}_{t,F}|\hat{\mathbf{y}}_s, qp_s, qp_t, up) = \mathcal{N}(\mu_t, \sigma_t)$ where up is a variable that specifies if latents super-resolution is required.

E5. *Latents Features Entropy Encoding*: Generate the SRQH-F bitstream by entropy encoding $\mathbf{y}_{t,F}$ using the estimated parameters μ_t, σ_t.

The entropy modeling procedure in SRQH closely resembles that of JPEG PCC, as both approaches utilize a hyperprior to estimate a Gaussian prior for the latents. The key distinction in SRQH is the use of $\hat{\mathbf{y}}_s$ as the hyperprior, rather than the hyper-latents $\hat{\mathbf{z}}_t$ employed in JPEG PCC.

The decoder, after decoding $\hat{\mathbf{y}}_s$, can obtain $\hat{\mathbf{y}}_t$ through the following sequence of steps:

D1. *Upsampled Coordinates Probability Estimation*: If $sf_s \neq sf_t$, compute the candidate coordinates $\tilde{\mathbf{y}}_{t,C}$ from $\hat{\mathbf{y}}_{s,C}$ and estimate $P(\tilde{\mathbf{y}}_{t,C}|\hat{\mathbf{y}}_s, qp_s)$ using RQuLPE-C.

D2. *Coordinates Entropy Decoding*: Entropy decode the ground truth $GT(\tilde{\mathbf{y}}_{t,C})$ from the SRQH-C bitstream using the estimated probability distribution $P(\tilde{\mathbf{y}}_{t,C}|\hat{\mathbf{y}}_s, qp_s)$

D3. *Empty Coordinates Pruning*: Refine $\tilde{\mathbf{y}}_{t,C}$ by pruning candidates that are not actual points based on $GT(\tilde{\mathbf{y}}_{t,C})$, yielding $\mathbf{y}_{t,C}$.

D4. *Latents Features Probability Estimation*: Use the RQuLPE-F model to estimate μ_t, σ_t for the latents $\hat{\mathbf{y}}_t$.

D5. *Latents Features Decoding*: Entropy decode $\hat{\mathbf{y}}_{t,F}$ from the SRQH-F bitstream using μ_t, σ_t.

D6. *Higher Resolution And/Or Quality PC Reconstruction*: Reconstruct the final PC $\hat{\mathbf{x}}_t$ using the JPEG PCC decoder applied to $\hat{\mathbf{y}}_t$ followed by the SR if specified in the coding parameters.

This process can be iterated as needed to generate multiple enhancement layers, with each new layer serving as the base for the subsequent one.

C. ASSUMPTIONS ON THE CODING PARAMETERS

To drive the design of the SRQH algorithm some assumptions on the coding parameters were introduced based on the most commonly used coding configurations for JPEG PCC. These assumptions serve to reduce the complexity of SRQH while maintaining its effectiveness. The key constraints are as follows:

1) SRQH and RQuLPE were designed to handle scaling factors that are powers of 2. Scaling factors that are not powers of 2 can be used as coding parameters for JPEG PCC, but are rarely, if ever, considered. Moreover, this

constraint effectively corresponds to pruning the deepest levels of the octree representing the point cloud.

2) The ratio of scaling factors between consecutive layers is restricted to either 1 (as in SQH) or 2. Indeed, a ratio 2^n can be achieved by applying SRQH n times thus upscaling the latents by the correct factor. Although the theoretical ratio between scaling factors for consecutive rate points could be as large as the point cloud's resolution, ratios of 4 or greater are rarely used in practice due to the substantial disparity in rate and distortion values between source and target PCs. Furthermore, as evidenced in Fig. 3a, even a ratio of 2 significantly impacts latent alignment, suggesting that larger ratios would further complicate the estimation of probability distributions for target latents.

These constraints are reasonable tradeoffs (as will be shown in Section V) that allow to effectively cover the most common coding configurations used in JPEG PCC while maintaining a manageable level of complexity in the design of the RQuLPE model.

D. COORDINATES CODING

The process of applying SRQH when $sf_s \geq sf_t$ begins with encoding the information necessary to decode the coordinates $\boldsymbol{y}_{t,C}$. This is accomplished using RQuLPE-C, which is trained to predict the occupancy probabilities for a given set of target voxels. By using transposed sparse convolutions, every source occupied voxel is subdivided into 8 different target voxels (equivalent to creating a new level for the octree), whose coordinates are given by:

$$\hat{\boldsymbol{y}}_{t,C} = \{2 \cdot \boldsymbol{c} + [i,j,k] \,|\, \boldsymbol{c} \in \boldsymbol{y}_{s,C}, [i,j,k] \in \{0,1\}^3\}. \quad (2)$$

Any of the resulting voxels could be empty or occupied in the higher resolution grid, and for this reason a ground truth label $GT(\boldsymbol{c})$ is computed as:

$$GT(\boldsymbol{c}) = \begin{cases} 1 & \text{if } \boldsymbol{c} \in \boldsymbol{y}_{t,C} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

At this point, the RQuLPE-C model is trained using the cross-entropy loss between its predicted probabilities $P(\hat{\boldsymbol{y}}_{t,C})$ and the ground truths $GT(\hat{\boldsymbol{y}}_{t,C})$. Then, at coding time, the estimated probability distribution $P(\hat{\boldsymbol{y}}_{t,C})$ serves as the entropy model to encode the ground truth $\tilde{\boldsymbol{y}}_{t,C} = GT(\hat{\boldsymbol{y}}_{t,C})$.

The RQuLPE-C model, as illustrated in Fig. 5, is implemented as a sparse convolutional network that predicts $P(\hat{\boldsymbol{y}}_{t,C}|\hat{\boldsymbol{y}}_s, qp_s)$. It takes as input the source latents $\hat{\boldsymbol{y}}_s$ and the source quality parameter qp_s which is transformed into an embedding by feeding its one hot encoding $OH(qp_s)$ into an MLP. The embedding is concatenated with $\hat{\boldsymbol{y}}_s$ and fed into the main network branch, consisting of two sparse convolutional layers and followed by an Inception-ResNet layer. The choice of a relatively simple architecture for RQuLPE-C is justified by the negligible size of the bitstream required to encode $\boldsymbol{y}_{t,C}$ ($< 4\%$) compared to that needed for $\boldsymbol{y}_{t,F}$. More complex models, such as autoregressive ones,

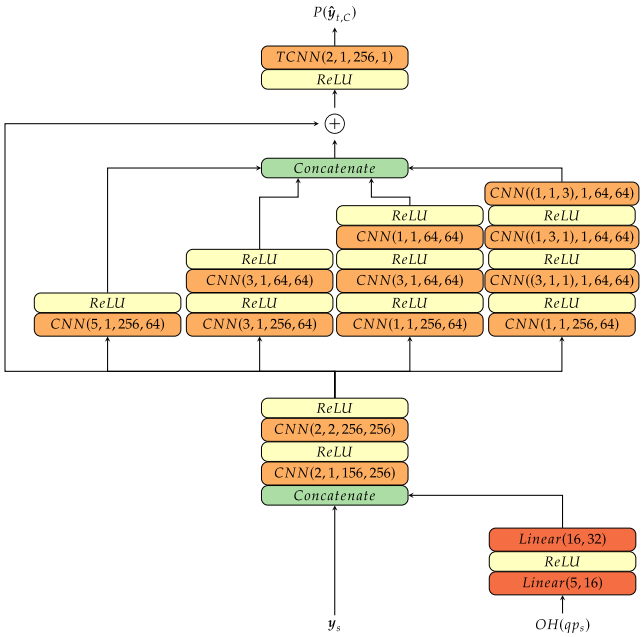


FIGURE 5. Scheme of the RQuLPE-C network used to encode the latents coordinates losslessly.

were deemed unnecessary for this task. This approach enables efficient encoding of the coordinate information necessary for upscaling while maintaining a balance between model complexity and performance.

E. FEATURES CODING

Using low-quality latents as a base layer to encode higher-quality ones requires designing a model that can predict $P(\boldsymbol{y}_t|\hat{\boldsymbol{y}}_s)$. In this case, this probability distribution is modeled as a Gaussian $\mathcal{N}(\mu, \sigma)$ as in [14]. This requires a model, from now on referred to as RQuLPE-F, that can handle latents with possibly different resolutions and/or different qualities. The existing QuLPE model from [7] proves inadequate for this task due to its sparse U-Net architecture, which constrains input and output resolutions to be identical. Two potential architectural approaches emerge:

1) A dual-model approach, with one model that handles source and target latents obtained with different qualities and one model that handles latents with different resolutions. This would result in a high number of total network parameters, which is already considerably high (22M in the QuLPE model [7]).
2) A unified architecture approach with a single model that handles all possible latents configurations. This allows reducing to the minimum the additional number of parameters required to achieve joint resolution and quality scalability due to effective usage of the learned parameters.

Given these considerations, the development focused on a novel architectural approach that minimizes additional parameters while enabling joint resolution and quality

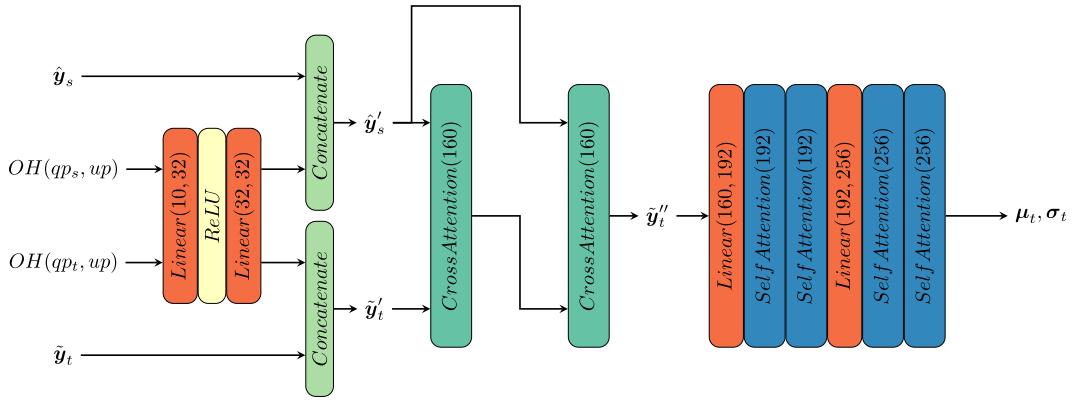


FIGURE 6. Scheme of RQuLPE-F.

scalability. Motivated by recent results in many different fields including PC coding, an architecture based on the attention layers proposed for Point Transformer v2 (PTv2) [28] was adopted instead of sparse convolutions. The reason for this choice is that these architectures do not require matching input and output coordinates, which enables a single model to handle various combinations of resolutions and qualities simultaneously. As demonstrated in Section V, transformer-based models can achieve comparable or superior performance to sparse convolution models with a reduced overall parameter count, enhancing computational efficiency.

A PTv2 layer takes two sparse tensors as inputs. When these are identical then the layer is referred to as a self-attention layer, while if they are different (e.g., one represents the source latents while the other the target latents), then it is referred to as a cross-attention layer. Using cross-attention proves particularly effective since it allows using known information (the source latents) to improve the estimated probability distribution for the target parameters. A key feature of the cross-attention layer is its ability to consistently output a tensor with dimensions matching those of the second input in terms of point count and coordinates.

Under these considerations the steps taken by the RQuLPE-F model (see the architecture in Fig. 6) to predict the probability of the target latents features $\boldsymbol{y}_{t,F}$ are:

1) *Coarse estimate for the target latents*: Generate an initial estimate $\tilde{\boldsymbol{y}}_t$ of the target latents. In particular for each coordinate in $\boldsymbol{y}_{t,C}$ (which is already available through the previous lossless encoding step using RQuLPE-C), the features of its nearest neighbors in $\hat{\boldsymbol{y}}_s$, denoted as $\kappa_{y_s,F}$, are averaged to obtain $\tilde{\boldsymbol{y}}_t$.
2) *Embedding of the coding parameters*: Embed the quality parameters qp_s, qp_t, and a boolean up, indicating if $\frac{sf_s}{sf_t} = 2$, into an embedding vector computed by feeding the one-hot encodings of the coding parameters into a shared multilayer perceptron (MLP), following the approach introduced in [7].
3) *Integration of the latents and the embedding*: Integrate $\hat{\boldsymbol{y}}_s$ and $\tilde{\boldsymbol{y}}_t$ with the corresponding embeddings via concatenation to obtain $\hat{\boldsymbol{y}}'_s$ and $\tilde{\boldsymbol{y}}'_t$.
4) *Refinement of the target latents estimate*: Refine the estimate $\tilde{\boldsymbol{y}}'_t$ through cross-attention with the source latents $\hat{\boldsymbol{y}}'_s$ to obtain $\tilde{\boldsymbol{y}}''_t$
5) *Prediction of the latents probability*: Process $\tilde{\boldsymbol{y}}''_t$ through some self attention layers to predict the parameters μ_t, σ_t describing the probability distribution of the target latents

This specific architectural choice enables the model to seamlessly handle scenarios where $\boldsymbol{y}_{s,C} = \boldsymbol{y}_{t,C}$ (implying $sf_s = sf_t$) as well as cases where $\boldsymbol{y}_{s,C} \neq \boldsymbol{y}_{t,C}$ (indicating $sf_s \neq sf_t$).

The next subsections will elaborate further on each specific step and their implementation details.

1) RQuLPE-F INPUTS

One of the inputs to the model is the initial estimation of the target parameters $\tilde{\boldsymbol{y}}_t$ which was obtained by averaging the latents of the nearest neighbors. Alternative approaches for this initial estimation were explored, such as distance-weighted averaging schemes, these variations did not yield significant improvements in the final rate-distortion performance and were thus discarded.

As additional inputs the model takes the coding parameters qp_s, qp_t, up, however, while the quality parameters qp_s and qp_t are directly input to the model, the resolution information is simplified to the boolean up. This design choice allows for scalability across the most commonly used scaling factor (sf) values in JPEG PCC, which are typically powers of 2. The decision to handle only the case where $sf_s = 2sf_t$ is based on the observation that larger ratios lead to poorly aligned latents, which the model struggles to exploit effectively. This suggests that employing multiple enhancement layers with a scaling factor ratio of 2 is a more efficient strategy than training a model to handle ratios exceeding 2.

The use of a boolean up flag instead of explicit scaling factors is motivated by the understanding that the critical information is whether the latent resolutions differ, rather than their specific values. This is caused by the fact that point distributions (and consequently, latent representations) at various resolutions are heavily influenced by the original

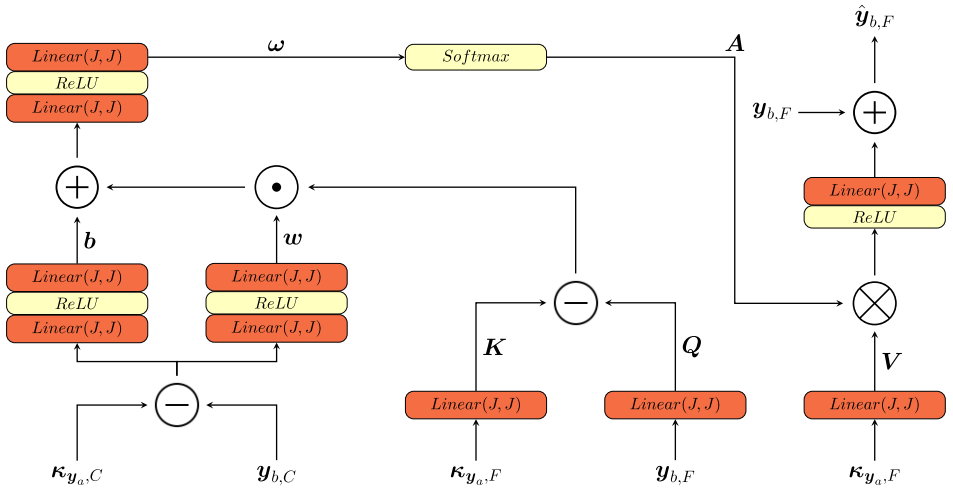


FIGURE 7. Scheme of the adopted attention layer.

point distribution, which is unknown to the decoder. Therefore, explicit knowledge of sf_s and sf_t is unlikely to provide significant benefits and might potentially lead to overfitting.

2) ATTENTION LAYERS ARCHITECTURE

The attention architecture adopted in this work is depicted in Fig. 7. The particular design of the layer allows for the exploitation of information from the first input $\boldsymbol{y}_a$ to enhance the representation of the second input $\boldsymbol{y}_b$. Through the computation of keys and values from $\boldsymbol{y}_a$ and queries from $\boldsymbol{y}_b$, the architecture enables the generation of a residual update for $\boldsymbol{y}_b$ based on a linear combination of values in $\boldsymbol{y}_a$.

In the architecture, $\boldsymbol{K}_{\boldsymbol{y}_a} = (\boldsymbol{K}_{\boldsymbol{y}_a,C}, \boldsymbol{K}_{\boldsymbol{y}_a,F})$ represents the nearest neighbors in $\boldsymbol{y}_a$ for each point in $\boldsymbol{y}_b$, with the neighboring relationship determined in the coordinate domain. While maintaining the core operations proposed in the original PTv2 work [28], this implementation extends beyond the original model by incorporating both self-attention and cross-attention mechanisms. Furthermore, it employs vector attention instead of grouped vector attention, with the number of selected groups set equal to the size of the vectors.

The choice of vector attention is particularly motivated by the nature of the latents produced by JPEG PCC, which are not computed using grouped vector attention. These latents lack specific structural organization along the channel dimension, making arbitrary channel grouping potentially suboptimal for overall performance. Therefore, processing each channel independently is considered more appropriate for this specific application, as it better aligns with the inherent characteristics of the JPEG PCC latent representations. This tailored approach to transformer architecture design enables more effective handling of JPEG PCC latents, potentially leading to improved RD performance.

A key design decision in all attention layers present in the RQuLPE-F model is the use of 5 neighbors for the k-nearest neighbors (KNN) algorithm. This choice was taken after empirical testing revealed that increasing the neighbor count beyond 5 did not yield significant performance improvements but did increase computational complexity.

F. TRAINING AND EVALUATION

The training and validation procedures for SRQH incorporate updates from the JPEG Pleno PCC Common Training and Testing Conditions [5], reflecting modifications to both training and test sets to encompass a broader range of point cloud characteristics. These updated datasets, detailed in [5] and [22], were employed for training and evaluating both RQuLPE-F and RQuLPE-C components.

The latent representations were generated using JPEG PCC with quality parameters $qp \in 1, 2, 3, 4, 5$ and scaling factors $sf \in 1, 2, 4$ for each training and validation block. RQuLPE-C and RQuLPE-F are trained independently, as RQuLPE-C's lossless coding objective for latent coordinates allows the use of ground truth during RQuLPE-F training. Training and validation point clouds were segmented into $128 \times 128 \times 128$ blocks, an increase from the previous $64 \times 64 \times 64$ dimension, to better accommodate downscaling operations in conjunction with the intrinsic downscaling introduced by the analysis transform of JPEG PCC.

During training, at each gradient update step, a tuple (qp_s, qp_t, sf_s, sf_t) is selected for each training PC block with uniform probability, and the corresponding latents $\hat{\boldsymbol{y}}_s$, $\boldsymbol{y}_t$ are loaded accordingly from memory. This sampling strategy ensures comprehensive coverage of all possible configurations encountered during inference. The validation phase implements an exhaustive evaluation across all parameter combinations, ensuring consistent validation loss measurements throughout the training epochs.

Parameter combination selection differs between RQuLPE-C and RQuLPE-F, reflecting their distinct operational requirements:

- RQuLPE-C: $qp_s \in 1, 2, 3, 4, 5, sf_s = 2sf_t$, as the model specifically addresses latent resolution upscaling scenarios, disregarding $\boldsymbol{y}_{t,F}$.
- RQuLPE-F: $qp_s \leq qp_t + 1, 1 \leq sf_s/sf_t \leq 2$, accommodating all permissible parameter combinations.

When training RQuLPE-F, the quality parameter variations were constrained to $qp_s \leq qp_t + 1$ since values outside this range are rarely used in JPEG PCC. While the condition $qp_s > qp_t$ can occur when $sf_s > sf_t$ (as downscaling often necessitates decreasing quality parameters for consecutive rate points), Fig. 3a indicates that the correlation between latents with different scaling factors and decreasing QPs is generally low. Preliminary tests revealed that training outside this range negatively impacts coding performance without providing substantial benefits.

RQuLPE-C is trained with the loss function

$$\mathcal{L}(\boldsymbol{y}_{t,C}, \hat{\boldsymbol{y}}_s) = \mathcal{H}(\boldsymbol{y}_{t,C}|\hat{\boldsymbol{y}}_s, qp_s) \tag{4}$$

while RQuLPE-F was trained to minimize

$$\mathcal{L}(\boldsymbol{y}_t, \hat{\boldsymbol{y}}_s) = \mathcal{H}(\boldsymbol{y}_t|\hat{\boldsymbol{y}}_s, \boldsymbol{y}_{t,C}, qp_s, qp_t, up) \tag{5}$$

The absence of a distortion component aligns with the objective of lossless encoding of $\boldsymbol{y}_t$ with minimal bit consumption. The optimization process utilizes the Adam optimizer with an initial learning rate of 10^{-3}, implementing an exponential decay factor of 10 following 7 epochs without improvement. The training procedure incorporates early stopping, thus terminating when the validation error remains stagnant for 10 consecutive epochs.

V. PERFORMANCE ASSESSMENT FOR JPEG-PCC WITH SRQH

This section presents the experimental setup and the performance assessment of the proposed SRQH implemented on top of JPEG PCC.

A. TEST DATASET

The experimental evaluation utilized the JPEG Pleno PCC test dataset (Fig. 8), adhering to the Common Training and Test Conditions [5]. The test dataset comprises twelve point clouds, categorized into solid, dense, and sparse based on the MPEG-defined average local point density criteria, as detailed in Table 1.

B. CODING CONFIGURATIONS

The coding configurations utilized for coding the test dataset with JPEG PCC and RQuLPE are documented in Table 1. These were derived for JPEG PCC by Guarda et al. [3] by optimizing the PCQM metric [29] while meeting the target rates specified in the CTTC. Regarding block size configuration, the coding configurations in Table 1 were obtained with a block size of 128. However, when using different SF parameters, since the blocks are partitioned after downscaling, the latents would correspond to different regions of the PC, thus preventing RQuLPE from working correctly. For this reason, to provide a fair comparison between JPEG PCC and RQuLPE, the block sizes used to code the PCs with JPEG PCC were set to values equivalent to the ones enforced by SRQH. Specifically, the block size was defined as $128 \cdot sf_4/sf_1$, where sf_1 and sf_4 denote the scaling factors for the initial and final rate points respectively, ensuring a consistent block size of 128 at the final rate point.

An analysis of the configurations reveals that the vast majority of cases for JPEG PCC align with the constraints outlined in Section IV. A single exception is observed for the *Bouquet* point cloud, where the first and second rate points exhibit $sf_s = 4$, $sf_t = 1$ and $qp_s = qp_t + 2$, slightly deviating from the specified constraints. To address this isolated case, the configurations for SRQH were adjusted to ensure full compliance with the established constraints, as reflected in Table 1. Importantly, this adjustment was implemented without compromising the integrity of the original coding configurations. The modification involved the inclusion of a single additional RD point to facilitate a smooth transition between two configurations with $sf_s = 4, sf_t = 1$, while all other RD points remained unaltered.

This scenario underscores the robustness and flexibility of the implemented constraints. Their ability to accommodate the optimal configurations for the entire test set, with only minimal adjustments required in a single instance, demonstrates their practical applicability and effectiveness in real-world scenarios. The constraints thus prove to be sufficiently versatile to address the diverse requirements of point cloud coding across the test dataset.

C. METRICS

The assessment of decoded point clouds employed point-to-point PSNR (PSNR D1) and point-to-plane PSNR (PSNR D2) as quality metrics, while the bitrate was quantified using bits-per-(original)-point (bpp). The rate-distortion (RD) performance comparison against other anchors was evaluated using Bjontegaard delta rate and delta PSNR metrics.

D. BASELINES AND ANCHORS

The codec chosen as a baseline is JPEG PCC v4.0, i.e., the non-scalable codec serving as basis for the proposed SRQH.

Additionally, another solution was evaluated, called SRQH-hybrid which is a hybrid scalable algorithm that selectively employs QuLPE for quality scalability ($sf_s = sf_t$) and RQuLPE for resolution scalability ($sf_s \neq sf_t$). This serves as an ablation study to investigate the potential advantages of utilizing specialized models for each scalability type versus a unified approach. For this comparative analysis, the QuLPE model underwent complete retraining using the updated training dataset.

Among the most widespread codecs, the ones that provide scalability for geometry coding are the conventional G-PCC and the learning-based SparsePCGC [8] and Unicorn, [9], [10]; unfortunately, the source code and trained models were not publicly available during the development of this work, thus preventing any comparison.

FIGURE 8. JPEG Pleno PCC test dataset. From left to right starting from the first row there are: Arco Valentino, CITIUSP, ULB Unicorn, House without roof, Boxer, Thaidancer, Bouquet, Soldier, EPFL, Facade 000009, Saint Michael, Shiva.

TABLE 1. Coding parameters for JPEG PCC expressed as the tuple qp, sf, SR where $SR \in \{T, F\}$ decides if the super-resolution module in JPEG PCC is used/not used.

Type	PC	model	r1	r2	r3	r4	r5
Solid	StMichael	JPEG PCC	2, 4, T	3, 2, T	3, 1, F	5, 1, F	
		SRQH	2, 4, T	3, 2, T	3, 1, F	5, 1, F	
	Bouquet	JPEG PCC	5, 4, T	3, 1, F	4, 1, F	5, 1, F	
		SRQH	5, 4, T	**4, 2, T**	3, 1, F	4, 1, F	5, 1, F
	Soldier	JPEG PCC	3, 4, T	3, 2, T	3, 1, F	4, 1, F	
		SRQH	3, 4, T	3, 2, T	3, 1, F	4, 1, F	
	Thaidancer	JPEG PCC	2, 4, T	2, 2, T	2, 1, F	4, 1, F	
		SRQH	2, 4, T	2, 2, T	2, 1, F	4, 1, F	
Dense	Boxer	JPEG PCC	1, 4, T	2, 2, T	5, 2, T	4, 1, F	
		SRQH	1, 4, T	2, 2, T	5, 2, T	4, 1, F	
	House wo roof	JPEG PCC	2, 4, T	3, 2, T	4, 1, F	5, 1, F	
		SRQH	2, 4, T	3, 2, T	4, 1, F	5, 1, F	
	CITIUSP	JPEG PCC	1, 4, F	4, 4, F	4, 2, F	5, 2, T	
		SRQH	1, 4, F	4, 4, F	4, 2, F	5, 2, T	
	Facade 00009	JPEG PCC	2, 4, T	3, 2, T	4, 1, F	5, 1, F	
		SRQH	2, 4, T	3, 2, T	4, 1, F	5, 1, F	
Sparse	EPFL	JPEG PCC	1, 4, F	4, 4, F	4, 2, F	5, 2, F	
		SRQH	1, 4, F	4, 4, F	4, 2, F	5, 2, F	
	Arco Valentino	JPEG PCC	1, 4, F	3, 4, F	4, 2, F	5, 2, F	
		SRQH	1, 4, F	3, 4, F	4, 2, F	5, 2, F	
	Shiva	JPEG PCC	2, 4, F	3, 2, F	5, 2, T	4, 1, F	
		SRQH	2, 4, F	3, 2, F	5, 2, T	4, 1, F	
	ULB Unicorn	JPEG PCC	2, 4, F	3, 4, F	5, 4, F	4, 2, F	
		SRQH	2, 4, F	3, 4, F	5, 4, F	4, 2, F	

For this reason, most of the codecs chosen as anchors do not provide any form of scalability. The chosen conventional anchors are:

1) G-PCC Octree v23.
2) V-PCC Intra v24.

while the chosen learning-based anchors are:

1) GRASP-Net [20].
2) PCGCv2 [17].

Among these only G-PCC provides scalability (in particular resolution scalability) while all the other solutions are non-scalable. To compare scalable and non-scalable solutions the RD curves for the scalable solutions were obtained by setting the quality metric for a specific rate point as the maximum possible quality obtainable with that bitstream.

E. PERFORMANCE ASSESSMENT

This section presents the results obtained by the proposed solution against the baselines and the other anchors.

1) SRQH VERSUS JPEG-PCC-BASED CODING SOLUTIONS

Firstly, in order to understand how much SRQH affects RD performance, it will be compared with non-scalable JPEG PCC and SRQH-hybrid. The RD curves obtained from the three solutions can be seen in Fig. 9 where their average over the whole test set is shown.

The results demonstrate that SRQH maintains performance comparable to JPEG PCC, with SRQH and SRQH-hybrid exhibiting nearly identical average performance. This achievement is particularly significant as it indicates that

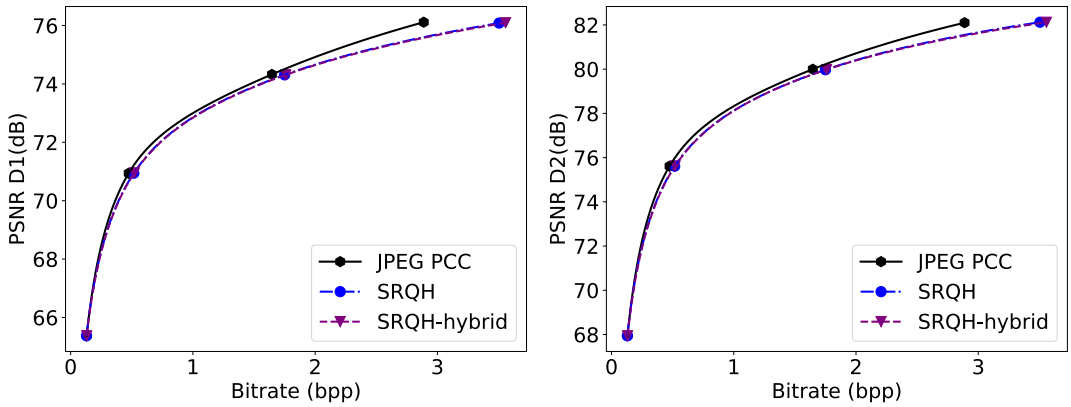


FIGURE 9. Average over the test set of the RD curves for JPEG PCC based solutions.

TABLE 2. Bjontegaard metrics of SRQH with reference to JPEG PCC based solutions.

PC	Type	JPEG PCC				SRQH-hybrid			
		PSNR D1		PSNR D2		PSNR D1		PSNR D2	
		BD-Rate	BD-PSNR	BD-Rate	BD-PSNR	BD-Rate	BD-PSNR	BD-Rate	BD-PSNR
Saint-Michel	Solid	7.37	-0.34	7.21	-0.39	**-0.35**	**0.02**	**-0.26**	**0.01**
Bouquet		21.12	-0.52	24.09	-0.70	**-0.21**	**0.01**	**-0.04**	**0.00**
Soldier		1.57	-0.08	1.50	-0.09	0.03	-0.00	0.04	-0.00
Thaidancer		5.22	-0.25	5.27	-0.33	**-0.18**	**0.01**	**-0.17**	**0.01**
Boxer	Dense	6.66	-0.13	5.65	-0.26	0.19	-0.00	0.12	-0.01
House without roof		8.62	-0.17	9.55	-0.27	**-0.06**	**0.00**	**-0.19**	**0.01**
CITIUSP		**-1.10**	**0.02**	**-0.71**	**0.01**	**-6.95**	**0.22**	**-6.95**	**0.29**
Facade		7.00	-0.19	5.58	-0.18	**-0.03**	**0.00**	**-0.09**	**0.00**
EPFL	Sparse	6.84	-0.17	6.87	-0.23	2.11	-0.05	2.17	-0.07
Arco Valentino		3.94	-0.12	2.98	-0.10	**-0.05**	**0.00**	**-0.20**	**0.01**
Shiva		6.59	-0.13	7.25	-0.20	**-0.06**	**0.00**	0.14	-0.01
ULB Unicorn		10.38	-0.18	9.94	-0.32	**-1.72**	**0.05**	**-0.44**	**0.04**
Avg		7.02	-0.19	7.10	-0.25	**-0.61**	**0.02**	**-0.49**	**0.02**
Avg (Dense)		5.30	-0.12	5.02	-0.18	**-1.71**	**0.06**	**-1.78**	**0.07**
Avg (Sparse)		6.94	-0.15	6.76	-0.21	0.07	**0.00**	0.42	-0.01
Avg (Solid)		8.82	-0.30	9.52	-0.37	**-0.18**	**0.01**	**-0.10**	**0.01**

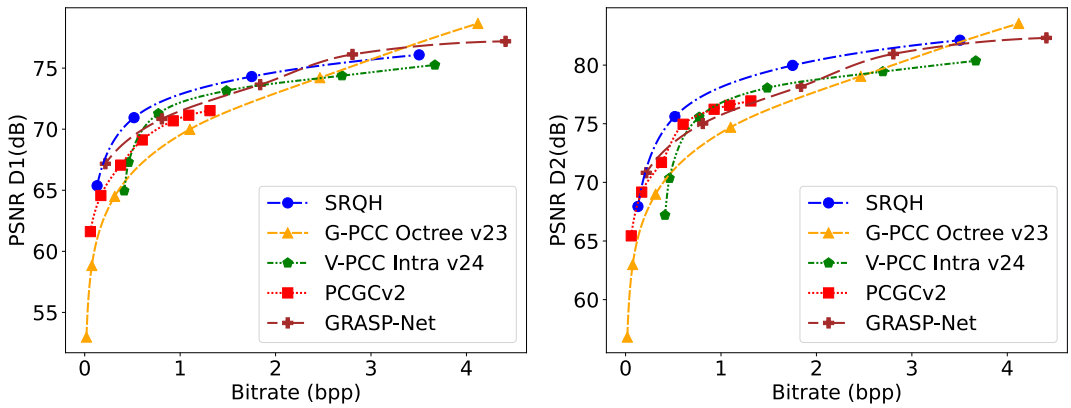


FIGURE 10. Average RD curves when comparing against anchors that are not based on JPEG PCC.

SRQH successfully implements full resolution and quality scalability while incurring minimal rate overhead compared to the non-scalable JPEG PCC solution.

The Bjontegaard metrics in Table 2 quantify the performance differences, revealing that the price to be paid for scalability is within 5-9% across different PC categories which is acceptable for such an important feature. The most challenging case is observed for the *Bouquet* point cloud, attributable to the fact that two consecutive configurations were not compliant with the constraints set before, since $sf_s/sf_t = 4$. This required adding an extra enhancement layer, thus increasing the price for scalability; additionally,

TABLE 3. BD-Rate of SRQH with reference to G-PCC and V-PCC.

PC	Type	G-PCC Octree v23				V-PCC Intra v24			
		PSNR D1		PSNR D2		PSNR D1		PSNR D2	
		BD-Rate	BD-PSNR	BD-Rate	BD-PSNR	BD-Rate	BD-PSNR	BD-Rate	BD-PSNR
Saint-Michel	Solid	**-77.33**	**7.11**	**-63.59**	**5.59**	**-42.23**	**3.34**	**-39.12**	**3.89**
Bouquet		**-76.97**	**6.17**	**-66.44**	**4.92**	**-39.47**	**2.53**	**-36.92**	**2.65**
Soldier		**-82.85**	**9.31**	**-73.15**	**7.99**	**-23.24**	**1.76**	**-24.57**	**2.06**
Thaidancer		**-82.54**	**8.56**	**-69.35**	**7.32**	**-4.64**	**0.21**	0.16	-0.06
Boxer	Dense	**-67.82**	**3.02**	**-73.11**	**5.63**	35.95	-0.58	30.27	-0.93
House without roof		**-51.11**	**2.44**	**-51.31**	**2.70**	13.73	-0.19	**-26.35**	**1.13**
CITIUSP		**-24.27**	**1.08**	**-10.77**	**0.49**	18.30	-0.54	19.59	-0.69
Facade		**-28.77**	**1.38**	**-45.71**	**2.66**	571.35	-5.77	158.94	-4.20
EPFL	Sparse	**-20.69**	**0.52**	**-1.52**	-0.07	**-15.89**	**0.40**	**-2.01**	-0.01
Arco Valentino		**-17.22**	**0.40**	**-5.59**	-0.21	**-62.90**	**2.85**	**-71.87**	**3.88**
Shiva		**-15.59**	-0.05	**-5.25**	-0.16	**-53.46**	**2.04**	**-68.26**	**4.14**
ULB Unicorn		65.75	-2.71	**-18.21**	**0.47**	75.37	-2.30	**-4.66**	-2.21
Avg		**-39.95**	**3.10**	**-40.33**	**3.11**	39.40	**0.31**	**-5.40**	**0.80**
Avg (Dense)		**-42.99**	**1.98**	**-45.23**	**2.87**	159.83	-1.77	45.61	-1.17
Avg (Sparse)		3.07	-0.46	**-7.64**	**0.01**	**-14.22**	**0.75**	**-36.70**	**1.45**
Avg (Solid)		**-79.92**	**7.79**	**-68.13**	**6.46**	**-27.39**	**1.96**	**-25.11**	**2.14**

TABLE 4. BD-Rate of SRQH with reference to GRASP-Net and PCGCv2.

PC	Type	PCGCv2				GRASP-Net			
		PSNR D1		PSNR D2		PSNR D1		PSNR D2	
		BD-Rate	BD-PSNR	BD-Rate	BD-PSNR	BD-Rate	BD-PSNR	BD-Rate	BD-PSNR
Saint-Michel	Solid	13.73	-0.68	14.32	-0.80	4.94	-0.15	**-2.45**	**0.22**
Bouquet		61.58	-2.13	62.69	-2.32	33.01	-0.77	28.17	-0.74
Soldier		8.49	-0.30	10.75	-0.37	**-19.60**	**1.53**	**-21.41**	**1.94**
Thaidancer		**-31.64**	**2.20**	18.62	-0.77	**-12.89**	**1.45**	**-14.23**	**1.85**
Boxer	Dense	**-18.24**	**1.56**	44.90	**1.60**	0.29	**0.01**	**-7.72**	**0.41**
House without roof		**-54.36**	**1.39**	**-19.14**	**0.79**	**-0.91**	-0.03	**-14.72**	**0.47**
CITIUSP		**-57.63**	**2.80**	31.71	-0.76	47.93	-1.43	48.06	-1.69
Facade		**-44.45**	**1.51**	**-6.36**	**0.53**	8.05	-0.25	**-20.06**	**0.76**
EPFL	Sparse	6.59	**0.11**	**-1.79**	**0.51**	24.33	-0.79	2.58	-0.38
Arco Valentino		**-42.71**	**2.15**	**-33.68**	**1.95**	8.07	-0.88	5.79	-0.81
Shiva		35.25	**0.04**	51.33	**0.03**	19.96	-0.54	12.64	-0.41
ULB Unicorn		**-92.78**	**11.83**	**-74.33**	**5.82**	206.61	-2.90	89.15	-1.50
Avg		**-18.01**	**1.71**	8.25	**0.52**	26.65	-0.40	8.81	**0.01**
Avg (Dense)		**-43.67**	**1.81**	12.78	**0.54**	13.84	-0.43	1.39	-0.01
Avg (Sparse)		**-23.41**	**3.53**	**-14.62**	**2.08**	64.74	-1.28	27.54	-0.78
Avg (Solid)		13.04	-0.23	26.60	-1.06	1.36	**0.51**	**-2.48**	**0.82**

the resulting configurations have $qp_s > qp_t$, thus a lower latent alignment.

The comparable performance between SRQH and SRQH-hybrid suggests that employing separate models for quality ($sf_s = sf_t$) and resolution ($sf_s \neq sf_t$) scalability offers no substantial benefits. In fact, SRQH slightly outperforms SRQH-hybrid on average, suggesting that exposure to diverse parameter configurations during training provides a beneficial regularizing effect. This finding favors the adoption of the proposed single unified RQuLPE model, which achieves similar or slightly better rate-distortion performance with significantly fewer parameters (4.7M parameters versus QuLPE's 22M parameters). This additionally proves the effectiveness of PTv2 for this task which achieves similar performance with fewer network parameters than a much larger model based on sparse convolutions.

As a final test SRQH was also compared with a naive scalable version of JPEG PCC where the bitstreams for each configuration in Table 1 were concatenated together to achieve scalability. This solution yields 14.32% higher bitrates on average (BD-rate D1) w.r.t. SRQH and 54% additional rate when considering the full bitstream (i.e. the one relative to the highest rate point that allows decoding all chosen coding configurations). This shows that SRQH is an effective solution for providing scalability in JPEG PCC.

2) SRQH VERSUS ANCHORS

The performance comparison of SRQH against the anchor codecs, as illustrated in Fig. 10, shows that despite being a scalable solution, SRQH performs better or on par with all the considered anchors. To better assess the relative performance between the solution and the anchors and to provide a more consistent evaluation, Bjontegaard-Delta (BD) metrics (BD-Rate, BD-PSNR) were employed, providing a more reliable basis for comparison despite their dependence on polynomial interpolation quality.

Table 3 presents the BD metrics for SRQH in comparison with the standardized codecs (G-PCC and V-PCC), while Table 4 showcases the BD metrics for SRQH in comparison with the learning-based codecs (GRASP-Net and PCGCv2).

The analysis reveals that SRQH significantly outperforms G-PCC on solid and dense PCs, while showing slightly inferior performance on sparse PCs, primarily due to G-PCC's exceptional results on ULB Unicorn. In comparison with V-PCC, SRQH demonstrates superior performance on solid and sparse PCs, though V-PCC achieves better RD performance (D1 PSNR) on all four dense PCs in the test set.

Regarding learning-based codecs, SRQH outperforms PCGCv2 on dense and sparse PCs, with marginally lower performance on solid PCs. GRASP-Net shows comparable performance on solid PCs and superior results on dense and sparse PCs, indicating its effectiveness as an enhancement layer addressing G-PCC's limitations.

A critical distinction to note is that all aforementioned methods, except G-PCC, lack scalability. In practical scenarios where scalability is required, SRQH offers a significant advantage since it enables decoding point clouds at multiple resolution/quality levels from a single bitstream.

F. COMPLEXITY ANALYSIS

The impact of the RQuLPE model in scalable coding was assessed through comprehensive encoding/decoding time and memory consumption measurements. The evaluation was conducted on hardware comprising a single L40s GPU and 4 cores of an AMD EPYC 9224 CPU, with each PC processed 20 times, excluding extreme measurements to eliminate outliers. To better assess the computational complexity, the PCs were coded with $qp \in \{1 \dots 5\}$ since coding PCs at different scales leads to very different coding times. To obtain interpretable complexity metrics, testing focused on quality parameter variations ($qp \in 1, 2, 3, 4, 5$) at original resolution ($sf = 1$) with a block size of 128, excluding super-resolution effects to avoid accounting for postprocessing (which is equal for both JPEG PCC and SRQH) in the complexity evaluation. This configuration was applied to both old and new test datasets, providing a robust sample size for complexity assessment.

The evaluation framework measured four key temporal metrics:

- $t_{enc,jpeg}(i)$: JPEG PCC encoding time, for non-scalable streams, required to encode the PC at quality i.
- $t_{enc,SRQH}$: SRQH encoding time for the full scalable bitstream.
- $t_{dec,jpeg}(i)$: JPEG PCC decoding time, for non-scalable streams, required to decode the PC at quality i.
- $t_{dec,SRQH}(i)$: time required by SRQH to decode the PC at quality i from the scalable stream.

The relative computational overhead introduced by SRQH was quantified through

$$t_{enc,extra} = \left(\frac{t_{enc,SRQH}}{\sum_{i=1}^{5} t_{enc,jpeg}(i)} - 1 \right) \cdot 100 \quad (6)$$

$$t_{dec,extra}(i) = \left(\frac{t_{dec,SRQH}(i)}{t_{dec,jpeg}(i)} - 1 \right) \cdot 100 \quad (7)$$

A key distinction exists between encoding and decoding processes: encoding requires full PC decoding at each rate point for distortion optimization, while decoding necessitates processing only the base layer and relevant enhancement layers without reconstructing the PC at each quality. For this reason, if quality i needs to be decoded then the base layer and $i-1$ enhancement layers need to be decoded. Consequently, SRQH execution time increases linearly with quality level i, suggesting a linear relationship between decoding time and enhancement layer count.

Experimental results showed that $t_{enc,extra} = 9.95\%$, additionally, the decoding time analysis, visualized in Fig. 11, confirms the expected linear growth with quality level, where each enhancement layer processing adds approximately 20% to the base JPEG PCC decoding time. This value is higher than the average increase in encoding time since, at the encoder side, the time for distortion optimization is not negligible and it reduces the relative impact of SRQH w.r.t. JPEG PCC (a similar effect would be seen also if SR was introduced).

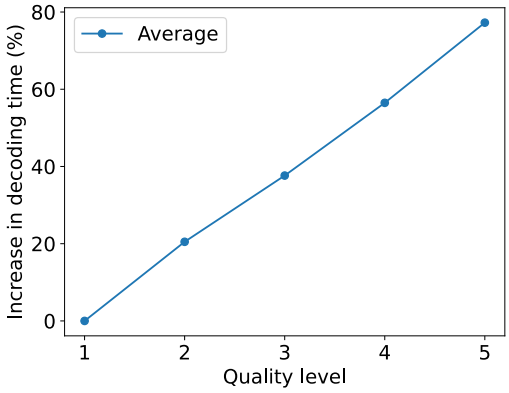


FIGURE 11. Extra decoding time, in percentage, required by SRQH w.r.t. JPEG PCC.

Additionally, the peak GPU memory usage across the test set was measured for RQuLPE, QuLPE, and JPEG PCC. This measurement was conducted to ascertain the increased memory requirements of a PTv2-based architecture compared to one employing sparse convolutions, and evaluate whether RQuLPE introduces a bottleneck when adding scalability to JPEG PCC. Across the entire test set, RQuLPE exhibits a peak memory usage of at most 5.6 times that of QuLPE, a difference attributed to the use of vector attention. However, RQuLPE still consumes 57.4 times less memory than JPEG PCC, a consequence of operating in the latent space, which has a significantly lower resolution than the original PC. These results indicate that, from a memory footprint standpoint, SRQH does not constrain JPEG PCC, and the additional encoding and decoding times introduced by SRQH could be mitigated by increasing the number of blocks encoded in parallel with RQuLPE.

VI. CONCLUSION

This paper introduces SRQH, a novel joint resolution and quality scalability scheme implemented and validated for geometry coding in the JPEG PCC standard. SRQH enables the encoding of point clouds into scalable bitstreams, supporting compressed representations at various compression qualities and resolutions. The proposed method demonstrates minimal to no rate-distortion performance loss compared to non-scalable JPEG PCC, indicating that the added scalability functionality incurs a negligible rate cost.

A key advantage of SRQH is its implementation in the latent space, which circumvents common drawbacks associated with spatial domain scalability, such as residual sparsity and the necessity to decode point clouds at each target quality. This approach results in limited additional complexity relative to the JPEG PCC baseline. Moreover, SRQH enhances the capabilities of SQH by incorporating resolution scalability alongside quality scalability while reducing the required network parameters. The modular nature of SRQH, implemented through the RQuLPE model, allows for flexible usage of the feature. Users can still utilize JPEG PCC in a non-scalable manner when scalability is not required, maintaining backward compatibility and versatility.

The latent space alignment principle underlying SRQH is readily achievable through sequential training, making this approach adaptable to other learning-based codecs.

Future research directions include integrating SRQH with JPEG-AI to enable attribute domain scalability in JPEG PCC and exploring additional applications of latent alignment beyond scalability. Preliminary findings suggest that low-quality latents may serve as superior side information compared to hyper-latents in certain scenarios, potentially leading to improvements in current entropy models. Additionally, extending support for arbitrary block sizes across various enhancement layers would enhance the algorithm's adaptability in practical applications.

ACKNOWLEDGMENT

After the first draft, the text in the various sections of the manuscript (except the Abstract) was improved using Claude 3.5 Sonnet through the Perplexity AI web app. After this process, the article underwent multiple sets of reviews from the authors and was thus further improved and modified.

REFERENCES

[1] E. Camuffo, D. Mari, and S. Milani, "Recent advancements in learning algorithms for point clouds: An updated overview," *Sensors*, vol. 22, no. 4, p. 1357, Feb. 2022.

[2] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai, "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, no. 1, p. 13, 2020.

[3] A. F. R. Guarda, N. M. M. Rodrigues, and F. Pereira, "The JPEG pleno learning-based point cloud coding standard: Serving man and machine," 2024, *arXiv:2409.08130*.

[4] A. Seleem, A. F. R. Guarda, N. M. M. Rodrigues, and F. Pereira, "Deep learning-based compressed domain multimedia for man and machine: A taxonomy and application to point cloud classification," *IEEE Access*, vol. 11, pp. 128979–128997, 2023.

[5] *Common Training and Test Conditions for JPEG Pleno Point Cloud V2.1*, document ISO/IEC JTC 1/SC29/WG1 N100841, PCQ, 103rd JPEG Meeting, Apr. 2024.

[6] *Scalability Support in V-PCC*, document ISO/IEC JTC1/SC29/WG1 N19156, 129th MPEG Meeting, Bruxelles, Belgium, Jan. 2020.

[7] D. Mari, A. F. R. Guarda, N. M. M. Rodrigues, S. Milani, and F. Pereira, "Point cloud geometry scalable coding with a quality-conditioned latents probability estimator," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2024, pp. 3410–3416.

[8] J. Wang, D. Ding, Z. Li, X. Feng, C. Cao, and Z. Ma, "Sparse tensor-based multiscale representation for point cloud geometry compression," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 5, pp. 1–18, May 2022.

[9] J. Wang, R. Xue, J. Li, D. Ding, Y. Lin, and Z. Ma, "A versatile point cloud compressor using universal multiscale conditional coding—Part I: Geometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 1234–1248, Jun. 2023.

[10] J. Wang, R. Xue, J. Li, D. Ding, Y. Lin, and Z. Ma, "A versatile point cloud compressor using universal multiscale conditional coding—Part II: Attribute," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 1, pp. 252–268, Jan. 2024.

[11] *Final Call for Proposals on JPEG Pleno Point Cloud Coding*, document ISO/IEC JTC1/SC29/WG1 N100097, Jan. 2022.

[12] *Call for Proposals for AI-Based Point Cloud Coding*, document ISO/IEC JTC 1/SC 29/WG 2 N365, 146th MPEG Meeting, Rennes, France, Apr. 2024.

[13] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," in *Proc. Int. Conf. Learn. Represent.*, Jan. 2018, pp. 1–23.

[14] J.-H. Kim, B. Heo, and J.-S. Lee, "Joint global and local hierarchical priors for learned image compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 5992–6001.

[15] M. Quach, G. Valenzise, and F. Dufaux, "Improved deep point cloud geometry compression," in *Proc. IEEE 22nd Int. Workshop Multimedia Signal Process. (MMSP)*, Sep. 2020, pp. 1–6.

[16] B. Bross, Y.-K. Wang, Y. Ye, S. Liu, J. Chen, G. J. Sullivan, and J.-R. Ohm, "Overview of the versatile video coding (VVC) standard and its applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3736–3764, Oct. 2021.

[17] J. Wang, H. Zhu, H. Liu, and Z. Ma, "Lossy point cloud geometry compression via end-to-end learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 12, pp. 4909–4923, Dec. 2021.

[18] G. Liu, J. Wang, D. Ding, and Z. Ma, "PCGFormer: Lossy point cloud geometry compression via local self-attention," in *Proc. IEEE Int. Conf. Vis. Commun. Image Process. (VCIP)*, Dec. 2022, pp. 1–5.

[19] A. F. R. Guarda, N. M. M. Rodrigues, and F. Pereira, "Point cloud geometry scalable coding with a single end-to-end deep learning model," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 3354–3358.

[20] J. Pang, M. A. Lodhi, and D. Tian, "GRASP-Net: Geometric residual analysis and synthesis for point cloud compression," in *Proc. 1st Int. Workshop Adv. Point Cloud Compress., Process. Anal.*, Oct. 2022, pp. 11–19.

[21] M. Ulhaq and I. V. Bajić, "Scalable human–machine point cloud compression," 2024, *arXiv:2402.12532*.

[22] *Verification Model Description for JPEG Pleno Learning-Based Point Cloud Coding V4.0*, document ISO/IEC JTC 1/SC29/WG1 N100709 REQ, 102nd JPEG Meeting, San Francisco, CA, USA, Jan. 2024.

[23] A. F. R. Guarda, N. M. M. Rodrigues, and F. Pereira, "Point cloud geometry and color coding in a learning-based ecosystem for JPEG coding standards," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2023, pp. 2585–2589.

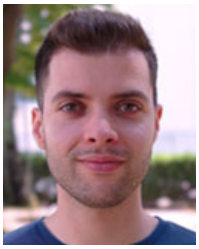
[24] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal ConvNets: Minkowski convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3070–3079.

[25] *V-PCC Codec Description*, document ISO/IEC JTC 1/SC29/WG1 N91058, 3rd MPEG 3D Graph. Coding Meeting, Apr. 2021.

[26] *JPEG AI Verification Model 1.0 Description*, document ISO/IEC JTC 1/SC29/WG1 N100332, 97th JPEG Meeting, Oct. 2022.

[27] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.
[28] X. Wu, Y. Lao, L. Jiang, X. Liu, and H. Zhao, "Point transformer v2: Grouped vector attention and partition-based pooling," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2022, pp. 33330–33342.
[29] G. Meynet, Y. Nehmé, J. Digne, and G. Lavoué, "PCQM: A full-reference quality metric for colored 3D point clouds," in *Proc. 12th Int. Conf. Quality Multimedia Exper. (QoMEX)*, May 2020, pp. 1–6.

DANIELE MARI received the B.S. degree in information engineering and the M.S. degree in ICT for internet and multimedia from the University of Padova, Italy, in 2019 and 2021, respectively. He is currently pursuing the Ph.D. degree in information engineering in Padua. Additionally, he has spent six months as a Visiting Ph.D. Student with Instituto Superior Técnico, Universidade de Lisboa, Portugal, in 2023. He has authored several publications in top conferences and journals in this field. His main research interests include learned point cloud and image coding.



ANDRÉ F. R. GUARDA (Member, IEEE) received the B.Sc. and M.Sc. degrees in electrotechnical engineering from Instituto Politécnico de Leiria, Portugal, in 2013 and 2016, respectively, and the Ph.D. degree in electrical and computer engineering from Instituto Superior Técnico, Universidade de Lisboa, Portugal, in 2021. He has been a Researcher with Instituto de Telecomunicações, since 2011, where he currently holds a postdoctoral position. He has authored several publications in top conferences and journals in this field and is actively contributing to the standardization efforts of JPEG and MPEG on learning-based point cloud coding. His main research interests include multimedia signal processing and coding, with particular focus on point cloud coding with deep learning.

NUNO M. M. RODRIGUES (Senior Member, IEEE) received the degree in electrical engineering, in 1997, the M.Sc. degree from Universidade de Coimbra, Portugal, in 2000, and the dual Ph.D. degree from Universidade de Coimbra, and in collaboration with the Universidade Federal do Rio de Janeiro, Brazil, in 2009. He is currently a Professor with the Department of Electrical Engineering, School of Technology and Management, Politécnico de Leiria, Portugal, and a Senior Researcher with Instituto de Telecomunicações, Portugal. He has coordinated and participated as a Researcher in various national and international funded projects. He has supervised three concluded Ph.D. theses and several M.Sc. theses. He is the co-author of a book and more than 100 publications, including book chapters and papers in national and international journals and conferences. His current research is focused on deep learning-based techniques for point cloud coding and processing. His research interests include several topics related with image and video coding and processing, for different signal modalities and applications.

SIMONE MILANI (Member, IEEE) received the Laurea degree in telecommunication engineering and the Ph.D. degree in electronics and telecommunication engineering from the University of Padova, Padua, Italy, in 2002 and 2007, respectively. He was a Visiting Ph.D. Student with the University of California at Berkeley, Berkeley, CA, USA, in 2006. He was a Consultant with STMicroelectronics, Agrate, Italy. He was a Postdoctoral Researcher with the University of Udine, Udine, Italy, the University of Padova, and the Politecnico di Milano, Milan, Italy, from 2007 to 2013. From 2013 to 2020, he was an Assistant Professor with the Department of Information Engineering, University of Padova, where he is currently an Associate Professor. His research interests include digital signal processing, image and video coding, 3-D video processing and compression, joint source-channel coding, robust video transmission, distributed source coding, multiple description coding, and multimedia forensics.

FERNANDO PEREIRA (Fellow, IEEE) received the degree in electrical and computer engineering, and the M.Sc. and Ph.D. degrees from Instituto Superior Técnico, Technical University of Lisbon, in 1985, 1988, and 1991, respectively. He is currently with the Department of Electrical and Computers Engineering, Instituto Superior Técnico, Universidade de Lisboa, and Instituto de Telecomunicações, Lisbon, Portugal. He has been one of the key designers of the JPEG Pleno and JPEG AI standardization projects. He has contributed more than 300 papers in international journals, conferences, and workshops, and made several tens of invited talks and tutorials at conferences and workshops. His research interests include video analysis, representation, coding, description and adaptation, and advanced multimedia services. He was an IEEE Distinguished Lecturer, in 2005, and elected as an IEEE Fellow, in 2008, for "contributions to object-based digital video representation technologies and standards." He has been elected to serve on the IEEE Signal Processing Society Board of Governors in the Capacity of Member-at-Large, in 2012 and from 2014 to 2016. Since 2013, he has been a EURASIP Fellow for "contributions to digital video representation technologies and standards." He has been elected to serve on the European Signal Processing Society Board of Directors, from 2015 to 2018. He has been the IEEE Signal Processing Society Vice-President for Conferences, from 2018 to 2020, and the IEEE Signal Processing Society Awards Board Member, in 2017. He was a recipient of the 2023 Leo L. Beranek Meritorious Service Award. He was a recipient of the 2023 EURASIP Meritorious Service Award. Since 2015, he has been an IET Fellow. He has also held key leadership roles in numerous IEEE Signal Processing Society conferences and workshops, mostly notably serving twice as the ICIP Technical Chair in two continents, Hong Kong, in 2010, and Phoenix, in 2016. He has been the MPEG Requirements Subgroup Chair and is the JPEG Requirements Subgroup Chair. He has been an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, *IEEE Signal Processing Magazine*, and *EURASIP Journal on Image and Video Processing*, and an Area Editor of *Signal Processing: Image Communication* journal. From 2013 to 2015, he was the Editor-in-Chief of IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING.