Point Cloud Reconstruction From Truncated Geometry-Based Streams

Diogo C. Garcia, Andre L. Souto, Gustavo P. Sandri, Tomas M. Borges, Ricardo L. de Queiroz

***Abstract*—Geometry-based point cloud compression (G-PCC) has been rapidly evolving in the context of international standards. Despite the inherent scalability of octree-based geometry description, current G-PCC attribute compression techniques prevent full scalability for compressed point clouds. In this paper, we present a solution to add scalability to attributes compressed using the region-adaptive hierarchical transform (RAHT), enabling the reconstruction of the point cloud using only a portion of the original bitstream. Without the full geometry information, one cannot compute the weights in which the RAHT relies on to calculate its coefficients for further levels of detail. In order to overcome this problem, we propose a linear relationship approximation relating the downsampled point cloud to the truncated inverse RAHT coefficients at that same level. The linear relationship parameters are sent as side information. After truncating the bitstream at a point corresponding to a given octree level, we can, then, recreate the attributes at that level. Tests were carried out and results attest the good approximation quality of the proposed technique.**

***Index Terms*—Point cloud compression, geometry-based point cloud compression, G-PCC, truncated bitstream.**

I. Introduction

A Point cloud (PC) is a three-dimensional structure usually represented by a collection of points or volume elements (voxels) described by their geometry, given as the (x, y, z) coordinates of the points, and by the points attributes, which may be color, normal vectors and reflectance, among others. Recently, PC compression (PCC) research has intensified [1]–[6] and the Motion Picture Expert Group (MPEG) is in the process of finalizing two PCC standards [7], [8], based on purely geometrical techniques (G-PCC) or on existing video compression standards (V-PCC). PCs can be used to represent three-dimensional scenes, where the points describe the hull of objects therein, and are used in, for example, autonomous navigation [9], [10], heritage preservation [11], entertainment, and telepresence [12].

The wide range of applications may result in different requirements in terms of quality and resolution. Consider the example illustrated in Fig. 1, where a PC is combined with other three-dimensional elements to compose a scene. If the PC is placed near to the user's viewing point, the composition of the scene would require it to be rendered at a higher resolution when compared to the PC being placed further away. For instance, autonomous cars can anticipate data along their path, large heritage preservation sites can be more-easily rendered, and virtual reality users can be placed in large landscapes. On the user side, the PC can be downloaded and fully reconstructed and then downsampled to the required resolution. However, if the encoded bitstream could offer some degree of scalability, the user would be able to only download and process the required amount of the bitstream in order to reconstruct the PC at the desired resolution.

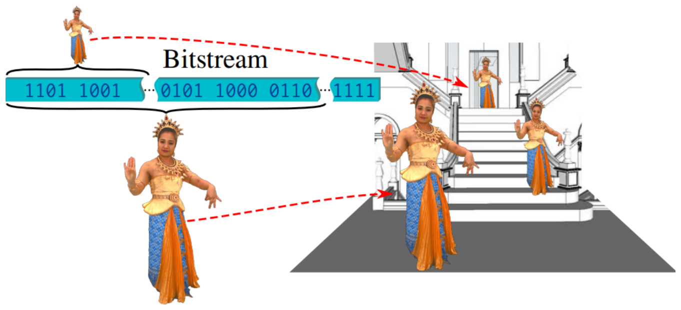


Figure 1: Reconstruction of the PC from a truncated bitstream. Subsequent portions of the bitstream are used to add details and resolution to the reconstructed PC.

Spatial scalability was one of the original requirements for G-PCC, so that the bitstream would have a layered structure for coarser approximations, and with each layer being used to predict the next one [13]. However, such a requirement is only partially addressed by G-PCC. The geometry representation already offers scalability by using octrees [14], wherein the PC geometry is inherently encoded as successive improvements on the geometry resolution, starting from a single block, successively dividing blocks into eight smaller blocks. The octree bitstream signals to the decoder which of these new blocks are occupied. For the attributes, G-PCC is currently based on the region-adaptive hierarchical transform (RAHT) [1], [2] or on the predicting/lifting transform [3], [15]. G-PCC offers partial spatial scalability, by using overlapping slices to encode regions of a PC at different fidelity levels, without inter-layer prediction [16].

In this paper, we propose a solution to reconstruct a PC from a truncated portion of the encoded bitstream containing the data from the octree-encoded geometry and RAHT-encoded

Work partially supported by CNPq under grants 88887.600000/2021-00 and 301647/2018-6.

D. Garcia is with the Gama Engineering College, University of Brasilia, Brazil, e-mail: diogogarcia@unb.br

A. L. Souto was with the Electrical Engineering Department at University of Brasilia, Brazil.

G. Sandri is with the Federal Institute of Brasília (IFB), e-mail: gustavo.sandri@ifb.edu.br

T. M. Borges is with the Electrical Engineering Department at University of Brasilia, Brazil, e-mails: tomas@divp.org.

R. de Queiroz is with the Department of Computer Science, University of Brasilia, Brazil, e-mail: queiroz@ieee.org

Digital Object Identifier: 10.14209/jcis.2023.9

attributes. The use of a truncated portion of the bitstream can save bandwidth in data transmission and avoid re-encoding the entire PC for each resolution.

II. Point cloud representation and coding

A. Point cloud geometry and attributes

A PC is usually represented by an unordered list that indicates the three-dimensional positions and corresponding attributes. A colored PC, for instance, can be represented in the RGB color space by the list $\{x_k, y_k, z_k, R_k, G_k, B_k\}$. Each position is unique and comprised in a cube of size $M \times M \times M$, where $M = 2^D$ and D is considered the depth of decomposition. Therefore $0 \leq x_k, y_k, z_k < M$.

The depth can be further decomposed by each dimension into $L = 3D$ *levels* of decomposition. Consider the two-dimensional PC depicted in the upper left region of Fig. 2, consisting of 8 voxels in 4×4 space ($M = 4$ and $D = 2$). Its representation is given by the following list:

$$\begin{bmatrix} 1, 0, a_0 \\ 2, 0, a_1 \\ 3, 1, a_2 \\ 1, 2, a_3 \\ 3, 2, a_4 \\ 1, 3, a_5 \\ 2, 3, a_6 \\ 3, 3, a_7 \end{bmatrix} \tag{1}$$

The geometry (spatial position of the voxels) and the attribute spaces in a PC present different characteristics, so that they are usually separately encoded. In this paper, geometry compression is taken as a given, so that we focus on the attribute compression based on the RAHT.

B. Region-adaptive Hierarchical Transform

RAHT is a variation of the Haar transform, and it uses attribute values of a node at a lower level of the octree to predict the attributes of the nodes at the next level. For simplicity, assume there is just one attribute to be encoded per voxel (point), be it a color component, reflectance or else. Neighboring voxels are paired and transformed into low- and high-pass coefficients. The low-pass ones are further combined at each step with neighboring low-pass coefficients, repeating the process, until the entire space is traversed.

At a given level, two low-pass coefficients about to be paired and transformed represent averages over different numbers of voxels, which render different weight values in the transformation matrix. Each weight indicates the number of voxels that were actually involved to generate that low-pass coefficient.

Two neighbor low-pass coefficients at level $\ell + 1$, F' and F'', are combined through an orthogonal transform to form a low- and a high-pass coefficient, F and G, at level ℓ,

$$\begin{bmatrix} F \\ G \end{bmatrix} = \mathbf{T} \begin{bmatrix} F' \\ F'' \end{bmatrix}, \tag{2}$$

where,

$$\mathbf{T} = \begin{bmatrix} a & b \\ -b & a \end{bmatrix}. \tag{3}$$

Let w' and w'' be the respective weights of the input coefficients, then

$$a = \sqrt{\frac{w'}{w' + w''}} \quad \text{and} \quad b = \sqrt{\frac{w''}{w' + w''}}. \tag{4}$$

Note that $a^2 + b^2 = 1$, $\mathbf{T}$ is orthogonal and the above equation can be inverted using $\mathbf{T}^{-1} = \mathbf{T}^T$.

Since RAHT is applied recursively, we need to keep track of indices for level, and spatial coordinates. So, we let the two neighbor low-pass coefficients at level $\ell + 1$, be $F(\ell + 1, 2x, y, z)$ and $F(\ell + 1, 2x + 1, y, z)$, which are combined through $\mathbf{T}$ to form a low- and a high-pass coefficient, $F(\ell, x, y, z)$ and $G(\ell, x, y, z)$, at level ℓ. Let $w(\ell+1, 2x, y, z)$ and $w(\ell+1, 2x+1, y, z)$ be the respective weights of the input coefficients. We then rewrite the transform as:

$$\begin{bmatrix} F(\ell, x, y, z) \\ G(\ell, x, y, z) \end{bmatrix} = \mathbf{T} \begin{bmatrix} F(\ell + 1, 2x, y, z) \\ F(\ell + 1, 2x + 1, y, z) \end{bmatrix}. \tag{5}$$

As a toy example, reconsider the two-dimensional PC shown in Fig. 2, where coefficients were labeled $\{a_i\}$ instead of $F(\ell, x, y, z)$ and $G(\ell, x, y, z)$ to make the image more readable. It can be seen that neighboring voxels are progressively grouped in pairs: for instance, coefficients a_6 and a_7 are used to create coefficients b_0 and c_0.Coefficients h_0 and i_0, g_0, g_1, e_0, e_1, e_2 and c_0 are then quantized and entropy-coded, and the process is reversed at the decoder side.

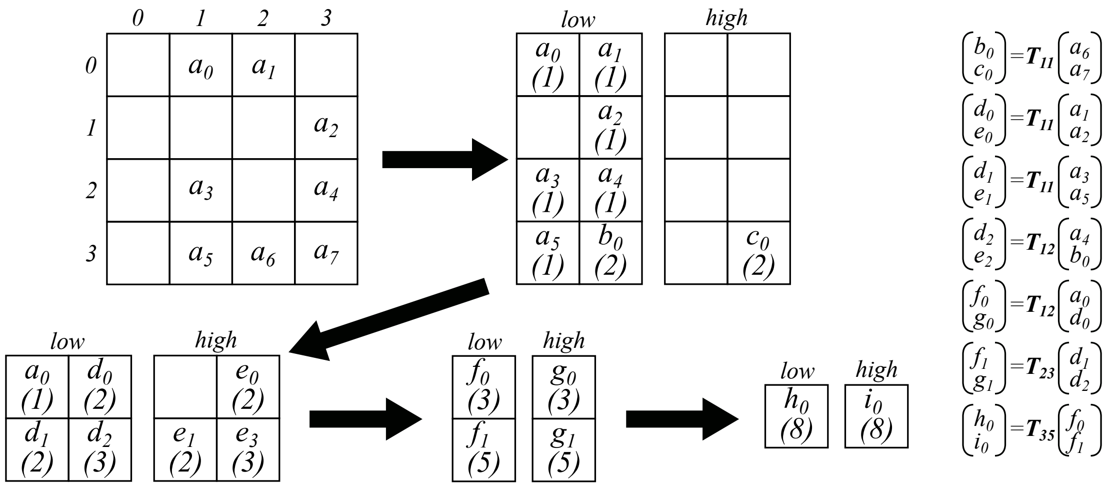
C. Truncated inverse RAHT

Let the PC have N occupied voxels, laid in a cubic grid of $2^D \times 2^D \times 2^D$ voxels. The set of $\{G(\ell, x, y, z)\}$ are the $N - 1$ RAHT coefficients which are encoded along with the overall DC $F(0, 0, 0, 0)$. The forward RAHT starts from the voxels (tree leaves at the L-th level, or $\{F(L, 2x, y, z)\}$) generating low-pass coefficients which are laid in a voxel grid of level $L - 1$ ($\{F(L - 1, x, y, z)\}$). The process is recurred until we traverse all the way to the tree root at level 0, generating the overall DC value $F(0, 0, 0, 0)$ for the entire PC.

At the decoder, we start with $F(0, 0, 0, 0)$ and $G(0, 0, 0, 0)$, along with the weights $w(1, 0, 0, 0)$ and $w(1, 0, 0, 1)$, to calculate $F(1, 0, 0, 0)$ and $F(1, 0, 0, 1)$. From the start we need the weights for the whole tree. If we truncate the tree at level $L - K$ and know the geometry up to level $L - K$ we would be unable to reconstruct the set of $\{F(L - K, x, y, z)\}$ because the weights for all levels below $L - K$ still depend on the geometry at levels above $L - K$. Hence, RAHT is not scalable.

III. Proposed solution

Assume we have truncated data composed of the geometry information up to level $L - K$ and all the RAHT coefficients up to level $L - K$, i.e. $F(0, 0, 0, 0)$ and $\{G(\ell, x, y, z),\ 0 \leq \ell < L - K\}$. If we decode the data as an $L - K$-level PC, the lower resolution geometry would provide incorrect weights $\{w'(\ell, x, y, z)\}$ for the given coefficients, which were computed using the correct weights $\{w(\ell, x, y, z)\}$. For example, all $w'(L - K, x, y, z) = 1$ since it is the last level of the truncated PC, which is most definitely not the case for the original PC with K further levels.



$\begin{pmatrix}b_0\\c_0\end{pmatrix}=T_{11}\begin{pmatrix}a_6\\a_7\end{pmatrix}$
$\begin{pmatrix}d_0\\e_0\end{pmatrix}=T_{11}\begin{pmatrix}a_1\\a_2\end{pmatrix}$
$\begin{pmatrix}d_1\\e_1\end{pmatrix}=T_{11}\begin{pmatrix}a_3\\a_5\end{pmatrix}$
$\begin{pmatrix}d_2\\e_2\end{pmatrix}=T_{12}\begin{pmatrix}a_4\\b_0\end{pmatrix}$
$\begin{pmatrix}f_0\\g_0\end{pmatrix}=T_{12}\begin{pmatrix}a_0\\d_0\end{pmatrix}$
$\begin{pmatrix}f_1\\g_1\end{pmatrix}=T_{23}\begin{pmatrix}d_1\\d_2\end{pmatrix}$
$\begin{pmatrix}h_0\\i_0\end{pmatrix}=T_{35}\begin{pmatrix}f_0\\f_1\end{pmatrix}$
low
high

Figure 2: Two-dimensional example to illustrate the grouping to form the region-adaptive hierarchical transform (RAHT): a voxel map is progressively transformed into low- and high-pass maps by grouping horizontal neighbors, and then vertical neighbors. The transforms are indicated in the right side of the image, the voxel weights are indicated along with the labels.

Assume Eq. (5) is inverted with incorrect weights $\{w'(\ell,x,y,z)\}$, using the transpose of say $\mathbf{T}' = \left[\begin{smallmatrix} a' & b' \\ -b' & a' \end{smallmatrix}\right]$ instead of $\mathbf{T}$, where a' and b' stem from incorrect weights, and let $\hat{F}(\ell+1,2x,y,z)$ and $\hat{F}(\ell+1,2x+1,y,z)$ be the incorrectly inverted coefficients. Then,

$$\begin{bmatrix} \hat{F}(\ell+1,2x,y,z) \\ \hat{F}(\ell+1,2x+1,y,z) \end{bmatrix} = (\mathbf{T}')^T \mathbf{T} \begin{bmatrix} F(\ell+1,2x,y,z) \\ F(\ell+1,2x+1,y,z) \end{bmatrix}$$

$$= \begin{bmatrix} a' & -b' \\ b' & a' \end{bmatrix} \begin{bmatrix} a & b \\ -b & a \end{bmatrix} \begin{bmatrix} F(\ell+1,2x,y,z) \\ F(\ell+1,2x+1,y,z) \end{bmatrix}$$

$$= \begin{bmatrix} aa'+bb' & a'b-ab' \\ ab'-a'b & aa'+bb' \end{bmatrix} \begin{bmatrix} F(\ell+1,2x,y,z) \\ F(\ell+1,2x+1,y,z) \end{bmatrix}$$

$$= p \begin{bmatrix} F(\ell+1,2x,y,z) \\ F(\ell+1,2x+1,y,z) \end{bmatrix} + q \begin{bmatrix} F(\ell+1,2x+1,y,z) \\ -F(\ell+1,2x,y,z) \end{bmatrix}, \tag{6}$$

where $p = aa'+bb'$ and $q = a'b-ab'$. Since all transform entries are positive, it is clear that $q < p$. If one assumes that neighbor color attributes are similar, such that $F(\ell+1,2x+1,y,z) = F(\ell+1,2x,y,z)+\epsilon$, then $\hat{F}(\ell+1,2x,y,z) = (p+q)F(\ell+1,2x,y,z)+q\epsilon \approx \alpha F(\ell+1,2x,y,z)$, for some scalar α. Such an error is propagated through the RAHT tree from the root to level $L-K$ and we expect such a linear relation in between incorrectly and correctly reconstructed coefficients.

Let $X(\ell,x,y,z)$ be the voxel attribute at the same position of $F(\ell,x,y,z)$ obtained by downsampling the L-level PC from full-resolution voxels $\{F(L,x,y,z)\}$ down to level ℓ. From the above discussion, we expect a trend to a linear approximation from one variable to the other as

$$X(\ell,x,y,z) \approx \alpha_\ell \hat{F}(\ell,x,y,z) + \beta_\ell, \tag{7}$$

where the β_ℓ term was added to better adjust to the model's imperfections. Figure 3 shows the relationship among pairs $X(\ell,x,y,z)$ and $\hat{F}(\ell,x,y,z)$ for different conditions for a couple of PCs encoded at particular bit-rates specified in MPEG's G-PCC common test conditions (CTC). This relation pattern has held for all PCs and conditions we have tested in Sec. IV. We, then, calculate, at the encoder side, not only the $\{F(\ell,x,y,z)\}$ and $\{G(\ell,x,y,z)\}$ for all levels $0 \le \ell < L$, but we also calculate $\{\hat{F}(\ell,x,y,z)\}$ and $\{X(\ell,x,y,z)\}$ for $0 \le \ell \le L-K$. The encoder computes α_ℓ and β_ℓ using least-squares for all levels $0 \le \ell \le L-K$. The $2(L-K+1)$ parameters are encoded and sent as side information to the decoder, which is generally a very small penalty.

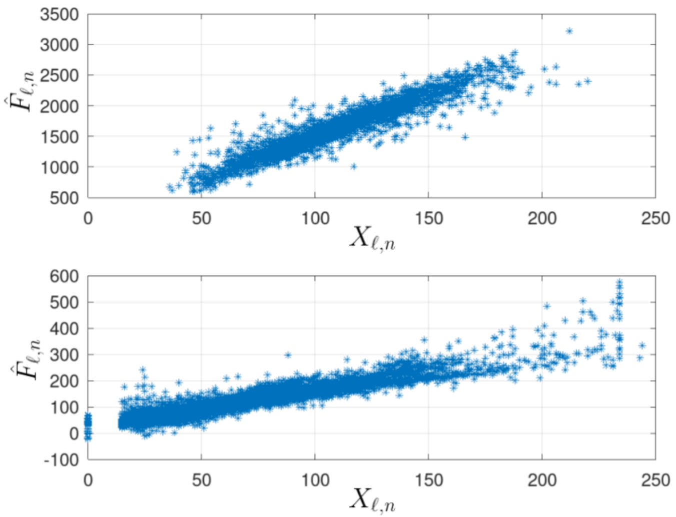


Figure 3: Relationship between attribute values $X(\ell,x,y,z)$ of downsampled point clouds and truncated inverse RAHT coefficients $\hat{F}(\ell,x,y,z)$, for sequences *Longdress_vox_10*, frame 1300, CTC rate 1, downsampled by 16 (top plot), and *Ford_03_q1mm*, frame 0100, CTC rate 6, downsampled by 512 (bottom plot).

It is important to note that the G-PCC standard does not interlace the geometry and attribute information on a level basis, as each stream is separately included in the bitstream.

Table I: Summary of information of the tested PCs.

	Point cloud	L	# voxels	Attributes
(a)	Longdress_vox_10_1051 [17]	10	765821	RGB
(b)	Loot_vox_10_1000 [17]	10	784142	RGB
(c)	Redandblack_vox_10_1450 [17]	10	729133	RGB
(d)	Soldier_vox_10_0536 [17]	10	1059810	RGB
(e)	ford_01_q1mm-0100	18	80265	Reflec.
(f)	ford_02_q1mm-0100	18	79882	Reflec.
(g)	ford_03_q1mm-0200	18	87590	Reflec.
(h)	qnxadas-junction-approach_000001 [18]	18	31279	Reflec.
(i)	qnxadas-junction-exit_000001 [18]	18	26842	Reflec.
(j)	qnxadas-motorway-join_000001 [18]	18	29424	Reflec.
(k)	qnxadas-navigating-bends_000001 [18]	18	26041	Reflec.

In order to allow for the desired scalability in this standard, the proposed method expects stream interlacing on a level basis. The proposed G-PCC decoder must be able to truncate the geometry and attribute information until the desired downsampling level, inverse transform the truncated RAHT coefficients, and then apply the first-order approximation. Figs. 4 and 5 illustrate our method.

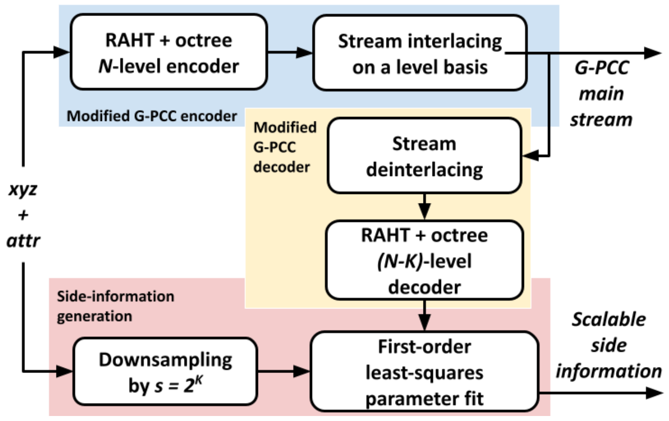

RAHT + octree N-level encoder
Stream interlacing on a level basis
Modified G-PCC encoder
G-PCC main stream
Modified G-PCC decoder
Stream deinterlacing
RAHT + octree (N-K)-level decoder
xyz + attr
Side-information generation
Downsampling by s = 2^K
First-order least-squares parameter fit
Scalable side information

Figure 4: Truncated geometry-based encoder.

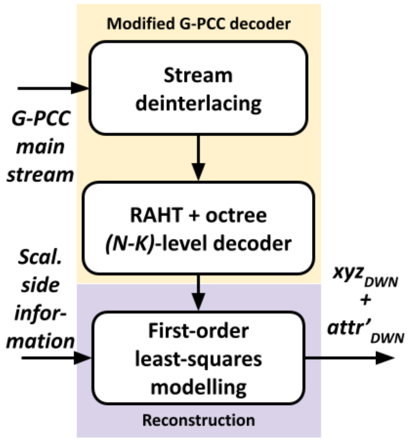


Figure 5: Truncated geometry-based decoder.

IV. Experimental results

In order to test the proposed solution, we selected PCs with different densities, levels, and attributes (RGB color or reflectance), as described in Table I. We tested different truncation points, corresponding to removing K octree levels from the PC, i.e., downsampling by a factor of $s = 2^K$. Our distortion metric computes Y-PSNR [19], [20], i.e. the PSNR of the luminance channel or reflectance. We start from a compressed bitstream and compare the resulting PC to the original PC after resolution reduction (downsampling) of K levels. The downsampling is carried out by removing octree levels of the geometry and by averaging the attributes at each level. Two methods are compared: the one obtained by full-stream reconstruction (*full-stream*) followed by down-sampling of K levels, against our algorithm to reconstruct $L - K$ levels of the PC from the truncated bitstream (*proposed*)[1]. The rate was calculated as the number of bits actually transmitted, i.e. the full bitstream size for *full-stream* method against the truncated bitstream size for the *proposed* one. Encoding and decoding, in both cases, were based on G-PCC Test Model (TMC13) version 12.0 [21], [22], since we are enhancing the codec's capabilities, instead of proposing a different codec. The encoding bit-rates were set according to the six target-rates from G-PCC's CTC [23].

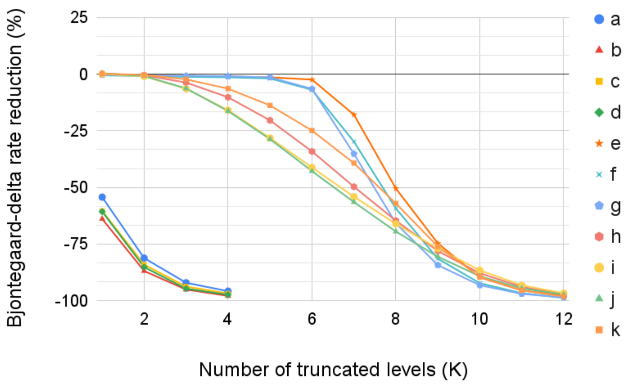


Figure 6: BD-rate reduction for several values of K (truncated levels) for several PCs, which are labeled as in Table I.

For a given PC, as we set K we can vary the encoding rate obtaining rate-distortion (RD) curves for each method (*full-stream* vs. *proposed*) from which one can calculate the Bjøntegaard-delta (BD) [24] rate reduction. Figure 6 shows the BD-rate achieved for many PCs at different truncation points, indicating that the proposed method not only reduces the rate, but also offers a better objective quality at said rate. For the sequences with reflectances as attributes (specially sequences "e" through "g"), there is little to gain at smaller values of K, because they are much sparser than the other sequences. For sequences "a" through "d", which are denser, K values higher than 4 are not very useful (see Fig. 7). Although impressive results were obtained, it is important to mention that reductions come with a caveat, since they do not capture the fact that a truncated bitstream cannot achieve the same distortion levels as full-stream decoding. This is illustrated in the in the RD curves in Fig. 7, where each point corresponds to a CTC rate. For example, for $K = 1$, the proposed method PSNR's is

[1]To the best of our knowledge, there is no other openly available solution.

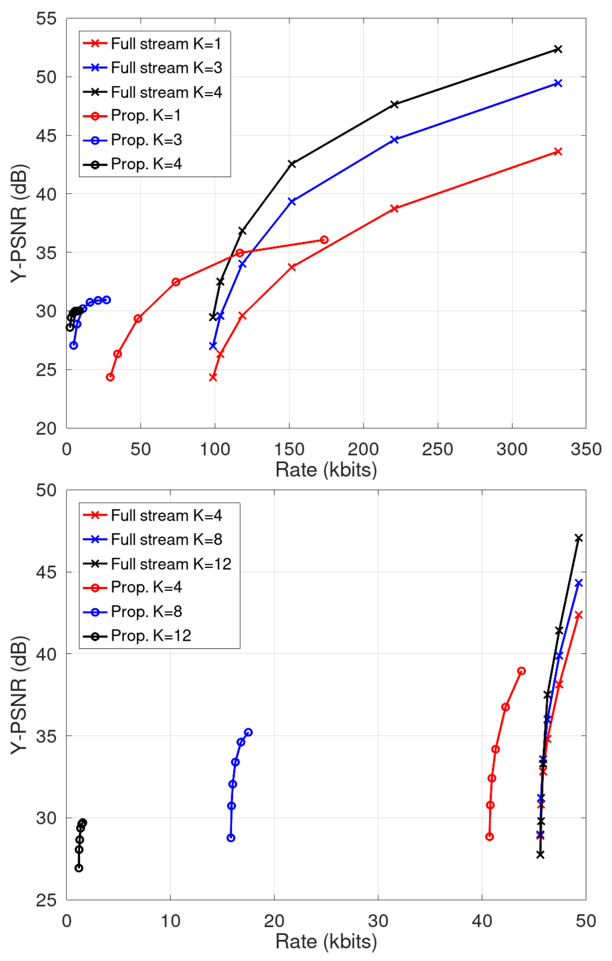


Figure 7: Representative RD curves for both methods at different levels of downsampling. Top plot is for *Longdress_vox10_1300* and the bottom one is for *qnxadas-junction-approach*.

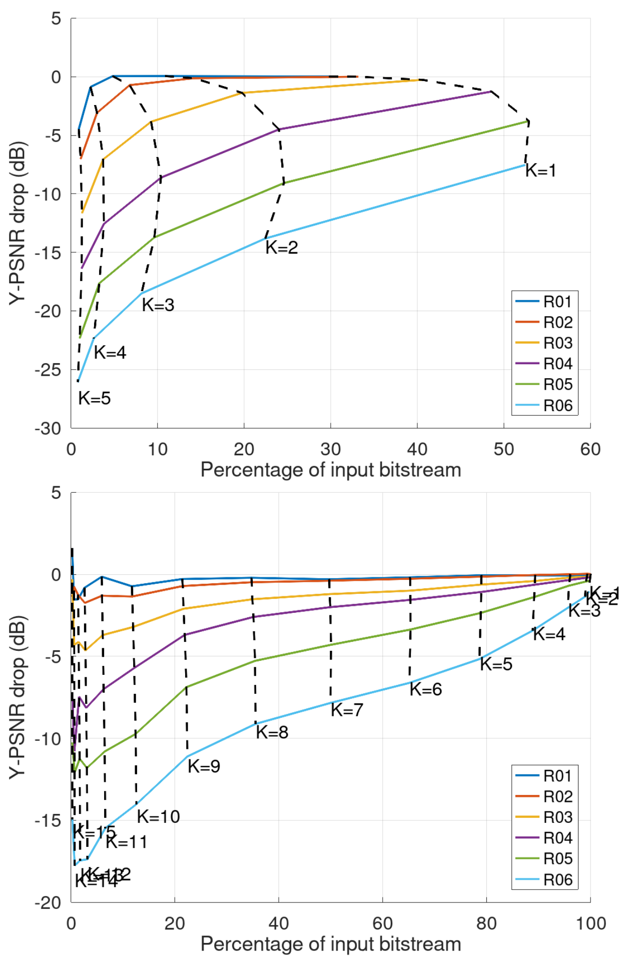


Figure 8: Bit-rate reduction and corresponding Y-PSNR drop for PCs *Longdress_vox10_1300* (top) and *qnxadas-junction-approach* (bottom), at different downsampling factors and encoding bit-rates. The dotted curves indicate different rates for the same K.

less than 36 dB for sequence *Longdress_vox10_1300*, and for $K = 4$ there is not much of a difference between CTC rates, indicating that higher K values become impractical for this sequence.

In yet another way we can present results, Fig. 8 relates the percentage of the bitstream that is used against the drop in Y-PSNR (dB) for the *proposed* method against the *full-stream* one. In these, one can see the curves for either varying the encoding rate for a given value of s (or K), or vice-versa. In all these curves one may appreciate there are sweet spots where the large reduction in rates and small reduction in quality may be of interest to given application developers. For instance, it can be seen that there is a direct relationship between the Y-PSNR drop and the CTC rate, as G-PCC coding already highly degrades the attributes at smaller rates (R01), and this fact can be explored when choosing the CTC rate and K values.

The *proposed* method has a performance trade-off, as the calculation of least-squares parameters α_ℓ and β_ℓ for each octree level increases the encoder's complexity. According to the application, one may choose to calculate parameters for all levels, or just the most significant ones. Sequence *ford_01_q1mm-0100*, for example, may be downsampled by 2^7 before the number of occupied voxels significantly reduces, so that calculating $\alpha_1 - \alpha_6$ and $\beta_1 - \beta_6$ is not very practical (see Fig. 6).

Figs. 9 and 10 offer subjective results for sequences *Longdress_vox10_1300*, *Loot_vox10_1000* and *Redandblack_vox10_1450* after downsampling by 2 and 8, respectively. In both Figs., the first, second and third lines present results after (a) the proposed method, (b) full-stream G-PCC coding and (c) no coding, and for the first two methods, rate 6 of the G-PCC's CTC was applied. Basically, even though the proposed method offers substantial rate reduction over the full-stream method, it is very hard to tell the difference between the decoded versions.

V. Conclusions

We have proposed an algorithm to reconstruct a G-PCC-coded PC from a partial (truncated) bitstream. Based on a linear relationship approximation relating the downsampled

PC to the truncated inverse RAHT coefficients at that same level, the downsampled PC can be estimated, enabling lower-resolution "previews" without full-stream reconstruction. This is no substitute for a truly scalable coder, but it can be useful to save transmission bandwidth when, for example, rendering objects farther away for autonomous cars or virtual reality applications. A number of tests were carried out to demonstrate the quality of the reconstruction. In many situations, one can save a large percentage of the original bitrate at a small distortion penalty, at the cost of higher encoder complexity and of limiting the maximum achievable quality.

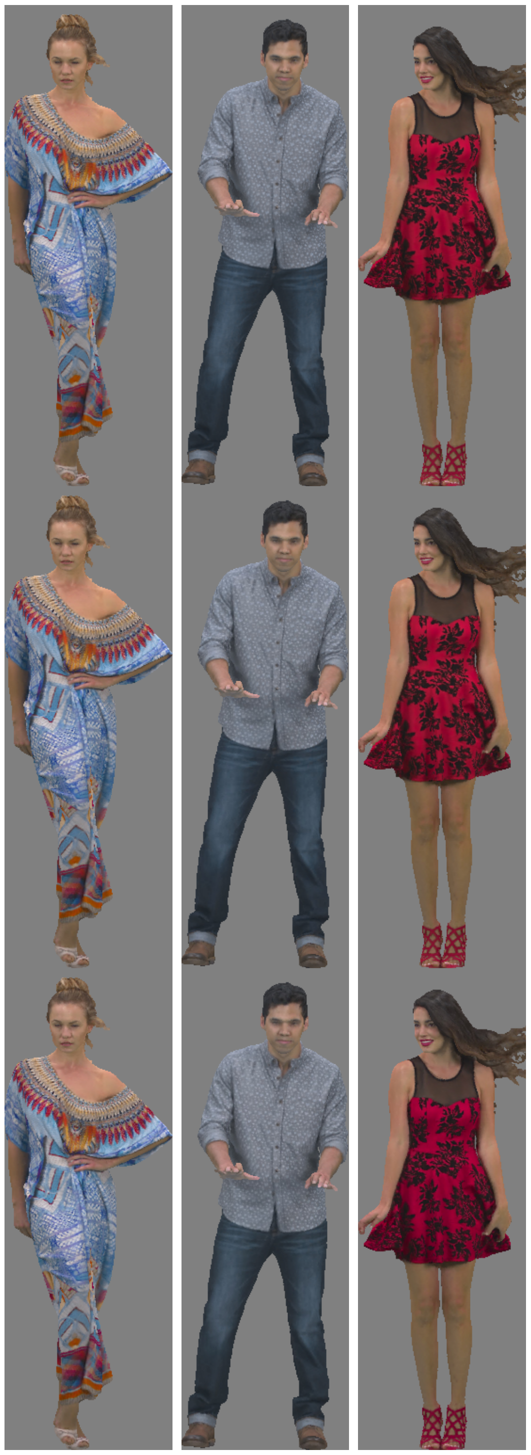


Figure 9: Subjective results for sequences *Long-dress_vox10_1300*, *Loot_vox10_1000* and *Redand-black_vox10_1450*. The first, second and third lines present results after downsampling by 2 ($K = 1$) of: (a) the proposed method, (b) full-stream G-PCC coding and (c) no coding. For the first two methods, rate 6 of the G-PCC's CTC was applied.

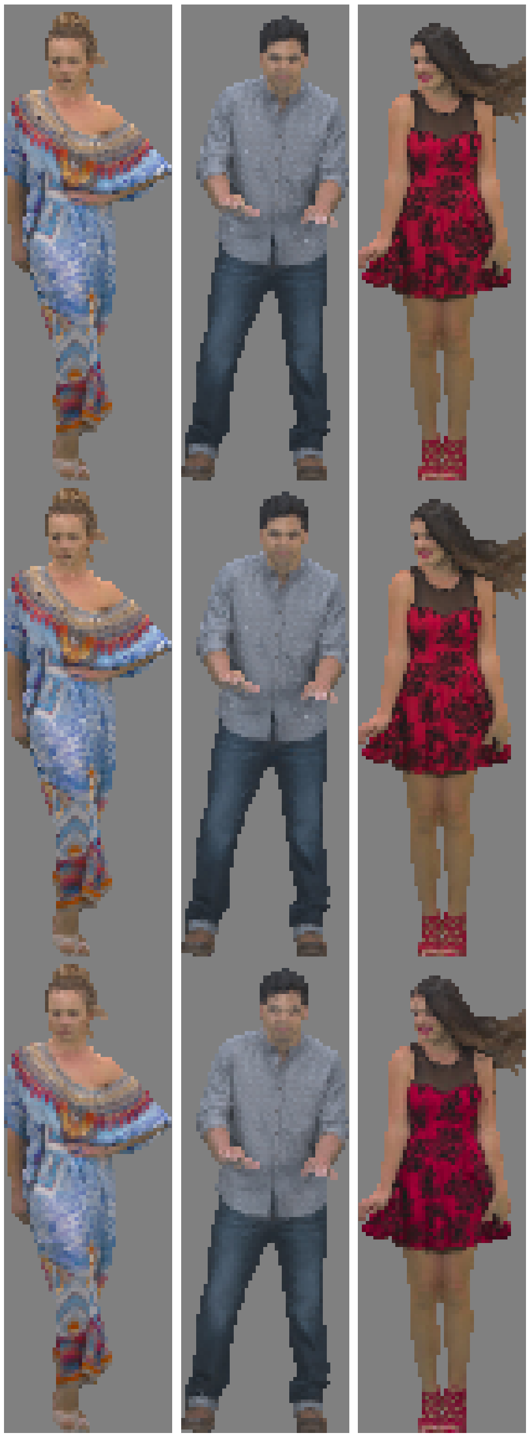


Figure 10: Subjective results for sequences *Longdress_vox10_1300*, *Loot_vox10_1000* and *Redandblack_vox10_1450*. The first, second and third lines present results after downsampling by 8 ($K = 3$) of: (a) the proposed method, (b) full-stream G-PCC coding and (c) no coding. For the first two methods, rate 6 of the G-PCC's CTC was applied.

References

[1] R. L. de Queiroz and P. A. Chou, "Compression of 3D point clouds using a region-adaptive hierarchical transform," *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3947–3956, Aug. 2016.

[2] S. Lasserre and D. Flynn, "On an improvement of RAHT to exploit attribute correlation," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Tech. Rep. m47378, Mar. 2019.

[3] K. Mammou, A. Tourapis, J. Kim, F. Robinet, V. Valentin, and Y. Su, "Proposal for improved lossy compression in TMC1," ISO/IEC MPEG JTC1/SC29/WG11, San Diego, US, Input contribution m42640, Apr. 2018.

[4] G. Sandri, R. D. Queiroz, and P. A. Chou, "Compression of plenoptic point clouds using the region-adaptive hierarchical transform," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, oct 2018. doi: 10.1109/icip.2018.8451367

[5] X. Zhang, P. A. Chou, M.-T. Sun, M. Tang, S. Wang, S. Ma, and W. Gao, "A framework for surface light field compression," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, oct 2018. doi: 10.1109/icip.2018.8451269

[6] R. L. de Queiroz, C. Dorea, D. C. Garcia, R. U. Ferreira, D. R. Freitas, R. Higa, I. Seidel, and V. Testoni, "Differential plenoptic point cloud coding for V-PCC," ISO/IEC MPEG JTC1/SC29/WG11, Tech. Rep. input document WG7M55145, Oct. 2020.

[7] 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 Transactions on Signal and Information Processing*, vol. 9, p. e13, 2020.

[8] S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuća, S. Lasserre, Z. Li, J. Llach, K. Mammou, R. Mekuria, O. Nakagami, E. Siahaan, A. Tabatabai, A. M. Tourapis, and V. Zakharchenko, "Emerging MPEG standards for point cloud compression," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 133–148, 2019.

[9] F. Hidalgo, "ORBSLAM2 and point cloud processing towards autonomous underwater robot navigation," in *Global Oceans 2020: Singapore – U.S. Gulf Coast*. IEEE, oct 2020. doi: 10.1109/ieeeconf38699.2020.9389096

[10] M. Yoshioka, N. Suganuma, K. Yoneda, and M. Aldibaja, "Real-time object classification for autonomous vehicle using LIDAR," in *2017 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*. IEEE, nov 2017. doi: 10.1109/iciibms.2017.8279696

[11] D. Groux-Leclet, J. Lentremy, G. Caron, and E. Mouaddib, "Efficient reproduction of heritage buildings: a new way to exploit 3d point cloud slicing : An example with the amiens gothic cathedral," in *2018 3rd Digital Heritage International Congress (DigitalHERITAGE) held jointly with 2018 24th International Conference on Virtual Systems & Multimedia (VSMM 2018)*. IEEE, oct 2018. doi: 10.1109/digitalheritage.2018.8810106

[12] S. Orts-Escolano, C. Rhemann, S. Fanello, W. Chang, A. Kowdle, Y. Degtyarev, D. Kim, P. L. Davidson, S. Khamis, M. Dou, V. Tankovich, C. Loop, Q. Cai, P. A. Chou, S. Mennicken, J. Valentin, V. Pradeep, S. Wang, S. B. Kang, P. Kohli, Y. Lutchyn, C. Keskin, and S. Izadi, "Holoportation: Virtual 3d teleportation in real-time," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, ser. UIST '16. New York, NY, USA: Association for Computing Machinery, 2016. ISBN 9781450341899 p. 741–754.

[13] 3DG, "PCC Requirements," ISO/IEC MPEG JTC1/SC29/WG11, Gwangju, KR, Approved WG 11 document W17353, Jan. 2018.

[14] D. Meagher, "Geometric modeling using octree encoding," *Computer Graphics and Image Processing*, vol. 19, no. 2, pp. 129–147, Jun 1982.

[15] MPEG 3DG, "PCC Test Model Category 3 v0," ISO/IEC MPEG JTC1/SC29/WG11, Macau, CN, Approved N17249, Oct. 2017.

[16] MPEG 3DG, "Status on addressing the g-pcc requirements," ISO/IEC MPEG JTC1/SC29/WG11, virtual, MPEG Requirements N00052, Jan. 2021.

[17] E. d'Eon, B. Harrison, T. Myers, and P. A. Chou, "8i Voxelized Full Bodies, version 2 – A Voxelized Point Cloud Dataset," ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG), Geneva, input document m40059/M74006, January 2017.

[18] D. Flynn, S. Lasserre, and G. Martin-Cocher, "PCC Cat3 test sequences from BlackBerry—QNX," ISO/IEC MPEG JTC1/SC29/WG11, Ljubljana, input document m23647, Jul. 2018.

[19] MPEG 3DG, "MPEG 3DG and Requirements: Call for proposals for point cloud compression v2," ISO/IEC MPEG JTC1/SC29/WG11, Hobart, AU, Approved WG 11 document N16763, April 2017.

[20] P. A. Chou and M. Krivokuca, "Concerns about Objective Metrics for Point Cloud Compression," ISO/IEC MPEG JTC 1/SC 29/WG 11, Macau, Input document to AhG on Point Cloud Compression m41809, October 2017.

[21] 3DG, "G-PCC codec description v9," ISO/IEC MPEG JTC1/SC29/WG7, Online, Approved WG 7 document N00005, Nov. 2020.

[22] 3DG, "G-PCC test model v12 user manual," ISO/IEC MPEG JTC1/SC29/WG7, Online, Approved WG 7 document N00011, Oct. 2020.

[23] 3DG, "Common Test Conditions for G-PCC," ISO/IEC MPEG JTC1/SC29/WG11, Online, Approved WG 11 document N19584, July 2020.

[24] G. Bjontegaard, "Calculation of average PSNR differences between RD curves," ITU, Tech. Rep. VCEG-M33, Apr. 2001.