Adaptive Entropy Coding of Graph Transform Coefficients for Point Cloud Attribute Compression

Thibaut Meyer, Dominik Mehlem, and Christian Rohlfing
Institut für Nachrichtentechnik, RWTH Aachen University, Germany
thibaut.meyer@ient.rwth-aachen.de

***Abstract*—A point cloud's attributes constitutes most of its information content. This is why their efficient compression is of great importance when designing a compression scheme. In this paper, the entropy coding stage of an existing compression method is replaced using Context-based Adaptive Binary Arithmetic Coding (CABAC), which is already largely used for video compression applications. Additionally, the DC transform coefficients are quantized with Differential Pulse Code Modulation (DPCM), yielding better performance at lower bit rates. By using predictive coding and coding techniques adapted to the signal's characteristics, Bjøntegaard Delta rate savings of up to 31.38% are observable in comparison to previously used entropy coding methods.**

***Index Terms*—3D point cloud compression, color attributes, Graph Fourier Transform, entropy coding, CABAC**

I. Introduction

The number of real world applications that consume 3D data, such as point clouds, as well as the availability of 3D capturing technology are both steadily increasing. In their raw and uncompressed form, point clouds, because of their irregular structure, are extremely data intensive. This sparked interest in developing methods that enable the efficient representation, storage and transmission of point clouds. Point clouds are comprised of points that carry geometry and optionally attribute data. Some methods compress the point clouds' geometry and attributes simultaneously, such as novel neural network-based methods, e.g. [1]. However, another kind of popular end-to-end point cloud compression schemes transmits the geometry to the decoder and the attribute data independently from one another. G-PCC [2], a point cloud coding standard developed by the Moving Pictures Expert Group (MPEG), decorrelates the attributes' signal using transform coding [3]. Other techniques used to derive an efficient representation of the attributes are based on the Graph Fourier Transform (GFT) [4], such as originally in [5], or more recently as in [6] or [7].

The entropy coding of the resulting GFT coefficients has been mostly realized using adaptive Run-Length Golomb-Rice (RLGR) coding [8]. As an alternative, we consider the latest variant of Context-based Adaptive Binary Arithmetic Coding (CABAC), that was made available with the most recent video standard Versatile Video Coding (VVC) [9]. We show that changes to the entropy coding scheme can lead to improvements that reduce the total necessary bitrate of the compression pipeline. Furthermore, to enhance the compression efficiency at lower bitrates, the GFT DC coefficients are quantized by Differential Pulse Code Modulation (DPCM), which predicts the current DC value given the previously coded one. This paper extends [5] and [7] with the following contributions:

- The entropy coding of the quantized GFT coefficients is conducted by Context-based Adaptive Binary Arithmetic Coding (CABAC) [10], replacing RLGR.
- Quantization of the GFT DC coefficients using Differential Pulse Code Modulation (DPCM).

This paper is structured as follows: The existing point cloud attribute coding methods are summarized in Section II. CABAC and its recent improvements are presented in Section III. Section IV explains in detail the new entropy coding techniques that represent this paper's contributions. Evaluation experiments and finally conclusions are discussed in Section V and VI, respectively.

II. Attribute Coding using Graph Fourier Transform

A point cloud consist of its geometry V and attributes A, which are distributed over its collection of N points. In order to use the Graph Fourier Transform (GFT), a set of points needs to be transformed into a graph, from which we can derive the GFT transformation matrix. This is computationally expensive and thus not practicable for big graphs representing an entire typical point cloud. This is why an input point cloud is split into L blocks using one of various partitioning techniques. In this work, k-means is used for partitioning, as proposed by [7].

Each resulting partition, which contains N_l points and is represented by its geometry V_l and attributes A_l, is turned into a graph $\mathcal{G}_l$ in the same manner as proposed in [5]. Given $U^\top$, the GFT's transform matrix, a signal embedded on $\mathcal{G}$'s nodes is projected to the GFT domain as

$$C_l = U^\top A_l, \tag{1}$$

where the signal represents here the attributes A_l of a single block. The GFT's transform matrix $U^\top$ is computed given V_l, as detailed in [11] and summarized in e.g. [5]. Using this transform for each block, the attributes A_l yield the GFT coefficients C_l, which are then quantized as $\tilde{C}_l = Q(C_l)$, and input to the entropy coding stage, detailed in Section IV. Note that the GFT coefficients C_l of a given block l are comprised of one DC coefficient $c_{l,0}$ and one or more AC coefficients $c_{l,1...N_l}$ if the block possesses more than one point ($N_l > 1$).

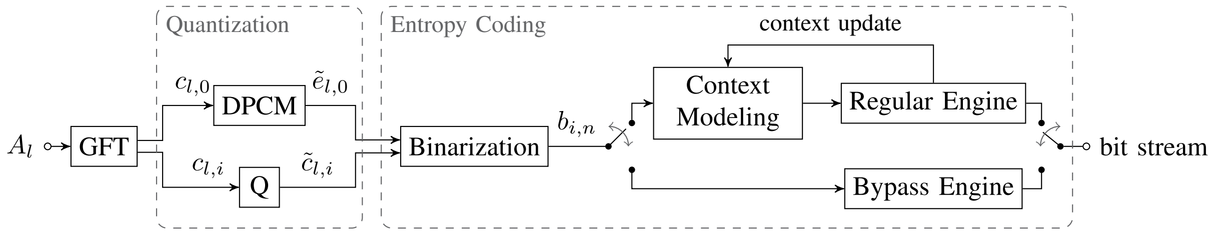

Fig. 1: Proposed point cloud attribute compression scheme with separate quantization of DC ($c_{l,0}$) and AC ($c_{l,i}$, $i > 0$) coefficients. CABAC is used as the entropy coding method.

They follow a geometric distribution and so do the quantized levels $\tilde{C}_l$.

III. Context-based Adaptive Binary Arithmetic Coding (CABAC)

In this section, Context-based Adaptive Binary Arithmetic Coding (CABAC) [10] in its most recent version (as used in VVC) is briefly summarized. For a more detailed discussion, refer to e.g. [12]. As depicted in the entropy coding block in Fig. 1, CABAC consists of four steps: First, integer symbols are binarized, yielding binary codewords (sometimes referred to as bin-strings). Second, given information extracted from previously (de-)coded data, a probability model, also called context, is assigned to each binary value (bin). The estimated probability is then further used in the third step, the binary arithmetic coding (BAC) engine. Alternatively, the context modeling and BAC can be skipped for given bins by utilizing the faster bypass mode, which codes them with fixed probability values of $p = 0.5$.

CABAC stems from Advanced Video Coding (AVC) [13] and was used in the same variant in AVC's successor High Efficiency Video Coding (HEVC) [14]. During the standardization of VVC [9], some of the core functionality of CABAC was further improved, namely the probability estimation for each context model: While the CABAC variant for AVC and HEVC uses a finite state machine to model the symbol probability, the recent variant proposed for VVC utilizes two probability estimators in parallel [12]. Both probability estimates are arithmetically derived, using two different adaptation rates: α_0 and $\alpha_1 < \alpha_0$, which allows for the simultaneous estimation of both faster and slower varying probabilities respectively. Given the two estimates, the final estimate is chosen as their average value. This multi-hypothesis probability estimation procedure is detailed in [15].

Binarization

Since the arithmetic engine of CABAC is restricted to code only binary values, integer inputs are binarized in a first step. Typical choices for binarization schemes are prefix-free codes such as fixed-length (FL), truncated unary (TU), or exponential-Golomb codes of order k (EG-k) [16]: The FL binarization is the binary representation of the to-be-binarized non-negative integer v with a fixed number of bits n_{FL}. The TU binarization of a non-negative integer v yields a sequence of v ones and a terminating zero, which is omitted only for the maximum value of v. The EG-k binarization consists of a prefix and a suffix: The prefix value $v_{\mathrm{p}} = \lfloor \log_2 \left(v/2^k + 1 \right) \rfloor$ is encoded with a unary code, yielding $n_{\mathrm{p}} = v_{\mathrm{p}} + 1$ bits. The suffix value $v_{\mathrm{s}} = v + 2^k \left(1 - 2^{v_{\mathrm{p}}}\right)$ is encoded with a fixed length code of $n_{\mathrm{s}} = v_{\mathrm{p}} + k$ bits.

Context Modeling

Context modeling exploits statistical dependencies between the to-be-coded bin and previously coded data. Typically, contexts are chosen depending on the position of each bin in the bin-string, the position of the symbol, or the preceding symbol's value. Each context model is associated with two probability estimates, which are continuously updated for each coded bin. We refer to [10], [12] for further reading.

Last Significant Coefficient Position

Since coefficients at higher frequencies tend to be quantized to zero, we adopt a coding technique from HEVC/VVC [12], [16]. The position of the last significant coefficient of each block is binarized and sent to the decoder. By doing so, the transmission of the remaining non-significant coefficients can be avoided, sparing bitrate. The last significant coefficient is binarized and the corresponding context models are chosen the same way as in [12].

IV. Coding scheme

This paper proposes the following changes with respect to the baseline [7]:

- The GFT DC and AC coefficients of each block are quantized separately. Quantization of the DC GFT coefficients is carried out by utilizing Differential Pulse Code Modulation (DPCM) [17].
- For efficient entropy coding of all elements (namely the GFT DC and AC coefficients as well as the k-means clusters), we propose to replace RLGR with CABAC. We adapt both binarization and context modeling to the statistics of the GFT coefficients and keep the third part of CABAC, the arithmetic coding engine, untouched.

Differential Pulse Code Modulation

It can be assumed that points in neighboring blocks often contain similar attributes, which yields similar DC values for neighboring blocks. This dependency is utilized for efficient transmission of the GFT DC coefficients by means of a simple version of Differential Pulse Code Modulation (DPCM) [17]

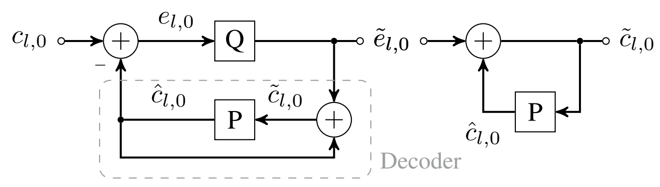

Fig. 2: DPCM scheme for quantization of DC coefficients with quantization denoted with "Q" and prediction with "P".

presented in Fig. 2. The DPCM encoder contains the decoder (closed loop structure), where the previous output values $\tilde{c}_{l,0}$ are fed into a prediction block, yielding prediction estimates $\hat{c}_{l,0}$. To account for cases where the prediction fails, the encoder calculates the prediction error as $e_{l,0} = c_{l,0} - \hat{c}_{l,0}$. This error is quantized and transmitted to the decoder, where it is added to the prediction estimates, yielding the output values $\tilde{c}_{l,0} = \hat{c}_{l,0} + \tilde{e}_{l,0}$. Since it is the mean values of each block that are comparable to one another, the prediction of the decoded DC value of current block l is selected as a scaled version of the DC value of previous block $l-1$: $\hat{c}_{l,0} = \sqrt{\frac{N_l}{N_{l-1}}}\tilde{c}_{l-1,0}$.

In [7], the scan-order of the cluster centers, which dictates the order of the underlying partitions, is determined using Morton space-filling curves (also known as z-order curves). Since the DC coefficient of the previously coded partition is used for prediction, we propose to replace Morton by Hilbert space-filling curves, which were shown to best preserve the spatial correlations of a point cloud's attributes [18]. By using both DPCM and Hilbert curves, it is expected that the DPCM coding loop will have access to a better DC coefficient prediction and thus provide higher coding efficiency.

Binarization

With CABAC being a binary encoder, our symbols need to be binarized before they can be encoded. In this paper, truncated unary (TU) and exponential-Golomb codes of order k (EG-k) are evaluated. Note that TU and EG-k are only able to encode non-negative integers. Since the levels $\tilde{C}$ to-be-encoded can be negative, they are transformed as follows:

$$\tilde{c}_i \leftarrow \begin{cases} 2\tilde{c}_i - 1 & \text{if } \tilde{c}_i > 0, \\ -2\tilde{c}_i & \text{if } \tilde{c}_i \leq 0, \end{cases} \tag{2}$$

mapping positive and negative levels to odd and even values respectively. Note that the very probable level $\tilde{c}_i = 0$ is assigned a bin-string of the shortest length of one.

Context Modeling

Our context modeling strategies are employed for both the bin-strings resulting from TU-binarization and for the prefix part of EG-k generated codes (which is also a TU-coded bin-string). EG-k suffixes are coded using CABAC's bypass engine. Binarization of symbol $\tilde{c}_i$ at position i in its block yields bins $b_{i,n}$ at position n in the corresponding bin-string. In context modeling, each bin $b_{i,n}$ is assigned a unique context identifier $\mathrm{ctx}_{i,n}$. In this paper, we evaluate three different context modeling approaches as follows:

1) "Binary Arithmetic Coding" (BAC): Coding every bin $b_{i,n}$ with the same context, which reduces CABAC to a binary arithmetic coder without information-dependent context modeling: $\mathrm{ctx}_{i,n} = 0$.
2) "Bin Position" (BP): Coding $b_{i,n}$ depending on its position n in the bin-string. To limit the number of contexts, we clip n by n_R:

$$\mathrm{ctx}_{i,n} = \begin{cases} n & \text{if } n < n_R, \\ n_R & \text{else.} \end{cases} \tag{3}$$

This results in a total of $n_R + 1$ contexts. Note that for $n_R = 0$, this context model becomes equivalent to BAC.
3) "Bin & Symbol Position" (BPSP): Selecting $\mathrm{ctx}_{i,n}$ depending on bin position n as well as i, the position of the binarized symbol $\tilde{c}_i$ inside its block. Four different intervals are defined with edge values $i_0 < i_1, < i_2$, as described in Table I. In total, $4\,(n_R + 1)$ contexts are used. Note that for $i_0 = i_1 = i_2 = 0$, this context model becomes equivalent to BP.

TABLE I: Context selection based on bin & symbol positions.

i	Context identifiers $\mathrm{ctx}_{i,n}$	
	$n < n_R$	$n \geq n_R$
$i < i_0$	$n + n_R + 1$	$2n_R + 1$
$i_0 \leq i < i_1$	$n + 2(n_R + 1)$	$3n_R + 2$
$i_1 \leq i < i_2$	$n + 3(n_R + 1)$	$4n_R + 3$
$i_2 \leq i$	n	n_R

V. Evaluation

The "8iVFBv2" data set [19] is used to verify the efficiency of the coding scheme. It is comprised of four point cloud sequences "Loot", "Longdress", "Soldier" and "Redandblack" which present varying degrees of color complexity. Each point cloud contains 800K points on average, and $T = 30$ frames chosen uniformly over the sequences are used for evaluation.

Prior to entropy coding, the GFT coefficients C are derived using the same method and parameters as in [7] i.e the attributes are converted to the YUV color space and partitioning is performed using $k = 1500$ centers. Moreover, the GFT coefficients and partition centers are uniformly quantized using quantization step sizes $q \in [1, 64]$ and a fixed step size $q_{\text{centers}} = 10$, respectively.

For our evaluations, the coefficients of each block are concatenated component-wise and are then fed to the CABAC coder sequentially in one pass. In order to provide a fair comparison, [7]'s RLGR-based entropy coding scheme, which codes the Y, U and V components separately from one another, has been modified to adopt the described coding method. This will constitute our baseline method.

Preliminary Experiments

As a first experiment, RLGR is kept as the encoding method and only the quantization scheme for the GFT DC

TABLE II: BD rate changes in % with respect to [7]. BAC ($n_R = 0$) or BP ($n_R > 0$) context models.

	Loot			Longdress			Red&black			Soldier			Average		
n_R	TU	EG-0	EG-1	TU	EG-0	EG-1	TU	EG-0	EG-1	TU	EG-0	EG-1	TU	EG-0	EG-1
0	-30.22	-25.95	-10.47	-21.67	-15.39	-1.85	-33.02	-28.47	-14.48	-24.05	-18.45	-2.89	-27.24	-22.06	-7.42
2	**-31.70**	-31.75	-13.95	**-23.20**	-23.04	-6.77	**-34.27**	-34.23	-17.97	**-25.60**	-25.52	-7.31	**-28.69**	-28.64	-11.50
4	-31.67	-32.13	**-14.09**	-23.15	-23.85	-7.10	-34.20	-34.65	-18.13	-25.53	-26.16	-7.55	-28.64	-29.20	-11.72
6	-31.56	-32.16	-14.09	-23.02	**-23.94**	-7.12	-34.07	-34.70	**-18.15**	-25.39	**-26.23**	**-7.57**	-28.51	**-29.26**	**-11.73**
8	-31.49	**-32.17**	-14.09	-22.93	-23.95	**-7.13**	-33.98	**-34.71**	-18.15	-25.29	-26.23	-7.57	-28.42	-29.26	-11.73
10	-31.44	-32.17	-14.09	-22.86	-23.95	-7.13	-33.92	-34.71	-18.15	-25.22	-26.23	-7.57	-28.36	-29.26	-11.73

TABLE III: BD rate changes in % with respect to [7]. BPSP context model with EG-0.

n_R	Loot	Longdress	Red&black	Soldier	Average
0	-26.28	-15.70	-28.77	-18.81	-22.39
2	-33.50	-24.79	-35.86	-27.41	-30.39
4	-34.12	-25.97	-36.56	-28.41	-31.26
6	**-34.18**	-26.12	-36.64	-28.54	-31.37
8	-34.18	**-26.13**	**-36.65**	**-28.55**	**-31.38**
10	-34.18	-26.13	-36.65	-28.55	-31.38

coefficients is replaced with DPCM. As a useful addition, the Morton-ordering is replaced by Hilbert-ordering. While the use of Hilbert space-filling curves over Morton curves provides limited gains of 0.04%, DPCM is able to substantially improve the coding efficiency. These results are enhanced when the two techniques are combined yielding 1.76% of bitrate reduction. This indeed shows that subsequent blocks, the order of which is improved by using Hilbert curves, carry redundant information regarding their DC attribute values.

When replacing RLGR with CABAC, the quantized error signal of the DC coefficients and the partition centers are binarized using EG-0 and entropy-coded with a context model that assigns different contexts depending on the bin's position and the bin of the previously coded symbol at the same position in the bin-string [20]. This context model did not perform well for the AC coefficients and is hence not considered here. The subsequent experiments focus on coding the AC coefficients since they constitute the majority of the bit stream.

Moreover, it has been determined that the highest values for CABAC's adaptation rates of the context probability estimators $\alpha_0 = 2^{-2}, \quad \alpha_1 = 2^{-5}$, which provide the fastest adaptation, yield the best results and are thus, later used for all contexts.

Binarization

Truncated Unary (TU) and Exponential Golomb Codes of order $k \in \{0, 1\}$ are tested for varying values of n_R while using the context model BP. Table II shows that substantial gains can be obtained for every binarization scheme. Exponential Golomb with values of k higher than 0 is not able to compete since symbols representing low amplitude values are not as efficiently represented. Context model BAC ($n_R = 0$) provides already significant gains on its own, which are best when using the TU binarization, yielding 27.24% BD-rate savings. These results are further improved when increasing n_R, where TU becomes optimal for $n_R = 2$. However, EG-0 is able to achieve savings of up to 29.26% on average for $n_R = 6$ and is thus chosen for subsequent experiments. Larger values $n_R > 6$ only increase the complexity of the context model while providing no benefits.

It is worth noting that these results were obtained while coding the position of the last significant coefficient per block (as described in Section III). Turning this feature off worsens the results across the board e.g. EG-0 with $n_R = 6$, only provides 25.62% BD rate savings, which is a loss of 3.64%.

Context Modeling

The BPSP context model, which uses the interval boundaries i_0, i_1 and i_2 is investigated. The logarithmically spaced values of $i_0 = 10$, $i_1 = 50$, $i_2 = 100$ were experimentally found to be suitable. BPSP assigns a different set of $(n_R + 1)$ contexts to the bins of a coefficient $\tilde{c}_i$ depending on its position i inside the block, achieving a kind of frequency bining. This is beneficial, as an additional 2.12% rate savings are achieved for $n_R = 8$, as shown in Table III. This shows that subsequent blocks exhibit statistical dependencies over their frequency components and that CABAC is able to exploit them.

VI. Conclusion

An efficient adaptive entropy coding scheme, aimed at graph transform coefficients representing point cloud attributes, is presented. The point cloud blocks are first ordered using Hilbert space-filling curves, which increases the spatial dependencies of subsequent blocks. Predictive coding of the DC coefficients is then performed using DPCM and the resulting error signal is entropy coded using CABAC. A combination of binarizations and context modeling techniques has been investigated to code the AC coefficients using CABAC as well. These results have been compared to and then integrated into the baseline method, in order to provide an efficient end-to-end compression scheme. On average, Bjøntegaard Delta rate saving of 31.38% could be achieved.

However, some areas could be further researched: As done in the VVC standard, the adaptation rates and the initial probabilities per-context could be individually optimized. Context modeling strategies that depend on previously coded symbol values should be further investigated as well.

Acknowledgment

The authors would like to thank the reviewers for their thoughtful comments that helped to improve this manuscript.

References

[1] X. Sheng, L. Li, D. Liu, Z. Xiong, Z. Li, and F. Wu, “Deep-PCAC: An end-to-end deep lossy compression framework for point cloud attributes,” *IEEE Transactions on Multimedia*, vol. 24, pp. 2617–2632, 2022.

[2] “Final Draft: Information technology – Coded representation of immersive media – Part 9: Geometry-based point cloud compression,” International Organization for Standardization, Geneva, CH, Standard, 2022.

[3] 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, 2016.

[4] A. Sandryhaila and J. M. F. Moura, “Discrete signal processing on graphs: Graph Fourier transform,” in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 6167–6170.

[5] C. Zhang, D. Florêncio, and C. Loop, “Point cloud attribute compression with graph transform,” in *2014 IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 2066–2070.

[6] E. Pavez, B. Girault, A. Ortega, and P. A. Chou, “Region adaptive graph Fourier transform for 3D point clouds,” in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 2726–2730.

[7] T. Meyer, M. Meyer, D. Mehlem, and C. Rohlfing, “Attribute-aware partitioning for graph-based point cloud attribute coding,” in *Proc. of International Picture Coding Symposium PCS ’22*. San Jose, USA: IEEE, Piscataway, Dec. 2022.

[8] H. Malvar, “Adaptive run-length/Golomb-Rice encoding of quantized generalized Gaussian sources with unknown statistics,” in *Data Compression Conference (DCC’06)*, 2006, pp. 23–32.

[9] 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 Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3736–3764, Aug. 2021.

[10] D. Marpe, H. Schwarz, and T. Wiegand, “Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620–636, Jul. 2003.

[11] A. Ortega, P. Frossard, J. Kovavcević, J. M. F. Moura, and P. Vandergheynst, “Graph signal processing: Overview, challenges, and applications,” *Proceedings of the IEEE*, vol. 106, no. 5, pp. 808–828, 2018.

[12] H. Schwarz, M. Coban, M. Karczewicz, T.-D. Chuang, F. Bossen, A. Alshin, J. Lainema, C. R. Helmrich, and T. Wiegand, “Quantization and entropy coding in the versatile video coding (VVC) standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3891–3906, 2021.

[13] T. Wiegand, G. Sullivan, G. Bjøntegaard, and A. Luthra, “Overview of the H.264/AVC video coding standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[14] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, “Overview of the high efficiency video coding (HEVC) standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 12, pp. 1649–1668, Sep. 2012.

[15] A. Alshin, E. Alshina, and J. Park, “High precision probability estimation for CABAC,” in *2013 Visual Communications and Image Processing (VCIP)*, 2013, pp. 1–6.

[16] M. Wien, *High Efficiency Video Coding – Coding Tools and Specification*. Berlin, Heidelberg: Springer, Sep. 2014.

[17] J.-R. Ohm, *Multimedia Signal Coding and Transmission*. Springer Heidelberg/Berlin, 2015.

[18] J. Chen, L. Yu, and W. Wang, “Hilbert space filling curve based scan-order for point cloud attribute compression,” *IEEE Transactions on Image Processing*, vol. 31, pp. 4609–4621, 2022.

[19] E. d’Eon, B. Harrison, T. Myers, and P. A. Chou, “8i Voxelized Full Bodies – A Voxelized Point Cloud Dataset,” ISO/IEC JTC1/SC29/WG1 input document M74006 and ISO/IEC JTC1/SC29/WG11 input document M40059, Geneva, Doc., Jan. 2017.

[20] M. Bläser, C. Rohlfing, Y. Gao, and M. Wien, “Adaptive coding of non-negative factorization parameters with application to informed source separation,” in *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP ’18*. IEEE, Piscataway, Apr. 2018, pp. 751–755.