3D point cloud compression

Chao Cao

INSTITUT POLYTECHNIQUE DE PARIS

TELECOM SudParis

IP PARIS

NNT : 2021IPPAS015

Thèse de doctorat

Compression d'objets 3D représentés par nuages de points
(3D point cloud compression)

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom SudParis

École doctorale n°626 Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat: Informatique

Thèse présentée et soutenue à Évry de soutenance, le 14/12/2021, par

Chao Cao

Composition du Jury :

Livio DE LUCA Directeur de recherche, CNRS (l'UMR CNRS/MCC MAP)	Président
Lu YU Professeure, College of Information Science and Electronic Engineering	Rapporteur
Marc ANTONINI Directeur de recherche, CNRS (I3S)	Rapporteur
Ralf SCHAEFER VP Standards R&I, InterDigital, Inc.	Examinateur
Titus ZAHARIA Professeur, Télécom SudParis (SAMOVAR)	Directeur de thèse
Marius PREDA Maître de conférences, Télécom SudParis (SAMOVAR)	Co-Encadrant de thèse

To all the suffering we take in life

I keep telling myself

You are nothing but everything

Institut Polytechnique de Paris

91120 Palaiseau, France

ACKNOWLEDGEMENTS

"A journey of a thousand miles begins with a single step" is a common saying from the ancient Chinese philosopher Laozi and describes perfectly my status when I have just finished this meaningful step as a Ph.D. graduate. None of the remarkable experiences will happen without the help of many people.

Firstly, I would like to express my sincere gratitude to my supervisor Prof. Titus ZAHARIA for the continuous support during my study and research. He helped me greatly with his outstanding insights academically and professionally. I could not imagine having a better supervisor who would guide and help me refine all my articles with great patience over and over.

I would also like to thank my co-supervisor Prof. Marius PREDA, who provides me the opportunity to participate in MPEG meetings and gives me massive help during the study. I can always rely on his insightful comments and encouragement when I have troubles. Meanwhile, he always inspires me with his foresight and immense knowledge in my research.

Meanwhile, I am also thankful to the members of my thesis committee: Prof. Livio DE LUCA and Prof. Marco CAGNAZZO, for their useful comments and kind suggestions. They helped me to find the insufficient part in my work and encouraged me to delve into it.

Moreover, I would like to thank all members of the jury: Prof. Lu YU and Prof. Marc ANTONINI for accepting being my thesis reviewers, and for their attention and thoughtful comments. I also thank Mr. Ralf SCHAEFER for accepting being my thesis examiner.

Sincere thanks to my fellows in the department ARTEMIS: Abhaya-Dhathri ARIGE, Christian TULVAN, Léa SAUNIER, Minderis KRIR, Nathan HUBENS, Nicolas HOFFMANN, Rania BENSAIED, Tan-Khoa MAI, Traian LAVRIC, Veronica SCURTU, Zied LAHIANI for all the interesting scientific discussions and academic arguments. It is an honor to collaborate with you all.

I owe my deepest gratitude to the people who helped me in Télécom SudParis. Special thanks to Evelyne TARONI, who cared for me warmly and helped me in the administrative aspects with her kindest heart. Thanks also go to Veronique GUY, Zahia ZENDAGUI, Walid BENAMEUR, Xavier VASSEUR for their kind assistance and help.

Last but not the least, I would like to thank my beloved family, my parents, and my lovely wife for supporting me with faith and providing continuous encouragement during my entire study. This accomplishment would not have been possible without their assistance.

Institut Polytechnique de Paris

91120 Palaiseau, France

TABLE OF CONTENT

LIST OF TABLES

Institut Polytechnique de Paris

91120 Palaiseau, France

LIST OF FIGURES

LIST OF ABBREVIATIONS

fps	frames per second
3D PCC	3D Point Cloud Compression
MPEG	Moving Picture Expert Group
V-PCC	Video-based Point Cloud Compression
G-PCC	Geometry-based Point Cloud Compression
MLS	Mobile Laser Scanner
LiDAR	Light Detection And Ranging
TOF	Time-Of-Flight
SfM	Structure from Motion
MVS	Multi-View Stereo
RGB-D	Red, Green, Blue - Depth
DL	Deep Learning
LOD	Level Of Details
ROI	Region Of Interests
ISO	International Organization for Standardization
IEC	International Electrotechnical Commission
ICP	Iterative Closest Point
KD	K-Dimensional
bpp	bits per point
PCA	Principal Component Analysis
VR	Virtual Reality
AR	Augmented Reality
MR	Mixed Reality
RDO	Rate-Distortion Optimization
JPEG	Joint Photographic Experts Group
LZW	Lempel–Ziv–Welch
XOR	exclusive disjunction operator
GLA	Generalized Lloyd Algorithm
CSRBF	Compactly Supported Radial Basis Function
GMM	Gaussian Mixture Models
KLT	Karhunen-Loeve Transform
DCT	Discrete Cosine Transform
RAHT	Region-Adaptive Hierarchical Transform
GFT	Graph Fourier Transform

GPT	Gaussian Process Transform
PSNR	Peak signal-to-noise ratio
HEVC	High Efficiency Video Coding
PCL	Point Cloud Library
TM	Test Models
TMC	Test Model Category
K	thousand
M	million
RD	Rate-Distortion
RA/INTER	Random Access
AI/INTRA	All Intra
CfPs	Call for Proposals
CU	Coding Unit
PU	Prediction Unit
TU	Transform Unit
MV	Motion Vector
CTU	Coding Tree Units
GOF	Group of frames
SOR	Statistical Outlier Removal
DOF	Degrees Of Freedom
SVD	Singular Value Decomposition
LZMA	Lempel–Ziv–Markov chain algorithm

Institut Polytechnique de Paris

91120 Palaiseau, France

ABSTRACT

With the rapid growth of multimedia content, 3D objects are becoming more and more popular. Most of the time, they are modeled as complex polygonal meshes or dense point clouds, providing immersive experiences in different industrial and consumer multimedia applications. The point cloud, which is easier to acquire than mesh and is widely applicable, has raised many interests in both the academic and commercial worlds.

A point cloud is a set of points with different properties such as their geometrical locations and the associated attributes (e.g., color, material properties, etc.). The number of the points within a point cloud can range from a thousand, to constitute simple 3D objects, up to billions, to realistically represent complex 3D scenes. Such huge amounts of data bring great technological challenges in terms of transmission, processing, and storage of point clouds.

In recent years, numerous research works focused their efforts on the compression of meshes, while less was addressed for point clouds. We have identified two main approaches in the literature: a purely geometric one based on octree decomposition, and a hybrid one based on both geometry and video coding. The first approach can provide accurate 3D geometry information but contains weak temporal consistency. The second one can efficiently remove the temporal redundancy yet a decrease of geometrical precision can be observed after the projection. Thus, the tradeoff between compression efficiency and accurate prediction needs to be optimized.

We focused on exploring the temporal correlations between dynamic dense point clouds. We proposed different approaches to improve the compression performance of the MPEG (Moving Picture Experts Group) V-PCC (Video-based Point Cloud Compression) test model, which provides state-of-the-art compression on dynamic dense point clouds.

First, an octree-based adaptive segmentation is proposed to cluster the points with different motion amplitudes into 3D cubes. Then, motion estimation is applied to these cubes using affine transformation. Gains in terms of rate-distortion (RD) performance have been observed in sequences with relatively low motion amplitudes. However, the cost of building an octree for the dense point cloud remains expensive while the resulting octree structures contain poor temporal consistency for the sequences with higher motion amplitudes.

An anatomical structure is then proposed to model the motion of the point clouds representing humanoids more inherently. With the help of 2D pose estimation tools, the motion is estimated from 14 anatomical segments using affine transformation.

Moreover, we propose a novel solution for color prediction and discuss the residual coding from prediction. It is shown that instead of encoding redundant texture information, it is more valuable to code the residuals, which leads to a better RD performance.

Although our contributions have improved the performances of the V-PCC test models, the temporal compression of dynamic point clouds remains a highly challenging task. Due to the limitations of the current acquisition technology, the acquired point clouds can be noisy in both geometry and attribute domains, which makes it challenging to achieve accurate motion estimation. In future studies, the technologies used for 3D meshes may be exploited and adapted to provide temporal-consistent connectivity information between dynamic 3D point clouds.

Institut Polytechnique de Paris

91120 Palaiseau, France

CHAPTER I. INTRODUCTION

Institut Polytechnique de Paris

91120 Palaiseau, France

1.1. Background and motivation

Recent advances in computer graphics have made it possible to create realistic digital representations of 3D objects and surroundings, allowing real-time interactions with users. Such representations use various mathematical models to describe both geometry and attribute information (e.g., colors, materials, lighting behavior, etc.) of the objects. Among them, 3D meshes became highly popular and have been extensively used in the gaming and 3D filming industries. Nevertheless, the creation of high-fidelity content requires enormous computational resources for storage, transmission, processing, and visualization purposes. This problem penalizes their deployment on light, mobile devices.

More recently, due to the progress of advanced 3D scanning technologies, a different approach, based on point cloud representations has emerged. A point cloud is a simple data structure, defined as a set of unorganized points in the 3D space, usually represented in a Cartesian coordinate system with (x, y, z). It can carry different attributes such as colors, normals, and reflectances while being used to represent both static and dynamic 3D objects.

The point cloud representations are well-suited for various industrial applications, including 3D scanning and modeling, environmental monitoring, agricultural and forestry, autonomous driving, and so on. It can fulfill different requirements in terms of precision, point density, and related attributes.

Thus, for hyper-realistic visualization applications, point clouds with high density (up to millions of 3D points) are required, which raises massive demands in terms of computational and memory requirements. For example, in the case of autonomous navigation applications, rather sparse point clouds are used, but high precision is required and needs to be preserved. On the other hand, for applications that involve dynamic point clouds, the supplementary temporal dimension explodes the related bandwidth and needs to be appropriately considered.

Within this context, the development of efficient compression methods and technologies, considering various application constraints, has become a crucial challenge.

1.1.1. Background

1.1.1.1. Definition

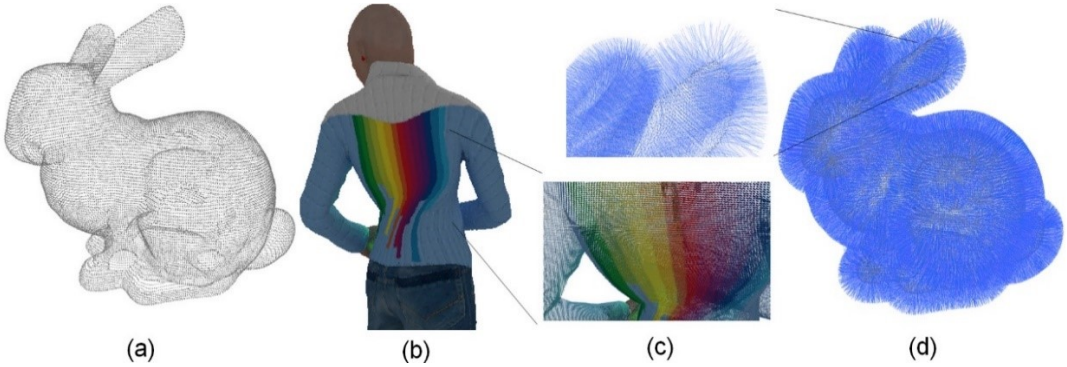


Fig. I-1. (a) Point cloud "Stanford Bunny" (Stanford Bunny, n.d.) with geometry information, (b) queen dataset produced by Technicolor (J. Ricard , C. Guède, R. Doré, n.d.) with the color information, and (c) Zoomed detail for pictures in (b) and (d). (d) "Stanford Bunny" with normal information denoted with the blue line.

By definition, a 3D point cloud is a set of points $\{P_i\}_{i=1}^{n}$, embedded in the 3D space and carrying both geometry information described by its Cartesian coordinates and various attributes (i.e., photometric properties) such as colors or normals as shown in Fig. I-1. The term "cloud" reflects the unorganized nature of the set (Otepka et al., 2013). That is, the point clouds are unstructured, with no coherence/order relations between the points.

1.1.1.2. Acquisition

With the advancement of 3D data acquisition technologies, it is becoming easier and faster to obtain point clouds from real-world objects. However, the various technologies also bring different types of “noise” into the data representation simultaneously. The “noise” here refers to either undesired or error information acquired during the capturing process. Some distortions are caused by external reasons, such as the inevitable movement of the mobile laser scanner (MLS). Some are the errors generated by the reconstruction algorithms. It is important to understand the basic concept of acquisition technologies so that effective and efficient compression can be achieved to the maximum extent.

In general, 3D point cloud data are captured using a) 3D laser scanning and b) photogrammetry technologies.

a) 3D laser scanning

1) Time-of-flight and phase-shift

The 3D laser scanning technology is also known as Light Detection And Ranging (LiDAR). It captures the 3D environment using two different range measurement techniques: time-of-flight (TOF) and phase-shift.

The TOF-based scanner emits a laser pulse and measures the traveling time of the reflected pulse. Meanwhile, the phase-shift-based scanner emits an amplitude modulated continuous wave (AMCW) and measures the phase shift between the emitted and reflected signals (Q. Wang et al., 2020). One main tradeoff between TOF scanners and phase-based scanners is between the speed of acquisition and the dynamic range. The TOF scanners hold a wider range of measurement as close as a meter, and up to a kilometer. On the other hand, the phase-based ones obtain higher ranging accuracy and acquisition speed, being able to capture hundreds of thousands of points per second.

2) Operating platform

Furthermore, the scanners can be classified into three categories, depending on their operational platforms as terrestrial laser scanner (TLS), airborne laser scanner (ALS), and mobile laser scanner (MLS).

The TLS is normally mounted on a tripod positioned in a fixed location on the ground during the scanning of surroundings. Thus, it is also known as a ground LiDAR scanner. It provides the highest stability and ranging accuracy since it stays still during the acquisition operation. Yet, it takes a great amount of scanning time and requires the environment to be static as well.

The ALS, representing the scanner mounted on the aircraft, is intended to scan the large-scale objects/scenes and is also known as an aerial LiDAR. It is capable of capturing the point cloud data for terrains of the city with relatively low accuracy (Hansen et al., 2015).

The MLS, also known as mobile LiDAR, is mounted on a mobile platform such as a vehicle or a drone to dynamically scan the environment while moving. The vibration of the vehicle leads to inconstant scanning conditions and can cause geometrical shifts of the captured 3D points.

However, the usage of LiDAR sensors has been limited due to their high cost compared to the ones using photogrammetric approaches.

b) Photogrammetry methods

Recently, various photogrammetric algorithms and technologies have been developed and have gained a great amount of popularity due to the ease of access to the required equipment. First, a collection of overlapping 2D images are captured from different locations representing different parts of the target object (Lu & Lee, 2017). Then, photogrammetric algorithms are leveraged to estimate the relative location of 2D images and reconstruct them into a 3D point cloud. The most popular algorithms are the structure from motion (SfM) algorithm and the multi-view stereo (MVS) algorithm.

In addition, when available, the RGB-D (Red, Green, Blue - Depth) cameras take advantage of both methods. It is a fusion of an RGB color camera and a depth sensor. The 3D point cloud is in this case generated by combining the 2D locations of the rasterized color pixels and the corresponding depth values.

To summarize, the LiDAR-based 3D scanning techniques can capture the point cloud data directly but the devices have usually a high cost. The scanners are normally mounted on objects and thus lack portability compared to the ones using photogrammetry. As a consequence, the dynamically captured point clouds can be deformed or translated, which brings difficulties to registration and compression.

Meanwhile, the sensors using photogrammetric techniques have been widely used in many interactive mobile applications due to their ease of access and deployment. Depending on the utilized algorithm, the generated point clouds can be incomplete, overlapped, or sometimes chaotic.

Though with several limitations, the acquisition technologies are evolving rapidly in recent years. It is foreseeable that shortly, the scanners may be compact enough to be integrated into a mobile phone. At the same time, with the advancement of deep learning (DL) techniques, faster reconstruction and better denoising of the point clouds can be achieved.

1.1.2. Motivation

The increasing growth of point cloud usage has brought challenges related to the storage and transmission of point cloud data. Different compression technologies are eagerly required to support various requirements. For example, the point cloud can be sparse or dense, so the compression algorithm should be adapted accordingly. The point cloud can be static or dynamic representing its variability over time. In addition, the compression can be either lossless or lossy meaning whether the information can be fully reversed or not.

A large number of works such as (Gumhold et al., 2005), (Y Huang et al., 2006), (R L de Queiroz & Chou, 2016), (Cohen et al., 2016a) have been proposed to reduce the spatial redundancy of the static point cloud. A few works such as (J Kammerl et al., 2012), (Dorina Thanou et al., 2015), (Dorina Thanou et al., 2016), (Moreno et al., 2017), (L. Li et al., 2019), (L. Li et al., 2020), and (L. Li et al., 2021) focus on exploiting the temporal correlations between dynamic point clouds. With the advancement of computer techniques, point clouds with higher density and variety can be processed. However, it is still challenging to efficiently transmit, visualize and store large-scale objects or high-fidelity content especially when such 3D point clouds are also dynamic. In addition, different functionalities such as LOD (level of details), random access, ROI (region of interests), are in great demand to realize real-time visualization of the 3D point clouds.

The interest in developing efficient PCC methods raised rapidly in recent years. 3D PCC standards are under development by the ISO (International Organization for Standardization) / IEC (International Electrotechnical Commission) MPEG group. The standard includes two approaches, published as the final international standard in 2020 and 2021 respectively, as two parts: the so-called V-PCC and G-PCC.

V-PCC is a toolbox of video-based compression methods and aims at providing low complexity decoding capabilities for applications that require real-time decodings, such as virtual/augmented reality and immersive communications. On the other hand, G-PCC is considered to provide efficient lossless and lossy compression for the deployment of autonomous driving, 3D maps, and other applications that exploit LiDAR generated point clouds (or similar content).

In our work, we had the opportunity to participate in the MPEG meetings during the V-PCC standardization process. We thus managed to analyze the limitations of the ongoing test model and to propose several contributions. Since the V-PCC solution was under development in the MPEG 3DG (3D Graphics) group and was updated continuously, our contributions were as well adapted over time in order to cope with the up-to-date V-PCC test model.

1.2. Outline of the thesis and contributions

The rest of the manuscript is structured as follows. Chapter II proposes a state of the art survey of PCC methods, highlighting main principles and analyzing related advantages and limitations. A study of DL-based methods is provided as well. Part of the work is published in (Cao et al., 2019), (Cao et al., n.d.), and (Cao et al., 2021).

Chapter III proposes a more in-depth presentation of the MPEG V-PCC and G-PCC models, adopted in our future developments. Some elements of experimental evaluation are also presented and discussed here. Their analysis makes it possible to identify bricks in the processing chain that are responsible for performance degradation in certain situations, and thus to define some axes of future enhancement.

Chapter IV introduces a first methodological contribution, which concerns an RDO-based octree segmentation method for V-PCC. The method recursively segments the 3D space into an octree structure adapted to a by-part, affine motion model. An RDO (Rate-Distortion Optimization) process is carried out to determine if further segmentation is needed for each octree node.
This work is involved in a joint project and is registered as a US patent numbered 1888-75602 P42966US1.

Chapter V introduces a second contribution, which consists of a combination of the 2D pose estimation technique and 3DPCC. A skeleton-based, affine motion model is here used for motion estimation purposes. The 3D point cloud is first projected onto a set of 2D images to exploit existing 2D pose estimation technologies. The resulting 2D labels are then integrated into 3D, making it possible to detect the anatomical segments of the point cloud. Finally, motion estimation and compensation are applied to the segments using affine transformation. Part of the work is published in (CAO et al., 2020).

A third contribution is introduced in Chapter VI. It concerns an optimized version of the skeleton-based technique previously proposed and notably aims at achieving a better prediction for dynamic point clouds with high motion amplitudes. Here, an iterative closest point (ICP) algorithm is used to improve the accuracy of the motion estimation process. The prediction scheme is also enhanced: the predictor frame is always transformed to match with the closest point cloud frame (generally, the previous frame). In this way, the motion estimation is more accurate.

Chapter VII describes our fourth contribution, related to the coding of photometric attributes. Instead of consuming most of the bits (e.g., the texture video occupies up to 80% of the resources in the encoding process in R05 condition) to encode texture video, we propose to encode the color residuals computed from the predicted and reconstructed colors. By doing so, we show that significant gains in terms of RD performances can be achieved for all the tested sequences.

Finally, Chapter VIII presents the general conclusions and opens some perspectives of future work. Our publications, patent, and standardization input documents are listed below.

List of publications:

Journal article:

[1] C. Cao, M. Preda, V. Zakharchenko, E. S. Jang and T. Zaharia, "Compression of Sparse and Dense Dynamic Point Clouds--Methods and Standards," in Proceedings of the IEEE, doi: 10.1109/JPROC.2021.3085957.

Conference proceedings:

[1] C. Cao, M. Preda, and T. Zaharia, "3D point cloud compression: A survey," in Proceedings - Web3D 2019: 24th International ACM Conference on 3D Web Technology, 2019, pp. 1–9, doi: 10.1145/3329714.3338130.

[2] C. Cao, C. Tulvan, M. Preda, and T. Zaharia, "Skeleton-based motion estimation for Point Cloud Compression," in 2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP), Sep. 2020, pp. 1–6, doi: 10.1109/MMSP48831.2020.9287165.

Invited online journal article:

[1] C. Cao, M. Preda, and T. Zaharia, "What's new in Point Cloud Compression?," doi: 10.33552/GJES.2020.04.000598.

Patent:

US patent numbered 1888-75602 P42966US1.

Standardization input documents:

[1] Marius Preda, Christian Tulvan, Chao Cao, Merged results of PCC CfP, ISO/IEC JTC1/SC29/WG11 MPEG2017/m41501, October 2017, Macau, CN

[2] Chao CAO, Christian Tulvan, Marius Preda, Validation process and results of the submitted data for the responses to the Point Cloud Compression CfP, ISO/IEC JTC1/SC29/WG11 MPEG2017/M41830, October 2017, Macao, CN

[3] Chao CAO, Christian Tulvan, Marius Preda, Proposal for adaptive orientation of projection planes, ISO/IEC JTC1/SC29/WG11 MPEG2018/M42560, April 2018, San Diego, USA

[4] Chao CAO, Christian Tulvan, Marius Preda, Updates on adaptive orientation of projection planes method, ISO/IEC JTC1/SC29/WG11 MPEG2018/M43512, July 2018, Ljubljana, SI

[5] Marius Preda, Alexis Tourapis, Rajan Joshi, Chao Cao, Khaled Mammou, Madhukar Budagavi, Encoding per patch 3D transforms, ISO/IEC JTC1/SC29/WG11 MPEG2019/m47347, March 2019, Geneva, CH

[6] Chao CAO, Marius PREDA, Draft anchor result for draft V-PCC Verification Tests, ISO/IEC JTC1/SC29/WG 7 MPEG2021/ m57591, July 2021, Online

[7] Chao CAO, Marius PREDA, Anchor codec description and profiles for verification test, ISO/IEC JTC1/SC29/WG 7 MPEG2021/ m58286, October 2021, Online

CHAPTER II. 3D POINT CLOUD COMPRESSION: STATE OF THE ART

In this Chapter, a comprehensive overview of the 3D PCC state-of-the-art methods is proposed, to understand the status of the research in the field. The various families of approaches are identified, analyzed, and represented with the help of a so-called "Universe Map". We classify the compression techniques using such multi-dimensional representation in three main families: 1) 1D traversal methods, 2) 2D projection methods, and 3) 3D decorrelation methods. Emerging technologies, based on deep learning methodologies, are described as well. A final discussion opens some perspectives for further studies on 3D PCC.

Part of the work presented in this chapter has been published in (Cao, Preda, and Zaharia 2019), (Cao, Preda, and Zaharia n.d.), and (Cao et al. 2021).

Institut Polytechnique de Paris

91120 Palaiseau, France

2.1. The 3D PCC "Universe Map" for methods

The various 3D PCC state-of-the-art techniques are represented under the form of a so-called "Universe Map", illustrated in Fig. II-1. The "Universe Map" provides a multi-dimensional representation and involves three main levels. The highest level describes the dimension among which the redundancy is reduced. Three different modes can be here considered: 1D traversal (line space), 2D projection (plane space), and 3D decorrelation (sphere space). The second level presents how the geometry and attribute information is compressed. Three different approaches can be considered, which correspond to decomposition into LODs, clustering methods, and transform-based techniques. Interleaving with both the first two levels, the third one concerns the data structure involved in processing the geometry information. Various types of tree-based representations can be considered here, including binary trees, octrees, and other types of predictive spanning trees.

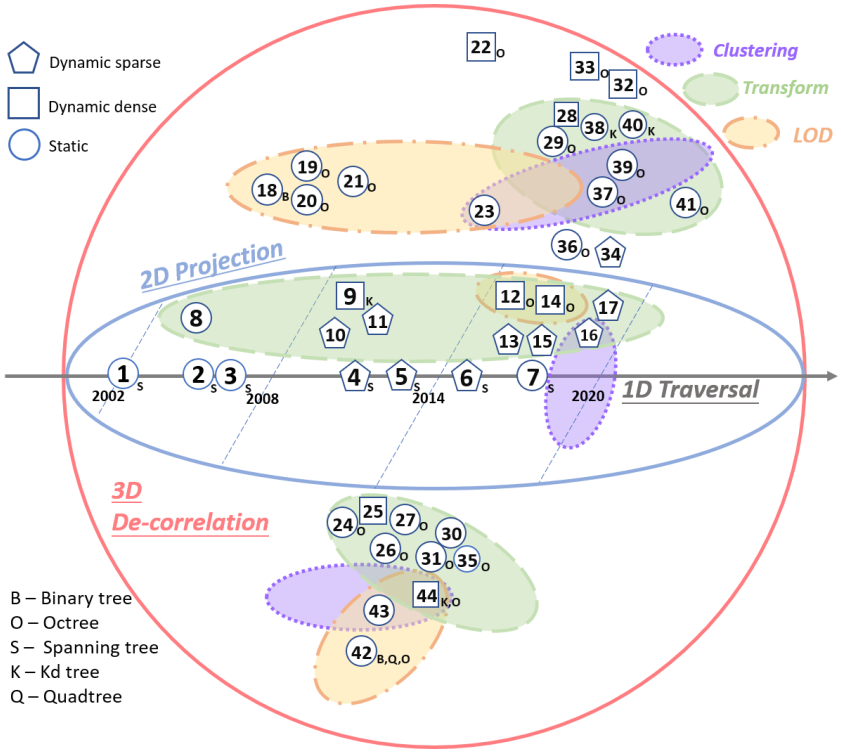


Fig. II-1. "Universe Map" for 3D PCC methods.

As shown in Fig. II-1, the number enclosed in circle, square, or pentagon is an identifier of the compression method described in the reference article, denoting the corresponding citing order in this overview. The list of references for Fig. II-1 is provided in Table II-1 for fast indexing. Meanwhile, the enclosing shape illustrates the type of input point clouds. The position of the shape presents the corresponding level and utilized approaches such as LODs (long dash-dot ellipse), clustering (round-dot ellipse), and transform-based techniques (long-dash ellipse). Moreover, the methods are presented from left to right with an ascending time order and characters representing the employed tree structure are shown as subscripts.

For instance, the square-shape reference (Sebastian Schwarz et al., 2019) numbered 44 contains the "K, O" subscript and also stays in the overlapped region of the three approaches within the 3D sphere space. Thus, the expression here is that this approach is a 3D decorrelation compression method that consists of LODs, clustering, and transform-based approaches for dynamic dense point clouds while taking advantage of k-dimensional (KD)-tree and octree.

Table II-1 Reference list of the methods for Fig. II-1.

N°	Reference	N°	Reference	N°	Reference	N°	Reference
1	(Gandoin & Devillers, 2002)	**12**	(Golla & Klein, 2015)	**23**	(Y Fan et al., 2013)	**34**	(Tu et al., 2017)
2	(Gumhold et al., 2005)	**13**	(Tu et al., 2016)	**24**	(C Zhang et al., 2014)	**35**	(Shao, Zhang, Li, Fan, et al., 2018)
3	(Merry et al., 2006a)	**14**	(Mekuria et al., 2017)	**25**	(D Thanou et al., 2015)	**36**	(D C Garcia & de Queiroz, 2018)
4	(Mongus & Žalik, 2011)	**15**	(Kohira & Masuda, 2017)	**26**	(Cohen et al., 2016b)	**37**	(K Zhang et al., 2018)
5	(Isenburg, 2013)	**16**	(Sun et al., 2019)	**27**	(Cohen et al., 2016a)	**38**	(Shao, Zhang, Li, Li, et al., 2018)
6	(Masuda & He, 2015)	**17**	(Tu et al., 2019b)	**28**	(D Thanou et al., 2016)	**39**	(Xu et al., 2018)
7	(Bogoslavskyi & Stachniss, 2016)	**18**	(Waschbüsch et al., 2004)	**29**	(Ricardo L. De Queiroz & Chou, 2016)	**40**	(Q. Zhang et al., 2018)
8	(Ochotta & Saupe, 2004)	**19**	(Schnabel & Klein, 2006)	**30**	(L. Wang et al., 2017)	**41**	(Gu et al., 2020)
9	(J.-M. Lien et al., 2009)	**20**	(Yan Huang et al., 2006)	**31**	(Ricardo L. De Queiroz & Chou, 2017b)	**42**	(Kathariya et al., 2018)
10	(Wiman & Yuchu, 2009a)	**21**	(Y Huang et al., 2008)	**32**	(Diogo C. Garcia & De Queiroz, 2018)	**43**	(Navarrete et al., 2018)
11	(Im et al., 2010)	**22**	(Julius Kammerl et al., 2012)	**33**	(R L de Queiroz & Chou, 2017)	**44**	(Sebastian Schwarz et al., 2019)

Let us now detail each family of methods involved starting with the 1D methods.

2.2. 1D methods: geometry traversal

Unlike in the mesh compression field, there is no connectivity information available for point clouds that can provide the geometry correlation between points to help to predict neighboring points.

In the case of 1D prediction methods, the underlying principle consists of constructing tree-based connectivity, by exploiting the neighborhood relations induced by the native geometric distances between the points in the cloud. Various traversal orders of the resulting trees can then be used to convert the geometry data into a 1D, prediction-adapted signal.

In (Gumhold et al. 2005), a prediction tree (Fig. II-2 (a)) is proposed to provide the geometry correlation between points. It is based on a greedy strategy which consists of determining the branches that minimize the prediction residuals, with respect to a breadth-first traversal order of the tree. The resulting residuals are finally binary coded with the help of a valence-based arithmetic encoder.

The single-resolution method introduced in (Merry et al., 2006b) aims at compressing the geometry information. Motivated by the work presented in (Gumhold et al. 2005) and (Taubin & Rossignac, 1998), the technique uses multiple geometry predictors to take into account the various correlations between neighboring points. Four different symbols (*B−base*, *L−left*, *R–right,* and *F−forward*) in Fig. II-2 (b) indicate the direction of the predictors of the child nodes starting from a root node. An additional *T (Terminal)* code terminates the list of child predictors and implicitly specifies the valence of the children so that the points starting from the root node can be reordered and predicted using such symbols.

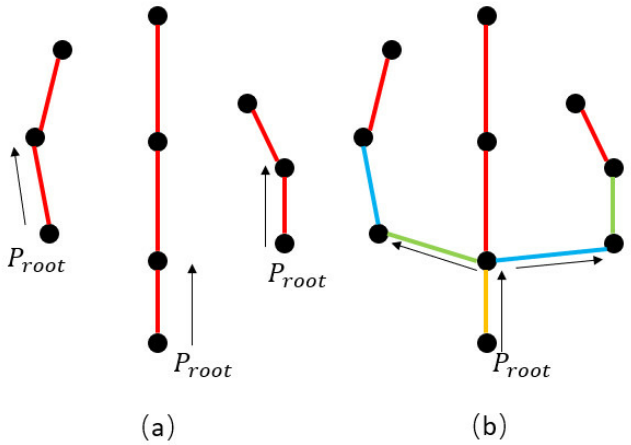


Fig. II-2. The proposed spanning tree example using different predictors (B(yellow), L(green), R(blue), F(red)).

Table II-2 summarizes the various 1D traversal compression approaches. Here, we have included two additional approaches, originally intended for 3D mesh compression but which can be also directly applied for 3D point cloud geometry compression. The first one is the reference KD-tree method introduced in (Gandoin & Devillers, 2002), while the second concerns the generic predictive coder in (Waschbüsch et al., 2004).

Table II-2 Summary of 1D traversal compression techniques.

Algorithm	Compression rates (bpp) for geometry	Compression rates (bpp) for attributes	Input type (datatype)	Lossless support	Remarks
Progressive lossless coder (Gandoin and Devillers 2002)	14.8 to 20.1 (in lossy mode)	-	Static (float)	Yes	Lossless data required domain such as medical research; Progressive; KD-tree based; Support lossless coding
Generic progressive coder (Waschbüsch et al. 2004)	13.59 to 21.22	2 for simple color; 11 for normal	Static (float)	No	Progressive decoding; Predictive coding for position, color, and normal
Prediction-tree coder (Gumhold et al. 2005)	7.29 to 17.41	-	Static (float)	No	Streaming; Efficient greedy prediction; Non-guaranteed point order
Linear & lateral predictor coder (Merry, Marais, and Gain 2006b)	6.45 to 21.92 (in Axial mode)	-	Static (float)	No	Laser range scanning; streaming; Suitable for real-time applications; Poor on non-regularly sampled or sparse point sets

The compression rates reported in Table II-2 are expressed in bpp. It should be underlined that the term "lossless support" reveals the reported condition in each reference without considering the potential of the methods. It is here applied to the situation where the input data may be voxelized (or, equivalently, represented with integer values). More importantly, it can be recovered identically by the decoder. We can observe that the 3D point cloud-dedicated methods outperform the previous mesh-based techniques.

Although the tree-based traversal methods offer the advantage of relatively simple implementation, the compression performances are quite limited because such algorithms do not fully take into account the 3D spatial correlations. Thus, exploiting higher dimensional correlations becomes necessary.

2.3. 2D methods: Projection and mapping onto 2D planar domains

In this case, the principle consists of converting the 3D point cloud into 2D images/videos by projection or mapping. Then, existing image/video coding technologies are used for compression purposes.

A wavelet-transform-based compression method using an atlas of height fields is proposed in (Ochotta & Saupe, 2004) to exploit image coding techniques. Inspired by (Pauly & Gross, 2001) and (Sander et al., 2003), a heuristic split-and-merge partition of the point set is performed using the mean least-squares approximation approach (Fig. II-3 (a)). The partitioned cluster as the one in Fig. II-3 (b) is parametrized as a height field, over the corresponding PCA (Principal Component Analysis) computed base plane. Each parametrized height field is then projected onto the base plane to generate a 2D image, as illustrated in Fig. II-3 (c). The obtained image is further compressed with the wavelet-based shape-adaptive image coding technique described in (S. Li & Li, 2000). Finally, a rate-distortion optimization (RDO) step ensures progressive rates, adapted for transmission over the network with various bandwidths.

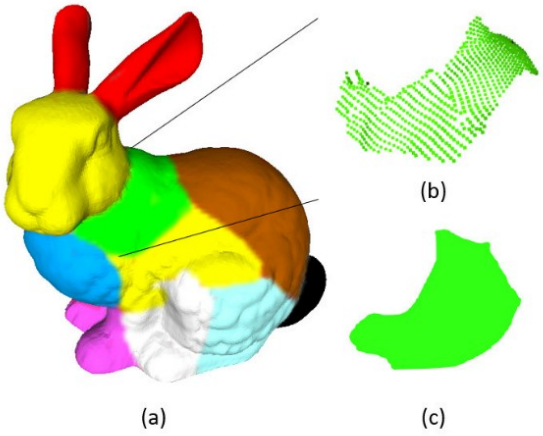


Fig. II-3. (a) Example of clustered Stanford bunny. (b) the clustered 3D points and (c) the projected height field over the respective base plane.

Another mapping-based method is introduced in (Daribo et al., 2012). Here, a 2D grid pattern consisting of horizontal and vertical lines is projected onto the 3D model, according to a set of viewpoints. For each viewpoint, this procedure yields a set of 3D curves. Considering the resulting 3D curves as coding units, compression is achieved by predictive coding using a spanning tree of the points' positions.

Several other approaches also adopt a multi-view representation strategy. The principle consists of representing the point cloud as a set of depth images, corresponding to projection according to various angles of view. Based on the multi-view video coding algorithms introduced in (Merkle et al., 2007), which exploit both temporal and inter-view statistical dependencies under a predictive framework, several coding schemes targeting real-time compression, are proposed in (Mekuria et al., 2017) and (J. M. Lien et al., 2010).

A combination of octree and projection-based methods is presented in (Ainala et al., 2016). Here, the residuals obtained from planar projections are exploited as an enhancement layer within the framework of a coarse octree coding scheme. The points in the enhancement layer are projected onto a 2D plane, determined with the help of a PCA for each cell of the considered octree. Then, the geometry and color attributes are predictively coded. In addition, the resulting prediction residuals are entropy encoded with the help of a learning-based technique, which leads to improvements in both compression performance and computational complexity. Authors show that the method achieves significant gains in compression ratio when compared to the JPEG (Joint Photographic Experts Group) developed compression approach introduced in (Mekuria et al., 2017).

The combination of the octree-based geometry compression and image-coding-based color has become in the last years an emerging research direction. Within this context, a patch-based method, so-called V-PCC, has been introduced in (Sebastian Schwarz et al., 2019). The main philosophy behind V-PCC is to leverage existing video codecs for compressing the geometry and texture information of a dynamic point cloud. A detailed description of the V-PCC methodology is provided in Chapter III-3.1.

V-PCC decomposes the input point cloud into a set of patches. First, a normal vector is associated with each point, with the help of a local approximation of the planar surface. Then, an initial clustering of the points is determined by grouping neighboring points with similar orientations (with respect to a given threshold), within a given search radius. The clustering is then refined iteratively by updating the neighboring points in a certain range that have similar normal vectors, to obtain smooth cluster borders and to minimize the associated mapping distortion. As the points gathered in the resulting clusters have similar normal vectors, a reference orientation can be associated with each patch, which corresponds to the mean value of the individual normal vectors.

Once the partition of the point cloud into patches is accomplished, the set of patches are orthogonally projected onto a 2D plane corresponding to the reference normal vector. The projection plane is discretized according to a regular sampling grid at a given resolution to obtain a pixelized projection image representation. Both depth and attribute (color) information are retained in the resulting projection image.

Finally, the geometry and attribute images are generated by assembling the orthogonally projected 2D patches within a global image. The patches are padded into the image in descending order of size. The pixels will be noted as "occupied" once filled with a patch. The bounding box information of the 2D patch is used to avoid intersection between patches while minimizing the non-occupied space between patches in the global image. The padding process makes it possible to obtain a smooth image with more spatial and temporal consistency, which is better suited for video coding. Table II-3 gives an overview of the projection-based approaches. The various techniques discussed here show that the efficiency of 3D PCC can be greatly improved by applying existing 2D image/video compression technologies.

Table II-3 Summary of 2D projection-based methods

Algorithm	Compression rates (bpp) for geometry (Peak signal-to-noise ratio (PSNR))	Compression rates (bpp) for attributes (PSNR)	Input type (datatype)	Lossless support	Remarks
Shape-Adaptive Wavelet Coder (Ochotta & Saupe, 2004)	1.20 to 3.41 (60dB to 87dB)	-	Static (float)	Yes	Efficient surface-based compression; Poor with complex surfaces
Image-based real-time coder (J. M. Lien et al., 2010)	Overall: 5000:1 to 50:1 (28dB to 31dB in geometry)		Dynamic (float)	Yes	Skeleton-based content in 3D tele-immersive environments; Inaccurate rigid-only motion estimation; Real-time for 5 to 6 fps (frames per second)
Grid-pattern-based predictive coder (Daribo et al., 2012)	10 to 18 (55dB to 99dB)	-	Dynamic (float)	No	3D one-shot scanning systems; Physical grid pattern; Significate gain by exploiting Spatio-temporal correlations
V-PCC coder (Sebastian Schwarz et al., 2019)	Overall: 0.01 to 0.07 (29dB to 39dB) 500:1 to 100:1		Dynamic (integer)	Yes	VR (Virtual Reality)/AR (Augmented Reality)/MR (Mixed Reality) content; Efficient compression performance; Easy to reuse with video encoding advancement

In addition, such approaches can benefit from the expected evolution of image/video codecs. In particular, the V-PCC encoder provides state-of-the-art performances, outperforming the other approaches, with compression rates of up to 500:1.

A third family of approaches, described in the following section, notably attempt to directly exploit full 3D correlations.

2.4. 3D methods: Direct exploitation of 3D correlations

2.4.1. Tree-based methods

Within this framework, let us first discuss the octree-based techniques, which have become highly popular for yielding progressive 3D representations. An octree is a 3D data structure where each node represents a 3D bounding box that is recursively decomposed into eight children's leaves.

A neighborhood-based predictor for the point cloud geometry is introduced in (Yan Huang et al., 2006). An octree is built according to the pre-defined depth of the tree structure. It starts with one root node representing the entire space containing the point cloud. At each level of the octree, 8 leaf nodes are generated as children. They correspond to a uniform subdivision of the parent 3D cell. To each child, a binary occupancy code is assigned to the node, with a value of 1 for occupied cells and 0 for void ones. The binary occupancy code can be entropy encoded directly in single-rate compression or used to find the varying points in a progressive coding scheme. Each level of the octree can be interpreted as a LOD with further subdivision of the space. The octree construction process is terminated when reaching a user-defined, maximum depth level, which is equivalent to specifying the number of quantization bits for the geometry representation. In addition, the octree model is providing a stable structure of the 3D space, that can be notably exploited for motion estimation/compensation purposes in sequential frames.

A representative octree-based lossless intra-frame (i.e., compression/prediction are operated within a single point cloud frame in the context of this manuscript) compression method for point-cloud geometry is presented in (Diogo C. Garcia & De Queiroz, 2018). The used octree representation is completely described by the octree sequence **t**, which defines the binary pattern of the octree decomposition, starting from the root node (depth traversal of the tree).

Two additional sequences are used and exploited for generating a set of contexts:

- the *parent value* **v**, representing the corresponding binary value of its parent and specifying the corresponding occupancy.

- the *bit position* **p**, defining the corresponding ordered octree position of the point.

The **t** sequence is finally encoded conditionally with respect to the contexts generated by the **v** and **p** sequences, with both an LZW (Lempel–Ziv–Welch) (Sayood, 2012) and an arithmetic encoder. Let us also note that contexts can be further exploited for temporal geometry prediction.

The experimental results reported demonstrating that the use of contexts makes it possible to achieve 2% (respectively 5%) gains when compared to the same arithmetic encoder (respect to LZW encoder), directly applied to the **t** sequence.

In (Julius Kammerl et al., 2012), the octree-based representation is adapted for streaming applications. At each frame, the point clouds are decomposed into octrees. Then, the octree structures of two consecutive frames are built and serialized into binary sequences separately representing the occupancy code (Fig. II-4). They are compared using an exclusive disjunction operator (XOR) to detect the changing part.

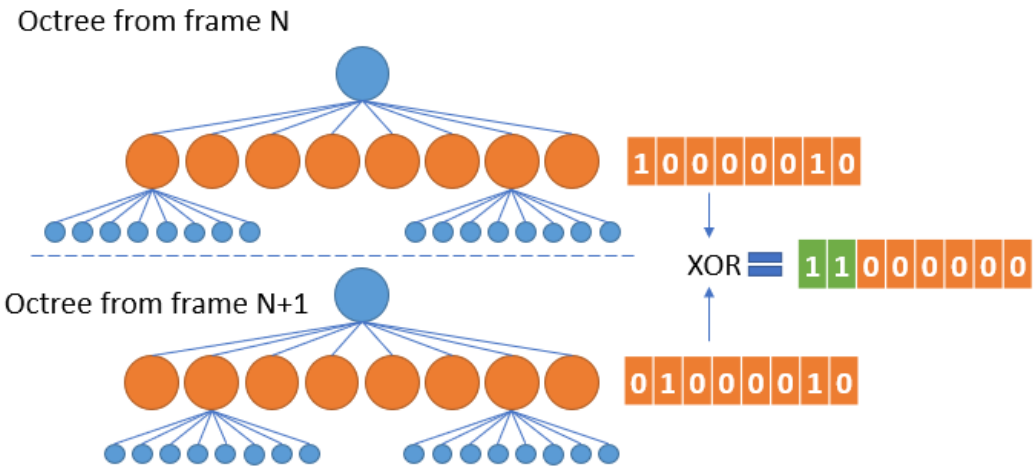


Fig. II-4. Illustration of the differential coding technique of consecutive octree data structures with XOR.

Finally, the resulting difference vector is entropy encoded. The method offers the advantage of a differential octree representation that saves computation and reduces memory requirements.

The lossless inter-frame compression method introduced in (D C Garcia & de Queiroz, 2017) also takes into account the temporal aspects of the 3D geometry sequence. The principle consists of reordering the points in the octree structure in the current 3D frame based on the prediction of the previous frame, before entropy encoding. In this way, it becomes possible to improve the temporal consistency of the points, notably for those that have similar properties and exhibit a relatively low motion. The experimental results reported show an average rate reduction of 30% when compared to the uncompressed octree representation. In addition, the approach yields average rate reductions of 24% and 22% when compared to the entropy-encoded octree using GZIP (Deutsch, 1996) and to the method presented in (Julius Kammerl et al., 2012), respectively.

Other tree-based methods exploit KD trees (J. M. Lien et al., 2010) or combinations of binary trees and quadtrees (Kathariya et al., 2018). The method introduced in (Bentley, 1975a) is a type of binary tree that is widely used for efficient neighbor search for the points' geometry because of its low computational complexity. In contrast with the octree-based approach, which builds a predictive-adapted structure over time, the KD-tree splits the space into half at each internal node. This induces a high dependency on the time-varying point cloud bounding box. Thus, it is considered particularly efficient for the intra coding mode. However, in the case of inter-frame coding, the KD-tree is computationally expensive to build.

In (Kathariya et al., 2018), the authors introduce a so-called binary-tree-quadtree structure. It uses a binary tree to split the point cloud after building a full octree to detect empty blocks and avoid encoding the full depth octree. Binary-tree blocks contain the local surfaces of the point cloud. Any local surface that exhibits flat characteristics is then projected onto a 2D plane and encoded with quadtrees. Other surfaces that are not flat enough are encoded with the full octree. The proposed scalable coding solution can efficiently compress point cloud data at variable rates, which leads to multiple LODs, corresponding to gradually increasing complexities of the 3D representation. This brings us into the discussion of the importance of disposing of multiple LODs since adaptive bitrate and quality are today highly demanded requirements for data transmission.

2.4.2. LOD-based methods

LOD-based methods are usually based on, or combined with a tree-structure decomposition. An early study (Y Huang et al., 2008) introduces a generic point cloud encoder, which creates an octree-based LOD structure. Here, the geometric center and the statistical averages of the normals and colors in each non-empty cell of the octree are estimated and represent the part of the model within each cell. Each level of the octree represents a LOD of the original 3D model. A novel adaptive color frame determination method is introduced to better exploit the color coherency. Using PCA, the coordinates in the RGB color space are transformed into a new frame where the correlation of the colors is more concentrated and is aligned with the eigenvectors from PCA. Therefore, adaptive color quantization is achieved with the Generalized Lloyd Algorithm (GLA). Similarly, GLA is also applied for clustering the points and generating a coarser LOD in (Y Fan et al., 2013).

Another LOD-based method, based on adaptive subsampling of the points in the cloud is presented in (Kitago & Gopi, 2006). A cost function, which integrates information about the redundancy and non-uniformity of the points, is defined and used to prioritize the points. The set of points is sub-sampled based on the dominant geometric features and the local sampling density of the model. The problem is then converted to a minimization of CSRBF (Compactly Supported Radial Basis Function) (Wendland, 1995) coefficients. The point's position and the relative normal information are used to estimate a surface along which the points are resampled and scattered in a way that minimizes the distortion of the reconstructed points. By selecting different cost functions, multiple LODs can be achieved with the help of a subsampling process, which takes into account various precisions of the reconstructed points.

A voxel-based, real-time, adaptive LOD approach is introduced in (Golla & Klein, 2015). The method can take into account the network condition as well as application-related parameters. Thus, the required responding time from the application and the network available bandwidth are used to compute on-the-fly a target compression ratio. The point cloud is decomposed into 2D images with a height map as in (Ochotta & Saupe, 2004) and then compressed by applying JPEG image coding. The voxels are resized into a higher or lower resolution in a way that ensures a trade-off between quality and compression ratio.

2.4.3. Clustering-based methods

The sparse voxel arrays are described as voxelized point clouds (Ricardo L. De Queiroz & Chou, 2017a). Voxels are acquired by decomposing the point cloud space into blocks, as illustrated in Fig. II-5, corresponding to a discretization of the 3D space on a regular grid, defined by given cell size.

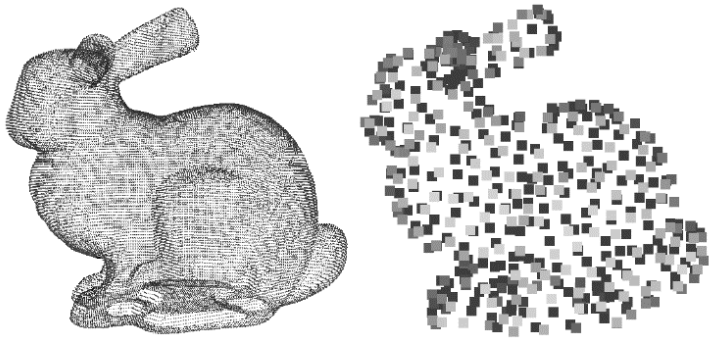


Fig. II-5. Point cloud object bunny (left) and voxelized object bunny (right).

Let us underline that the voxelization process leads to an integer representation of the geometric coordinates of the cloud points. By varying the step size chosen for the voxelization process, multiple LODs can be generated.

Leveraging on the graph-based motion estimation approach introduced in (D Thanou et al., 2015), the motion-compensation compression technique proposed in (Ricardo L. De Queiroz & Chou, 2017a) shows high efficiency and ensures real-time processing of voxelized point clouds. Similar results are also presented in (Loop et al., 2013)

The principle consists of extending the block-based, 2D video motion compensation techniques, widely used in various video compression standards, to 3D point clouds. In this case, 2D blocks are replaced by 3D cubes which are considered as "voxels". The matching procedure for consecutive frames determines the correspondence between voxels by minimizing the distance for both geometry and color information. The geometry residuals are considered as motion vectors and are associated with each voxel or voxel cube. Color residuals are considered as traditional color prediction errors.

Compared to the results presented in (Ricardo L. De Queiroz & Chou, 2016), where intra coding is used, the overall rate for geometry and color is about 3.7 bits per voxel for both cases. The approach in (Ricardo L. De Queiroz & Chou, 2017a) produces better color estimation but introduces some noise caused by geometry shifts. Optimization can be further investigated to select a fair tradeoff between geometry and color information.

Clustering-based methods have come into vision in recent years since the dense 3D point cloud can be segmented/clustered into different parts, based on color or motion information.

In (Q. Zhang et al., 2018), authors propose a dedicated point cloud color compression scheme based on hierarchical segmentation. Global color segmentation is first performed, to cluster all the points according to their color. To this purpose, the high-dimensional mean shift clustering algorithm (Georgescu et al., 2003) is here used. Then, a local segmentation is performed in the geometric space to guarantee the color consistency within the resulting patches. This property is notably useful for efficient intra prediction.

A similar method for color attribute clustering and encoding is proposed in (J.-M. Lien et al., 2009). Here, a skeleton-based motion estimation approach is used for deriving the kinematic parameters of a human body. The skeletons are generated offline with the method in (Cheung et al., 2003). The point cloud is clustered into point sets associated with the skeletons, to obtain a more accurate motion estimation.

Here, the rigid-body motion estimation is based on the ICP algorithm (Besl & McKay, 1992b), which has been also used for a similar purpose as shown in the article (Mekuria et al., 2017). The motion compensation residuals are stored within an atlas of residuals which is generated by projecting the residuals onto a regular 2-D grid embedded in a cylindrical coordinate system defined around the skeleton. They are finally compressed with the H.264 video codec.

The advantage of using a skeleton is that it can be used to interact with the objects in the virtual environment or to analyze the subject's movement. However, since the 3D point clouds are often noisy, the correlations between the points are not sufficient enough to support a unique skeleton registration in the 3D space. This issue can significantly penalize the compression process since inaccurate motion compensation can lead to 2D residuals that are inappropriate for efficient video coding.

Based on the geometric-driven approach introduced in (Morell et al., 2014), a clustering-based compression technique is introduced in (Navarrete et al., 2018). Both geometric and color features are considered for clustering the point cloud into a set of planar patches, by employing the method introduced in (Viejo & Cazorla, 2014). Then, a dedicated compression scheme is designed, based on the Gaussian Mixture Models (GMM). The principle consists of replacing the clustered points with each planar patch with the corresponding GMM approximation. Multiple LODs can be obtained by modifying the configuration of the model.

2.4.4. Transform-based methods

Transform-based methods attempt to extend the traditional signal/image transforms to point clouds. The difficulty here comes from the lack of topological information of the point cloud representation. In (Wiman & Yuchu, 2009b), a wavelet-based framework is proposed for LiDAR-like PCC, called WALZ (WAvelet LiDAR Zipper). Each coordinate component, X, Y, and Z, is independently wavelet-transformed as a 1D vector. In this way, highly correlated points, such as consecutively recorded points, will generate small and similar detail coefficients, which can be efficiently compressed.

A graph transform method is presented in (C Zhang et al., 2014) to decorrelate the attribute signal of the point cloud. Assuming that the point cloud is octree-decomposed, each occupied leaf node can be interpreted as a representative voxel as illustrated in Fig. II-6.

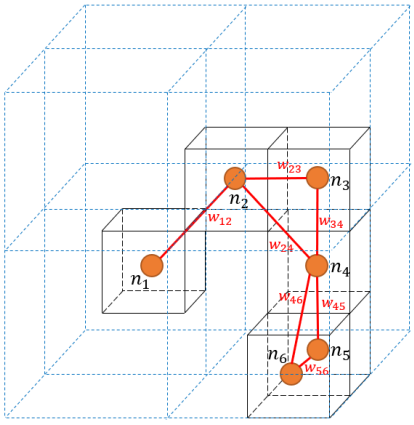


Fig. II-6. An example of constructed graph based on an octree-decomposed point cloud.

Here, n_i represents the i-th node of the graph, and w_{ij} is the weight value connecting the i-th and j-th nodes. The color signal defined on the constructed graph nodes can be transformed with the help of a precision matrix (Cha Zhang & Florêncio, 2013) and further entropy coded. Although the graph transform which is equivalent to Karhunen-Loeve Transform (KLT) in (Cha Zhang, Florencio, and Loop 2014) has better performance than traditional DCT (Discrete Cosine Transform) methods, it creates isolated sub-graphs when coding sparse point clouds. Moreover, the method is poorly adapted for inter-frame coding, since the graphs can vary significantly from one frame to another.

The so-called RAHT (Region-Adaptive Hierarchical Transform) approach introduced in (Ricardo L. De Queiroz & Chou, 2016) aims at improving color compression. A hierarchical, octree-based transform is here used. The voxels are grouped into a lower level of the octree to predict the colors of the nodes in the successive level. The process can be reduced to the Haar transform in the case where the prediction weights are uniform. The main advantage of the method is its low computational complexity, which makes it appropriate for real-time transmission.

The Graph Fourier Transform (GFT) proposed in (Dorina Thanou et al., 2016) is based on their previous work (D Thanou et al., 2015) and exploits both the spatial correlation and the temporal redundancy of the color and geometry sequences. As in (C Zhang et al., 2014), a graph is constructed from the initial 3D point cloud, with the help of an octree decomposition. A spectral representation of the resulting geometry and color signals is obtained with the help of the GFT which is defined through the graph Laplacian operator. The Gaussian Process Transforms (GPT) introduced in (Ricardo L. De Queiroz & Chou, 2017b) describes an approach to model the statistics of the point colors by assuming that they are samples of a stationary Gaussian process defined over an infinite 3D space. The covariance function of the Gaussian process is used to derive the covariance matrix of the colors, which in turn is used to obtain the KLT of the color signal.

In (Shao, Zhang, Li, Fan, et al., 2018), an optimized graph transform scheme for attribute compression is introduced. A KD-tree partition is here used to generate the graph. The Laplacian sparsity for graph transform is optimized with the help of an offline training stage. A Lagrangian, RDO-based quantization procedure is also considered. The experimental results show a significant gain on both transform efficiency and RD performance, for static point clouds. The applicability of the method to dynamic point clouds, under an inter-prediction mode, requires further studies.

Table II-4 summarizes the various 3D approaches discussed in this section. In a general manner, the fully 3D approaches offer useful solutions. In a majority of cases, they require integer/voxelized input representations.

Table II-4 Summary of fully 3D approaches.

Algorithm	Compression rates (bpp) for geometry (PSNR)	Compression rates (bpp) for attributes (PSNR)	Type of input (datatype)	Lossless support	Remarks
Generic progressive octree-based coder (Y Huang et al., 2008)	0.32 to 6.1 (53dB to 70dB)	Normal: 0.63 to 7.9 (18dB to 26dB); Color Y: 0.41 to 7.2 (23dB to 39dB)	Static (integer)	No	Potential use for lossless encoding
Hierarchical clustering coder (Yuxue Fan et al., 2013)	0.36 to 6 (49dB to 61dB)	Normal: 0.36 to 6 (19dB to 27dB)	Static (integer)	No	Low bandwidth network; Better reconstructed quality at low bitrates
Graph transform attribute coder (C Zhang et al., 2014)	-	0.16 to 5.36 (28dB to 52dB) for color Y	Static (integer) Voxelized	No	Efficient than Octree-based methods
RAHT coder (Ricardo L. De Queiroz & Chou, 2016)	-	0.85 to 2.5 (31.7dB to 39.8dB) for color Y	Static (integer) Voxelized	No	High computational efficiency; Real-time 3D video rate of 30 fps
GPT coder (Ricardo L. De Queiroz & Chou, 2017b)	-	0.54 to 2.11 (34.2dB to 41.8dB) for color Y	Static (integer) Voxelized	No	Gaussian processes model; Superior compression performance than RAHT
Hierarchical segmentation coder (Ke Zhang et al., 2018)	-	Overall 2.0 (51dB) for color Y	Static (integer)	Yes	Hierarchical segmentation based on attributes; Intra-cluster prediction method for lossless compression
Context-based intra coder (Diogo C. Garcia & De Queiroz, 2018)	Overall 1.95 (-)		Static (integer) Voxelized	Yes	Optimized contexts employed on octree for lossless coding
Differential octree coder (Julius Kammerl et al., 2012)	11:1 to 40:1	-	Dynamic (float)	No	Online compression; Streaming; Computation and memory-optimized
Image-based real time coder (Golla & Klein, 2015)	0.1 to 3.2 (65dB to 86dB)	-	Dynamic (float)	No	Online robotics applications; Low computational cost; Using JPEG2000
Graph-based motion estimation coder (D Thanou et al., 2015)	-	0.08 to 1.85 (34dB to 44.5dB) for color Y	Dynamic (integer) Voxelized	No	Accurate motion estimation; Graph transform
Optimized motion compensation coder (Ricardo L. De Queiroz & Chou, 2017a)	Overall 3.7 (-)		Dynamic (integer) Voxelized	No	Low bit rates targeted; Rate-distortion optimized motion compensation

However, in a majority of cases, such 3D approaches suffer from a relatively high computational complexity, since they require the creation of topological information to accurately describe the correlations between 3D points. This issue is particularly critical in the case of dynamic 3D point clouds. Moreover, in such cases, the additional 3D motion estimation/compensation procedure increases the computational burden. Let us observe that such approaches fail to achieve the compression performances offered by the 2D-based V-PCC approach, which fully takes advantage, pragmatically, of the efficiency of HEVC (High Efficiency Video Coding) 2D video codec. However, the motion estimation stage, performed in the 3D space, is much more precise than in the case of 2D, video-based approaches.

2.5. DL-based methods

Recently, the explosion of deep learning (DL) technologies has led to spectacular advances in various research fields, such as semantic classification (Qi et al., 2017), 3D reconstruction (Lin et al., 2018), object detection and tracking (Zhou & Tuzel, 2018). The 3D PCC field has also been impacted by this new paradigm, with numerous research works recently proposed in the literature (Tu et al., 2019b), (T. Huang & Liu, 2019), (Quach et al., 2019), (Tu et al., 2019a). Under this framework, two families of approaches can be identified, a first one geometric-driven and a second one texture-targeted.

2.5.1. Geometry-driven approaches

One of the first attempts of applying DL onto 3D PCC is presented in (Quach et al., 2019). A novel data-driven geometry compression method for static point clouds based on learned convolutional transforms and uniform quantization is here proposed. The model is a 3D convolutional auto-encoder composed of an analysis transform, followed by a uniform quantizer and a synthesis transform. Meanwhile, joint optimization of both rate and distortion using a trade-off parameter is performed during the training phase. In their recent work (Quach et al., 2020b), the impact of several parameters illustrated in (Quach et al., 2019) is evaluated through a series of experiments. Based on this evaluation, a hyper-prior model and deeper transforms, as well as fine-tuning of the loss function and adaptive thresholding have been proposed and managed to improve the compression performances.

Similarly, another early study (Guarda et al., 2019b) proposes to compress the geometry using an auto-encoder consisting of a small number of convolution and de-convolution layers for analysis and synthesis. The authors extend their efforts in (Guarda et al., 2019a) studying and analyzing the behavior and performance of their network in (Guarda et al., 2019b). Additionally, an alternative lower-complexity quantization approach combining implicit-explicit quantization is proposed in (Guarda et al., 2019a) to effectively reduce the number of DL models that need to be trained and stored.

A learning-based point cloud geometry compression method, so-called Learned-PCGC, is presented in (J. Wang et al., 2019). It consists of stacked 3D convolutions for the extraction of latent features and hyperpriors, a VAE structure for accurate entropy modeling of latent features, a weighted BCE loss function for training, and an adaptive thresholding scheme used for inferring the correct voxel classification. Several preprocessing steps are employed, including voxelization, scaling, and partition before feeding a model block-by-block to the network in a similar way as in (Quach et al., 2019).

Different from the previous methods, in (Yan et al., 2019), no voxelization is applied. Instead, raw point clouds are fed to the proposed architecture. The PointNet (Qi et al., 2017) approach is used to extract features from the unorganized 3D point cloud. Meanwhile, the synthesis transform is represented by a generative fully connected network.

2.5.2. Attribute-targeted approaches

A novel system for compressing point cloud attributes using DL is proposed in (Quach et al., 2020a). A 2D parameterization of the point cloud can be acquired by mapping the attributes from a point cloud onto a grid, making it possible to employ 2D image processing algorithms and compression tools. Inspired by (Yang et al., 2018), where the model is trained on a dataset to learn how to fold a 2D grid onto a 3D point cloud, instead, the method (Quach et al., 2020a) employs the folding network as a parametric function that maps an input 2D grid to points in 3D space. However, the lossy 3D-to-2D mapping introduces distortion for reconstruction and causes the low accuracy of the folding in geometrically complex parts of the point cloud.

Let us finally mention that two alternative architectures, including separate and joint compression of geometric and textural information for 3D point clouds, are presented and compared in (Alexiou et al., 2020). For the separate compression scheme, two cascading networks are trained separately to encode geometry and color individually. On the other hand, a unified model is proposed for jointly learning the point cloud encoding holistically, allowing fine-tuning for quality preservation per attribute.

The analysis of the state-of-the-art shows that there is a wide variety of 3D PCC approaches available today, dealing with the highly diverse nature of existing point cloud representations, going from sparse to dense, from static to dynamic, and from surface approximation to volumetric data. This analysis allows us to identify two main families of approaches:

- a purely geometric one, (as thee methods are based on octree decomposition), and
- a hybrid, geometry and video one, based on 3D-to-2D projections followed by 2D image/video coding.

At the same time, there is no solid proof that the DL-based methods achieve today a sufficient degree of maturity that can cover the various use cases addressed by traditional methods.

In terms of compression efficiency, the projection-based method is performant because it relies on highly optimized image/video coding advanced technologies. However, it works only when the initial point cloud is dense enough so that the 2D planes obtained by projection can be efficiently compressed by a 2D video codec. On the contrary, the geometric methods can handle both dense and sparse point clouds. Their main drawback is related to the additional computational cost introduced in the case of dynamic content (for more details, please refer to Chapter IV).

The variety of point clouds has been taken into consideration by the MPEG consortium, which initiated an activity of producing standards for 3D PCC.

After several years of experimenting with different methods, MPEG selected two approaches (projection-based and octree-based) and optimized them through a series of core experiments. They will be described in detail in Chapter III.

2.6. 3D PCC: What is missing?

In this chapter, the PCC approaches are considered as traditional ones such as 1D, 2D, 3D methods, and more recent ones exploiting DL technologies. All methods have their advantages and can be adapted accordingly to various applications. However, the coexisting limitations and challenges motivate us to explore what is still missing in both traditional and modern techniques.

2.6.1. The missing part of the traditional approaches

- *Direct exploration of the 3D correlation*

The advantage of the traditional approaches is that they can compress the 3D point cloud in a more general manner while providing promising performances. Nowadays, the 2D approaches represent the most well-suited ones for real-time mobile media applications using 3D point clouds. However, the 3D-to-2D projection can lead to incomplete recognition of the point cloud and are against the nature of its three-dimensional form. Meanwhile, a massive amount of works is required to balance the tradeoff between visual quality and transmission speed, notably for applications that require high precision and a real-time transmission at the same time, such as autonomous driving, robotics, and medical assistance.

- *Temporal compression for dynamic content*

With the advancement of acquisition techniques, 3D scenes and objects will become easier and faster to be captured. The 3D point cloud representation may become the next-generation image/video with a much denser resolution. Concerning the temporal aspect, it can be as dynamic as the 2D video today but in an unstructured way. This brings great challenges to correlate or to register between dynamic point clouds and to compress them. Yet only a few works such as (D Thanou et al., 2015) and (Dorina Thanou et al., 2016) have concentrated their focus on exploiting the temporal correlation between unstructured point clouds directly in 3D space. The development of effective and efficient temporal compression for the dynamic 3D point clouds remains a challenging task.

2.6.2. The missing part of the DL-based approaches

- *Attribute compression*

Since the DL techniques have recently been applied to the compression of 3D point clouds, not much work has been proposed for attribute compression. Since the attributes such as color, normal, reflectance, are tightly correlated to the geometry information, it is a must to design a joint learning process (Alexiou, Tung, and Ebrahimi 2020) to avoid cutting the connection between geometry and attribute information. There is still plenty of room for improvements in this field. To realize that, massive high-quality content is needed.

- *Temporal compression for dynamic content*

Similarly, most of the current DL-based methods are designed to compress single-frame 3D point clouds. Providing better compression performance in geometry information also holds the potential of greatly reducing the temporal redundancy between dynamic point clouds. Although some works such as (Owoyemi & Hashimoto, 2018), (Y. Wang et al., 2019), and (Liu et al., 2019) have been proposed in the field of classification and recognition using sequential point clouds as inputs, the compression aspect remains unexplored. Thus, the temporal convolutional network (TCN) (Bai et al., 2018) may be worth experimenting with.

2.7. MPEG 3D PCC standards

From the early 1990s, MPEG has been standardizing the compression standards for 3D graphic objects such as 3D meshes, animated objects, and point clouds. For the compression of static 3D mesh objects, MPEG-4 3D mesh coding (3DMC) was standardized in 1998. For generic and animated objects, the MPEG-4 animation framework extension (AFX) was standardized in 2003 and has been continuously extended until recently. The necessity of 3D PCC in MPEG was raised as early as 2013. MPEG began its exploration in 2014, which later generated much interest from the industry for immersive 3D services based on point clouds.

As we have seen, in recent years, numerous research works have considered the study of 3D PCC. However, the related constraints are treated in a rather heterogeneous manner by the various state-of-the-art approaches. The ISO/MPEG standardization committee has identified this bottleneck and launched a 3D PCC-dedicated activity in 2014. After several years of intensive developments, involving both academic and industrial actors, MPEG has issued two 3D PCC standards, as parts of MPEG-I Coded representation of immersive media. The approach adopted by MPEG is based on the optimization and extension of the most pertinent state-of-the-art methodologies while ensuring the coverage of an as-wide-as-possible field of applications and related functionalities. To this purpose, two distinct methodologies have been retained. The first one, called V-PCC, adopts a projection-based approach that makes it possible to take advantage of the highly optimized existing video codecs (e.g., HEVC). It mainly deals with highly dense 3D point clouds with uniform sampling and a low level of noise. The second one, called G-PCC, is more specifically dedicated to the management of sparse 3D point clouds. G-PCC is based on a purely geometric, octree-based decomposition approach and is useful for more diverse use cases being designed to handle a variety of point clouds ranging from human models to world-scale city maps.

The main advantage of V-PCC is the employment of efficient video technologies. Let us note that there is still room to further improve the coding efficiency of both G-PCC and V-PCC by developing non-normative tools.

The standardization of V-PCC and G-PCC is likely to be ongoing for some years because the first version of the standards mainly focuses on the provision of high compression efficiency. Extended functionalities, such as enhanced scalability, support of light-field data, increased bit-depths, and point density, are among the features that future versions of V-PCC and G-PCC may address.

Our work has been entirely carried out within the MPEG development framework and notably focused on the optimization of the V-PCC approach. One of the motivations of this choice concerns the significant enhancements that we may hope to achieve in particular when considering the temporal decorrelation aspects. For example, the current 40%-50% reduction (around 90% in traditional video compression) in the average bit rate of V-PCC can be further increased by improving the inter-coding mode. Moreover, the test materials of V-PCC are more accessible and license-free compared to the ones used by G-PCC which makes it more academic friendly. With this in mind, we decided to focus our study on improving the temporal compression of the V-PCC test model along with the MPEG standardization process.

The MPEG V-PCC and G-PCC methodologies are described in detail in the following chapter.

CHAPTER III. EXTENDED STUDY OF MPEG V-PCC AND G-PCC APPROACHES

In this chapter, we first detail the two MPEG V-PCC and G-PCC methodologies adopted. For each of them, an experimental evaluation of the related compression performances is also proposed. Afterward, we propose a finer analysis of the V-PCC limitations, carried out on an experimental basis. The considered V-PCC test model is version 1.0, and our analysis specifically concerns the inter-coded HEVC bitstream. By highlighting the poor prediction achieved between consecutive frames (which naturally leads to a high bit consumption), it is shown that the temporal consistency of the packed images needs to be optimized to enhance the V-PCC compression performances.

Institut Polytechnique de Paris

91120 Palaiseau, France

3.1. V-PCC methodology

The V-PCC approach transforms the 3D geometry and attributes data of the point cloud into a set of 2D patches. The patches are then mapped onto a predefined set of 2D planes through orthogonal projections, without self-occlusions, and with limited distortion. A mapping between the point cloud and a regular 2D grid is then obtained by packing the projected patches within the 2D domain. This mapping is then used to generate 2D images representing the point cloud geometry as well as its attributes. Since the mapping between the point cloud and the 2D grid is not bijective, an extra binary image, referred to as the occupancy map, is needed to distinguish between the filled (i.e., associated with a point) and the empty (i.e., not associated with any point) cells of the grid. Similar to the geometry and attribute video sequences, the occupancy map video sequence is compressed using a 2D-based image or video codec. Additional information, specifying per patch information (e.g., 3D/2D patch location and projection plane index) is required to reconstruct the 3D point cloud from the 2D geometry, attribute, and occupancy videos. To this purpose, V-PCC introduces a new codec specifically optimized to handle the patch information sub-stream. This sub-stream occupies a relatively small amount (less than 5%) of the overall bitstream. Additional information needed to synchronize and link the video and patch sub-streams is also signaled in the bitstream. All these components can be multiplexed into a single bitstream or encoded as separate tracks of an ISOBMFF (ISO base media file format) container (ISO/IEC, 2015).

Similar to video encoding standards, the MPEG V-PCC only specifies the decoding process. The encoding process is out of scope and it is left to an encoder manufacturer to create the most appropriate encoder that generates bitstreams conforming to the V-PCC specification. The V-PCC encoder diagram is illustrated in Fig. III-1.

Let us observe the intensive use of video codecs in the PCC coding architecture. Leveraging traditional video codecs to encode point clouds requires mapping the input point cloud to a regular 2D grid. The objective is to find a temporally-coherent, low-distortion, injective mapping, which would assign each point of the 3D point cloud to a cell of the 2D grid. Maximizing temporal coherency while minimizing distance/angle distortions enables the video encoder to take full advantage of the spatio-temporal correlation of the point cloud geometry and attributes signals. An injective mapping guarantees that all input points are captured by the geometry and attributes images and can be reconstructed without loss. By simply projecting the point cloud on the faces of a cube or a sphere does not guarantee lossless reconstruction due to auto-occlusions (i.e., auto-occluded points are not captured), and can generate in practice significant distortions. Therefore, a clustering process is performed before projection.

Let us now further detail the various stages involved.

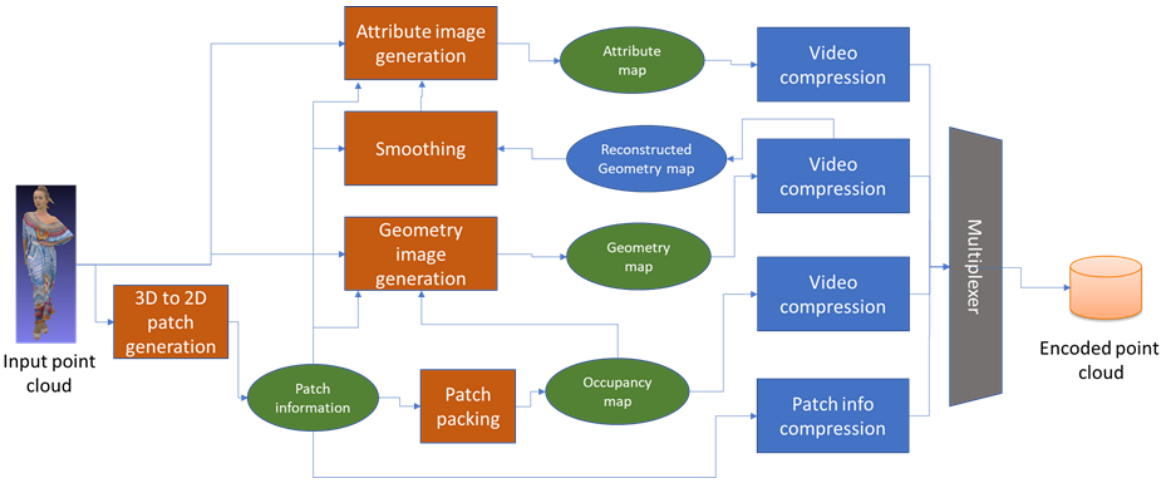


Fig. III-1. V-PCC encoder diagrams.

3.1.1. Patch generation

V-PCC decomposes the input point cloud into a set of patches, which can be independently mapped, through a simple orthogonal projection, onto a 2D grid without suffering from auto-occlusions and without requiring any re-sampling of the point cloud geometry. Furthermore, the patch generation process aims at generating patches with smooth boundaries, while minimizing their number and mapping distortions. Patch generation is outside the scope of the standard, however, an implementation of a heuristic segmentation approach, illustrated in Fig. III-2, is proposed as an example.

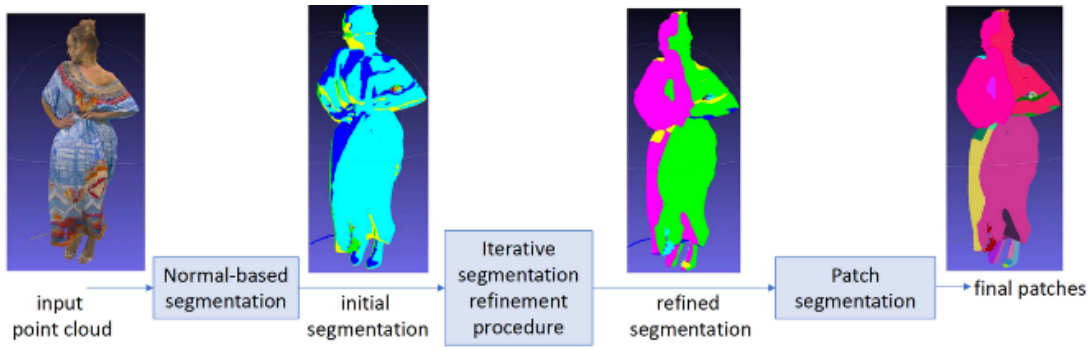


Fig. III-2. Patch Generation Process.

First, the normal at every point is estimated as described in (Hoppe et al., 1992). Initial clustering of the point cloud is then obtained by associating each point with one of the six-unit cube-oriented planes. More precisely, each point is associated with the plane that has the closest normal (i.e., maximizes the dot product of the point normal and the plane normal). The initial clustering is then refined by iteratively updating the clustered index associated with each point, based on its normal and the cluster indexes of its nearest neighbors. The final step consists of extracting patches by applying a connected component extraction procedure.

The number of generated patches largely depends on the complexity of the point cloud. Usually, less than a hundred patches are needed to capture more than 95% of the points. The remaining points are usually isolated points or isolated groups of points that would require a high number of patches to capture. Depending on the application constraints, the encoder may decide to discard the remaining points at the risk of introducing holes and cracks in the point cloud. To avoid such artifacts with a limited RD performance impact, V-PCC makes it possible to control the minimum number of points per patch, allowing the user to filter out isolated points or small groups of isolated points that may have a limited impact on visual quality. The encoder may also choose to store all or a subset of the remaining points into special patches, named unprojected or independent point patches. The cartesian coordinates of the points corresponding to an independent point patch are then directly embedded into the bitstream, without the use of any prediction method. The independent patch geometry and attribute values could be stored in the same video sequences as the regular patches or in separate ones.

3.1.2. Patch packing

The way of packing the different patches obtained by projection in a 2D frame is also not part of the standard but several methods have been experimented with during the development of the standard. The most efficient one iteratively tries to place patches in an image with size (W × H) pixels. Initially, the patches are ordered by their corresponding size. Then, the location of each patch is determined by an exhaustive search in raster scan order, of the first location that guarantees overlap-free insertion. More precisely, the bounding box of the current patch expressed in terms of T × T blocks, where T is the packing block size, should not contain any T × T block belonging to a previously placed patch. Such constraint makes it possible to determine for each T × T block the patch to which it belongs, by analyzing the 2D bounding boxes of all patches. To improve the fitting chances, different patch orientations are also allowed. In the case where there is no space available for the next patch, then the height H of the image is doubled and the insertion of the current patch is evaluated again. After the insertion of all patches, the final height of all the frames in the sequence is trimmed to the minimum needed value.

3.1.3. Geometry, occupancy Map, and attribute image generation

The patch packing procedure determines the projection of the 3D patches onto the 2D image canvas. A geometry image can then be generated, by inserting the depth values in the luminance channel of the image. However, a patch can have multiple points being projected onto the same 2D pixel location. To handle such cases, the V-PCC standard allows the encoder to use up to 16 layers to store overlapping points. In particular, let us assume that we have a set of points $H(u, v)$ being projected onto the same (u, v) location. V-PCC currently uses two layers. One layer, referred to as layer 0, stores the point from $H(u, v)$ with the lowest depth value $D0$. A second layer, referred to as layer 1, stores the point with the highest depth value within a user-defined interval $[D0, D0 + D]$. The user-defined interval size D represents the surface thickness and can be used to improve geometry coding and reconstruction.

V-PCC has the option to code the second layer as either a differential layer from the first layer or use its absolute value. Moreover, the layers can be coded in two separate video streams, or temporally interleaved into one single stream. The interval size D can be used to improve depth reconstruction since the reconstructed values must lie within the predefined interval. The second layer can be dropped altogether, in which case the decoder could generate the far layer from the near layer and the interval size D. Furthermore, both frames can be sub-sampled and spatially interleaved to improve reconstruction quality and coding efficiency.

The mapping of geometry from 3D to 2D does not use all the pixel positions in the geometry image, leaving some empty spaces. A binary image, also known as occupancy map, indicates with value 1 the used positions, and value 0 the empty ones. The occupancy map can be compressed at a lower resolution, which is determined by a user-defined parameter. This leads to higher compression efficiency but may affect the geometry reconstruction. Fig. III-3 shows an example of geometry, occupancy map, and attribute (e.g., texture) images.

Fig. III-3. Example of geometry (left), occupancy map (center), and attribute (e.g., texture) (right) images.

3.2. Experimental evaluation of V-PCC

To evaluate the performances of MPEG V-PCC, we have conducted several experiments using the TMs provided by the MPEG, with both the encoder and decoder. Let us note that better implementations of the encoder may be available in the future because the TMs are mainly designed to produce conformant bitstream and exclude many non-normative tools, which can improve coding efficiency. However, the results presented in this section provide a good illustration of the MPEG V-PCC capabilities.

The evaluation dataset comprises five sequences of dense point clouds (Fig. III-4). The sequences have numbers of points ranging from 800 thousand (K) to 1 million (M), with a frame rate of 30–50 fps. Each coordinate is stored in 10-bit precision. The texture information is stored with an 8-bit precision for each color channel.

Fig. III-4. Evaluation dataset from (E. d'Eon, B. Harrison, T. Myers, 2017) and (J. Ricard , C. Guède, R. Doré, n.d.).

For the geometry metrics, MPEG retained two PSNR-like measures, denoted by metrics D1 and D2, and corresponding to variants of the well-known Hausdorff distance. Both metrics measure the geometric distortion of a decoded point cloud B compared with a reference point cloud A by averaging all distances between each point in point cloud A and its nearest neighbor in point cloud B. The main difference between D1 and D2 is that D1 uses a point-to-point distance, whereas D2 uses a point-to-plane distance. The D1 and D2 distances are put under the form of a PSNR score for deriving rate-distortion (RD) curves. To compute the quality of the attributes, the same nearest-neighbor registration is used between point clouds A and B. The PSNR score is computed by considering the difference in attribute values for corresponding points. To globally compare RD performance of two codecs C1 and C2, MPEG uses BD-rates (Bjontegaard, 2001) (Fig. III-5).

First, the RD curves for both codecs are computed by encoding the same sequence with different parameters. Each encoding configuration corresponds to a point in the RD curve. The horizontal and vertical axes respectively correspond to bitrate and geometric/attribute quality (e.g., PSNR), respectively. The RD curve is created by interpolating between those points. The BD-rate provides a global relative performance measure, defined as the average (or integral) over the area between the two curves (green shade) of the bitrate difference between the two codecs for the same quality.

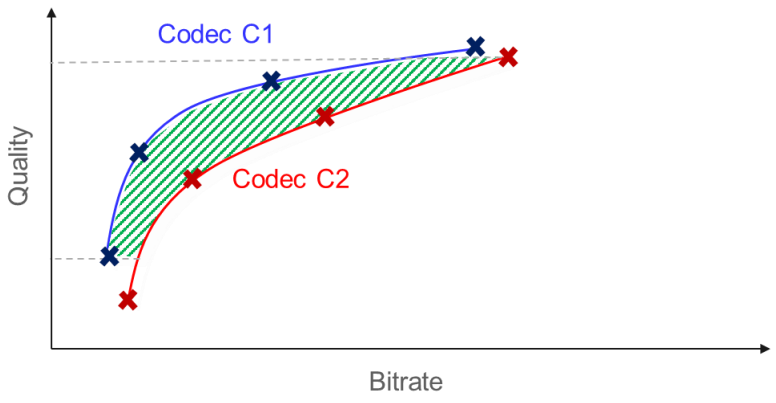


Fig. III-5. An example of the Rate-Distortion Metric noted as BD-Rate.

The V-PCC codec uses several video codecs to encode different bitstreams. Besides, it encodes some metadata for representing the relations between the 3D points and 2D patches. The composition of a V-PCC bitstream is measured in different coding modes in terms of percentages of the total bitstream size. A first example is illustrated in Fig. III-6, for the *longdress* sequence and two coding modes respectively denoted as Random Access (RA or INTER) and All Intra (AI or INTRA). In the AI configuration, each frame is independently encoded as a static point cloud with no reference to any other frames in the sequence. In the RA configuration, a hierarchical predictive structure is employed, based on the used video codec to provide predictions between pictures, which roughly reduces the bitrate by about 50%.

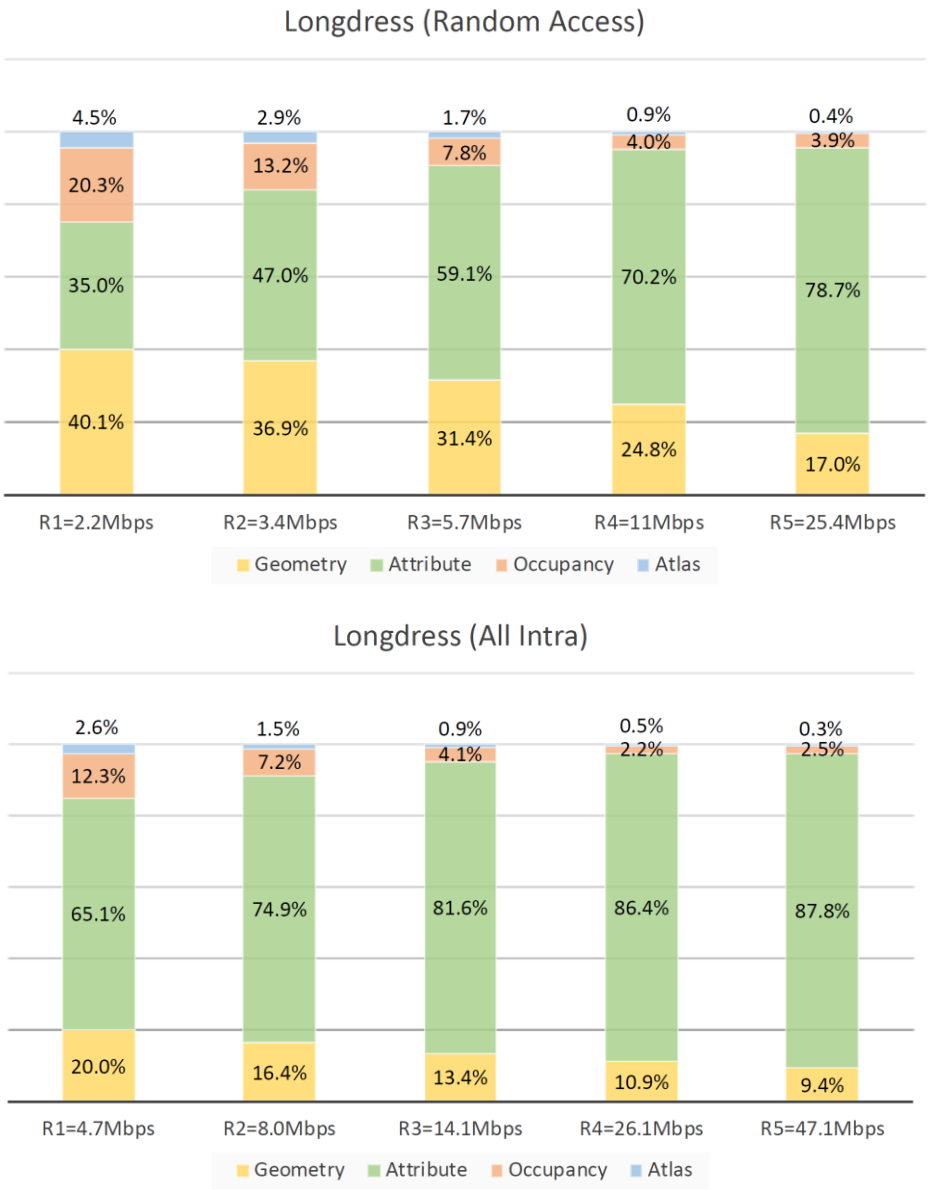
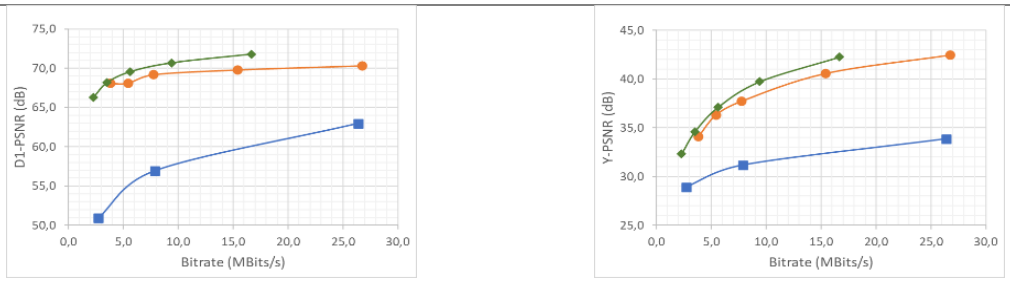


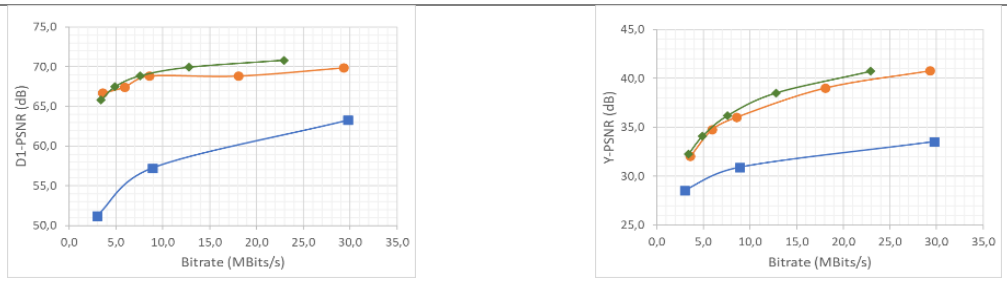
Fig. III-6. The composition of the bitstream components for the longdress sequence (the other sequences have similar distribution).

Fig. III-6 shows the contributions in the percentage of the total bitstream size of the V-PCC sub-streams corresponding to geometry, attribute, occupancy (all represented as 2D videos), and atlas parameters (represented as metadata). The results showed that 95%–99% of the total bitstream was dedicated to the video sub-streams (with both geometry and attributes), which were efficiently handled by the existing video codecs.

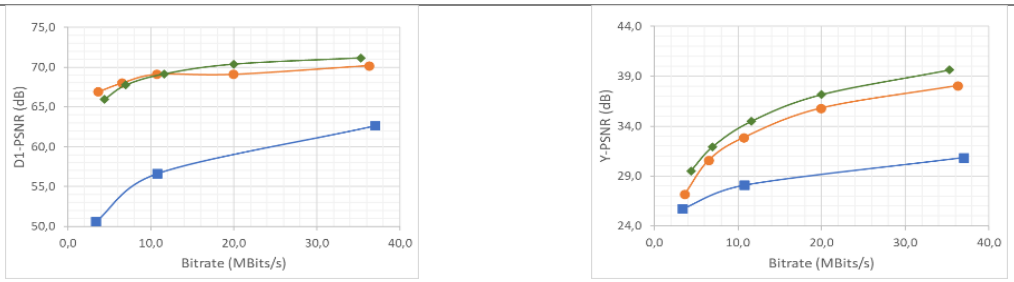
When V-PCC activity was initiated in 2017, the state-of-the-art point cloud codec was the one implemented in the PCL (Rusu & Cousins, 2011), based on octrees. For comparison, this codec (Mekuria et al., 2017) was used as a reference anchor method for benchmarking the performances of the approaches (3DG, 2015). In response to the MPEG CfPs (Call for Proposals) on 3D PCC techniques, eight different technologies have been proposed. The best among them—a projection-based method—was selected and transformed into TM v1. It can be considered as a prototype of the current V-PCC TM. During the development of the standard, 13 consecutive versions of TM have been developed, in order to achieve better performance. The optimization mainly concerns the refinement of 3D-to-2D projection, the improvement of patch packing and padding strategies, and the data signaling approach. In our work, we evaluated the progress of the encoding efficiency for the sequences in the evaluation dataset using V-PCC TM v1 as a benchmark and v11 as the one providing state-of-the-art performance. The results are illustrated in Fig. III-7, for the AI mode and in Fig. III-8, for the RA mode.



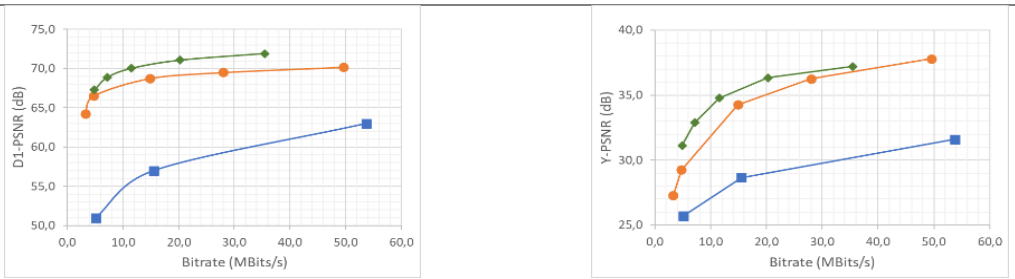
loot



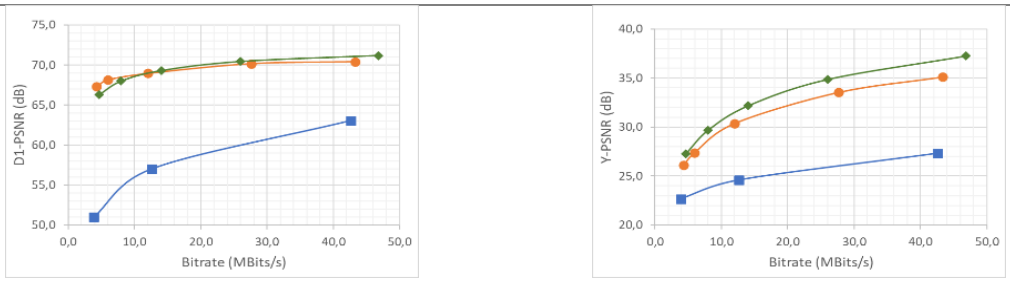
redandblack



soldier

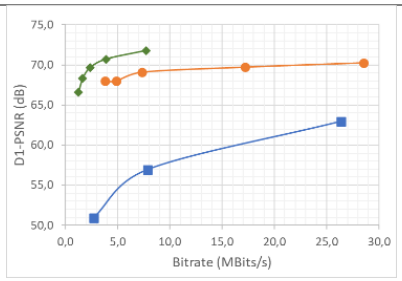


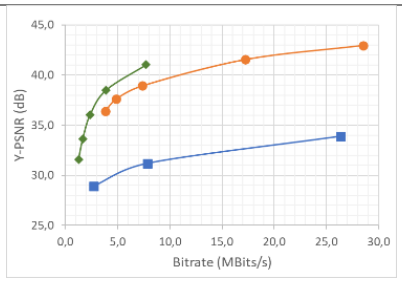
queen



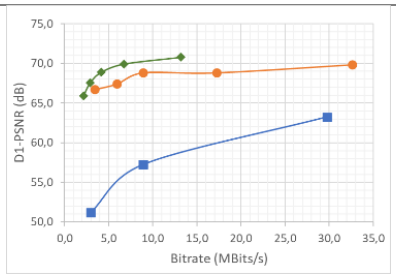
longdress

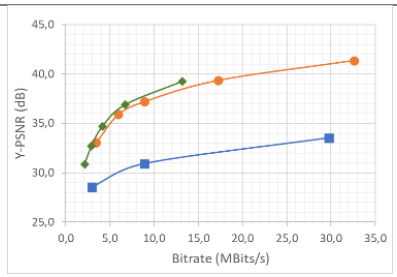
Fig. III-7. Geometry (left) and attribute (right) PSNRs comparison between anchors (blue), TM v1 (orange), and TM v11 (green) for the AI mode.



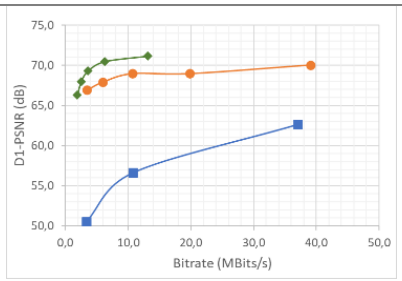


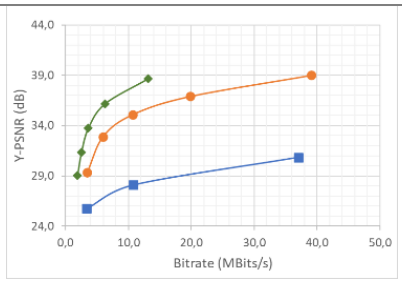
loot



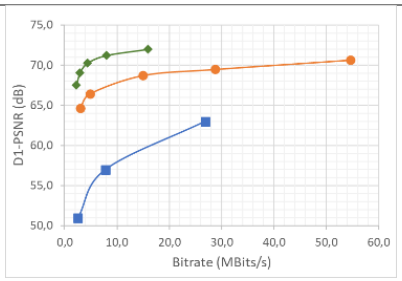


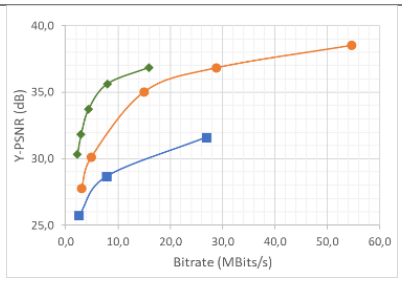
redandblack



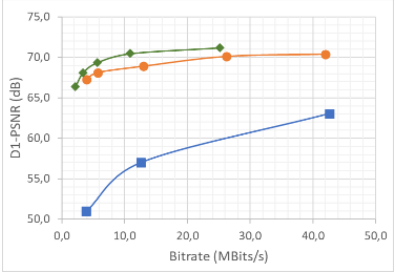


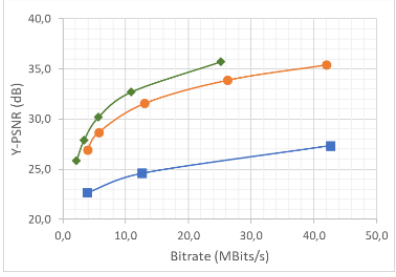
soldier





queen





longdress

Fig. III-8. Geometry (left) and attribute (right) PSNRs comparison between anchors (blue), TM v1 (orange) and TM v11 (green) for RA condition.

Fig. III-7 shows the RD curves of three codecs: the green curves correspond to V-PCC TM v11, named TMC2, the orange curves correspond to V-PCC TM v1 initially submitted as a response to MPEG CfP, whereas the blue curves correspond to the anchor codec. As expected, TM v1 and TM v11 outperformed the anchor codec, and TM v11 outperformed TM v1.

While the progress to the anchor codec is easy to observe in Fig. III-8, noticing the differences between TM v1 and TM v11 is nontrivial. Therefore, we illustrate the overall bitstream size reduction, integrated over the five encoding rates (BD-rates), for each sequence and the two modes (AI and RA) in Table III-1. This table should be interpreted as follows: considering a sequence, e.g., *redandblack*, the bitrate difference between the two TMs integrated over the five rating points was 28% and 58%, respectively, for the geometries in the AI and RA modes in favor of TM v11. The same interpretation was valid for the texture coding, 14% (6%), 11% (4%), and 46% (40%) for each color channel in the AI mode (RA mode).

Table III-1 The percentage of BD-rate reduction for geometry and texture in AI and RA modes.

AI / RA	**Geometry**	**Texture**		
Sequence	**D1**	**Y**	**Cb**	**Cr**
loot	45% / 74%	18% / 25%	8% / 18%	8% / 14%
redandblack	28% / 58%	14% / 6%	11% / 4%	46% / 40%
soldier	1% / 59%	23% / 32%	12% / 21%	10% / 10%
queen	52% / 81%	29% / 45%	59% / 69%	70% / 74%
longdress	15% / 52%	22% / 25%	32% / 39%	44% / 47%
Average	28% / 65%	21% / 27%	24% / 30%	36% / 39%

Notably, for all metrics (geometry and texture), global improvements were obtained from TM v1 to TM v11 for all sequences. For the geometry, the improvements were more significant for the RA condition than for AI. For the texture metrics, the behavior is more heterogeneous.

The V-PCC approach strongly relies on video codecs. However, the 2D frames obtained by projecting point clouds are not similar to the ones that video codecs usually handle. Since the clustering operation (and therefore the computation of the 2D patches) is performed for each frame individually (Notably, this is the TM behavior, and the clustering is non-normative), patches may not preserve the same index. Therefore, the 2D patches may not preserve the same location within the 2D atlas. Fig. III-9 illustrates two consecutive 3D point cloud frames for the *longdress* sequence (frames 12 and 13) and the corresponding texture maps obtained by projecting the point clouds.

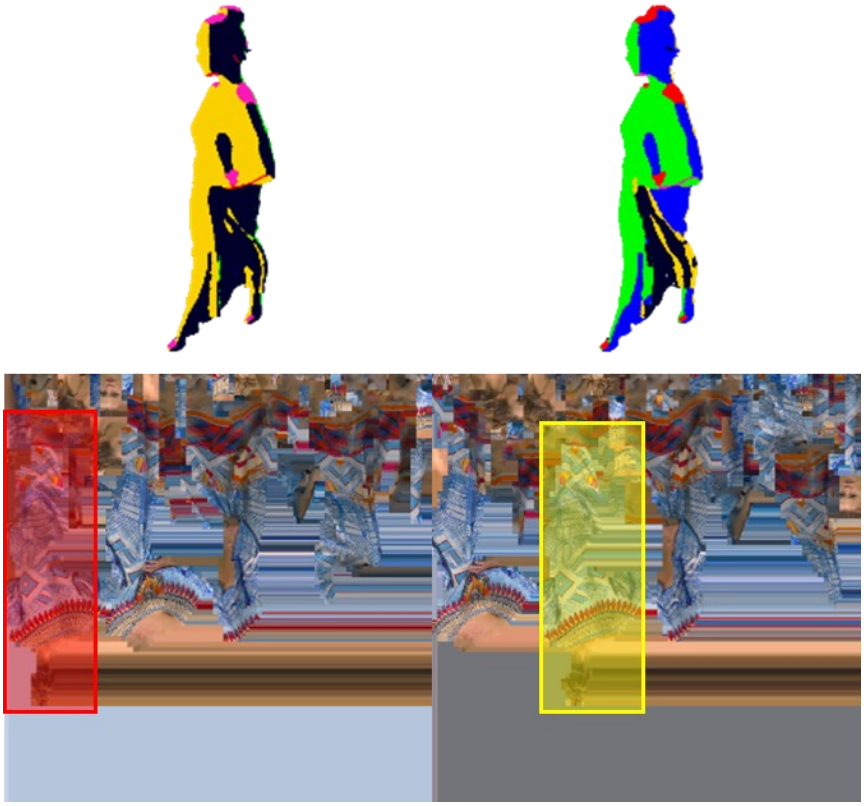


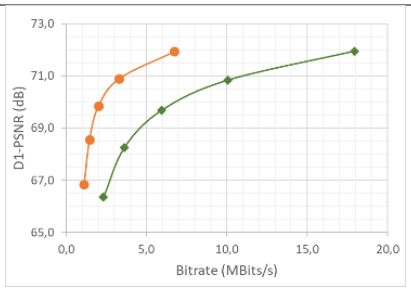
Fig. III-9. Two consecutive frames in 3D and the corresponding texture maps in 2D.

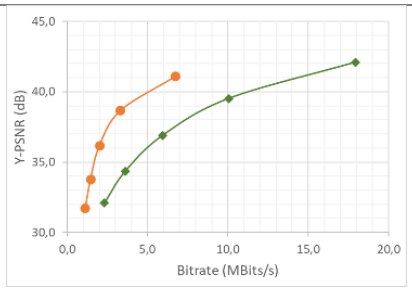
The first observation concerns the noisy structure of the 2D frames due to the patch-packing operation. We also observe some temporal inconsistency between 3D and 2D spaces. In particular, small 3D motions can lead to significant variations in the 2D positioning of patches. We evaluated the performances of the TM v11 in terms of how motion was addressed by comparing the results of the RA vs AI modes. The obtained results are presented in Fig. III-10.

As expected, the RA mode globally performed better. However, the difference is not as significant as in the case of natural image video encoders. The corresponding BD-rates reductions are summarized in Table III-2.

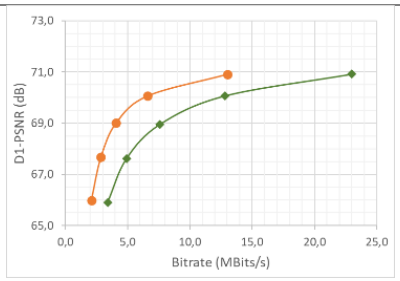
Table III-2 BD-rate reduction when RA mode is used instead of AI.

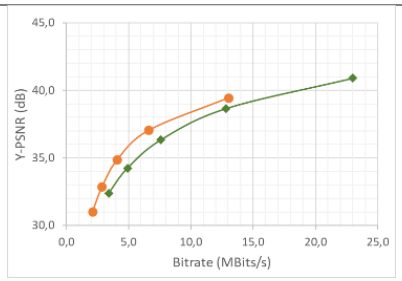
RA vs AI	**Geometry**	**Texture**		
Sequence	**D1**	**Y**	**Cb**	**Cr**
loot	56%	45%	56%	54%
redandblack	43%	22%	33%	18%
soldier	68%	61%	69%	68%
queen	42%	48%	58%	55%
longdress	58%	32%	41%	38%
Average	53%	42%	51%	47%



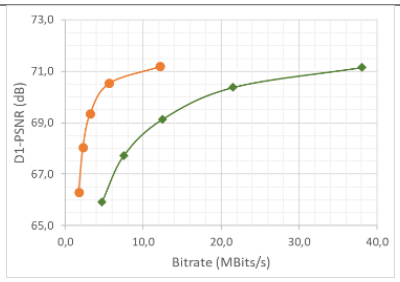


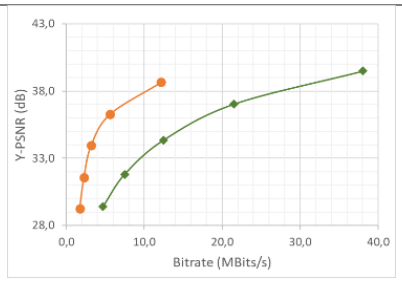
loot



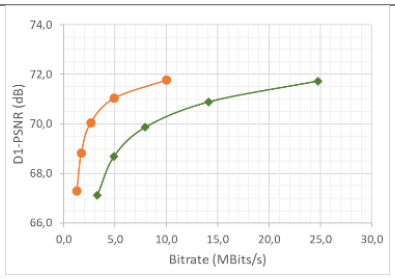


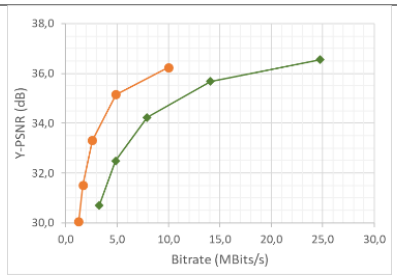
redandblack



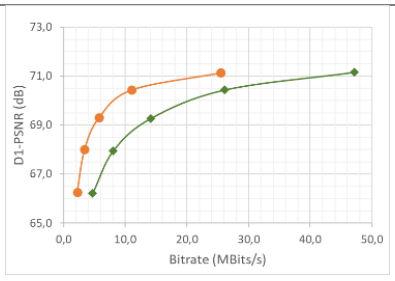


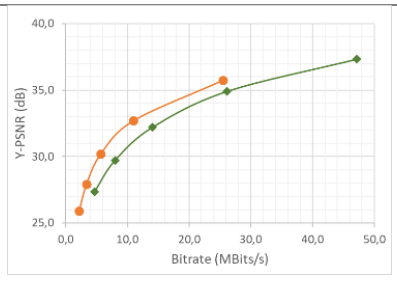
soldier





queen





longdress

Fig. III-10. Geometry (left) and attribute (right) PSNRs comparison between AI (green) and RA (orange) for TM v11.

The reduction in average was between 40% and 50%, which was less than the value of 90% that video codecs can usually obtain using RA mode on traditional video content. These results show that there is still room for improving the V-PCC inter-coding RD performance.

Apart from normative tools required in the standard specifications, it is possible to employ additional non-normative tools to improve the compression performances. Two examples are shown here illustrating the potential gain: (1) 3D motion estimation (L. Li et al., 2020) and (2) occupancy-based rate-distortion (L. Li et al., 2019). The results are reported in Table III-3 and Table III-4, respectively.

Table III-3 BD-rate reduction when Optimization with 3D Motion Estimation is applied.

	Geometry	**Texture**		
Sequence	**D1**	**Y**	**Cb**	**Cr**
loot	6%	9%	17%	17%
redandblack	6%	9%	13%	9%
soldier	15%	21%	30%	31%
queen	6%	7%	5%	6%
longdress	6%	7%	9%	9%
Average	8%	11%	15%	14%

The motion estimation was performed in 3D space by searching the nearest position of a point. Then, a motion vector was generated in 2D to guide the video codec in performing motion prediction for the attributes. For encoding dynamic content, a significant performance improvement can be expected with this tool.

Concerning the occupancy-based rate-distortion values, the results are reported in Table III-4. The main idea of this optimization is to ignore padded pixels while computing distortion in the video RDO block.

Table III-4 BD-rate reduction when Occupancy-based RDO is applied.

	Geometry	**Texture**		
Sequence	**D1**	**Y**	**Cb**	**Cr**
loot	23%	24%	21%	21%
redandblack	11%	13%	11%	13%
soldier	23%	27%	21%	22%
queen	21%	24%	19%	20%
longdress	9%	11%	10%	10%
Average	17%	20%	17%	17%

We can observe that, here again, significant improvement can be achieved. Other than the potential gain we may obtain from the additional non-normative tools, V-PCC could seamlessly leverage the expected improvements introduced by future video compression standards.

The MPEG V-PCC was designed for efficient coding of dense dynamic point clouds. To cover the numerous use cases for point clouds, MPEG developed, in parallel, a more generic technique, exploiting a geometry-based approach. The so-called G-PCC standard is presented in the following section.

3.3. G-PCC methodology

The general block diagram of the G-PCC encoder is illustrated in Fig. III-11. The G-PCC encoding is a sequential process, which first processes geometry and then the attributes. This behavior, useful for lossy coding, is directed by the need to first map the original attributes on the reconstructed (encoded and decoded) geometry and the encoding of these mapped attributes instead of the original ones.

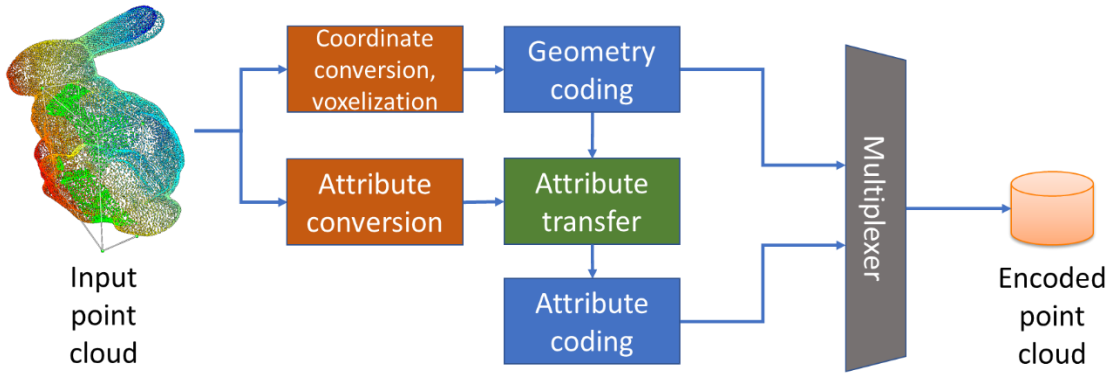


Fig. III-11. G-PCC encoder diagram.

3.3.1. Coordinate transformation and voxelization

The input point coordinates are represented in their coordinate system provided by the acquisition device and are generally expressed as floating-point numbers. In such cases, prior to G-PCC encoding, a conversion to integer values is performed. In order to normalize the original point cloud, the input world coordinate system is converted to the so-called Frame Coordinate System (FCS), represented on N bits for each of the 3 cartesian dimensions. Therefore, the bounding box of the original point cloud is geometrically aligned with the FCS while preserving the aspect ratio of the bounding box.

The G-PCC encoder processes integer values, therefore a voxelization is needed. The quantization process is applied to the points represented in FCS, converting the input bit-depth from N-bit (per dimension) to M-bit representation (where $M < N$). The values of N depend on the precision of the acquisition system and the values of M by the desired precision of the reconstruction.

Once the quantization process is applied, it is very likely that some new points are duplicated (they have the same quantized coordinate values). Such points are called duplicate points and they may have distinct attribute values (because the original un-quantized points have distinct attribute values). Different strategies can be used to compensate for the attribute accuracy loss due to the geometry quantization and a recoloring process is included in the encoding scheme.

3.3.2. Geometry coding

G-PCC includes two modes of encoding the geometry. The first one is using an octree decomposition of the 3D space and can potentially be complemented by a tool called *trisoup* used to terminate prematurely the octree decomposition. The second is a simpler predictive mode, where the points are sequentially traversed and the value of the current point is predicted from the previous (up to 3) ones. This second mode is mainly appropriate for objects acquired by lasers when the acquisition order can be used to predict the values of point coordinates. The overall scheme of these encoding modes for the geometry is presented in Fig. III-12.

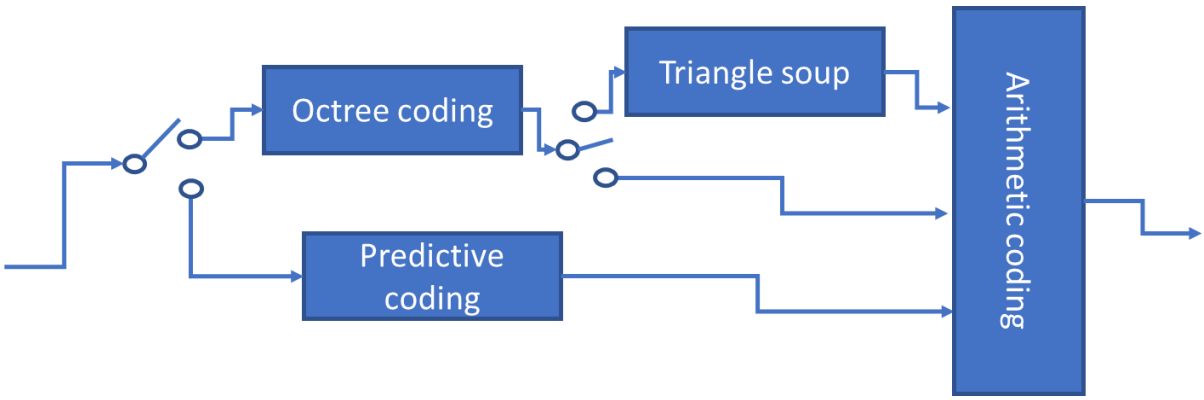


Fig. III-12. G-PCC geometry encoder diagram.

Let us now detail each of the processing blocks involved.

3.3.2.1. Octree coding

Once represented in M bits per each axis, each point position belongs to a cube, with coordinate values ranging from $(0, 0, 0)$ to $(2^M, 2^M, 2^M)$. The voxels in the cube may be empty or occupied by the points in the point cloud. The occupied voxels correspond to the quantized point locations. An octree decomposition is then carried out. As illustrated in Fig. III-13, a cube can be decomposed into eight sub-cubes of equivalent size. The cubes can be decomposed iteratively until the size of each sub-cube is the same as a voxel. A sub-cube that does not contain any occupied voxel will not be decomposed any further. Every node of an octree can be represented by either 0 (empty) or 1 (occupied). At each node decomposition in the octree, the eight child nodes are each represented by a binary value denoting its occupancy, which leads to an 8-bits pattern. The octree structure is then traversed in a breadth-first search manner, the nodes at the lower level are encoded earlier than those at the higher level. The 8-bits patterns of child nodes are then entropy coded with the help of an arithmetic encoder.

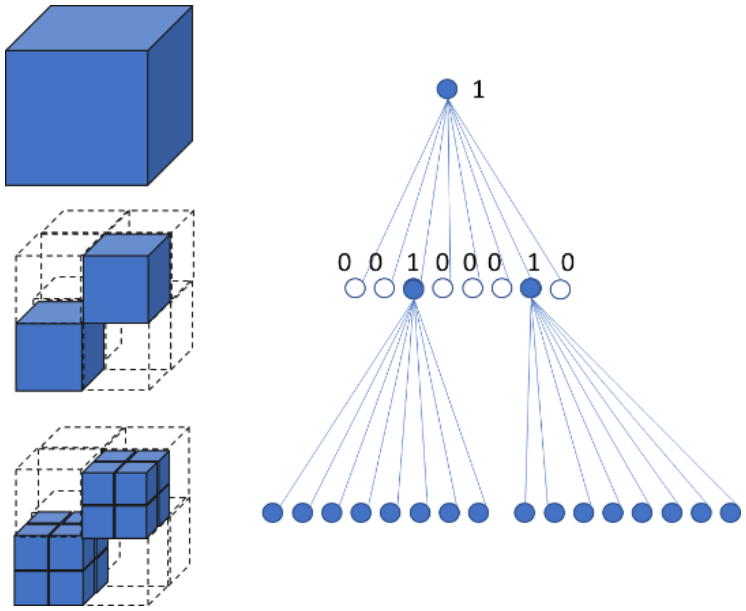


Fig. III-13. Octree decomposition of a cuboid.

In addition to the traditional octree decomposition, G-PCC includes specific encoding tools to optimize the octree analysis: the direct coding mode, the planar mode, the QTBT mode, and the angular/azimuthal mode.

a) Direct coding mode

Octree decomposition works effectively when there are many points in the given cube. When an object presents isolated points (which occurs in the case of sparse objects), the 8-bit pattern used to describe the child level is more expensive than simply encoding directly the voxel containing the isolated point, a mechanism called Direct Coding Mode (DCM).

Therefore, G-PCC exploits the correlation of parent-level node information to determine if DCM can be applied. More specifically, there are two conditions to determine the eligibility of DCM: 1) the parent node is the only occupied child and the grandparent is either the only occupied child or one of the two occupied children, and 2) the parent node is the only occupied child and the face-neighboring 6 cubes of the parent node are not occupied.

b) Planar mode and Quadtree/Binary tree (QTBT)

In the case of sparse point clouds, it is suboptimal to predict the coordinates from the neighbors. The planar mode is dedicated to points (even when they are far from each other) that are coplanar (e.g., ground, building walls, etc.). When such a case is identified, the number of possible cases used for prediction is reduced, as illustrated in Fig. III-14.

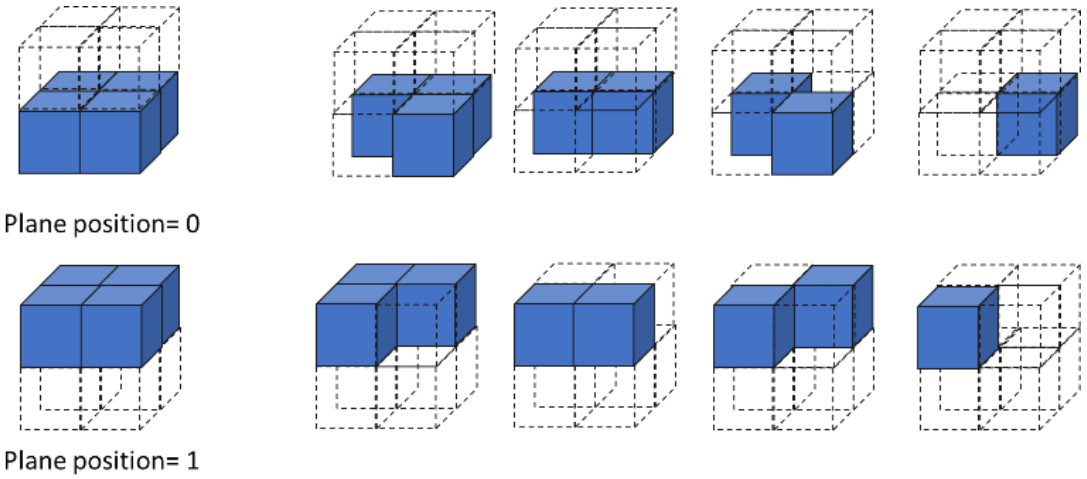


Fig. III-14. Example of planar mode where occupied nodes are in the same plane.

In the Planar mode, the occupancy pattern of the octree is compacted by the correlation from the neighborhood patterns. However, it is possible to express the distribution not by octrees but by Quad-trees or Binary-trees, thus making it possible to reduce the occupancy code to 4 or 2 bits. This signaling mode can be used for the partial areas of the point cloud, or from the specific depth level of an octree. Using QT or BT only for the simple pattern leaf nodes, the occupancy code can be more compact without losing accuracy (from 8 to 4 and 2 bits respectively, as illustrated in Fig. III-15).

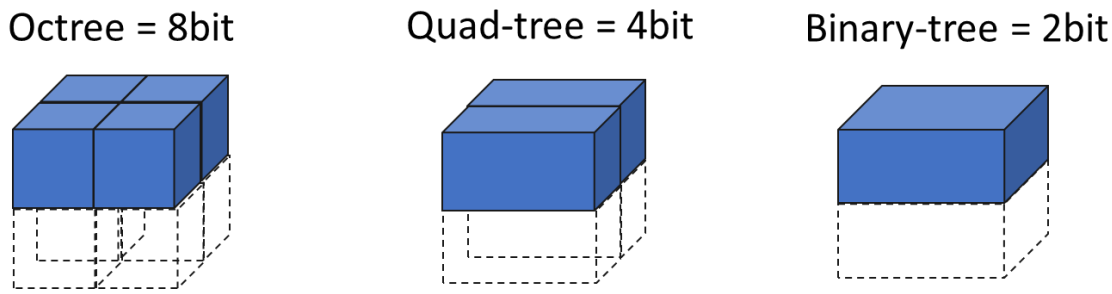


Fig. III-15. Encoding with QT/BT reduces the occupancy code.

The angular mode was introduced to improve the planar coding mode. It uses the child node angular distance from laser angles to improve compression of binary occupancy coding through the prediction of the plane position of the planar mode and the prediction of z-coordinate bits in IDCM nodes.

3.3.2.2. Triangle soup (Trisoup) coding

Trisoup-based geometry compression (TGC) is an additional geometry compression tool that can be used complementary to the octree decomposition. In an octree, an occupied leaf node represents the final resolution of point coordinates. In TGC, an occupied leaf node represents a 3D cube that may contain one or more points inside. Each leaf-level occupied 3D cube is represented as surfaces of triangle strips where each triangle can be formed with vertices on the edges of the 3D cube.

This triangle information is encoded instead of point coordinates belonging to the 3D cube. At the decoder side, an appropriate number of point coordinates is generated from the triangle surfaces. Depending on the formation of input point coordinates, there can be 3 to 12 vertices identified. With these vertices, there can be one to ten triangles in a 3D cube.

3.3.2.3. Predictive coding

Unlike the octree decomposition which recursively divides the bounding box, predictive geometry coding is a point-by-point coding method. This method is intended to cover the low-delay use cases, for example, to handle, in real-time, the input from 3D sensors. In this mode, the encoder orders a set of points as a prediction tree, as illustrated in Fig. III-16.

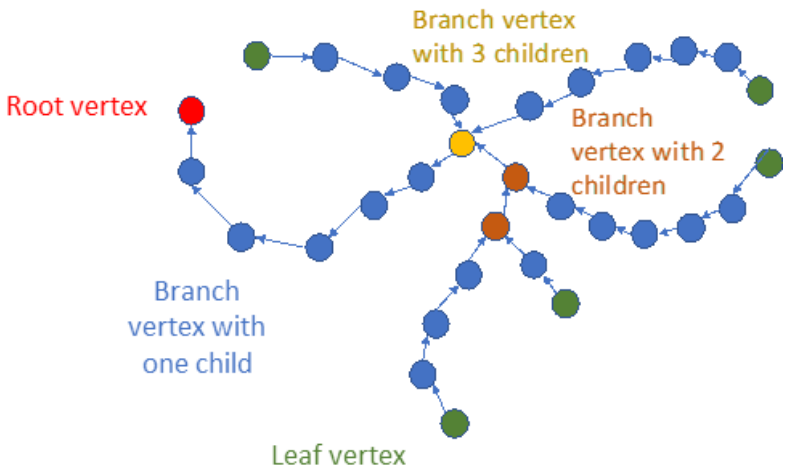


Fig. III-16. Example of the geometry prediction mode. A prediction tree can branch into up to three sub-trees at a point.

In this example, a tree starts from the red point then branches at the yellow point into three. Each vertex can be predicted only from its ancestors in the tree. Different colors are used to indicate the number of children for each branch vertex. The positions of the vertices are encoded by storing the chosen prediction mode and the obtained prediction residuals. Arithmetic coding is used to further compress the generated values. The performances of the encoder depend on the construction of the trees and on the *a priori* information (such as the scanning order) that can be used.

3.3.2.4. Arithmetic coding

The final step of the geometry encoder is the arithmetic encoder, contextualized with respect to the geometry encoding mode (octree, trisoup or predictive). For the octree mode, several improvements are provided in G-PCC. In order to maximize the gain of arithmetic coding, context-based probability modeling is used. There are two different ways to entropy code the 8-bit pattern: bit-wise and byte-wise context-based arithmetic coding. This context-based arithmetic coding is bypassed for some specific cases (such as DCM), where data is statistically not well-suited for entropy coding.

There are quite a few context models to efficiently encode 8-bit patterns in G-PCC. As illustrated in Fig. III-17, up to six neighbors of the parent node can be occupied. Therefore, 64 (i.e., 2^6) different neighbor configurations (NCs) are possible.

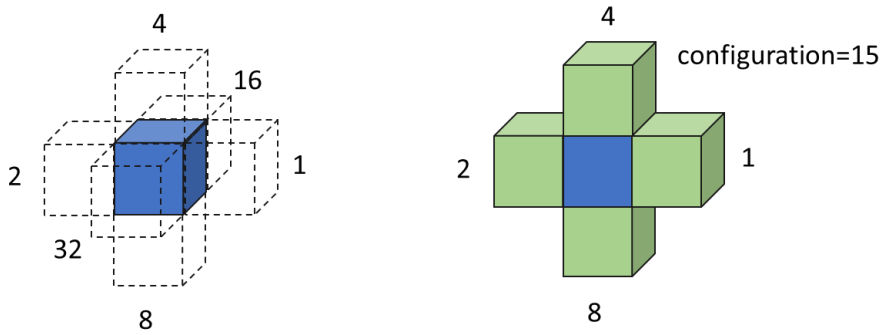


Fig. III-17. Neighbor configuration NC (left) and example for NC=15 (right).

When NC is zero, it means that the parent node does not have any occupied neighbor node. If the parent node has only one occupied child, it is subject to be encoded with DCM. Otherwise, the 8-bit pattern is encoded with the context of NC = 0.

In addition to NC, a further context refinement is carried out by adding the previously encoded bits in the 8-bit pattern of the same node which increases the number of contexts at 128 (2^8) multiplied by 64. G-PCC reduces the 128 possible combinations at 10 invariant cases by using symmetries and rotations.

An additional improvement of compression efficiency is obtained by exploiting encoded child nodes that are neighboring the current node as shown in Fig. III-18. By examining the already-encoded child nodes, it is possible to accurately determine whether an occupied neighbor block is truly occupied and thus connected to the current node (Fig. III-19).

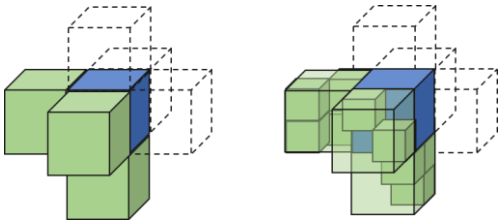


Fig. III-18. The exploitation of already encoded child nodes for further compression efficiency.

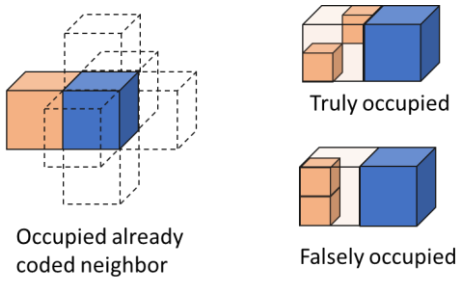


Fig. III-19. Determination of truly- or falsely-occupied neighbor.

3.3.3. Geometry reconstruction

The geometry reconstruction process decodes and reconstructs the point coordinates. The reconstruction does not mean a full decoding process since it does not contain any arithmetic decoding. The main reason for the geometry reconstruction is to use it for recalculation of attribute data according to the decoded geometry information. If the compression of position information is lossless, the geometry reconstruction and attribute transfer processes are not necessary and parallel (geometry and attribute) encoding can be conducted.

3.3.4. Attribute conversion, recoloring, and coding

In a point cloud, there may be many different attribute types (e.g., color, normal, reflectance, laser ID, etc.). However, the current G-PCC supports only color and reflectance data. The color attributes in G-PCC are represented in the YCbCr color space. A color space conversion is needed if the original acquisition color space is different. One can refer to the standard conversion process defined in related standards such as ITU-R BT.709 (International Telecommunication Union, 2002). For encoding the reflectance data, the same encoding path of color data is used, with the only difference being that the dimension of reflectance is one instead of three.

The attribute conversion process recomputes the attributes based on the reconstruction geometry when the compression of geometry is lossy. In this case, most point locations after encoding are different from those of the original input point cloud. Therefore, the values of attributes such as color and reflectance need to be recomputed. There are many possible ways to recolor the points, a simple one is to use the weighted average using the nearest points from the input point cloud. Since this process is done only at the encoder and is outside of the standardization scope, there is room for further improvement. Yet, good recoloring directly influences the performance of the encoder.

There are two methods to encode the attributes in G-PCC: RAHT and LoD, each one with some advantages and limitations with respect to the density of point clouds. The overall schema of these encoding modes for the attribute is presented in Fig. III-20.

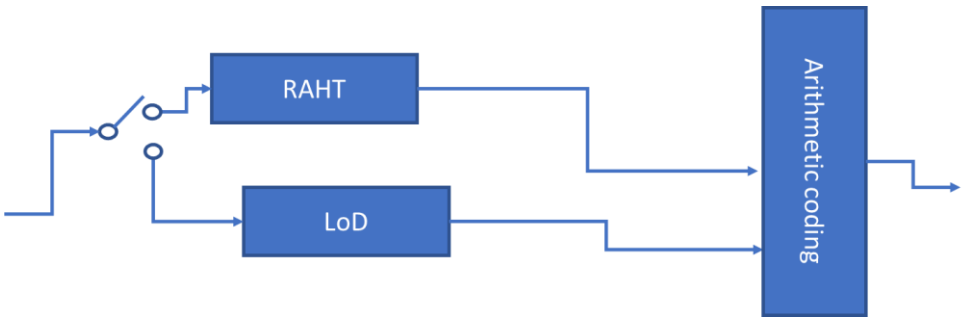


Fig. III-20. G-PCC attribute encoder diagram.

In the following sections, we provide the details for each of these blocs.

3.3.4.1. RAHT

RAHT (R L de Queiroz & Chou, 2016) is a 3D transform designed for the data that may contain empty locations in a 3D space. Point coordinates represented as an octree structure are transformed by RAHT. RAHT provides a hierarchical spatial transform similar to wavelet coding in image and video compression, which provides data compaction to the lowest frequency component. RAHT becomes a 3D version of the Haar transform if all the points in a given 3D cube are occupied. RAHT is performed in a hierarchical manner such that the current level transformation is performed on the lower level (or child level) transformed coefficients.

3.3.4.2. LOD generation

The LOD generation process aims at providing spatial scalability to G-PCC. The points are separated into layers (an Euclidian distance can be used, but the way in which the LODs are generated is out of the scope of the standard). The attributes associated with the points are predicted from points that are already encoded at the same layer or from the points at a lower layer. An example of LOD generation is illustrated in Fig. III-21. The LOD generation process has to generate a different encoding order of attributes from that of geometry. For example, a distance-based clustering is done in Fig. III-21. The LOD generation process is an encoder issue, which means that there can be different (and even more efficient) strategies to group points into layers. Once the order and grouping of points in layers are determined, the attribute values at each point are predicted and their differences are entropy coded using arithmetic coding.

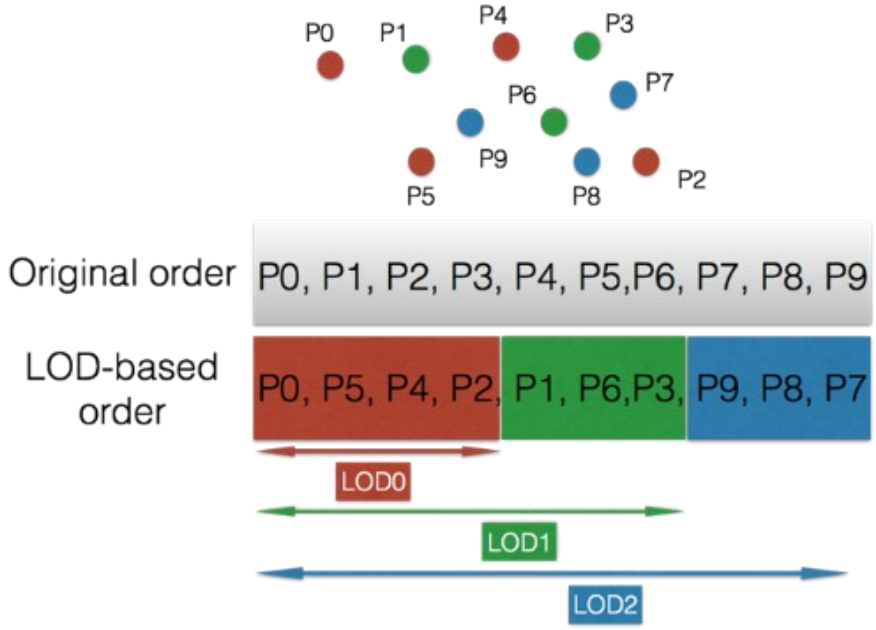


Fig. III-21. Example of LOD generation process.

The prediction process determines a prediction value of the current point from the nearest already encoded points. Searching the nearest points increases the encoder computational complexity as well as compression efficiency.

If there are k attribute values $(a_i, i = 0, \dots, k-1)$, these attribute values are predicted by a linear interpolation process based on the distances to the nearest neighbors of the current point i.

3.3.4.3. Quantization and arithmetic coding

The RAHT transformed coefficients and residual prediction values are uniformly quantized. The quantization is controlled by a global parameter denoted by QP, which is used just like in the previous MPEG AVC/HEVC/VVC standards. Finally, the quantized coefficients are entropy coded with run-length coding and binary arithmetic coding.

3.4. Experimental evaluation of G-PCC

The main difference between G-PCC and V-PCC is that G-PCC is designed to handle various point cloud categories, whereas V-PCC is optimized for dense surface-like point clouds. These point clouds are usually uniformly voxelized, where the coordinates are quantized into integer values and contain a low level of noise with limited bit-depth positions. The dataset for G-PCC evaluation includes individual frames from some of the sequences used for V-PCC evaluation and other categories of point clouds, which are described in the following paragraphs.

The first category is the one with static human models (single frames extracted from the sequences introduced in Chapter III-3.2. In this case, the point clouds are dense, smooth, and uniformly sampled. The bit depth for positions varies between 10 and 12. The number of points is between 800 K and 3 M points.

The second category is called “façade & buildings.” The density of the point clouds varies from sparse to dense with a medium to a high level of noise. The bit depth of such content can increase from 12 to 20 bits per coordinate. The number of points can reach 20 M points, and the textures are represented with 8 bits per color channel.

Façade 9 (C. Tulvan, A. Gabrielli, 2016) with around 1.6 M points

House without a roof (C. Tulvan, A. Gabrielli, 2016) with around 5 M points

Palazzo Carignano (GTI-UPM, n.d.) with around 4.2 M points

Arco Valentino (GTI-UPM, n.d.) with around 1.5 M points

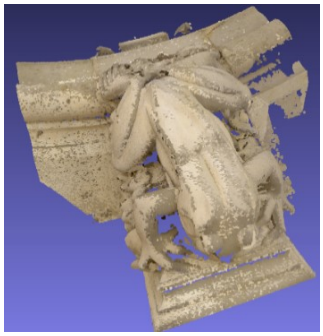
Fig. III-22. Objects from the "façades & buildings" category.

The third category, so-called "objects" contains various archeological, artistic, and ordinary objects. The point cloud density varies from sparse to dense, with a low to a high level of noise. Bit depth for positions varies from 11 to 20 bits. The number of points can be as low as 300 K and goes up to 64 M points. Some of these objects are illustrated in Fig. III-23.

Egyptian mask (C. Tulvan, A. Gabrielli, 2016) with around 0.2 M points

Statue Klimt (C. Tulvan, A. Gabrielli, 2016) with around 0.5 M points

Shiva (C. Tulvan, A. Gabrielli, 2016) with around 1 M points



Frog (C. Tulvan, A. Gabrielli, 2016) with around 3.6 M points

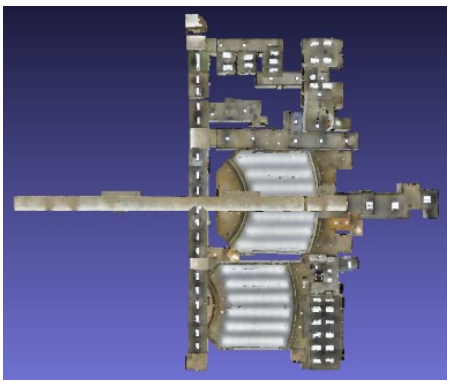
Head (C. Tulvan, A. Gabrielli, 2016) with around 14 M points

Unicorn (H.-L. Guillaume, T. Doneux, A. Schenkel, n.d.) with around 63 M points

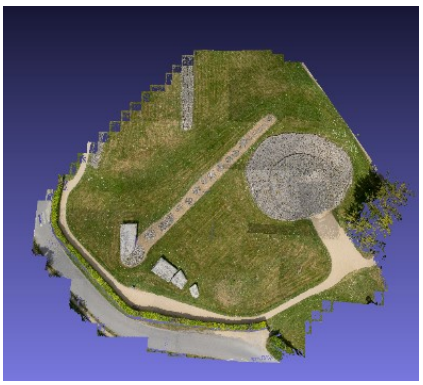
Fig. III-23. Objects from the "objects" category.

The fourth category is called “landscapes” (Fig. III-24). It contains large point clouds with 44–72 M points. The bit depth for positions varies from 14 to 20 bits per coordinate. The density varies from sparse to dense. The point clouds are usually tiled into smaller chunks before encoding in order to allow efficient navigation and RA.

Stanford 2 (Stanford 2 dataset n.d.) with around 47 M points



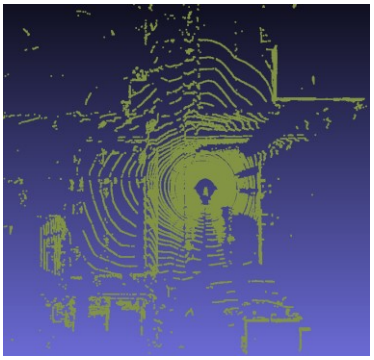
Stanford 4 (Stanford 2 dataset n.d.) with around 43 M points



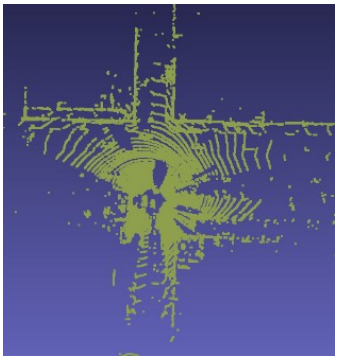
Landscape 14 (C. Tulvan, A. Gabrielli, 2016) with around 72 M points

Fig. III-24. Objects from the “landscapes” category.

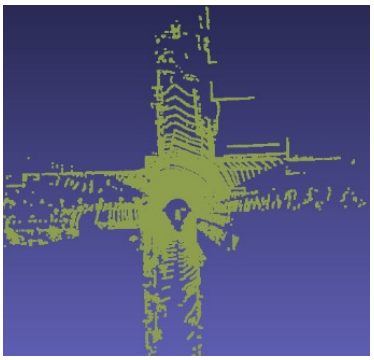
The fifth category, named “frame-based LiDAR,” contains point clouds captured by LiDAR scanners mounted on moving vehicles. The point clouds in this case are sparse, with a low to medium level of noise (Fig. III-25). The sampling is highly irregular and depends on the acquisition process implemented by the LiDAR scanner. The positions are quantized into 18 bits per coordinate. The objects include reflectance information associated with the point cloud. The number of points per frame varies between 27 and 84 K. The frame rate is between 5 and 10 fps.



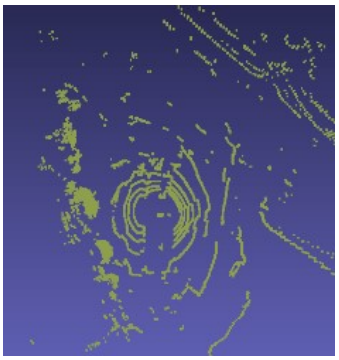
Ford 01 (Pandey et al., 2011) with around 100 K points per frame



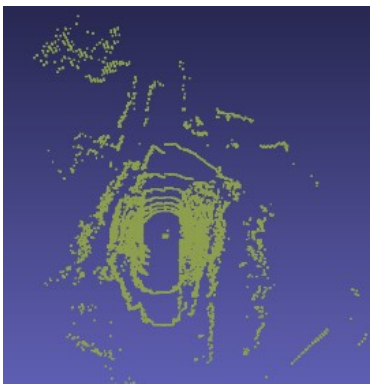
Ford 02 (Pandey et al., 2011) with around 100 K points per frame



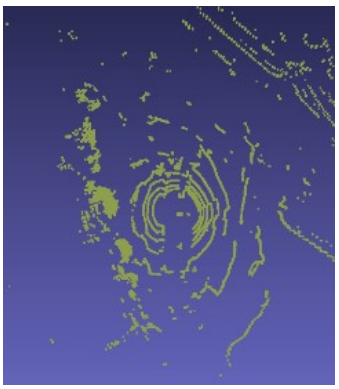
Ford 03 (Pandey et al., 2011) with around 32 K points per frame



Junction approach (BlackBerry Limited and QNX Software Systems Limited, 2018) with around 32 K points per frame



Motorway join (BlackBerry Limited and QNX Software Systems Limited, 2018) with around 32 K points per frame

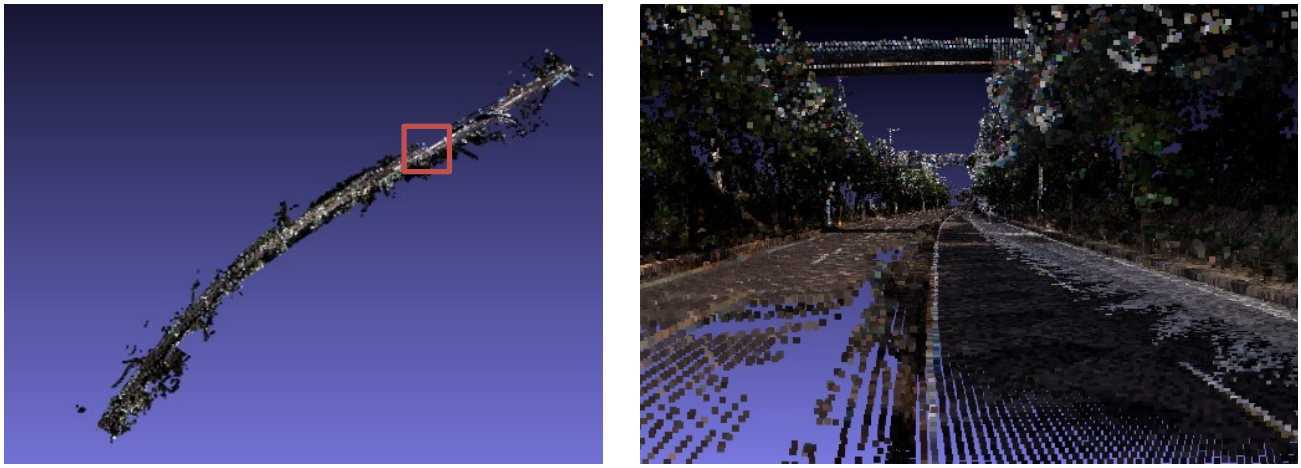


Navigating bends (BlackBerry Limited and QNX Software Systems Limited, 2018) with around 32 K points per frame

Fig. III-25. Objects from the "frame-based LiDAR" category.

The last category is called "fused LiDAR," containing large point clouds with 5–20 M points, which cover an extended geographic area (Fig. III-26). The point clouds are usually generated by fusing multiple frame-based LiDAR acquisitions into a single point cloud. The point clouds are sparse with a low to medium level of noise. The sampling is highly irregular. The positions are encoded with 20-bit precision. The point clouds have both texture and reflectance information.

City tunnel (R. Cohen, H. Ochimizu, D. Tian, 2017): entire scene (left) with around 19 M points and a local zoom (right)

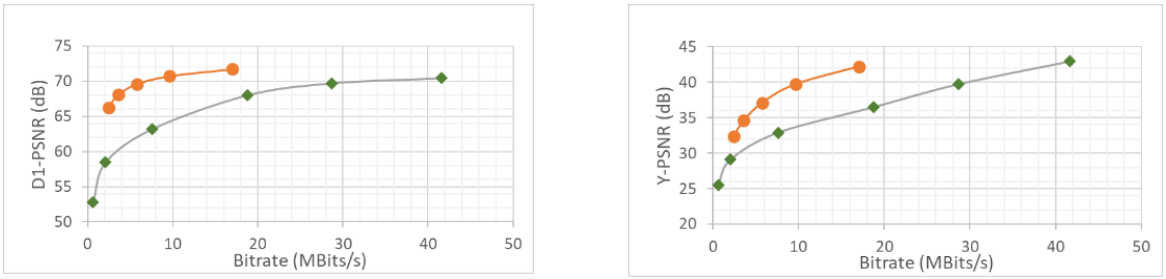


Overpass (R. Cohen, H. Ochimizu, D. Tian, 2017): entire scene (left) with around 5 M points and a local zoom (right)

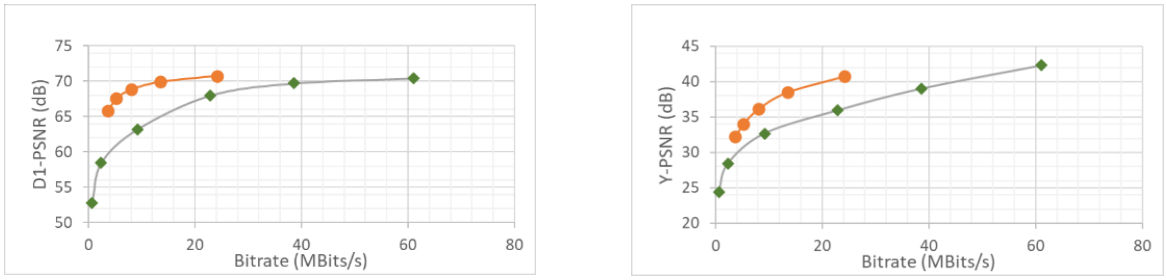
Fig. III-26. Objects from the "fused LiDAR" category.

The diversity of encoding tools in G-PCC allows addressing different use cases with different requirements.

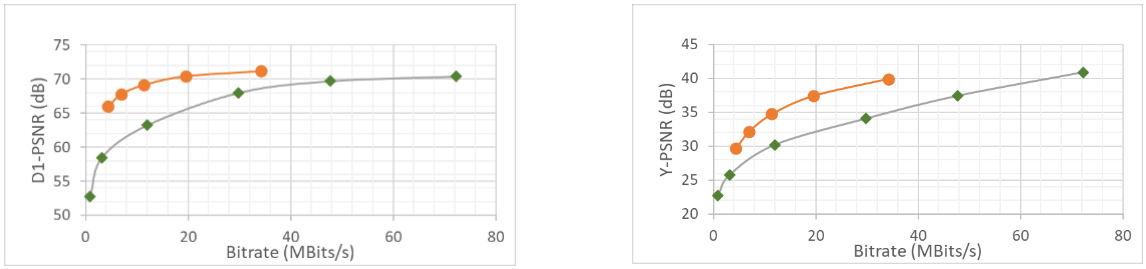
The first evaluation concerns dense and static objects, such as the human point clouds of the first category. The results obtained are presented in Fig. III-27. They confirm that, for this category, the V-PCC approach (as implemented in TM v11) outperformed G-PCC, for all the encoding rates considered.



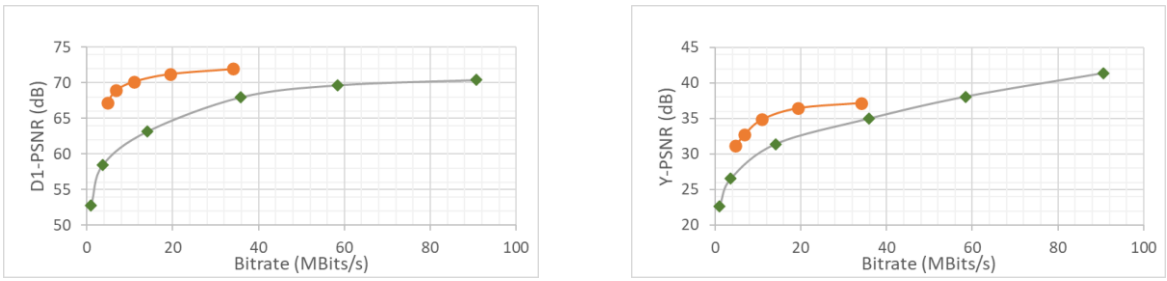
loot



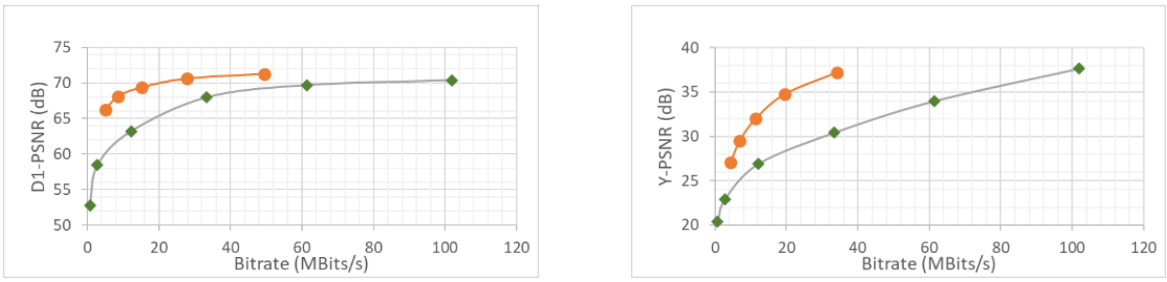
redandblack



soldier



queen

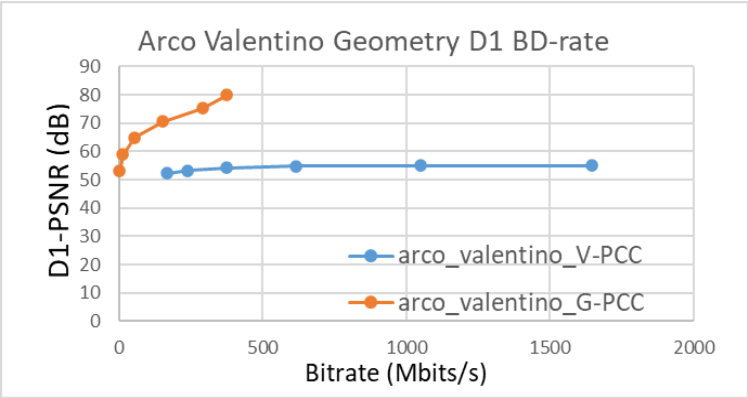


longdress

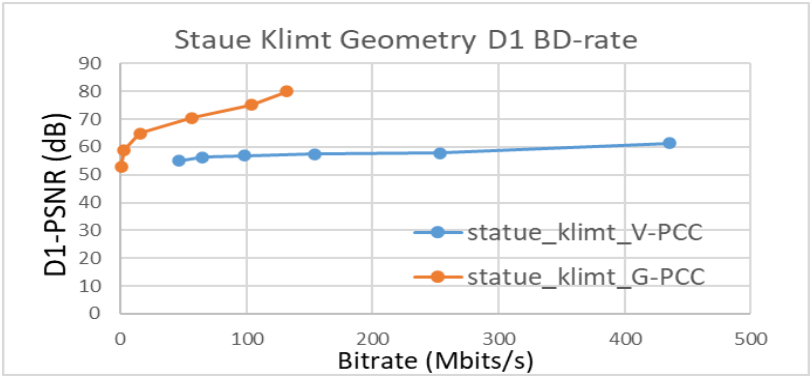
Fig. III-27. Geometry (left) and attribute (right) PSNRs comparison between V-PCC (orange) and G-PCC (green) for individual frames in the human category.

The second evaluation concerns the sparse objects from the second to fifth categories. Notably, for all cases, the number of missed points (3D points projected onto the same 2D pixel) is much larger than for the first category objects, V-PCC creating more patches and a bigger raw patch. For example, *House without a roof* has 4.8 M points, where 2.9 M (60%) points are classified as missed points and 430 2D patches are generated. An even smaller object, such as *Shiva* (1 M points), has 0.5 M (50%) points classified as missed points and 738 patches. This should be compared with only 0.4% of missed points for *longdress*, and 61 patches.

Despite the fact that V-PCC (and therefore 3D to 2D projection methods) encountered difficulties to deal with objects from the second to fifth categories, we have generated the RD curves for some objects (Fig. III-28).



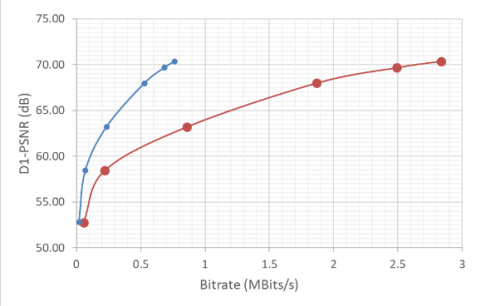
Arco Valentino (vox 12)



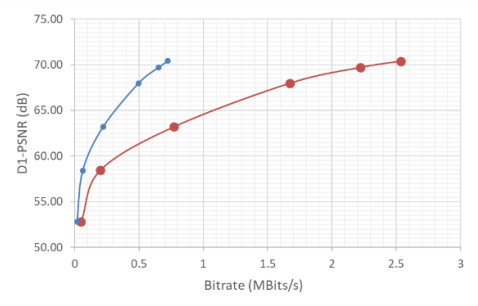
Klimt statue (vox 12)

Fig. III-28. Geometry PSNR comparison between G-PCC (orange) and V-PCC (blue) for the two categories.

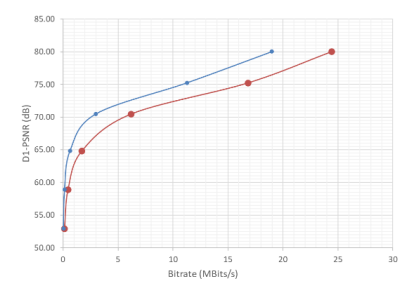
Since G-PCC offers a good coding performance for sparse objects from the second to fifth categories, we compared these performances with the ones of Draco, an open-source graphics codec provided by Google (*Draco*, n.d.). In these experiments, G-PCC is set in the octree mode for the geometry and *predlift* for attributes coding. The results are illustrated in Fig. III-29.



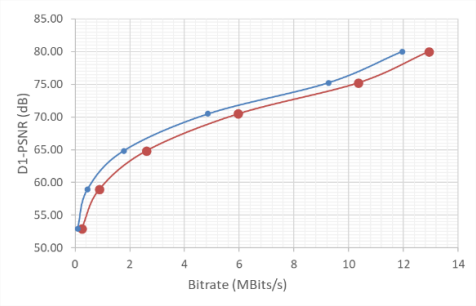
Longdress (vox 10)



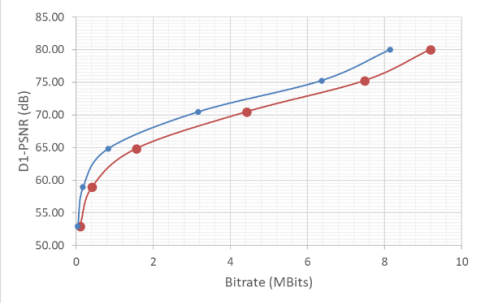
Redandblack (vox 10)



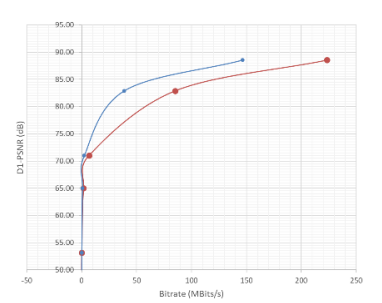
House without a roof (vox 12)



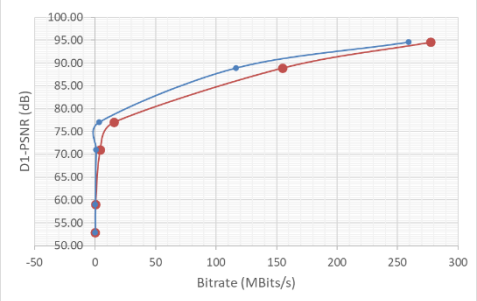
Arco Valentino (vox 12)



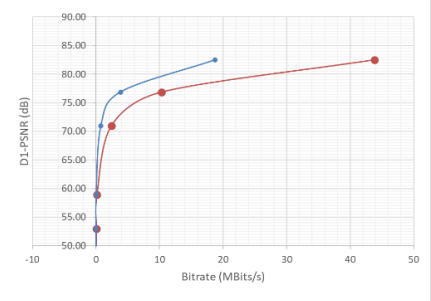
Shiva (vox 12)



Unicorn (vox 15)



Stanford area 2 (vox 16)



Landscape (vox 14)

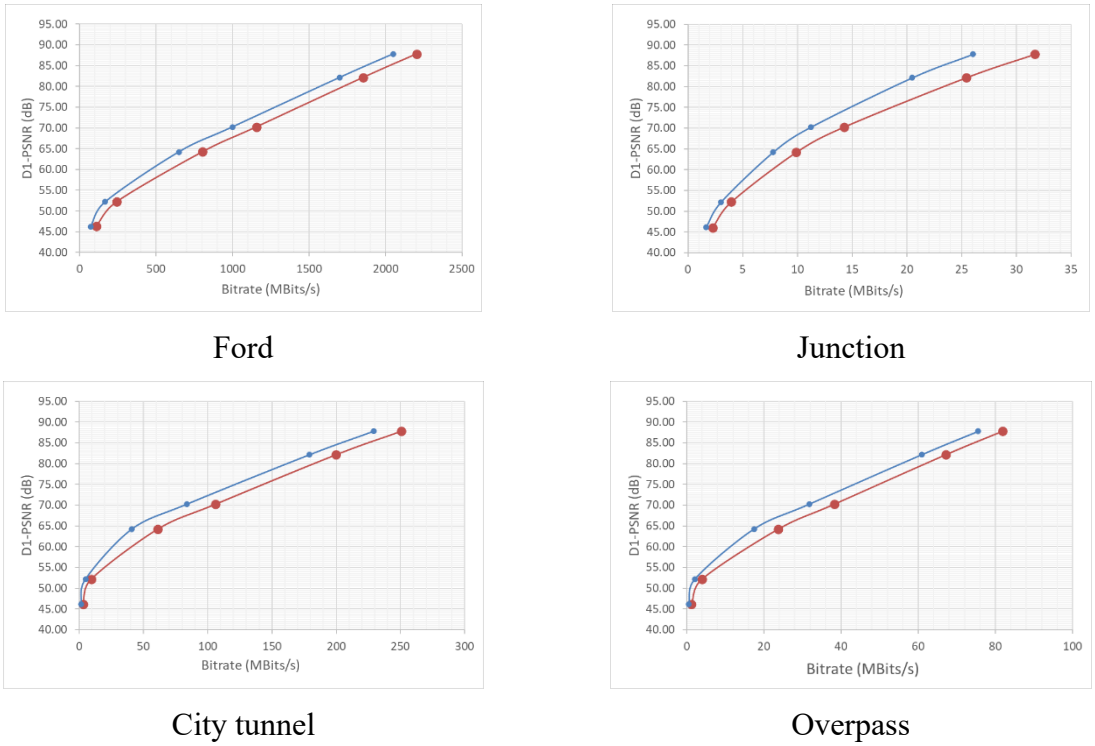


Fig. III-29. Geometry PSNR comparison between G-PCC (blue) and Draco (red) for the five categories.

We can observe that G-PCC outperforms Draco for all the objects and over all the coding rates considered.

In automotive and mapping applications, it is sometimes important to preserve and use the data acquired from scanners in a lossless mode. We compared the G-PCC lossless performance with respect to the raw (uncompressed) data representation and the Draco implementation. The results are presented in Fig. III-30 and Fig. III-31. Notably, an average of 61% reduction concerning raw data for geometry was obtained (38% reduction to Draco), and an average of 73% reduction with respect to raw data for reflectance was obtained (27% reduction with respect to Draco).

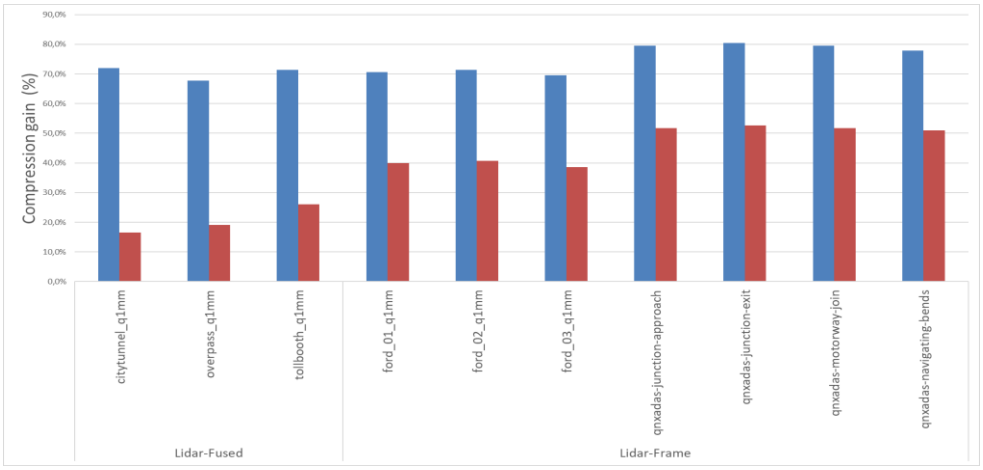


Fig. III-30. G-PCC TMC13-v11 vs. Raw (in blue) and vs Draco (in red) for lossless geometry compression.

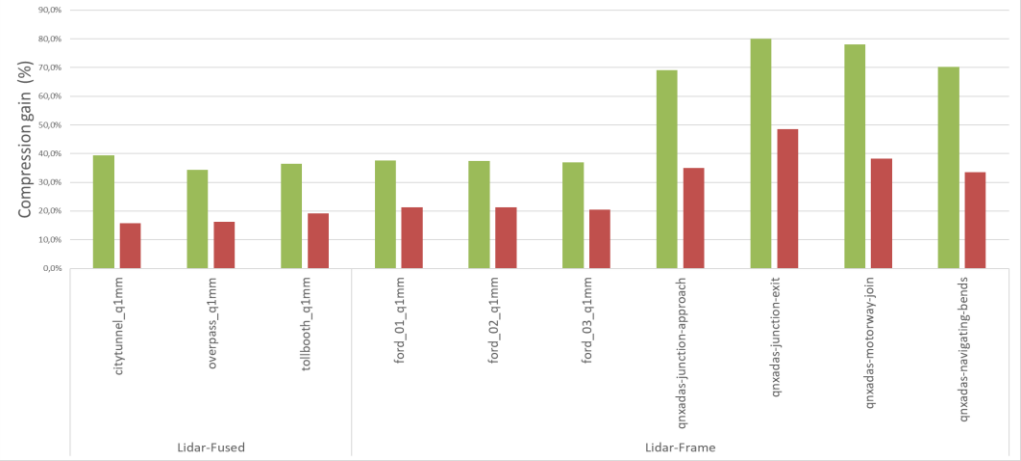


Fig. III-31. G-PCC TMC13-v11 vs. Raw (in green) and vs Draco (in red) for lossless reflectance compression.

The experimental results presented show that G-PCC offers outstanding, state-of-the-art performances in a majority of cases. However, in our opinion, there is a lot of room for future optimization related to the V-PCC approach, notably at the level of non-normative tools that can be considered beyond the current TM implementation. In particular, the RA (INTER) coding mode needs to be optimized. In order to better understand the limitations of the current model, we have performed a V-PCC dedicated experimental analysis that intends to identify the causes of the current V-PCC limitations.

3.5. Experiments on the V-PCC inter-coding mode

The main advantage of the V-PCC is that it utilizes the video codec (i.e., HEVC in this V-PCC test model) to compress efficiently the projected 3D point cloud under the form of a sequence of packed 2D videos. By doing so, the temporal correlation between the 3D points is transferred into a 2D form that is more compression-friendly. There are two modes utilized by the V-PCC codec, namely INTRA and INTER, where intra-frame and inter-frame prediction is used to compress the video respectively. In this thesis, the investigation is mainly focused on the INTER mode which is less studied in the literature and holds greater potentialities for improvements. In our experiments, 5 types of information including Bit Heat map, Coding Unit (CU) Structure, Prediction Unit (PU), Transform Unit (TU), and Motion vectors (MV) have been extracted from the HEVC video bitstream and analyzed using an open-source tool named “Gitl HEVC Analyzer” (*Gitl HEVC Analyzer*, n.d.). Several issues have been identified, which provide inspiration for our contributions that are presented in subsequent chapters. The experiments used the V-PCC version 1.0, with a dynamic voxelized point cloud dataset from (E. d’Eon, B. Harrison, T. Myers, 2017). For simplicity, this dataset will be noted as the *8i* dataset in the following context.

3.5.1. Bit Heat map

The Bit Heat map illustrates how the bits are consumed within each frame of the compressed video. An example is illustrated in Fig. III-32, where the red color represents higher bit allocation in the geometry and texture image respectively.

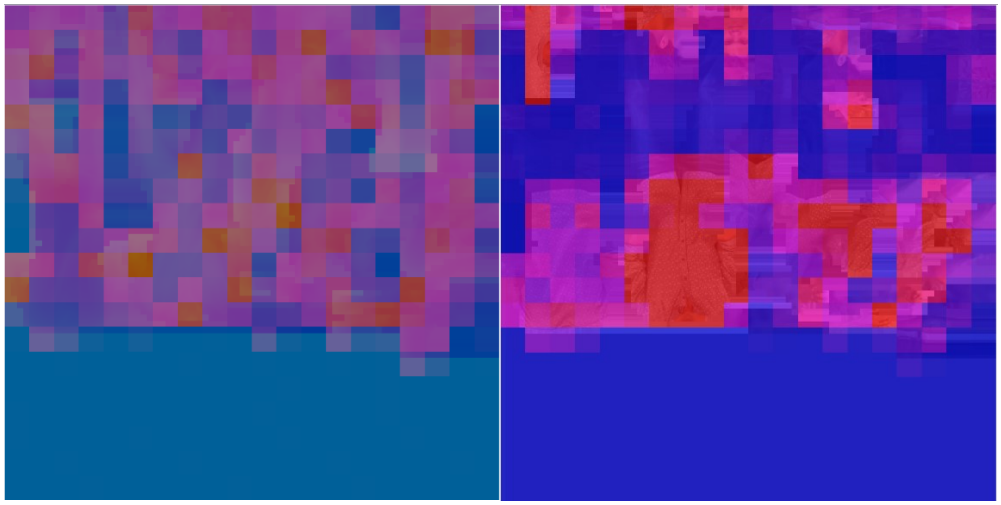


Fig. III-32. The bit heat map of encoded V-PCC geometry image Frame 2 (left) and texture image Frame 2 (right) in INTER mode for loot.

The texture image consumes more bits than the geometry image since the color pixel is composed of the three YUV components while the geometry pixel represents solely a single depth component (representing the distance between the 3D point and the corresponding projection plane). Although the video is encoded in the INTER mode of HEVC, a higher bit consumption has been observed at the boundary of the geometry patches and within some of the texture patches where the locations of these patches have changed compared to the ones in the previous frame.

It is observed that the generated 2D patches are not temporal consistent in terms of locations in the image. In order to understand the reason behind it, we have studied the partition into Coding Tree Units (CTUs), as described in the following section.

3.5.2. CU, PU, and TU

Differently from previous video coding standards, HEVC supports a highly flexible partitioning of a video sequence. Each frame of the sequence is partitioned into Tiles and/or Slices, which are further partitioned into CTUs. The CTU is the basic unit of coding, analogous to the Macroblock in earlier standards, and can be then divided into one or more CUs. The CUs are then divided into Pus (Prediction Units) of either intra-picture or inter-picture prediction types. Moreover, to code the prediction residual, a CU is divided into a quadtree of DCT TUs, which contains the coefficients for spatial block transform and quantization.

The PU and TU information has been analyzed along with the corresponding CU partition structure. An example of the CTU partition is illustrated in Fig. III-33 where the white line represents the CU division. The information of PUs is overlayed in Fig. III-33 as well.

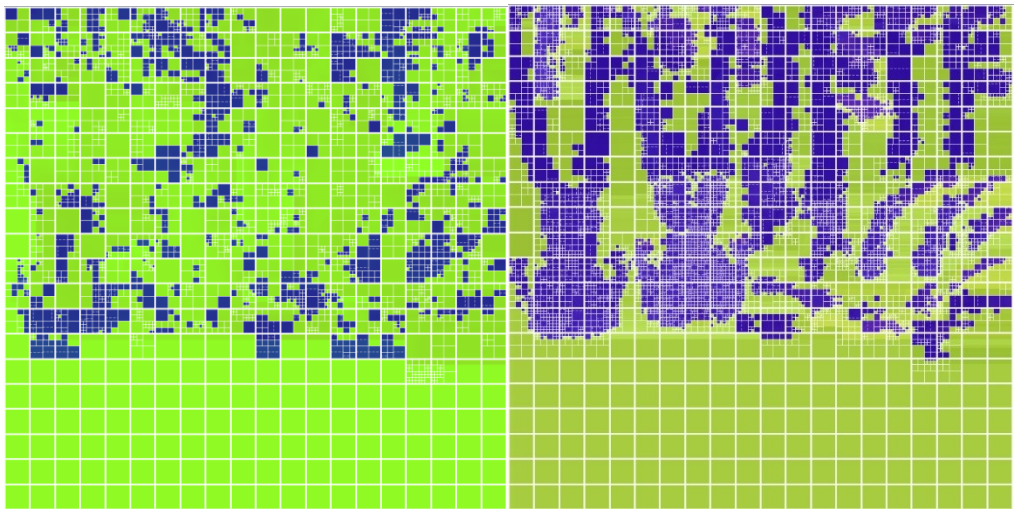


Fig. III-33. The Prediction Unit (Green/Yellow: INTRA prediction and Blue: INTER prediction) of the encoded V-PCC geometry image Frame 2 (left) and texture image Frame 2(right) in INTER mode overlayed with the Coding Unit structure (segmented by the white line) for the loot sequence.

The PUs in the Green/Yellow color represents the use of intra prediction while the Blue one represents the use of Inter prediction type. The CU division (in white line) shows the level of variation for the pixel values within the CTU. However, it is shown that with a similar level of CU partition, the texture image contains more INTER prediction units than the geometry image. In the geometry patches generation process, the minimum depth value is detected and is used to reduce the range of the depth values. This reduction has made the depth values between patches less temporal consistent. Moreover, it is observed that for both geometry and texture images, the patches with the same shape can be located in a different position when compared to the previous frame. Therefore, a better packing strategy is necessary to improve the temporal consistency of the patches, which can conduct a higher compression performance.

Similarly, the TU division is shown by the black lines in Fig. III-34. It can be seen that there exists the potential to improve the Intra/Inter prediction and reduce the number of residuals within the image.

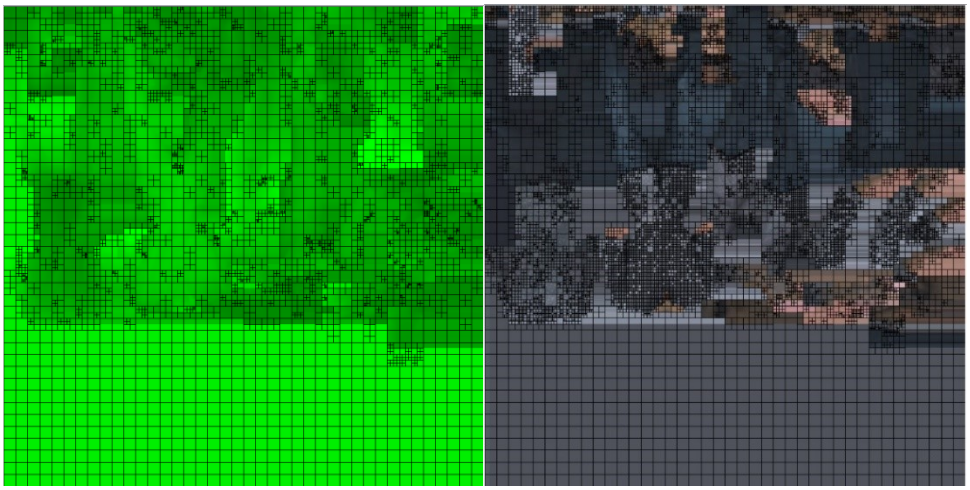


Fig. III-34. The Transform Unit (segmented by black lines) of the encoded V-PCC geometry image Frame 2 (left) and texture image Frame 2 (right) in INTER mode for loot.

Both the PU and TU division show that the INTER mode of HEVC is not fully exploited with the current patching and packing strategy resulting in a poor temporal consistency of the 2D video. To understand how exactly the patches are predicted with HEVC, the motion vectors are visualized and analyzed in the following experiment.

3.5.3. Motion vectors

A motion vector is a two-dimensional vector used for inter prediction that provides an offset from the coordinates in the decoded picture with respect to the coordinates in a reference picture (VCEG & Q.6), n.d.). As illustrated in Fig. III-35, in both geometry and texture images, there exist many motion vectors of large amplitude that correspond most of the time to erroneous matches. In the geometry image, the varying relative depth value for each patch has disconnected the temporal connection between the 2D geometry patches in consecutive frames. On the other hand, the varying location of the 2D texture patches leads up to a poor inter prediction.

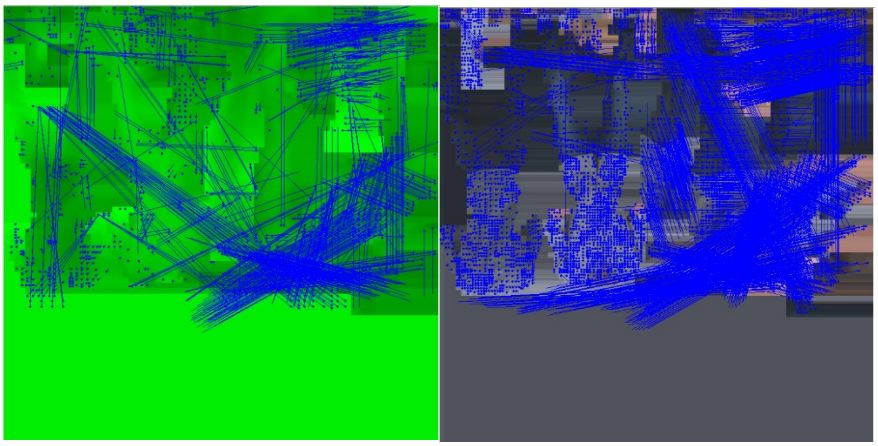


Fig. III-35. The Motion vectors (blue lines) of the encoded V-PCC geometry image Frame 2 (left) and texture image Frame 2(right) in INTER mode for loot.

3.5.4. Other problems

The packing of the generated 2D patches is following a descending order of the patch size, while minimizing the space of the packed image. However, as shown in the previous experiments, the packing strategy is not able to maintain the 2D locations of two similar-shape patches in continuous frames. Moreover, it is not prioritized to keep the continuity of the orientation of the generated 2D patches. This problem is illustrated in Fig. III-36, for the *queen* sequence.

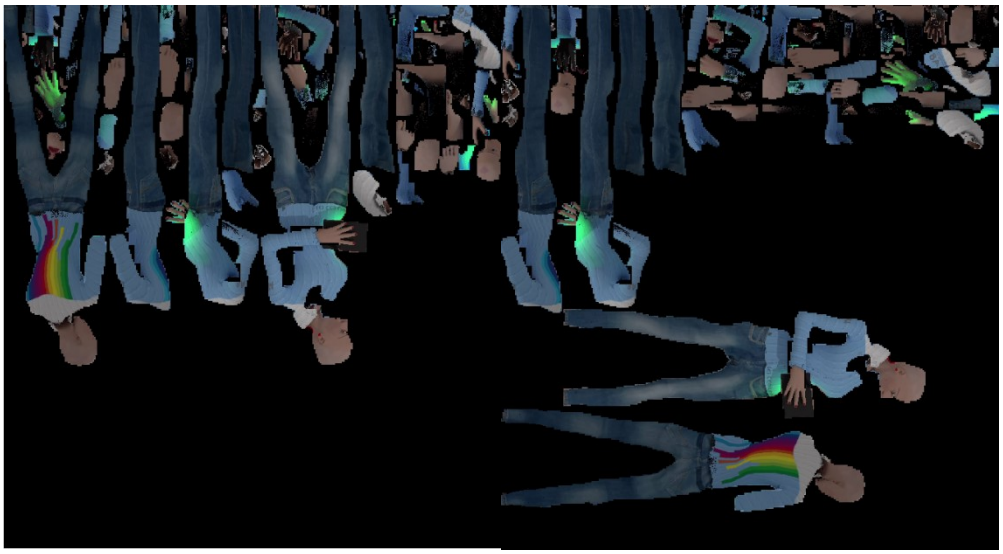


Fig. III-36. Frame 2 (left) and Frame 3 (right) of the encoded V-PCC texture image for the queen sequence.

Moreover, there are multiple layers in the point cloud with various colors in the *queen* sequence (J. Ricard , C. Guède, R. Doré, n.d.), as illustrated in Fig. III-37.

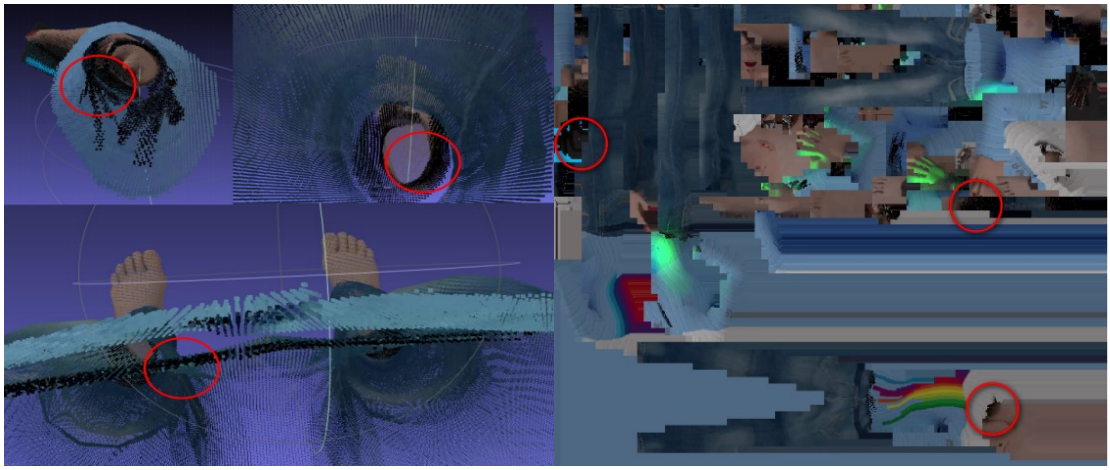


Fig. III-37. The multi-layered 3D points of queen (left) and the encoded V-PCC 2D texture image (right).

As a result, when the 3D points are projected onto a 2D plane according to the computed normal, the projected point on the same 2D pixel location can take different color values between successive frames. After projection and packing, the 3D geometry correlation between the points has been disconnected and brings difficulties to provide accurate prediction for both geometry and texture information.

3.6. Conclusion

In this chapter, we have described in detail the two MPEG V-PCC and G-PCC methodologies. A comparative experimental analysis of the related compression performances has also been presented. The obtained results show that G-PCC provides state-of-the-art results for a majority of point cloud types. In contrast, V-PCC outperforms G-PCC for the particular case of dense points clouds with a low level of noise and motion. However, V-PCC offers significant room for improvement. In order to understand the current limitations, we have performed a dedicated experimental analysis. It showed that the temporal coherency between the point cloud sequence is not been fully exploited by V-PCC. Thus, although the V-PCC takes advantage of the HEVC in INTER coding mode, the inter-frame prediction is not fully employed, for the following reasons:

1) The generation and packing of the 2D patches do not take into account the temporal aspects: even if the 3D points are identical, the packed 2D patches are assigned in different image locations or contain different information.
2) The 3D-to-2D projection has eliminated the 3D correlation of the points and thus cannot provide accurate predictions.

In order to overcome such limitations, we have first investigated how we can achieve a more 3D accurate prediction before the projection phase. The idea is to use the resulting correspondences to help improve the temporal consistency of the generated 2D geometry and texture videos. Under this framework, a compression method based on octree decomposition and RDO affine transformation is introduced in the following chapter.

CHAPTER IV. OCTREE-BASED RDO SEGMENTATION

In this chapter, an RDO-based octree segmentation method is proposed. The objective is to adaptively cluster the point cloud, with respect to both geometry and motion. The proposed method segments the 3D space into an octree structure so that motion estimation can be applied using affine transformation according to the different motion amplitudes of the sub-parts. Then, an RDO process is carried out to determine if further segmentation is needed for each octree node. The RDO is decided as a tradeoff between the bits required to build the octree structure and the resulting prediction error from the motion-compensated point cloud. After motion estimation and compensation, the 3D correlations between point clouds help to improve the temporal consistency of the projected 2D packed images. Thus, better compression performances are achieved for content with relatively low motion amplitudes. However, for sequences with higher motion amplitudes, the RDOs are hard to balance, leading to lower compression performances.

This work has been carried out in a joint project between Télécom SudParis and Apple and the proposed method has been registered as a US patent numbered 1888-75602 P42966US1.

Institut Polytechnique de Paris

91120 Palaiseau, France

As we have seen in the previous chapter, the 3D-to-2D projection used by the V-PCC test model disconnects the original correlation between the 3D points. It is thus difficult to preserve the temporal consistency of the shape and location for the generated V-PCC 2D patches. In order to preserve the original 3D correlation, an additional topological structure is needed. In our work, inspired also by the MPEG G-PCC model, we have retained an octree structure. The octree is used to segment the 3D space adaptively, according to a computed RDO score. The RDO process is proposed here to analyze the variation of the octree blocks as a prediction for the motion in the point cloud. Then, the octree cubes that contain high motion amplitudes are motion-compensated using affine transformation to locate the 3D correspondence between 3D cubes (points) in consecutive frames. This process can be interpreted as a mapping that is iteratively performed. Once the mapping is achieved, the original 3D correlation can be projected back onto 2D domain to help generating more consistent 2D patches and packed video. Let us now introduce the pipeline of the proposed method.

4.1. Pipeline

The RDO-based Affine Transform is adapted into the generic V-PCC pipeline to find the 3D point correspondences from successive frames n to n+1 or reversely from frames n+1 to n, as illustrated in Fig. IV-1.

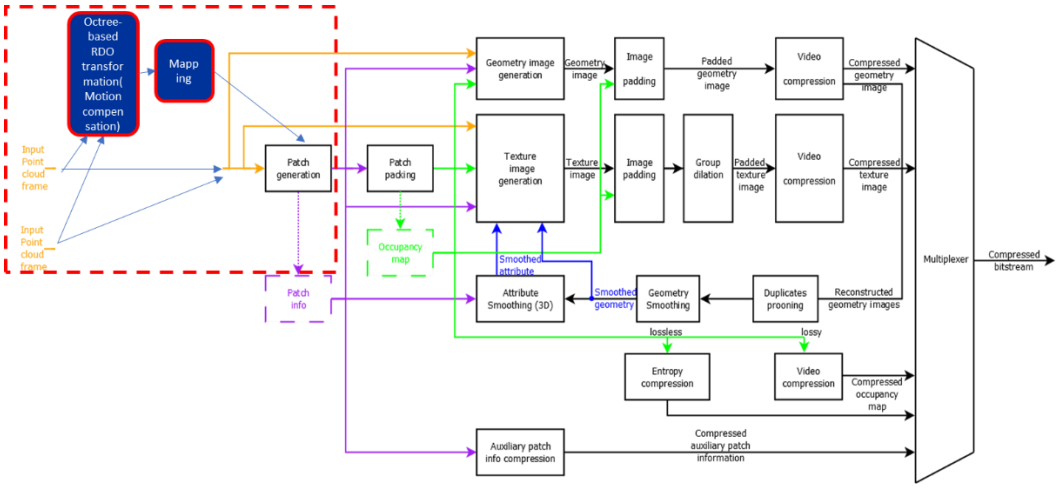


Fig. IV-1. The pipeline of the proposed method based on V-PCC.

The core of the method concerns the red dotted box in Fig. IV-1, which includes two processes: octree-based RDO transformation and mapping. It can be considered as a pre-processing step before applying the standard V-PCC encoding process. The main idea is to take advantage of an RDO-segmented octree structure and affine transformation to determine the 3D correspondence (mapping) between 3D points in successive frames of the sequence. The transformation matrices as well as the mapping vectors are discarded after their usage. In this case, no additional information is required for inclusion within the bitstream, which makes it an encoder-only modification.

4.2. RDO-based octree segmentation

The method first decomposes the input point cloud into levels of octree and the decision of further segmentation depends on the RDO step. For each frame considered as a reference, the affine transforms associated with each octree node are computed to transform the reference frame to the target frame. Initially, the entire reference frame is used to compute the error (point-to-point distances between the reference points and the transformed points), then sub-divided into a fixed number of segments (i.e., eight octree sub cubes). For each sub-divided segment, the errors are computed again to help determine if further segmentation is needed. This decision is given by the condition described in equation IV-(1):

$$Error_p + \text{RDO}\,\lambda * AT\_COST \geq \sum_{k=0}^{segments} Error_S + segments * \text{RDO}\,\lambda * AT\_COST \qquad \text{IV-(1)}$$

where:

- $Error_p$ is the computed affine motion compensation error of the segment before subdivision,
- $Error_S$ is the computed affine motion compensation error of the subdivided k^{th} segment in the octree,
- $segments$ denotes the number of subdivided segments (in this case, 8),
- RDO λ is a user-defined parameter,
- AT_COST measures the cost of an affine transform, hard-coded to the value = 12 * 32 (corresponding to 12 floating-point parameters defining a 3D affine transform)

The RD cost of the next sub-division is measured and compared with the saved current cost iteratively. If the RD cost of the current cost is lower than the one corresponding to a subsequent subdivision, it is considered that the balance between the rate and distortion is achieved and the RDO process is terminated.

Once the RDO completes, the affine transform is applied to the input frame and attempts to match with the reference frame for all the point cloud cubes in the octree. The transformed frame is motion-compensated to get geometrically close to the reference frame. The closest point in the transformed point cloud is considered as the corresponding point in the reference point cloud frame.

The selection of the λ parameter can affect the level of octree segmentation and the RDO results. An example is illustrated in Fig. IV-2 with different λ values considered for the RDO process. It can be observed that for lower values of the parameter λ, deeper octree decompositions will result. Thus, a higher λ should be set when the input point clouds contain higher motion amplitudes.

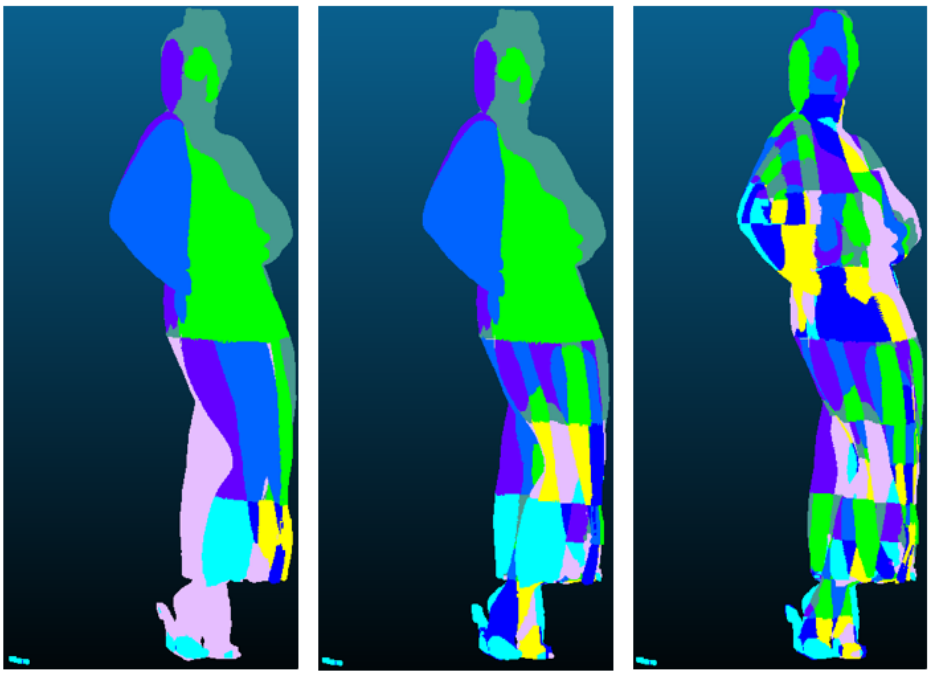


Fig. IV-2. Dataset longdress frame 1200 vs 1201 segmentation, RDO λ = 100 (a), 10 (b) and 1 (c).

In order to study the impact of different RDO λ values, Table IV-1, Fig. IV-3, and Fig. IV-4 show how the mean point-to-point distances between consecutive point cloud frames change with respect to the number of affine transformation matrices.

Table IV-1 Values for RDO λ, mean point-to-point distances, and the corresponding number of affine transformation matrices. (frames 1051 vs 1052, 1053, 1054, 1055, and 1056 of longdress)

	Frame 1052		**Frame 1053**		**Frame 1054**		**Frame 1055**		**Frame 1056**	
RDO λ	**Mean Distance**	**Affine Transforms**	**Mean Distance**	**Affine Transforms**	**Mean Distance**	**Affine Transforms**	**Mean Distance**	**Affine Transforms**	**Mean Distance**	**Affine Transforms**
200	1.23	8	1.92	20	2.15	33	2.42	39	2.20	71
100	1.23	8	1.52	51	1.73	71	2.17	65	1.89	103
50	1.08	25	1.29	88	1.41	121	1.84	112	1.69	137
40	1.00	37	1.23	100	1.36	132	1.81	119	1.69	137
25	0.98	43	1.10	132	1.24	171	1.69	158	1.41	228
10	0.90	75	0.97	205	1.13	231	1.52	250	1.19	361
5	0.85	112	0.88	296	1.04	315	1.45	326	1.00	556
1	0.75	292	0.70	719	0.82	789	1.23	783	0.76	1107

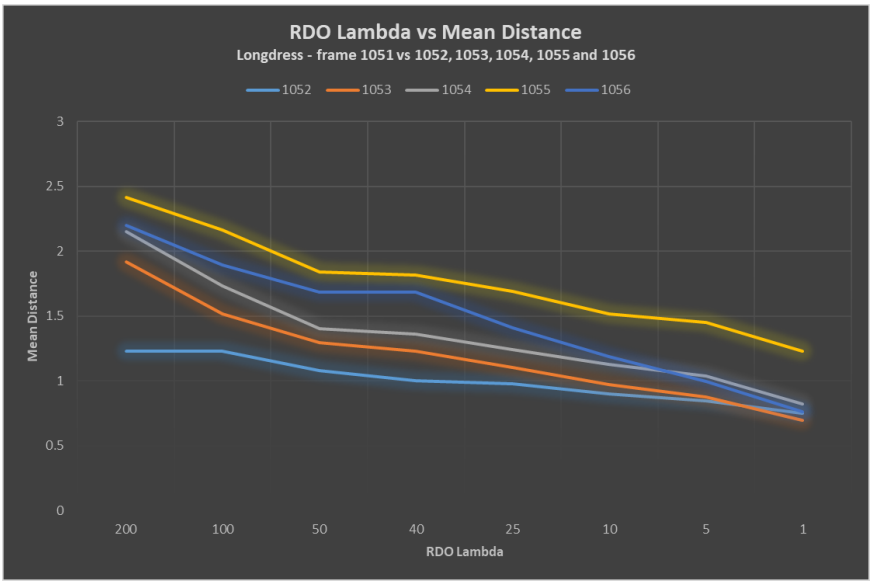


Fig. IV-3. Longdress - RDO λ vs Mean Distance. (frames 1051 vs 1052, 1053, 1054, 1055, and 1056)

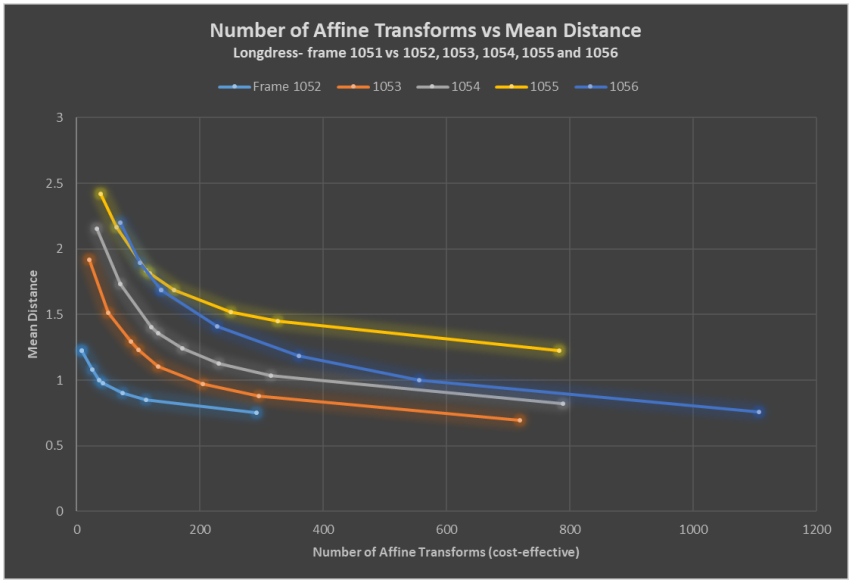


Fig. IV-4. Longdress frames 1051 vs 1052-1056 Mean distance vs Number of Affine Transform.

It can be seen from Table IV-1 and Fig. IV-3 that with the decrease of the RDO parameter λ, lower mean point-to-point distances can be achieved, at the cost of an increasing number of affine transformation matrices. It can be also observed from Fig. IV-4 that an "elbow" explaining the peak variance of the mean distance normally happens when 100-200 affine transformation matrices are used. After that, the decrease of the mean distance becomes relatively slow. In this proposal, the affine transformation matrices are discarded after building the 3D correlations. Thus, a relatively low value of RDO λ is preferred to achieve better estimation. Among the tested RDO λ values, "1" provides the lowest mean point-to-point distances but takes a great number of computational resources and time to reach the convergence. Thus, we select RDO λ as "10", "20", and "50" as a compromise of computing speed, without losing too much precision for estimating 3D correspondences between point clouds.

In sequential point cloud frames, the variation in terms of the number of points and their corresponding attributes can be significant. Different prediction structures can lead to different mapping results and have a huge impact on the efficiency of 3D PCC.

4.3. Prediction modes

The V-PCC test model processes each input point cloud frame individually and cluster the 3D points into patches without considering the 3D temporal correlation between the frames. For each frame, an initial clustering is launched to gather the points with the same orientation achieved by PCA. Then, the clusters are smoothed according to the orientations of their neighbors. Finally, the smoothed 3D patches are projected onto 2D images and compressed by video codecs.

In contrast, the proposed prediction process, illustrated in Fig. IV-5, exploits the resulting 3D correspondences determined as described in the previous section, to improve the temporal consistency of the obtained 2D image patches.

We propose different prediction modes to examine the accuracy of the mapping. The first one concerns prediction from a single reference frame all over the considered sequence. The second one exploits the sequential predictive approach, applied to each pair of consecutive frames throughout the sequence. This second approach yields the most accurate mapping and is illustrated in Fig. IV-5.

In the proposed method, a prediction structure noted as “IPPP” is used based only on the previous constructed frame. For example, the second frame “P” will be predicted from the intra-coded “I” frame and the third frame “P” will be predicted from the previously inter-coded “P” frame.

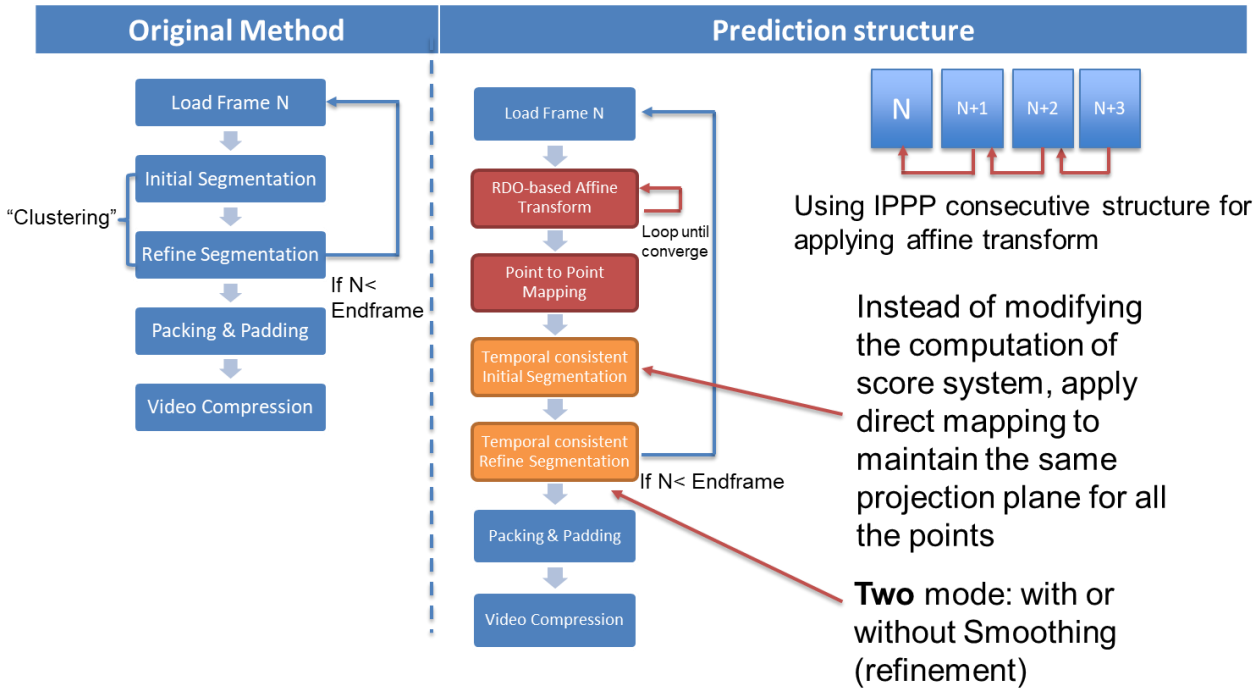


Fig. IV-5. Proposed prediction structure and operations (right) in comparison to the original method (left).

In the proposed prediction method, the RDO-based affine transformation transforms the target point cloud towards the reference point cloud to find the 3D correspondences between points. The 3D correspondences are used here to identify whether a point has been allocated to be projected with a different orientation or not compared to the last frame. If a certain amount of points in one patch has been detected holding a different orientation as in the previous frame, the generated patches are not temporal consistent or the sequence contains high motion amplitudes.

Once a point-to-point mapping is determined, the same "Initial Segmentation" and "Refine Segmentation" steps as in the traditional V-PCC approach are operated for the reference frame. The resulting 3D correspondences between points in consecutive point cloud frames are used here to achieve more temporal consistent normal information. More precisely, we map the orientation of the point in the reference point cloud to the one in the target frame so that the inconsistent allocation of the projection plane as shown in Fig. IV-6 can be avoided. In this way, more temporal consistent 2D patches in terms of shapes and locations can be achieved as shown in Fig. IV-7. The connection between the foot and leg are well preserved between frames producing a more compression-friendly video. The points near the contour of the 2D patches are affected the most since their orientations are more likely to change between frames when motion occurs.

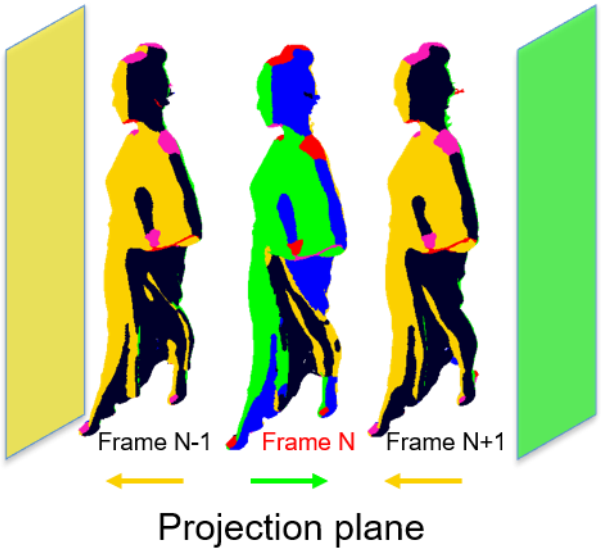


Fig. IV-6. Inconsistent allocations of projection plane for the same patch in consecutive frames.

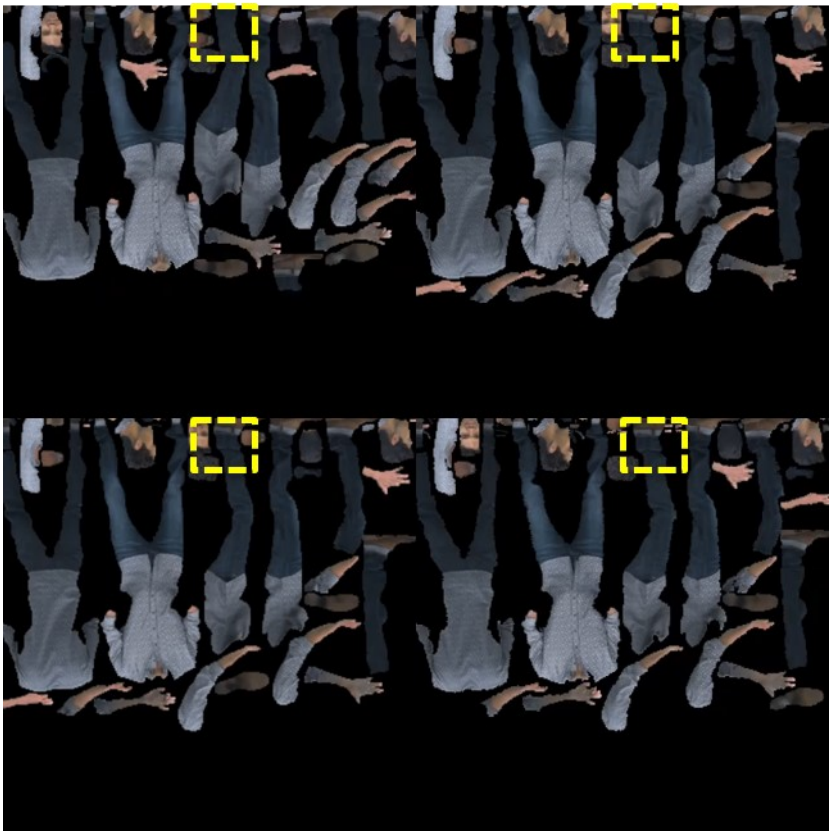


Fig. IV-7. The packed frame N (top left) and frame N+1(top right) using the V-PCC method, and the packed frame N (bottom left) and frame N+1(bottom right) using the proposed method.

It has been pointed out in Chapter III-3.5.4 that the compression performance is harmed by the inconsistent patch orientation between frames. By taking into account the normal information of points in different timestamps, the resulting projected 2D patches remain more temporally consistent in terms of both shapes and locations. As a result, the efficiency of video coding can be increased, as discussed in the following section.

4.4. Experimental results

We have carried out our experiments on the first 32 frames of the *8i* sequence using the proposed prediction scheme. The resulting performances in Fig. IV-8 are compared against the ones from the V-PCC test model version 2.0 in lossy inter coding mode.

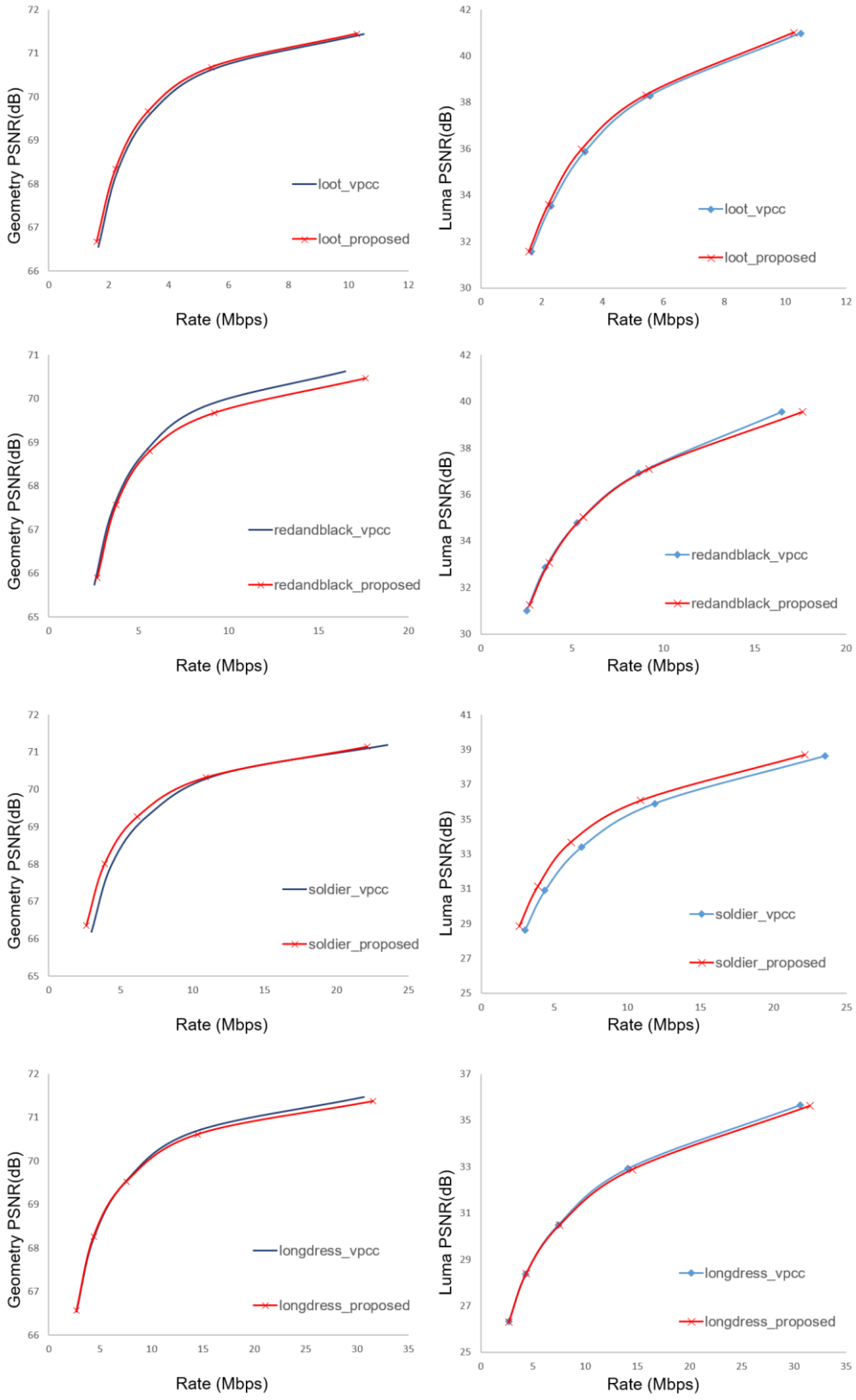


Fig. IV-8. The RD performances of the proposed method against the one from V-PCC test model v2.0.

The obtained results show that the proposed method outperforms V-PCC for the *loot* and *soldier* sequences, while V-PCC is preferable to compress *redandblack* and *longdress* in terms of both geometry (point-to-point distances) and color (Luma component) information. However, the curves are relatively close to each other for some of the sequences and it is hard to visually distinguish the difference. Thus, we propose to use the BD-Rate as shown in Fig. III-5 to calculate the coding efficiency between different codecs based on PSNR measurements.

The Bjøntegaard model is used to calculate the average PSNR and bit rate differences between two RD curves obtained from the PSNR measurement when encoding content at different bit rates. A curve through the data points from the RD curve is fit and the corresponding expression for the integral of the curve is described in equation IV-(2):

$$SNR = \mathrm{a} + \mathrm{b} * SNR + \mathrm{c} * SNR^2 + \mathrm{d} * SNR^3 \qquad \text{IV-(2)}$$

This can help us to find the average PSNR and bitrate difference over the whole range (both are shown in percentage for simplicity in our experiment). The corresponding BD-Rate values from Fig. 39 are illustrated in Table IV-2.

Table IV-2 Experimental results for the method.

BD-Rate [%]					
Tested sequence	**Point-to-point distance**	**Y**	**Cb**	**Cr**	**RDO λ**
loot	-7.95%	-6.79%	-7.55%	-7.98%	10
redandblack	7.22%	2.13%	-2.09%	1.51%	50
soldier	-9.42%	-13.45%	-11.70%	13.97%	0.3
longdress	1.66%	0.21%	-0.84%	0.06%	50

Negative values represent that reduction of required rate can be observed at the same level of visual quality, while positive percentages indicate a rate increase. These results confirm that the proposed method outperforms the V-PCC test model for *loot* and *soldier* but not for *redandblack* and *longdress* in BD-Rate. The reason is that both *redandblack* and *longdress* sequences contain higher motion amplitudes. Meanwhile, their motions are more complex (e.g., deforming clothes and moving hair parts) and cannot be predicted and mapped accurately by using affine transformation alone, without the use of any residuals.

4.5. Conclusion

In this chapter, we have proposed an RDO-based segmentation and mapping method for dynamic point cloud compression. A mapping stage is proposed to determine the 3D correlation between points in consecutive frames using affine transformation.

In this way, more temporal consistency of the generated 2D patches can be acquired, resulting in a higher video compression ratio. We show that, without changing too much the main methodology of V-PCC, limited gains can be achieved for content with low motion amplitudes.

The affine transformation is one of the potential candidates. However, in this chapter, it is shown that a great number of affine transformation matrices are needed to achieve accurate motion estimation when using octree-based segmentation. The tradeoff between the rate and distortion remains challenging to balance. Meanwhile, we have observed that within the same region, some octree cubes contain transformation matrices with similar values. This means that another structure that is less redundant than octree is needed. This inspired us to propose a novel skeleton-based data structure for motion estimation, described in the following chapter.

CHAPTER V. SKELETON-BASED MOTION ESTIMATION AND COMPENSATION

In this chapter, a 2D pose estimation technique is proposed. A skeleton-based model with 14 anatomical segments is here employed for motion estimation using a by-part affine transformation model. This choice is motivated by the specific application considered in this chapter, which is related to the 3D PCC compression of human-like objects. The 3D point cloud is first projected onto 2D images to exploit the 2D pose estimation technologies. Then, the resulting 2D labels are integrated into 3D, forming anatomical segments of the point cloud. After filtering and smoothing, motion estimation and compensation are applied to the segments using the considered affine transformation model. To the very best of our knowledge, this is the first approach in 3D PCC that takes advantage of 2D pose estimation techniques to help and improve 3D prediction. The proposed method outperforms the V-PCC method in low rate conditions when applied on dynamic point clouds with relatively small motion amplitudes. However, without adding the residuals, a limited amount of gain, in terms of BD rate, can be achieved for content with higher motion amplitudes.

Part of the work presented in this chapter has been published in (CAO et al. 2020).

Institut Polytechnique de Paris

91120 Palaiseau, France

5.1. Introduction

In the previous chapter, we have shown that the octree is not preferable for predicting human motion where the actions are always attached to a skeleton-based structure, as in the case of humanoid-like 3D point cloud avatars. Although there are several attempts such as using differential octrees (J Kammerl et al., 2012) and graph-based representations (Dorina Thanou et al., 2016), in a majority of cases, such approaches suffer from a relatively high computational complexity, which becomes critical in the case of dynamic 3D PCC.

In this chapter, we propose a skeleton-based motion estimation method, specifically designed for dynamic, human-based point clouds. We are trying to solve the following two problems: 1) massive synthetic skeleton data is needed to predict the motions within each point cloud frame where a huge amount of manual manipulations are needed (J. M. Lien et al., 2010); 2) performing motion estimation on dense point clouds usually requires down-sampling and voxelization (Dorina Thanou et al., 2016)), which may harm the visual resolution.

Thus, inspired by the methods introduced in (J. M. Lien et al., 2010) and (Raghuraman et al., 2013), we take advantage of the DensePose human pose estimation technique (Güler et al., 2018) to extract skeleton information from 2D projected images in an offline process. Then, the classified multi-view images containing the body part information are aggregated into a 3D point cloud. The points are further clustered into different point cloud segments which correspond to 3D body parts, with the help of an iterative filtering procedure. Affine transformation matrices are then computed for each 3D body part between consecutive point cloud frames to describe the motion of each point cloud segment. In this way, point-to-point mapping is achieved and the original resolution of the dense point cloud is well-preserved. Finally, the proposed motion estimation is integrated into the MPEG V-PCC test model. To the very best of our knowledge, this is a first attempt to achieve a skeleton-based motion estimation with affine transformation on digital humanoid avatars using point cloud representations.

For motion estimation purposes, different groups of frames (GOF) are selected, achieving various bit rates. Experimental results show that in low-rate conditions, the proposed method outperforms the V-PCC method achieving up to 83% bitrate reduction with an equivalent PSNR of point-to-point distance in the case of relatively small amplitude body motion. However, further studies are expected to overcome the inaccurate motion estimation in the case of highly challenging sequences, exhibiting strong non-rigid deformations (e.g., clothes and hair deformations).

The rest of the Chapter is organized as follows. Section 5.2 describes the proposed 3D skeleton extraction procedure. The motion estimation and compensation approach are detailed in Section 5.3. Section 5.4 details the experimental results obtained. Finally, Section 5.5 concludes the chapter.

5.2. 3D Skeleton Generation

Motion estimation has been widely employed and represents an essential element in any 2D video compression method. However, not so much work has been proposed specifically for human motion prediction in the case of 3D point clouds. The proposed method aims to take advantage of existing 2D DL (deep learning)-based pose estimation methods to extract 2D skeleton information to reconstruct a ready-for-animation 3D skeleton from 3D point clouds.

The advantage of such a skeleton-based approach comes from the fact that different motion estimations can be applied to each independent part over time. The proposed 3D skeleton generation pipeline is illustrated in Fig. V-1. It describes an offline processing stage, which generates the 3D skeleton information, to be further used for motion estimation and compression.

5.2.1. 3D/2D projection

As illustrated in Fig. V-1, the input 3D point cloud sequences are first projected onto multiple 2D grid planes and saved as RGB images. The projected images are considered as observations from the corresponding viewpoints. Four orthogonal viewpoints have been considered in our work, corresponding to the front, back, left, and right views.

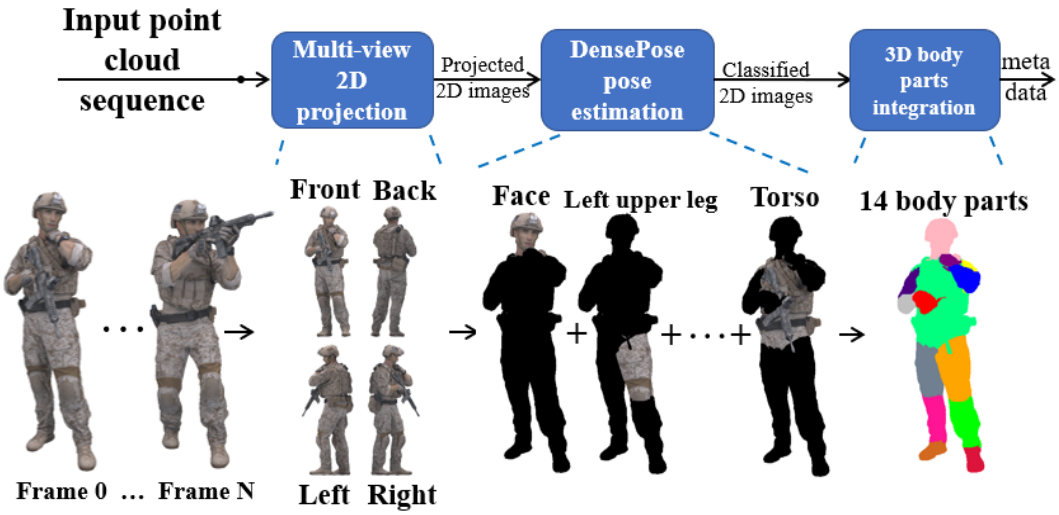


Fig. V-1. The pipeline of the proposed 3D skeleton generation process.

The projections from the top and bottom view are excluded here since they are redundant for the 2D human pose estimation process. The RGB images as well as the corresponding depth images are reserved for 3D point cloud reconstruction.

Though it is possible to use arbitrary viewpoints and generate the corresponding projection, the 3D pose integration encounters difficulties when more than four images from different views are used as input, because the 2D pose estimation is applied individually on the projected images. After 3D integration, inconsistent results in terms of multiple skeleton labels are observed for the generated 3D points. Moreover, the temporal consistency of 2D pose estimation becomes unstable when considering multiple views. Thus, solely the above-mentioned four orthogonal planes are retained, in order to reduce the impact of such problems on the reconstruction process.

5.2.2. Pose estimation

The resulting 2D projected images are then passed individually to the 2D human pose estimation process. The process is accomplished by a pre-trained region-based convolutional neural network (R-CNN) so-called DensePose (Güler et al., 2018). DensePose maps all pixels of the projected RGB images as surface patches of different human body parts.

Within each process, an input image from a certain view is considered. The human body is represented by 14 parts, illustrated in Fig. V-2, following the mechanism described in (Güler et al., 2018).

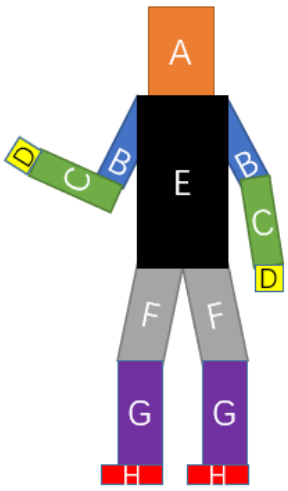


Fig. V-2. Predefined human skeleton model with 14 body parts.

Different body parts such as (A) head, (B) upper arms, (C) lower arms, (D) hands, (E) torso, (F) upper legs, (G) lower legs, and (H) feet are defined to describe a simplified version of the geometry and topology of the human skeleton. In this way, most of the human motion can be described with the affine transformation of such body parts. The body part information within each input image is then extracted and classified (they will be subsequently represented with different colors in our illustrations). The pixels that are not recognized as human body parts by DensePose are labeled as unknown. Finally, all the pixels that receive a valid label from different views are used to reconstruct the classified 3D points, denoting their corresponding 3D body parts, as described in the following section.

5.2.3. 3D pose integration

After the reconstruction of the point cloud, each 3D point may receive either multiple labels or no labels. Each label represents a body part to which the considered pixel is estimated to belong to. The following steps are proposed to ensure the consistency of the estimated skeleton in both spatial and temporal domains.

5.2.3.1. Majority vote

In the case where a 3D point includes multiple labels, a majority vote process is launched to determine the final label of the 3D point. In the case of using four orthogonal projection directions, most of the 3D points only contain one label denoting the corresponding body parts, which are usually correct. However, there are some conflictual cases where an equal amount of votes are observed. For example, a 3D point located near the arm part of the point cloud has two votes for its label, one correctly estimated from the front-view image as arm while another vote suggests the label as torso estimated from the left-view. Such points are then unlabeled and passed to the smoothing stage, along with the other non-labeled points.

5.2.3.2. Smoothing stage

The smoothing stage consists of a statistical outlier removal (SOR) filter and a greedy clustering process, specifically designed to both filter out the mistaken labels and fill the empty ones. The standard deviation is used here to represent the dispersion of a set of values (*i.e.*, the positions of the 3D points). Let ${P_i}^{x,y,z}$ be a vector containing three random variables representing the (x, y, z) coordinates of a random labeled 3D point. The standard deviations $\sigma_x, \sigma_y, \sigma_z$ for each coordinate of the N points within each point cloud are defined as described in equation V-(1):

$$\sigma_{x,y,z} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(P_i^{x,y,z} - \mu_{x,y,z})^2}, \text{ where } \mu_{x,y,z} = \frac{1}{N}\sum_{i=1}^{N} {P_i}^{x,y,z} \qquad \text{V-(1)}$$

Note that all three $\sigma_x, \sigma_y, \sigma_z$ dispersion values are computed for each point cloud sub-part, to measure the 3D closeness of the point to the corresponding estimated body part.

The Chebyshev's inequality (Alsmeyer, 2011) is used to filter out the points that are geometrically far from the mean position of the body part point cloud set. The inequality has great utility and can be applied to any probability distribution in which the mean and variance are known. Within each sub-part of the point cloud, it is guaranteed that a minimum of 75% of values must lie within two standard deviations of the mean and 89% within three standard deviations. This procedure is particularly useful for filtering out mistaken points.

After the SOR filter, all the unlabeled points are transmitted to a greedy clustering process. Since the body parts are always neighbors, a KD-tree-based nearest neighbor search (Bentley, 1975b) is applied here to group the unlabeled points. At first, a search region is defined for each unlabeled query point. Within the search region, the sums of different labels in the neighboring points are counted. The highest value stands for the most likely body part label to assign to the query point. The search region is enlarged adaptively. If there are not enough neighbors, the search window is enlarged, and the clustering process iterates until all the points are labeled and grouped into clusters.

After the 3D integration of the 2D skeleton information, each 3D point receives now a label representing the body part to which it belongs, for each frame of the point cloud sequence.

5.3. Motion estimation and compression

Rigid-transformation-based algorithms such as the ICP algorithm (Besl & McKay, 1992b) have shown to be an efficient method for estimating rigid-body motion (Simon et al., 1994). However, applying only rigid transformation is not sufficient to describe accurately the 3D point motion, notably in the case where the motion is caused by highly deforming objects such as clothes, skin, or hair. In our work, we have adopted an affine transformation prediction model.

5.3.1. Affine transformation

In addition to rotation and translation that are present in 3D rigid transformations, a 3D affine transformation holds 6 more degrees of freedom (DOF) which allows shearing and anisotropic scaling of the 3D objects. The additional 6 DOF increase the possibility of achieving better prediction for deformable motion. Any combination of such transformations can be described by a single four-by-four affine transformation matrix, under homogeneous coordinates. The transformation of point P with position (x, y, z) to (x', y', z') is defined as described in V-(2):

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \qquad \text{V-(2)}$$

Within the four-by-four 3D affine transformation matrix, the upper-left three-by-three sub-matrix represents the homogeneous component (*i.e.*, the combination of rotation and non-uniform scaling/shearing) and the last column of the matrix represents the translational part.

For each body sub-part, the affine transformation can be efficiently estimated with the help of an ICP-based process. In our experiments, 14 affine transformation matrices are computed for the corresponding 14 body parts in each predicted point cloud frame. The global motion estimation is achieved by merging the individual body part motions. The resulting affine transformation matrices are encoded and transmitted for applying motion estimation on the decoded point clouds.

5.3.2. V-PCC based motion estimation and compression

The proposed 3D motion estimation approach has been integrated into the V-PCC test model introduced in (S Schwarz et al., 2019), as illustrated in Fig. V-3.

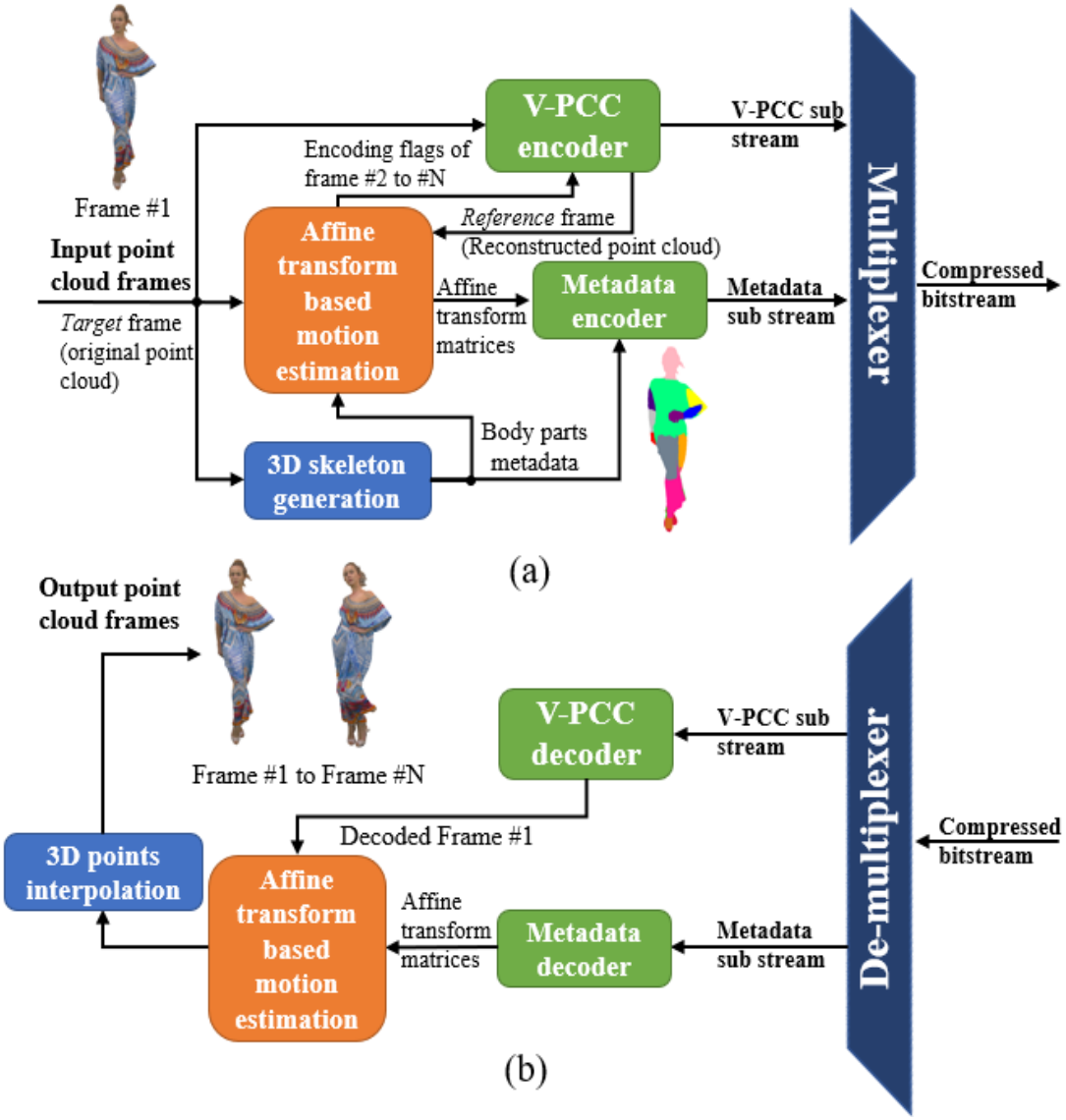


Fig. V-3. (a) Encoding process and (b) decoding process of the proposed V-PCC-based compression scheme with affine transformation-based motion estimation (N represents the size of the GOF).

On the encoder side, after the offline 3D skeleton generation, each point of the input point cloud contains a corresponding body part label. The body part labels are considered as metadata to serve for affine transformation-based motion estimation. Let us underline here that the motion estimation process is completely different from the one used in traditional 2D video compression. The proposed motion estimation indicates the computation of the affine transformation matrices between the *reference* frame and the *target* frame. The *reference* frame is affine-transformed to be geometrically as close as the *target* frame. A GOF structure is used here for rate control purposes, while an encoding flag is set to bypass the encoding of certain frames using V-PCC. Within each GOF encoding process, only the first frame is considered as a reference frame and encoded with the original V-PCC intra-coding process. The remaining $N-1$ frames in the GOF are motion estimated with the proposed method. The computed affine transformation matrices for each body part and the labels of the first point cloud frame within each GOF are transmitted to the metadata encoder and are then entropy encoded.

On the decoder side, the decoded affine transformation matrices are applied accordingly to the labeled point cloud segments for all the *target* frames within the GOF. Note that only motion estimation is applied on both the encoder and decoder sides. The corresponding motion compensation residuals are completely discarded by our approach. To fully motion compensate for each 3D point in the unstructured point clouds, massive residuals are needed. To overcome that, only motion estimation is applied to each body part as the simulation of skeleton-based motion. A simple linear interpolation process for 3D points is used between every two adjacent connected segments, as illustrated in Fig. V-4.

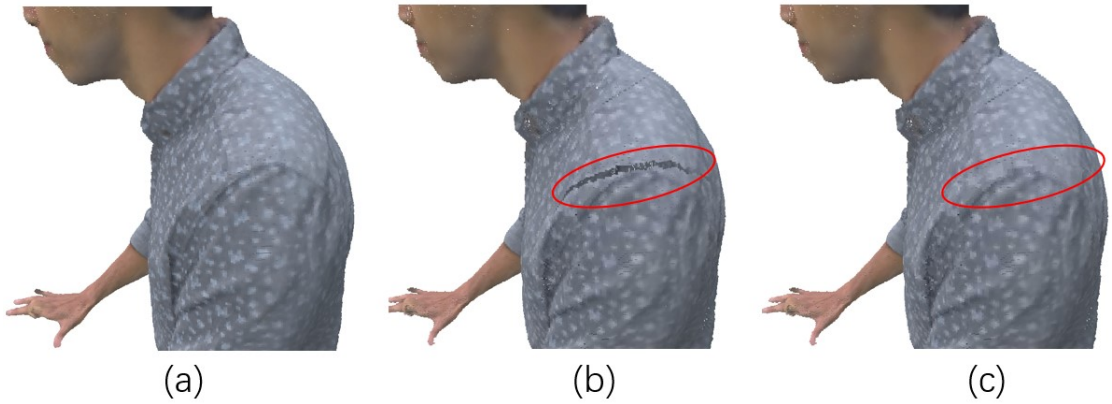


Fig. V-4. Example of the (a) original frame, (b) the estimated frame and, (c) the frame with interpolated points of loot.

Applying different affine transformation matrices on different body parts can cause the appearance of a gap as shown in Fig. V-4 (b). A 3D point interpolation process is used here to detect the edges and interpolate points between such edges. The color of the interpolated point is determined according to the distance to the corresponding two edges.

5.4. Experimental results

5.4.1. Results with skeleton-based segmentation

The proposed scheme is tested on part of the *8i* dataset (E. d'Eon, B. Harrison, T. Myers, 2017), *i.e.* the *loot, redandblack, soldier,* and *longdress* sequences containing more than 0.7 million colored points per frame The V-PCC test model version 8.0 (MPEG, n.d.) has been used in our experiments for the integration of the proposed motion estimation process. The comparison of compression performance concerns the V-PCC inter-coding mode. The results tested on the first 32 frames of the *soldier* dataset are illustrated with different GOF sizes. Five conditions aiming at different video encoding rates are tested using the V-PCC configurations as described in (3DG, 2020) where R01 stands for an encoding condition with the lowest bitrate and R05 corresponds to the highest one.

By using affine-transformation-based motion estimation on the clustered point clouds, the bitstream size is greatly reduced in the case where the motion presents relatively poor amplitudes, as illustrated in Fig. V-5, for the sequence *soldier*.

Fig. V-5. The decoded frame of soldier using V-PCC in R01 condition (left). The estimated frame of soldier using the proposed method with equivalent encoding cost (right).

The left image is an example of a decoded frame using the V-PCC approach with an average video encoding rate of around 1.61 Mbps. The right image is the corresponding decoded frame with the proposed method encoded with an equivalent rate of around 1.38 Mbps. Instead of encoding all the frames within the GOF structure, only the first frame is encoded with a higher video encoding rate. The rest of the frames are motion-estimated from the decoded first frame, with affine transformation. We observe that higher visual quality is achieved using the proposed method with around 2dB and 3dB gain in terms of mean PSNR scores of the point-to-point geometric distance and the Y component respectively.

The PSNR value is used in Fig. V-6 for geometry to evaluate the closeness of the point-to-point distance between the original point cloud and the decoded/estimated point cloud. Five GOF sizes are selected to simulate different network conditions.

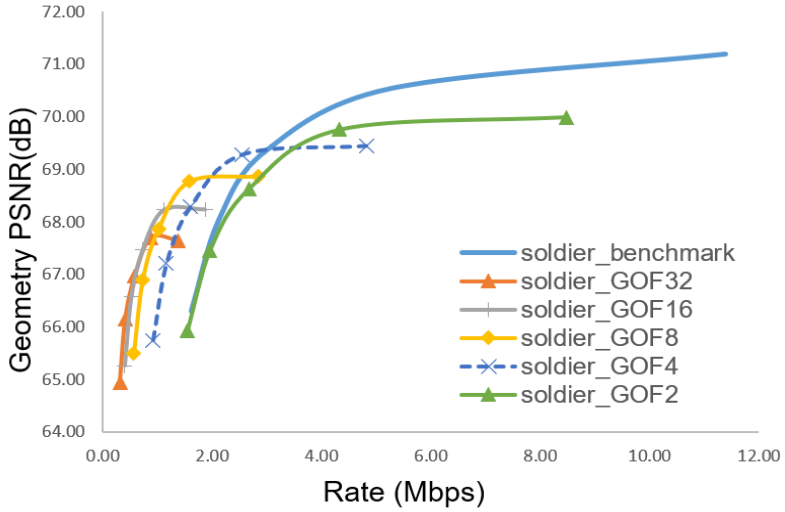


Fig. V-6. The geometry RD curves of the V-PCC (benchmark) method and proposed method tested on the first 32 frames of soldier.

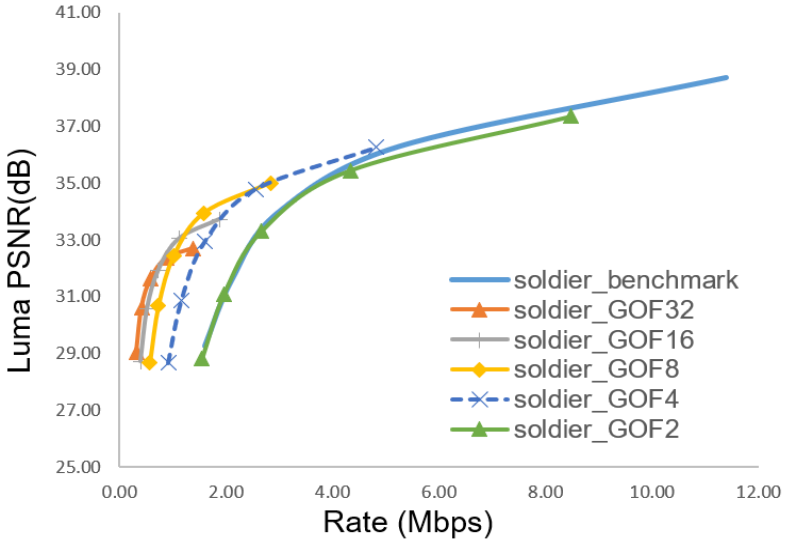


Fig. V-7. The luminance RD curves of the V-PCC (benchmark) method and proposed method tested on the first 32 frames of soldier.

We can observe, without surprise, that a larger value of the GOF size leads to more efficient 3D point cloud encoding. This is explained by the fact that for an entire GOF, solely the first, reference frame is completely encoded. For the rest of the frames, exclusively the affine parameters are transmitted. We can also observe that the proposed method outperforms the V-PCC approach, notably at low and very low bitrates.

The PSNR values obtained for the luminance component of the YUV color between the original point cloud and the decoded point cloud are shown in Fig. V-7. In our experiments, RGB-to-YUV420 color conversions are performed in all the testing conditions for color compression. Thus, only the results for the Y component, which is the most important, are reported. A similar conclusion can be driven here, with better color estimations at low bit rate conditions.

One of the advantages of the proposed method is that higher fidelity of the point cloud is preserved in terms of both geometry and color with equivalent encoding cost in some particular circumstances. With the proposed motion estimation, we can V-PCC encode the first frame solely in the R05 condition and estimate the remaining 31 frames in the GOF instead of encoding all the frames at a lower rate. Similarly, much lower bits are needed while achieving the same visual quality. The greatest bitrate reduction is observed when we use the GOF32 structure in *soldier* where motion with small amplitudes is observed. In this condition, the bitstream is reduced by 83% with the proposed approach when compared to the V-PCC method.

Though offering certain advantages in particular cases, the proposed method presents several limitations for more generic motion. The experiments on *loot, longdress* and *redandblack* show that the method fails to estimate some particular motions, such as facial expressions as well as clothes and hair deformations. The PSNR curves obtained for both geometry and color information for the *loot* sequence are presented in Fig. V-8 and Fig. V-9.

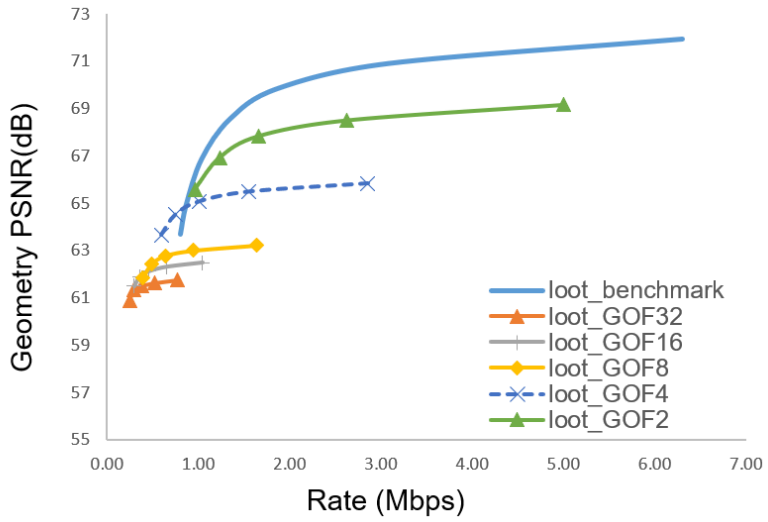


Fig. V-8. The geometry RD curves of the V-PCC (benchmark) method and proposed method tested on the first 32 frames of loot.

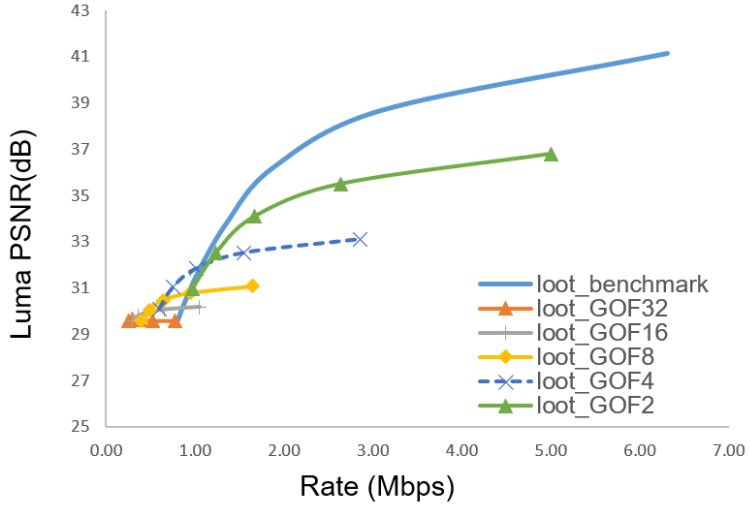


Fig. V-9. The luminance RD curves of the V-PCC (benchmark) method and proposed method tested on the first 32 frames of loot.

Though achieving a lower coding rate, the proposed method experiences a decrease in geometry and color PSNR values, achieving lower bitrates.

For content with more complex and higher motion amplitudes, no gain has been observed, as illustrated in Fig. V-10 for the *longdress* and *redandblack* sequences.

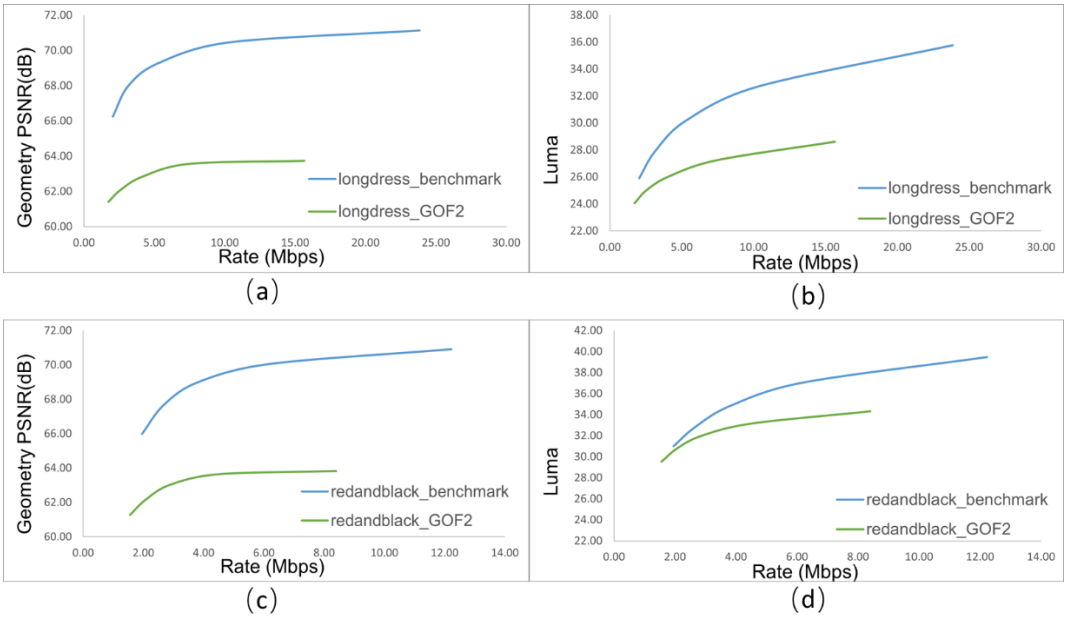


Fig. V-10. The geometry and luminance RD curves of the V-PCC (benchmark) method and proposed method tested with GOF2 configuration on longdress ((a) and (b)) and redandblack ((c) and (d)), respectively.

The poor performance of the proposed method is caused by the inaccurate motion estimation achieved by the affine transformation model. Note that the affine transformation matrices are determined between the segmented point clouds from consecutive frames where they may have great variation in terms of number and geometrical coordinates. The SVD (Singular Value Decomposition) approach (Golub & Reinsch, 1970) used to compute the affine transformation always tries to minimize the distance between the original point and the corresponding nearest neighboring point. However, when great motion is observed in the point cloud sequence, we observe quite often that the mapping is not correct by using only geometry information when searching for nearest neighbors.

In Fig. V-11 (a), the foot part of frame 1051 in *longdress* is illustrated in blue whereas frame 1052 is illustrated in red. The points should match with each other as shown in the yellow circles. However, when we try to find the corresponding points to compute the affine transformation matrices, the blue points on the instep of the foot are more likely to match with the red ones on the sole. As a result, the achieved affine transformation matrices transform the point cloud into a poorly estimated shape, illustrated in Fig. V-11 (b).

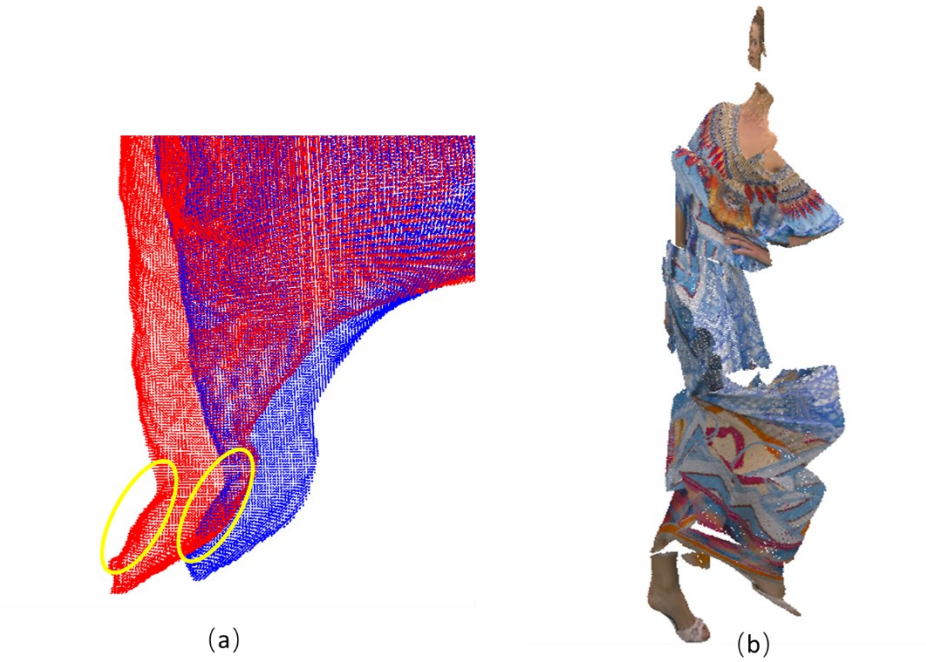


Fig. V-11. (a) Illustration for mapping problem using nearest neighbors. (b) An example of a poorly estimated point cloud of longdress.

Such limitations show that achieving accurate motion estimation by applying affine transformation on a roughly spatial segmented point cloud (*i.e.,* a human with only 14 skeletons in our experiments) remains a challenge. In particular, the varying and deforming clothes and hair remain the principal reason for the poor motion estimation, additionally affected by the inaccurate 3D skeleton generation (Fig. V-12 (a)).

Fig. V-12. (a) The generated 3D body parts of the tested sequences (the segments with incorrect labels are circled, from left to right: soldier, redandblack, longdress, and loot). (b) the original point clouds.

Since the skeleton generation system highly depends on DensePose and it is not able to distinguish between clothes, hair and other objects, some parts are wrongly labeled (red circles in Fig. V-12). The inconsistent labels result in misleading motion estimation. In this case, the affine transformation is applied to a poorly estimated segment and the skeleton-based structure is not preferable for estimating deformations caused by such components. Thus, a hybrid RDO procedure is expected in the future study to include both rigid transformation (for rigid structures as skeletons) and an affine transformation (for deforming hair and clothes) to overcome such a problem. Meanwhile, a smarter motion-based segmentation algorithm should be investigated as well.

5.4.2. Results with a clustering based on K-means++ algorithm

We have shown that the generated 3D anatomical segments are not accurate in terms of represented human body parts which penalizes the motion estimation process. Meanwhile, the segmentation does not contain smooth contours and causes the appearance of holes/gaps between the transformed body parts. To overcome this problem, we propose to experiment with a different clustering method using the K-means++ algorithm.

K-means++ (Arthur & Vassilvitskii, 2007) is an algorithm for choosing the initial values (or "seeds") for the k-means clustering algorithm (James and others, 1967). It attempts to avoid the occasional poor clustering problems observed from the standard k-means algorithm. The intuition behind this approach is to spread out the k initial cluster centers and this yields considerable improvements in the accuracy of clustering. With the k-means++ initialization, the algorithm is guaranteed to find a solution that is competitive to the optimal k-means solution.

In our proposal, a 3D point with a maximum/minimum coordinate value is selected as the first candidate of the k-means centroid. Then, the furthest point from the first candidate point is selected. The process iteratively selects the next centroid maximizing the distances between the new point and all the current candidates. To fairly compare the results with the skeleton-based approach, the same number of clusters (14 segments) has been selected. A comparison of the clustering of the skeleton-based method and the one using k-means++ partitioning is proposed in Fig. V-13.

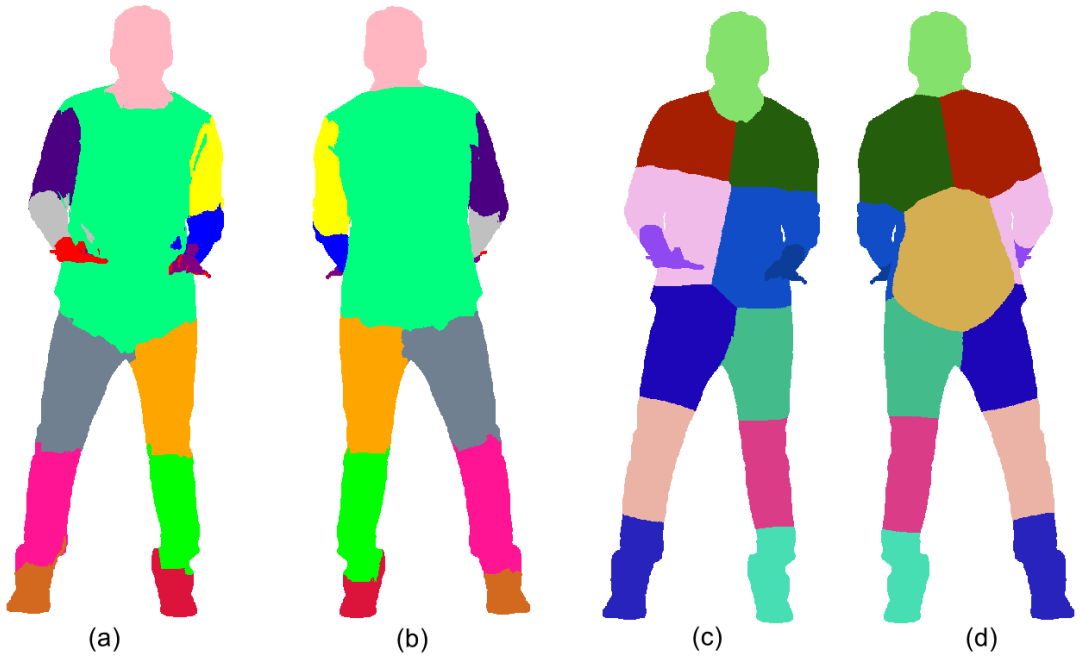


Fig. V-13. A comparison between the clustering of the skeleton-based method ((a) and (b)) and the k-means++ based method ((c) and (d)) with 14 segments.

As shown in Fig. V-13, the contours of the resulting clusters using the k-means++ based method are smoother than the ones using the skeleton-based method. To globally evaluate the impact of the segmentation on the overall compression performances, two different GOF configurations are retained, with 32 frames and 2 frames per GOF. The obtained RD curves of geometry and luminance information are illustrated in Fig. V-14 and Fig. V-15, respectively.

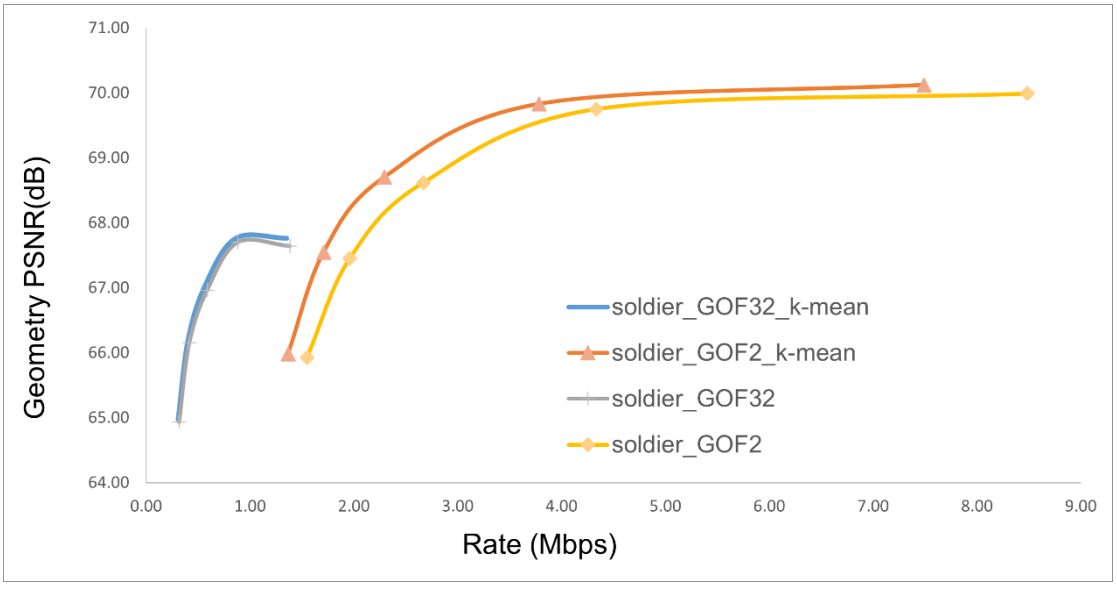


Fig. V-14. A comparison of geometry RD curves between the proposed skeleton-based method and the k-means method.

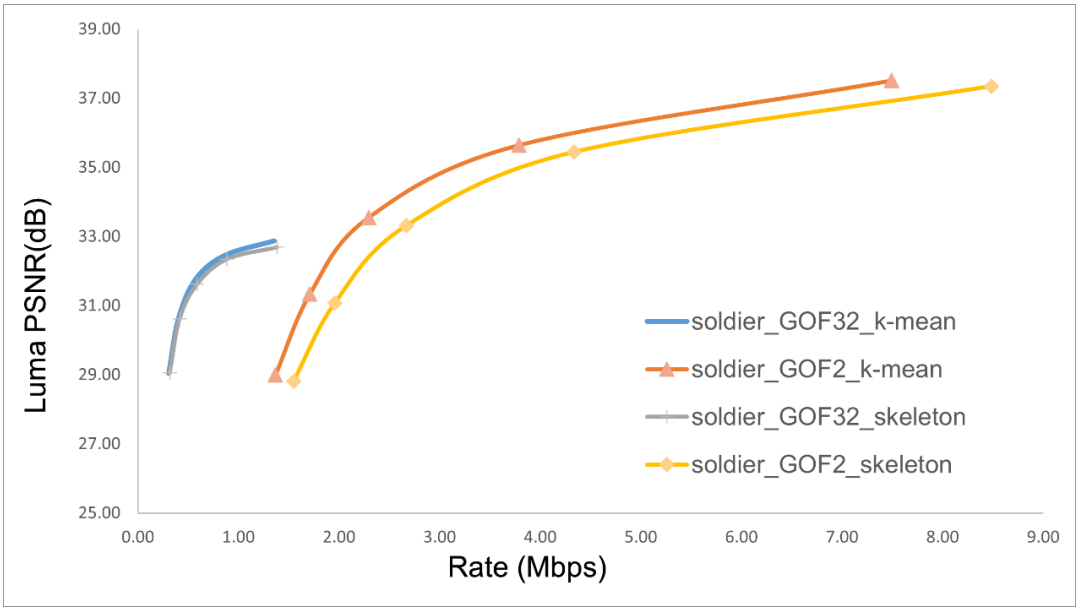


Fig. V-15. A comparison of luminance RD curves between the proposed skeleton-based method and the k-means method.

We can observe that with the same number of clusters, clusters with smoother contours can help the V-PCC encoder to generate smoother and more temporally consistent 2D patches, resulting in a more compression-friendly video. Thus, better visual quality can be achieved with the same rate as shown in Fig. V-14 and Fig. V-15. This result reveals that the performance of the skeleton-based method can be further improved with a smoother clustering.

5.5. Conclusion

In this chapter, a skeleton-based motion estimation method, for compressing dynamic human-shaped dense point clouds, has been proposed. By using a pre-trained DensePose R-CNN, all the 3D points in the point cloud are classified with a body part label and grouped into segments accordingly. The affine transformation matrices of different segments are then computed between consecutive point cloud frames for further motion estimation. A V-PCC-based compression scheme is proposed with motion estimation using different GOF sizes. Experimental results on five encoding conditions show that the proposed method outperforms the V-PCC method in low to very low bitrate conditions, when applied on dynamic point clouds with relatively small motion amplitudes.

Though presenting several limitations, the proposed method can greatly improve the visual quality of the decoded point clouds in low rate conditions for low-motion content. Such characteristic is beneficial for VR/AR applications, where the network bandwidth is limited and the content fidelity is in demand. Moreover, the skeleton-based method has many potentials in terms of useful features. Functionalities such as ROI and LODs, which are widely used in computer graphics, are expected in future studies. Finally, we have also shown that the compression performance can be further improved by clustering based on the k-means++ algorithm. Thus, further work is expected to refine the segmentation of the point cloud and to improve the motion estimation and compensation for point clouds with higher motion amplitudes.

CHAPTER VI. TEMPORAL PREDICTION USING ANATOMICAL SEGMENTATION

In this chapter, a novel prediction scheme is proposed for dynamic point clouds with higher motion amplitudes. It is based on our previous contribution, introduced in Chapter V, which has been here refined in different aspects. For example, an iterative closest point (ICP) algorithm is used to improve the accuracy of motion estimation. A smoothing and a 3D patch partitioning process are carried out to obtain more temporally-consistent 2D projected images. Moreover, in order to render more accurate the motion estimation process, we propose a novel prediction scheme using inverse affine transformation to predict from the previous reconstructed frame. The experimental results obtained show that a better prediction can be achieved compared to the previous approach for content with higher motion amplitudes.

Institut Polytechnique de Paris

91120 Palaiseau, France

6.1. Introduction

The method introduced in the previous chapter has shown interesting results for content that contains low motion amplitude, when coding in low to very low bit rate conditions. However, it is still challenging to balance the tradeoff between visual quality and compression ratio. Likewise, in our work, increasing the amount of affine transformation with a smarter segmentation approach may help us achieve a better prediction of motion. Yet, a huge number of resources are required to realize accurate motion compensation, especially for dynamic content. (e.g., *soldier* contains around 1 million points per frame with 30 fps).

In the previously proposed method, the first frame of the point cloud sequence has been selected as the reference frame for all the other point clouds within the GOF structure. The resulting prediction error grows rapidly as the coding continues while using only one fixed reference point cloud. To avoid such problems, a novel approach is proposed here. Instead of coding only one reference frame and transforming it into other frames, we propose to inverse the process. First, all the other frames within the GOF are transformed to get as close as possible to the reference frame. Then, the same V-PCC pipeline with modified smoothing, patching, and packing strategies is applied to each transformed frame so that the 2D packed images become more temporal consistent in terms of corresponding 2D shapes and location on the packed images. Finally, the inverse transformation is applied on the decoder side for reconstruction.

Moreover, we introduce a new dynamic 3D point cloud dataset created from animated human-shaped meshes. It contains three sequences with different types of motion amplitudes. Experimental results show that better compression performances can be obtained.

6.2. A novel dynamic 3D point cloud dataset

We have observed in the previous chapter that the points in the *8i* dataset contain geometrical shifts even when the model remains relatively static. The point cloud sequence is captured with multi-view cameras and each frame of the point cloud is generated using SfM-MVS photogrammetry. As the point cloud frames are reconstructed individually, the geometrical shift between points in consecutive frames is observed. In order to measure whether the influence of this artifact can be reduced, we have considered a novel point cloud synthetic dataset, created with the help of animated 3D meshes. The original 3D meshes downloaded from (https://renderpeople.com/free-3d-people/) contain the animation of three different human models with *walking*, *idling*, and *dancing* motions, respectively. Some example point cloud frames from such sequences are illustrated in Fig. VI-1.

Fig. VI-1. The 3D point cloud frames of sequence walking (top), idling (middle), and dancing (bottom) in the novel dataset.

The *walking*, *idling*, and *dancing* sequences contain 100, 612, and 1341 frames, respectively. Each frame is consists of around 0.7 million points with their 3D coordinates (x, y, z) and color attributes (R, G, B).

They are originally modeled using the 3DSMAX development platform, with the same number of pre-defined bones and joints. The built-in function in 3DSMAX helps us to convert the models into 3D meshes. The same parameter is selected to sample each 3D model with a different timestamp into an individual 3D mesh with around 8000 vertices and 16000 faces. Then, Meshlab (Cignoni et al., 2008) is used to compute the normal information for each vertex and to sample the 3D meshes into 3D point clouds using Poisson-disk sampling (Corsini et al., 2012). A voxelization step is executed here, where the sampled points are constrained to lie on a regular 3D voxel grid for each point cloud frame. Meanwhile, the spatial resolution is set to be the same as in the *8i* dataset, i.e. 1024x1024x1024 (also known as depth 10). Finally, the surface color information from the 3D meshes is transmitted, so that each point of the 3D point cloud contains color information that is estimated from the closest mesh surface. For simplicity, this novel dataset will be denoted as *renderpeople dataset*, since it is reproduced from the model created by https://renderpeople.com/.

6.3. Prediction structure

Several improvements are proposed in this contribution, including improved clustering of the 14 anatomy segments with temporal prediction between consecutive frames, more accurate motion estimation using Iterative Closest Point before affine transformation, and modified process of smoothing, patching, and packing strategy in the V-PCC coding method. An example of the prediction structure is illustrated in Fig. VI-2, using four consecutive frames (PC_n to PC_{n+3}) from the *walking* sequence.

The same process as the one described in Section 5.3.2 is performed here to obtain the 3D pose estimation. The PC_n is first projected in 2D with four orthogonal orientations and DensePose (Güler et al., 2018) is used, for each view, to segment the body silhouette in anatomical segments. Then, the segmented 2D pixels are back-projected onto corresponding 3D points, with the help of preserved depth information. After an iterative clustering and smoothing stage, the segmented PC_{n_seg} sequence is generated, carrying the labels associated with the resulting 14 anatomical segments.

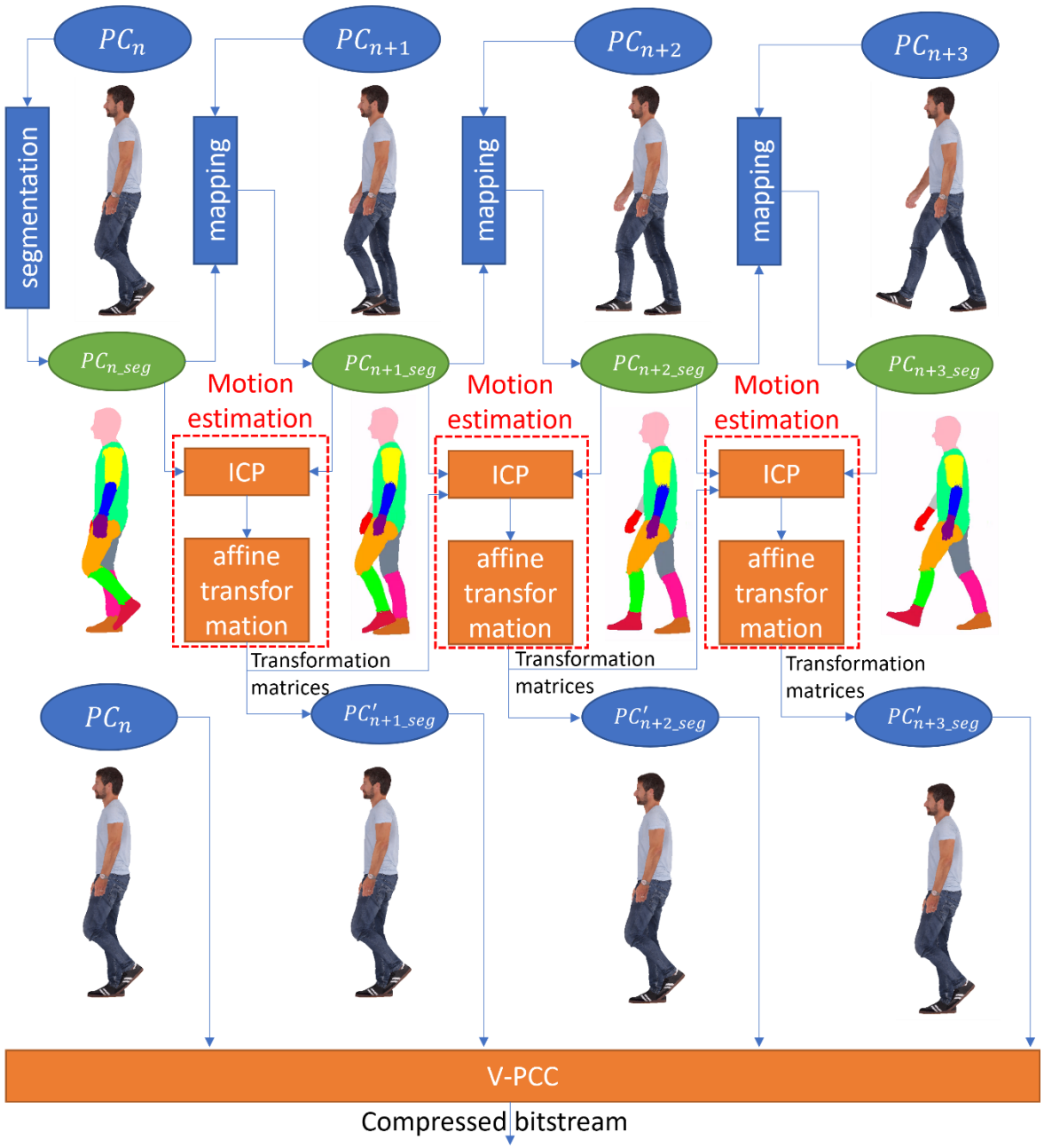


Fig. VI-2. An example of the proposed prediction structure coding four consecutive frames of walking.

In addition, several modifications of the V-PCC test model are proposed, in order to achieve more temporal consistency between the generated consecutive 2D images. First, the temporal consistency of the 3D segmentation process must be ensured. Thus, a mapping stage is required to build the correspondence between the PC_{n_seg} and PC_{n+1}The mapping is achieved by using the ICP algorithm, which first estimates the rigid motion of the body parts, followed by an affine transformation for estimating possible deformation of the body.

After that, motion estimation is applied to the mapped and reconstructed point cloud frames PC_{n_seg} and the PC_{n+1_seg}. The resulting affine transformation matrices are applied to compensate for the geometry of the reconstructed point clouds towards the reference frame. Meanwhile, the colors are assumed to be moving along with the geometry.

Finally, all the transformed point clouds, with a more similar shape as the untransformed PC_{n_seg}, are transmitted to the original V-PCC coding process with adapted patching and packing strategies.

The main idea behind this process is to obtain more temporal consistency for the dynamic point clouds in the transformed 3D space. The original patching method in the V-PCC test model takes advantage of both normal and geometry information to cluster the 3D points in each frame. However, it does not take into account the temporal consistency of the clustering result. By providing an anatomical segmentation followed by normal-based patching, it is more likely to obtain more similar 2D frames. Therefore, the video codec can perform better compression on the packed images. Let us now focus on the various improvements proposed.

6.4. Improved anatomy segmentation

6.4.1. Problem statement

As illustrated in Fig. V-12, the 3D segmentation of the various anatomical parts is not sufficiently accurate. The difficulty for labeling has been observed when different body parts are geometrically close to each other. These problems are caused by the inconsistent result generated by DensePose and by the fusion of the 3D point cloud using the resulting images.

6.4.1.1. Inconsistent output from DensePose

The 3D point cloud is first projected with different orientations to simulate various viewpoints onto 2D images. Such images are then fed to DensePose to output segmented 2D images according to 14 predefined anatomy body parts. However, the resulting 2D labeling result for the anatomy segments from DensePose output is not accurate and has brought challenges to label and cluster the 3D points correctly and consistently. As illustrated in Fig. VI-3, compared to the original input image on the left side, the right image contains less information near the contour of the head (sequence *loot*). This information is lost during the DensePose estimation process.

Fig. VI-3. The input image (left) to DensePose of the face part from loot and the segmented output image (right) from DensePose. (Red lines represent the contour of the head)

The reasons behind this result are the following:

1) DensePose is pre-trained with DensePose-COCO, a large-scale ground-truth dataset, with image-to-surface correspondences manually annotated on the 50K COCO images. However, the estimation obtained from DensePose cannot achieve per-pixel accuracy. Since we do not dispose of a ground truth annotation of our images, it is not realistic to retrain DensePose and improve the estimation performance without investing a huge amount of effort.

2) While the color of the pixel is similar to the background, it is confusing for DensePose to distinguish the difference between useful information and removable background.

The inconsistency of the DensePose generated segmentation results are not only spatial but also temporal. It can be observed by comparing the same body part with different views and the same body part between segmented images at different timestamps, as illustrated in Fig. VI-4.

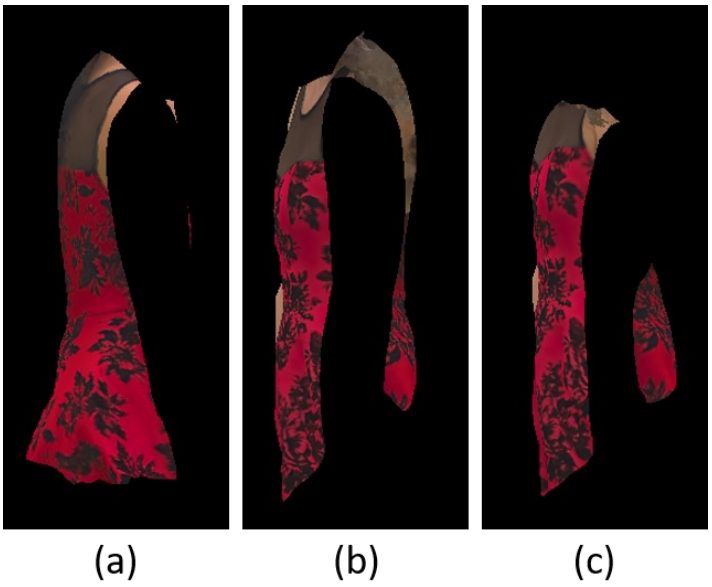


Fig. VI-4. The output image from DensePose of the segmented torso part of redandblack with (a) the right view of Frame 1, (b) the left view of Frame 1, and (c) the left view of Frame2.

In Fig. VI-4 (a), some pixels from the arm are considered as part of the torso while the hair is clustered into the torso as shown in Fig. VI-4 (b). The difference between the same *torso* part of the segmented images from Frame 1 (b) and Frame 2 (c) is distinct. This result is due to the great amount of motion observed in the *redandblack* sequence. Both spatial and temporal inconsistencies of the 2D labels brings difficulties for fusing them into the 3D space for sequential frames, especially for content with high motion amplitudes.

6.4.1.2. The difficulty of 3D fusion

After the human pose estimation step performed by DensePose, the 2D pixels are back-projected onto the 3D space using preserved depth information. Since some 3D points are projected onto images with different orientations, when they are projected back into the 3D space, multiple labels of the same 3D point can be achieved. Then a majority vote process is applied to keep only a single label representing the body part for that 3D point.

After that, a smoothing stage consisting of a SOR filter and a greedy clustering process is performed to filter out the poorly estimated labels and to fill the empty labels.

The 3D segmentation of the point clouds holds a strong impact on how accurate the motion can be estimated. The examples illustrated in Fig. VI-5, (a), (b), and (c) correspond to the successive smoothing stages, already introduced in Chapter V. However, the boundary between each anatomy segment is not smoothed while some points are poorly estimated on the feet of *longdress*. To overcome these limitations, we propose to use neighboring information for each point to smooth out the poorly estimated labels so that we can achieve better prediction as shown in Fig. VI-5 (d).

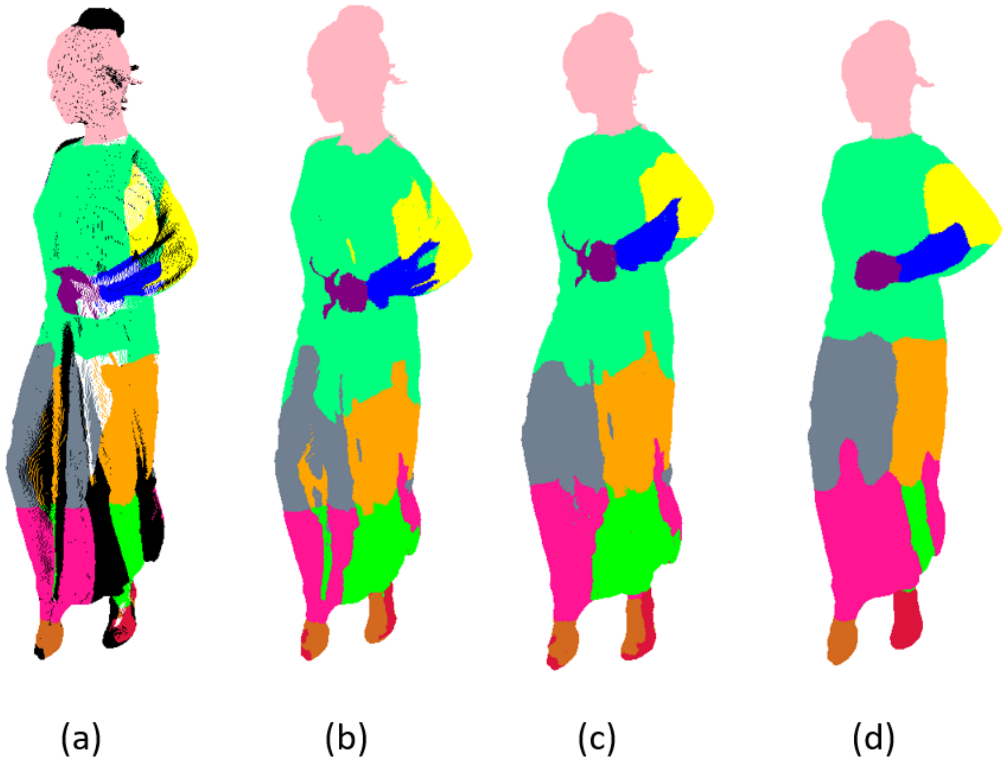


Fig. VI-5. The longdress with (a) integrated labels from DensePose after the majority vote. (b) The result after greedy filling using nearest neighbor search. (c) The SOR filtered result and (d) the result with the proposed improvements, based on neighbor information.

The proposed algorithm uses neighbor information and is described in the following section.

6.4.2. Smoothing using neighboring information

We propose to improve the smoothing stage by an iterative process using the neighboring information of each point using a KD tree. For each query point, a pre-defined number of neighboring points are determined. Then, the labels of the neighboring points are gathered to help decide whether to preserve the current label of the query point or assign a new label to it.

For example, to determine the label of a point "A", the current label of "A" and the ones of its neighbors are collected. If the current label of "A" does not match with the ones from its neighbors, a new label will be assigned to "A". The new label will be the one that occupies the largest proportion within the neighboring labels.

However, the exploited DL-based human pose estimation method is not able to accurately predict the human pose when deforming clothes and other various objects/accessories are covering the human body. As shown in Fig. VI-5 (d), there are still some poorly estimated points on the left lower leg part.

Moreover, different smoothing parameters (e.g., the number of points used for neighbor search and threshold for iteration) are tested and adapted for each sequence. This helps us to provide comprehensive clustering results for content with various motion amplitudes.

Instead of using the clustering result generated for each point cloud frame, the one from the first frame is used as a predictor for other frames, in order to keep the temporal consistency of the resulting labels. We use the proposed motion estimation based on affine transformation to find the correspondence between points in consecutive frames. As a result, the generated anatomical segments are more temporally consistent in terms of shape and number of points, which brings benefits for further estimation and compression.

6.4.3. Improved affine transformation with prior rigid ICP registration

When compared to a rigid transformation, the 3D affine transformation holds 6 additional DOFs, which increases the possibility of achieving better prediction for deformable motion. However, the motion of the anatomical segments previously determined is mainly rigid. It is reasonable to consider a rigid transformation step before the affine transformation one, in order to reduce the effect of unnecessary scaling and shearing. We have observed in the previous contribution that the reference point cloud can be over-sheared/scaled to fit the target one especially when the motion amplitude of that segment is large. As a consequence, the resulting affine-transformed point cloud contains a great number of duplicated points and distorted shapes than the original reference.

In order to overcome this limitation, we propose to include an ICP rigid registration step, prior to the computation of the affine transformation between reference and target point cloud. In this way, a better initialization can be achieved for computing the affine transformation. Thus, the over-sheared/scale problem can be avoided.

The ICP algorithm, first introduced in (Chen & Medioni, 1991) and (Besl & McKay, 1992a), is widely used to minimize the registration differences between two 3D point clouds. There exist many adapted ICP algorithms to solve different registration tasks. We have chosen to use the classic ICP algorithm from (Besl & McKay, 1992a), since we are not aiming to find a perfect registration but to reduce the effect of the unwanted shearing and scaling from poor estimation using affine transformation solely.

The ICP algorithm consists of the following steps, illustrated in the pseudo-code presented in Fig. VI-6.

```
threshold_min=max_value;
while ( threshold > threshold_min || Number of iteration < for example 20) do {
        ComputeCentroid (X, P);
        R = ComputeRotation (X, P);              (cf. equation VI-(13))
        T = ComputeTranslation (X, P);           (cf. equation VI-(14))
        (X_transformed, P_transformed) = ApplyTransformation (X, P, R, T);
        threshold = ComputeError (X, P, X_transformed, P_transformed);
        if ( threshold < threshold_min) {
                threshold=threshold_min;
        }
}
Output=output_final(R, T);
```

Fig. VI-6. Pseudo-code for the used ICP algorithm.

The ICP algorithm is usually used for two corresponding point sets, with equivalent numbers of points. In our case, it is notably not the case, since the number of points can be affected by the initial clustering result and by the temporal prediction of the generated anatomy segments. Let us denote the two point sets by the *reference* point set X and the *target* point set P:

$$X = \{x_1, x_2, \cdots, x_n\} \quad \text{VI-(1)}$$

$$P = \{x_1, x_2, \cdots, x_m\} \quad \text{VI-(2)}$$

We need to build the correspondence between the two point sets so that there is one corresponding point in X for each point in P. A KD tree structure is used here to search for the closest neighboring point for all the m points in P and the correspondence map is updated during the iterative ICP process.

The ICP algorithm is based on the least square's method for iterative calculation so that the sum of squared errors reaches a minimum:

$$\min_{R,t} J = \frac{1}{2}\sum_{i=1}^{m} ||\left(x_i - (Rp_i + t)\right)||^2 \quad \text{VI-(3)}$$

To find the corresponding rotation translation components R and t, the equation VI-(3) can be solved in the following steps:

(1) Compute the centroids for the *reference* and *target* set of points to simplify the objective function.

The centroids of the two sets of points can be defined as:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{VI-(4)}$$

$$\bar{p} = \frac{1}{m}\sum_{i=1}^{m} p_i \qquad \text{VI-(5)}$$

Equation VI-(3) can be simplified as follows:

$$\begin{aligned}\frac{1}{2}\sum_{i=1}^{m} ||\left(x_i - (Rp_i + t)\right)||^2 &= \frac{1}{2}\sum_{i=1}^{m} ||\left(x_i - Rp_i - t + (\bar{x} - R\bar{p}) - (\bar{x} - R\bar{p})\right)||^2 \\ &= \frac{1}{2}\sum_{i=1}^{m} ||\left(x_i - \bar{x} - R(p_i - \bar{p}) + (\bar{x} - R\bar{p} - t)\right)||^2 \\ &= \frac{1}{2}\sum_{i=1}^{m} \Big(||x_i - \bar{x} - R(p_i - \bar{p})||^2 + ||\bar{x} - R\bar{p} - t||^2 \\ &+ 2\left(x_i - \bar{x} - R(p_i - \bar{p})\right)^T (\bar{x} - R\bar{p} - t) \Big)\end{aligned} \qquad \text{VI-(6)}$$

Since the sum of $\left(x_i - \bar{x} - R(p_i - \bar{p})\right)$ is zero, the objective function can be simplified to:

$$\min_{R,t} J = \frac{1}{2}\sum_{i=1}^{m} (||x_i - \bar{x} - R(p_i - \bar{p})||^2 + ||\bar{x} - (R\bar{p} + t)||^2) \qquad \text{VI-(7)}$$

As shown in equation VI-(7), the first item is only related to the rotation matrix R and the translation matrix t can be obtained by setting the second item to zero once the rotation matrix R is computed.

(2) Calculate the rotation matrix R using the de-centroid points and compute the translation matrix t using the SVD algorithm.

The de-centroid coordinates of the points in the *reference* and *target* point sets can be represented as:

$$x_i' = x_i - \bar{x} \qquad \text{VI-(8)}$$

$$p_i' = p_i - \bar{p} \qquad \text{VI-(9)}$$

Substituting them into the first item in equation VI-(7), we obtain:

$$R^* = arg\min_R \frac{1}{2}\sum_{i=1}^{m} ||x_i^{'} - Rp_i^{'}||^2$$

$$= arg\min_R \frac{1}{2}\sum_{i=1}^{m} \left(x_i'^T x_i' + p_i'^T R^T R p_i' - 2x_i'^T R p_i'\right) \quad \text{VI-(10)}$$

Among them, only the third term is related to R since $R^T R = I$ as identity matrix.

The SVD algorithm (Golub & Reinsch, 1970), first define the following matrix:

$$W = \sum_{i=1}^{m} x_i' p_i'^T \quad \text{VI-(11)}$$

The decomposition of W is denoted by:

$$W = U \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} V^T \quad \text{VI-(12)}$$

where $U, V \in \mathbb{R}^{3\times3}$ are unitary, and $\sigma_1 \geq \sigma_2 \geq \sigma_3$ are the singular values of W. If $rank(W) = 3$, the optimal solution of $J(R,t)$ is unique and is given by:

$$R = UV^T \quad \text{VI-(13)}$$

$$t = \bar{x} - R\bar{p} \quad \text{VI-(14)}$$

Once the ICP process performed, the correspondence of the points between the *reference* and *target* frame for each anatomy segment is determined.

However, the various motions, including the deformation of the human skin cannot be accurately described with only rigid transformation. Thus, the same affine transformation process as in (CAO et al., 2020) is exploited after the ICP process to refine the prediction of the correspondence between the point sets.

In this proposed method, instead of bypassing the V-PCC encoding of the *target* frames, the inverse affine transformation matrices are applied to the *target* frames to get a prediction of the *reference* frame. In this way, we can encode the predicted frames using the V-PCC codec and obtain a better prediction of the *target* frames by affine transforming them back on the decoder side.

We have proposed several modifications to V-PCC so that better compression performance can be achieved. One of them concerns the modified partitioning strategy, described in the following section.

6.4.4. Modified partitioning strategy

The V-PCC codec projects the point cloud onto 2D planes so that various video coding technologies can be exploited to efficiently compress the 3D points as 2D pixels.

First, a PCA process is initialized to compute the normal attribute for each point in the point cloud. Then, the normals are refined using the neighboring information to provide smoothness among a cluster of points. A clustering process creates 3D patches containing neighboring 3D points with the same orientation. The clustering of 3D patches is iteratively refined by the smoothness of the points' orientation within those patches.

A comparison of the partitioned point cloud of the sequence *walking* is illustrated in Fig. VI-7. In the V-PCC partitioning approach, only the normal and geometry information from the current frame has been taken into account, without a global consideration of the movements of the 3D points among consecutive frames. The temporal prediction is applied on the 2D patches which are generated individually from every single frame by V-PCC. This causes poor consistency of the 2D patches in terms of both geometry and attribute values. In contrast, in the proposed method, the 3D point cloud is first clustered by the 14 anatomical segments, considering the motions that are attached to the skeleton structure. Then, the same process as in V-PCC is executed to compute the orientations of the points, to cluster the points into 3D patches. With more generated 3D patches, it is now possible to describe the prediction of the motion from each anatomical segment with an affine transformation matrix. A more temporal consistent patch can be generated by affine transforming the 3D patch in the *target* frame, and aligning it with the one in the *reference* point cloud. Similarly, the video coding benefits from the projected 2D patches with a more continuous shape.

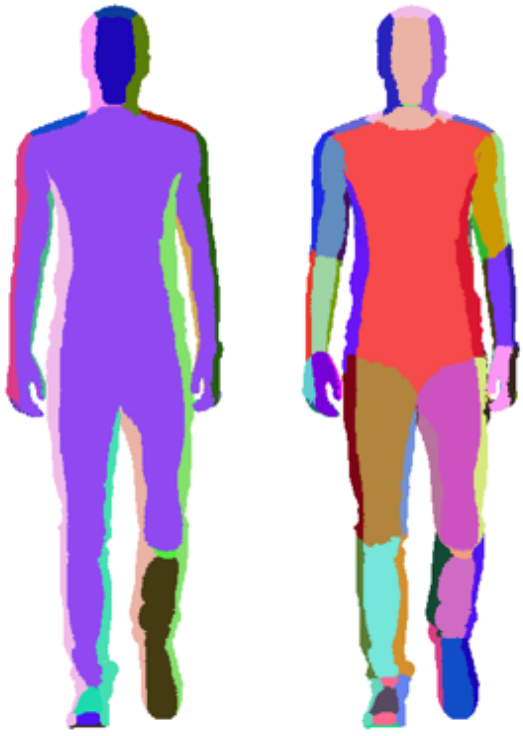


Fig. VI-7. The visualization of the segmented 3D patches using the V-PCC method (left) and the proposed method (right) for the point cloud from the sequence walking.

However, more resources are needed to code the resulting patches while the proposed method also generates more patches with a small number of points. The smoothness of the orientation for the neighboring points has been damaged by the initial clustering step which segments the point cloud into 14 anatomy parts as shown in Fig. VI-8. Thus, another local smoothing procedure is proposed to keep the continuity of the projecting orientation for the points in each anatomical segment.

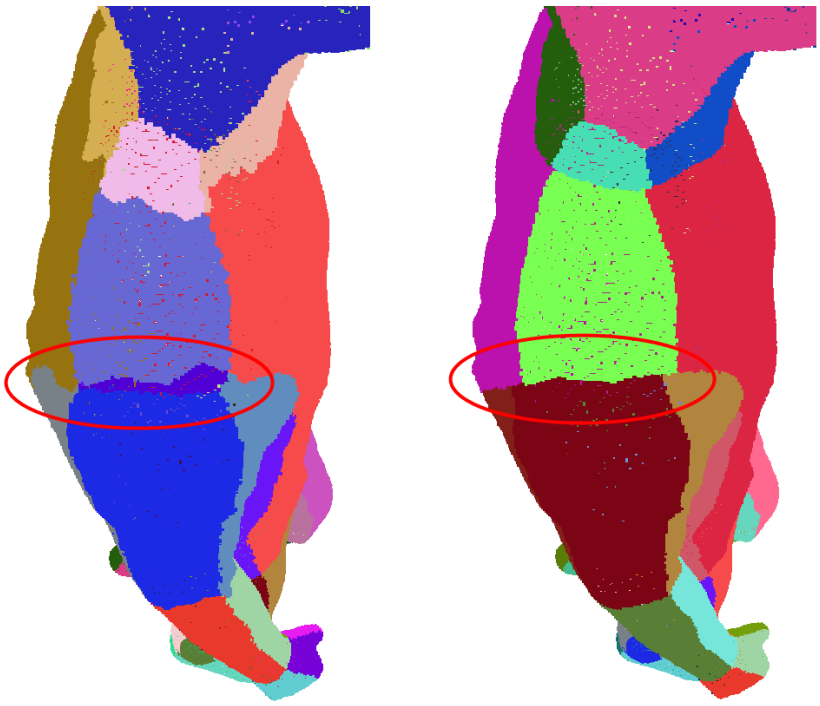


Fig. VI-8. The segmented 3D patches of walking before local smoothing (left image) and after local smoothing (right mage).

It can be observed that the 3D points in the circled purple patch have been re-clustered into neighboring patches. The resulting patches contain a smoother contour so that when they are projected onto the 2D image, fewer 2D patches with arbitrary shapes can be observed. This allows us to pack the 2D patches into a more compact and temporally consistent image and improves the compression performance of the proposed method.

A comparison of the packed 2D images for two consecutive frames using the V-PCC method and the proposed method is shown in Fig. VI-9. The 2D patches that are labeled with red and yellow bounding boxes represent the same patches in different frames.

As shown in Fig. VI-9 (a) and (c), in the V-PCC method, the patches have been reallocated to a different position during the encoding of two consecutive frames. Moreover, the yellow patch in Fig. VI-9 (a) has been split into two patches in Fig. VI-9 (c) (the positions of the patches in the previous frame are noted in dot-line rectangles). The reason is that the packing process in V-PCC gives priority to the 2D patches of a bigger size. Since only the normal information is considered during the 3D patch generation step in V-PCC, the generated 2D patches are less temporally consistent in terms of both shape and size, resulting in a varying packing order in consecutive frames.

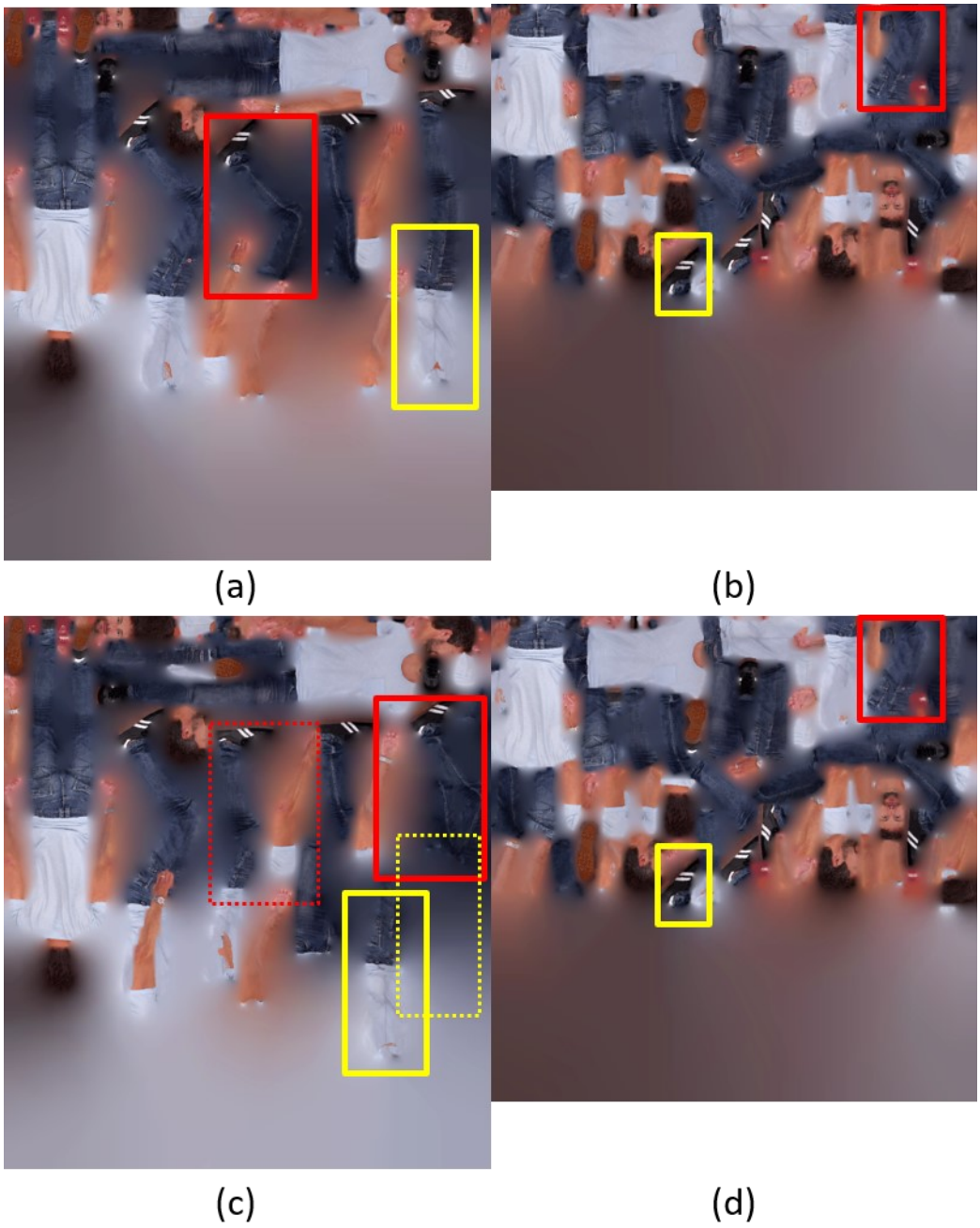


Fig. VI-9. The packed image of *walking* frames 0 and 1 ((a) and (c)) using V-PCC and the packed image of *walking* frames 0 and 1 ((b) and (d)) using the proposed method.

Compared to the patches in Fig. VI-9 (a) and (c), the ones in (b) and (d) remain relatively fixed positions in two successive frames. This allows the video codec to predict more easily the corresponding 2D patch between consecutive frames. Thus, lower values of the motion vectors and the prediction errors can be achieved resulting in a better compression performance for the video codec.

However, although the proposed segmentation method can achieve more temporal consistent packed images, it also introduces a much higher number of patches.

This leads to a significate increase in the requirement of resources for the metadata (i.e., the 2D location, the size, occupancy map of the patches, etc.). Moreover, the overall bit consumption of the proposed method can be higher than the V-PCC one. The comparison between the bit consumptions of both methods is illustrated in Fig. VI-10. These are the main reason why limited gains have been observed only in the *walking* sequence where the motion is mostly rigid as shown in the experimental results in Section 6.5.

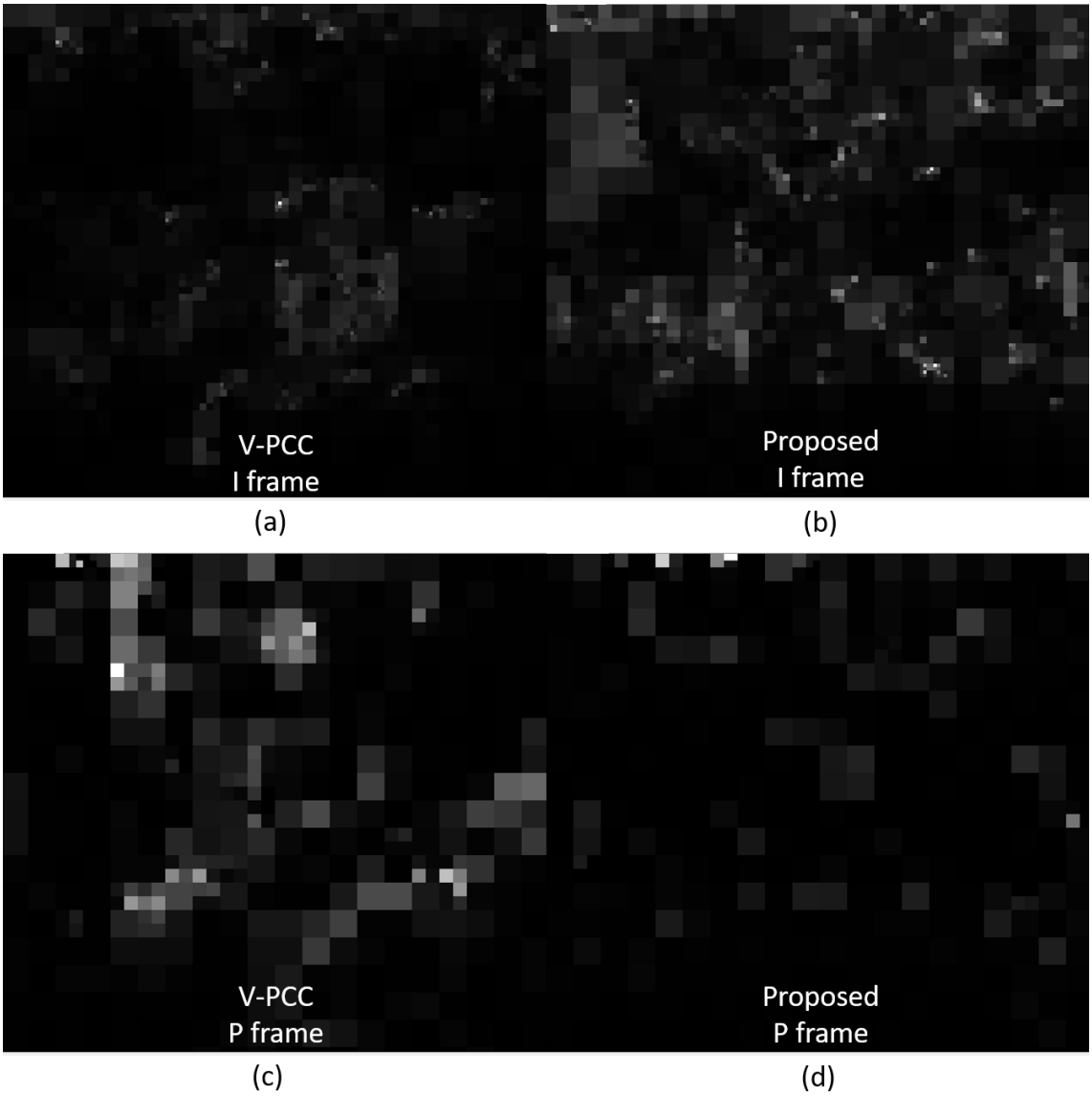


Fig. VI-10. The comparison of bit consumption between the I frames of (a) V-PCC and (b) the proposed method, and the one of bit consumption between the P frames of (c) V-PCC and (d) the proposed method.

It can be observed that more bits are consumed in the I frame in the proposed method compared to the V-PCC approach. In contrast, the P frame in the proposed method occupies fewer coding resources than the one in V-PCC. However, it has been observed in our experiment that the overall bits consumption is increasing in the proposed method compared to the V-PCC one. The increase of bits in the I frame is higher than the saved bits in the P frame. Moreover, the proposed segmentation also introduced more duplicated points when they are projected onto the 2D images.

As a result, these points are removed and lead to a decrease in the visual quality of the reconstructed point cloud frame.

6.5. Experimental results

The proposed method has been tested on both renderpeople and 8i datasets. In each tested sequence, 8 consecutive frames are used to measure the compression performance. The V-PCC version 8.0 (https://github.com/MPEGGroup/mpeg-pcc-tmc2) is used in our experiments for the integration of the proposed method. Similarly, as in the previous contribution, the comparison of the V-PCC inter-coding mode compression performances are evaluated with five conditions, corresponding to different video encoding rates. The compression performance of the proposed method is compared with the V-PCC method and the previously proposed method (cf. Chapter V) in the form of RD curves, illustrated in Fig. VI-11.

We can observe that the behavior of the proposed method is quite different among the tested sequences. The reason is that the amount of motion within each tested sequence is different and can have a great impact on the related compression performances. Our observations are summarized in Table VI-1.

Table VI-1 Characteristic of the motion for the tested point cloud sequence

Sequence name	Level of motion	Observations
walking	High	Motion observed mainly from arms, legs, and feet.
dancing	High	Complex motion observed from rapidly varying gesture
idling	Medium	Motion observed from slowly varying gesture
loot	Medium	Complex motion observed from the fingers
redandblack	High	Flying hair and flying dress observed
soldier	Low	A model gun is held by the *soldier*
longdress	High	Flying dress observed

To have a more distinct view of the amount of motion for each tested sequence, we have assigned a “High”, “Medium”, or “Low” label to each tested sequence, according to their motion amplitude. This label is subjectively determined and intends to provide a general idea about the quantity of motion that is present in each sequence.

In general, the V-PCC provides a higher PSNR value for both geometry and luminance components when coding at a higher bit rate. The previous method introduced in Chapter V encodes a single frame and uses that frame as a predictor for all the rest of the frames. By doing so, we can achieve a higher compression ratio for content with low motion amplitude without harming the visual quality.

The current proposal outperforms the previous one in both geometry and color in rate conditions for sequences with medium to high levels of motion amplitude as shown in Fig. VI-11 (a)-(h). However, the previous proposal remains more preferable for coding sequences with low motion amplitudes such as *soldier*, as shown in Fig. VI-11 (i) and (j).

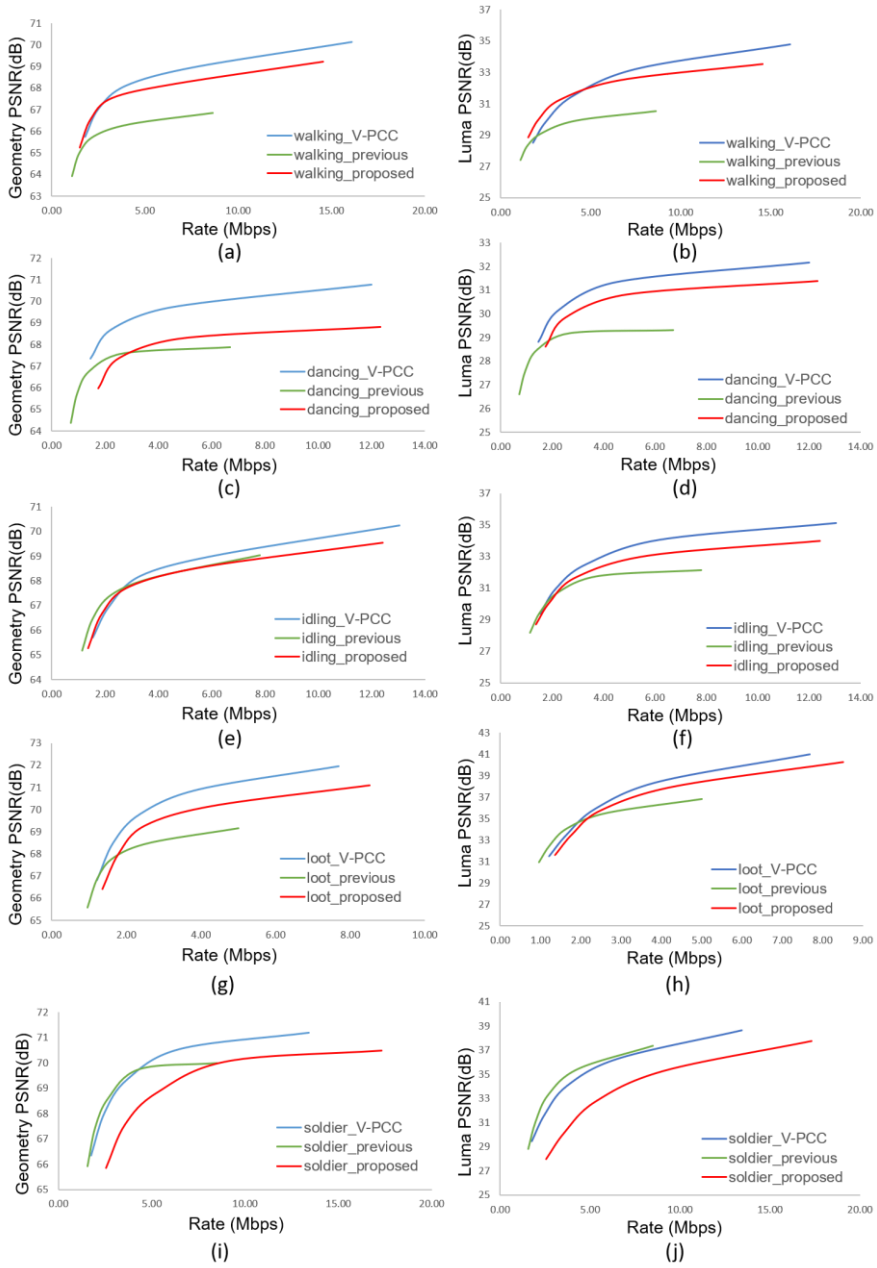


Fig. VI-11. The geometry RD curves ((a), (c), (e), (g), (i)) and the luminance RD curves ((b), (d), (f), (h), (j)) of sequence walking, dancing, idling, loot and soldier using the V-PCC method, the previously proposed method, and the current proposal, respectively.

However, the gain in terms of compression performance of the proposed method against V-PCC is quite limited. As illustrated in Fig. VI-11 (a) and (b), the proposed method achieves slightly higher PSNR values than the V-PCC method in geometry and luminance components in low rate conditions only for the sequence *walking*.

It is observed that the *walking* sequence contains more rigid motion than other sequences. Thus, the proposed method holds the potential in achieving better compression performance for dynamic point clouds with more predominant rigid motion. Table VI-2 and Fig. VI-12 show that a lower compression ratio can be achieved with comparable visual quality by the proposed method when compared to the V-PCC method in different conditions.

Table VI-2 Example values for the rate and PSNR when the V-PCC and the proposed method are used to compress walking in different rate conditions.

	V-PCC INTER R05	**Proposed R05**		**V-PCC INTER R01**	**Proposed R01**
Rate (Mbps)	16.09	14.56		1.80	1.52
D1 PSNR (dB)	70.13	68.22		65.75	65.24
Luma PSNR (dB)	34.80	33.52		28.52	28.84

It can be observed that the proposed method provides a comparable visual quality (0.51dB decrease in geometry and 0.32dB increase in luminance component) in a low rate condition (R01) with around 15% less required coding resources. Meanwhile, around 10% reduction in rate has been observed in a higher rate condition (R05) with a slight decrease in objective visual evaluation (0.91dB and 1.28dB decrease in geometry and luminance component respectively). An example of visual comparison between the reconstructed point clouds and the original uncompressed point clouds is presented in Fig. VI-12.

Fig. VI-12. Visual comparison between the original uncompressed point cloud frame 1001 of walking and the ones compressed with the V-PCC method and the proposed method in a low rate condition (R01).

It is worth pointing out that it is difficult to develop objective evaluation tools for measuring the quality of 3D content. They are designed to provide a certain level of guidance and can mismatch with the results from the subjective evaluation.

As shown in Fig. VI-12, no obvious visual difference can be observed between the V-PCC and the proposed method. In this case, the proposed method can be useful for use cases that occupy limited coding resources.

Note that it is challenging to balance the tradeoff between accurate 3D motion estimation and efficient compression. Predicting the variation of a great number of points, especially when non-rigid motion is observed, remains a highly challenging task.

Another important observation is that the tested dynamic point clouds usually contain some geometrical shifts of the points when the human remains still. An illustration of the computed point-to-point distances between two successive frames of the *soldier* sequence (*soldier_vox10_0536* and *soldier_vox10_0537*) is illustrated in Fig. VI-13.

Fig. VI-13. The point-to-point distances between point cloud frame soldier_vox10_0536 and frame soldier_vox10_0537.

Visually, no obvious motion can be observed from these two consecutive frames. However, the computed mean distance between the point clouds is around 0.15 which represents a small amount of motion. In this example, around 24% of the total points are observed as variations as shown in green color in Fig. VI-13. They are sparsely distributed over the human body which is different from a general human motion. They are caused by the individual process of capturing and reconstructing each single point cloud frame within the dynamic sequence. We have assumed that the color of each point will move along with the motion. Yet in this case, the motion cannot be described by an affine transformation. This problem is stated as “flickering geometry” and can be observed in all the tested sequences. With such noisy input, the motion estimation based on nearest neighbor search may result in finding the wrong corresponding points. This phenomenon is more obvious in the case where higher motion amplitudes are observed. The noise is entangled with the actual motion. Without a sampling algorithm specifically designed to strengthen the temporal relationship between 3D points, it is challenging to realize accurate predictions between dynamic, unstructured point clouds.

6.6. Conclusion

In this chapter, we have proposed a novel motion estimation approach for human-shape PCC. The main idea is similar to the previous contribution, introduced in Chapter V. However, the prediction structure is different in the sense that it always takes the previous point cloud frame as the predictor for motion estimation. By encoding all the motion-compensated frames, a better prediction has been achieved compared to the previous approach for content with higher motion amplitudes in higher rate conditions. We have shown that more temporal consistency of the generated 2D video is obtained compared to the ones in V-PCC. Experimental results have shown that the proposed method is more preferable to compress point cloud sequences with continuous rigid motion at low bit rate conditions in random-access mode, resulting in a higher luminance PSNR value. Nevertheless, more studies need to be done for accurate estimation of a complex motion such as the flying hair, clothes' deformation, and varying facial expressions.

With the increasing demand for high-fidelity 3D content, it is becoming important to explore the temporal compression for dynamic point clouds. The challenging part is, because of the unstructured nature of 3D points, the temporal correlations between consecutive point cloud frames are missing. On the other hand, due to the limitations of the current acquisition technology, the captured dense point clouds can be noisy in terms of the flickering of the geometrical positions and variation of the colors. This brings even more difficulties to build the correlation correctly between point clouds. To compensate for the visual quality, adding back the residuals in terms of distortion in both geometry positions and colors is necessary. In the next chapter, we will explain how the 3D residuals can be computed with a novel approach to estimate the colors, and discuss the difficulty we encountered with the support of some experimental results.

CHAPTER VII. A NOVEL COLOR COMPRESSION FOR POINT CLOUDS USING AFFINE TRANSFORMATION

In this chapter, the 3D motion compensation residuals coding is discussed. We propose a novel method to predict the colors of the latter point cloud frame from the previously encoded and reconstructed one. Instead of consuming most of the bits to encode texture video, which contains continuous and redundant color information, we propose to encode the color residuals computed from the predicted and original colors. By doing so, great gains in terms of RD performances can be achieved for all the tested sequences.

Institut Polytechnique de Paris

91120 Palaiseau, France

7.1. Introduction

In the previous two chapters, we have proposed motion estimation and compensation for dynamic point clouds using affine transformation based on the V-PCC structure. The raw residuals require high bandwidth but bring inequivalent gain in terms of visual quality. Due to the unstructured nature of 3D point clouds, the computed 3D residuals are not like traditional 2D video residuals which lay on fixed 2D grids. In the last two contributions, we assume that the attribute of each 3D point is consistent through time which means the color is not changing when the 3D point moved. However, we have shown in the last chapter there exists the “flickering geometry” problem. In this chapter, we also show that there also exist some variations in the colors of points located in the same geometrical location between consecutive frames. This problem can be stated as a “flickering color” problem. Meanwhile, the actual human motion in the form of variation between dynamic point clouds cannot be accurately described using only affine-transformation-based prediction. Thus, in this chapter, we propose to encode the geometry of the point clouds as in V-PCC and use several prediction methods for optimizing the color prediction. In this case, instead of transmitting all the color images, the 3D residuals are mapped and transmitted as 2D V-PCC videos. Only the first color image is reserved as a predictor for the latter color prediction. By encoding solely a small portion of residuals, corresponding to the most important values, with the help of some statistical filtering techniques, the visual quality greatly increases with a smaller cost than the V-PCC solution. Thus, higher compression performances have been achieved for all the tested sequences.

7.2. The residuals from both geometry and color

The “flickering geometry” phenomenon is illustrated in Fig. VII-1. Here, some geometrical variations (including the appearances of new points) of the points can be observed near the marginal zone of the feet, even if there is no apparent motion observed. Computing affine transformation on such two point clouds results in applying almost no transformation. This is why the RD performances obtained by V-PCC, which does not involve any 3D prediction stage, are always better than those obtained with the 3D affine transformation- prediction approach.

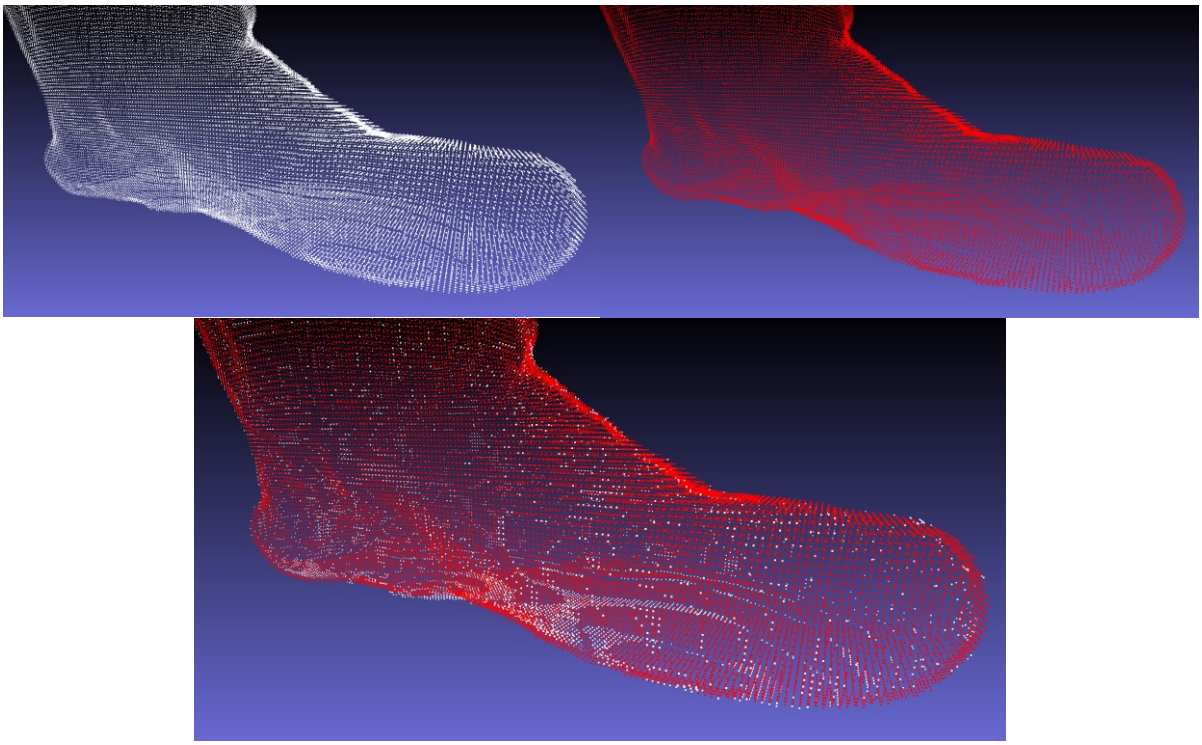
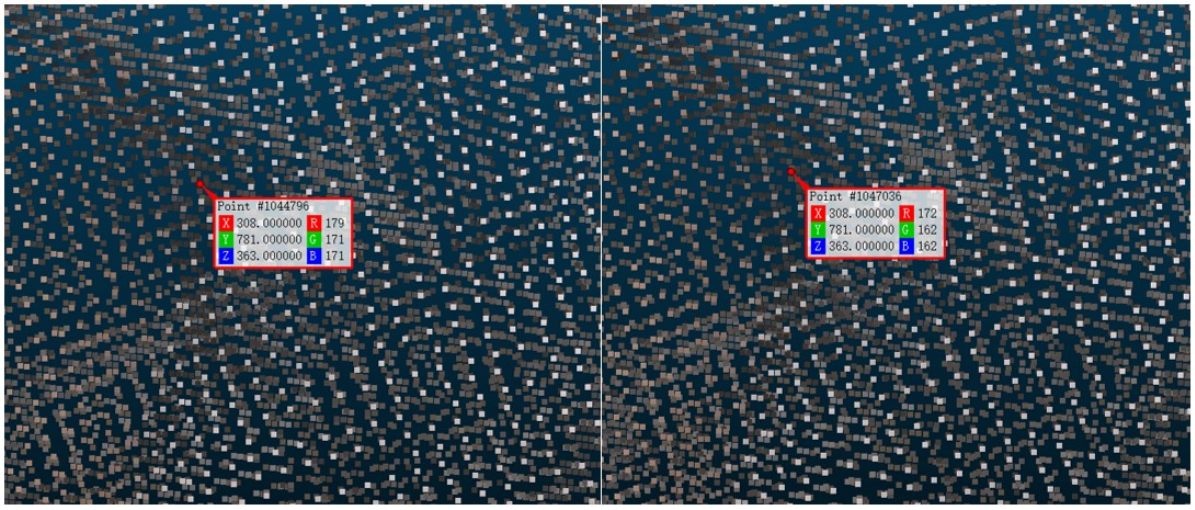


Fig. VII-1. The left foot point cloud of soldier frame 0536 (top left), frame 0537 (top right), and both point clouds overlapped (bottom).

Moreover, we have also observed that the flickering geometry has an impact on the colors of the 3D points as well, as illustrated in Fig. VII-2. Here, no motion from the point clouds representing successive frames is observed. However, the associated color exhibit significant variation. Such variations are coming either from the actual human motion or are introduced by the 3D reconstruction algorithm.

(a)



(b) (c)

Fig. VII-2. A single 3D point (a) locating on the arm of soldier with the same geometry but different color information in frame 0536 (b) and frame 0537 (c).

Due to the existence of both geometry and color flickering problems, two kinds of residuals of the predicted 3D points can be observed: a geometrical location shift composed of $(\Delta x, \Delta y, \Delta z)$ coordinates and a color shift $(\Delta r, \Delta g, \Delta b)$ in RGB color space. In our motion compensation predictive framework, the 3D points are affine-transformed with their associated color information. Thus, the residuals can be computed from the original 3D points and the affine transformed 3D points as described in equations (VII-(1)) and (VII-(2)).

$$\begin{pmatrix} x_{affine}^{ref} \\ y_{affine}^{ref} \\ z_{affine}^{ref} \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x^{ref} \\ y^{ref} \\ z^{ref} \\ 1 \end{pmatrix} \quad \text{VII-(1)}$$

$$\begin{aligned} \Delta x &= x^{target} - x_{affine}^{ref} \\ \Delta y &= y^{target} - y_{affine}^{ref} \\ \Delta z &= z^{target} - z_{affine}^{ref} \\ \Delta r &= r^{target} - r^{ref} \\ \Delta g &= g^{target} - g^{ref} \\ \Delta b &= b^{target} - b^{ref} \end{aligned} \quad \text{VII-(2)}$$

where $P^{ref} = (x^{ref}, y^{ref}, z^{ref})$ is a given point in the reference frame, $(r^{ref}, g^{ref}, b^{ref})$ its corresponding color values in the RGB color space, $P_{affine}^{ref} = (x_{affine}^{ref}, y_{affine}^{ref}, z_{affine}^{ref})$ the transformed point under the current affine transformation and $P^{target} = (x^{target}, y^{target}, z^{target})$ the corresponding point in the target frame, described by the $(r^{target}, g^{target}, b^{target})$ color information. The target point P^{target} is defined as the closest point (with respect to the Euclidian distance) to the affine-transformed one P_{affine}^{ref} in the target frame.

An example of the resulting geometry residuals is illustrated in Fig. VII-3, which provides a comparison with the V-PCC decoding errors. In the case of V-PCC, no 3D prediction has been performed. Here, the residuals are computed between initial and decoded geometries for each frame. In other words, the P_{affine}^{ref} points in equation (VII:7.2-(1)) are replaced by the decoded points in the current frame. In the case of V-PCC, the geometry errors are sparsely distributed. The method proposed in the previous chapter is used here to demonstrate the impact of affine compensation residuals. It can be observed that by affine transforming the segmented point clouds as body parts, numerous and relatively important geometry errors appear.

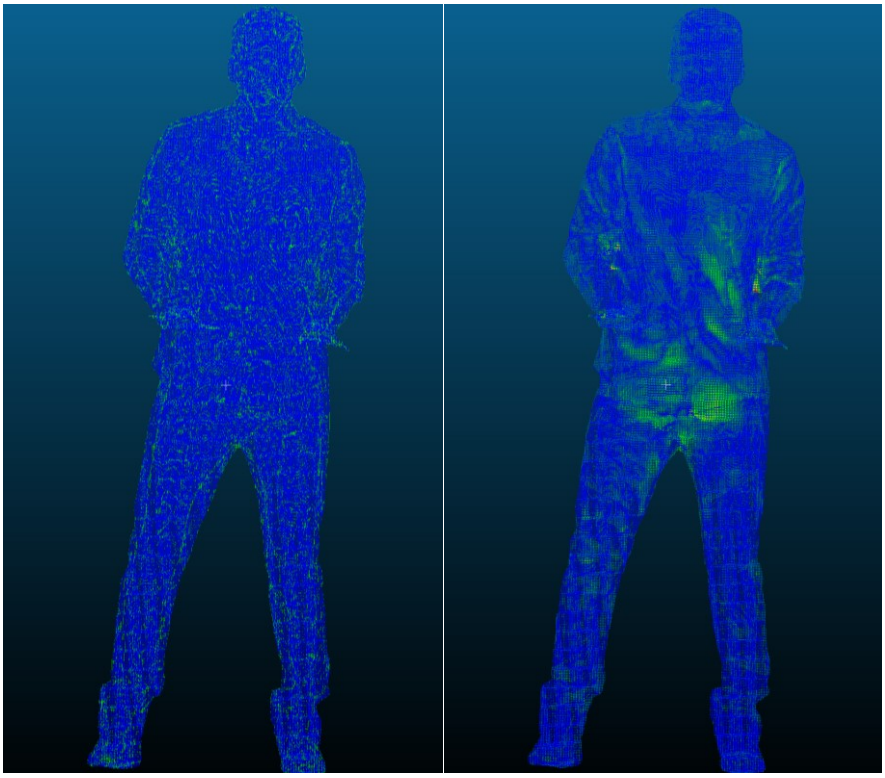


Fig. VII-3. The distribution of geometry prediction errors for (left) the V-PCC reconstructed point cloud (loot frame 1001), and (right) the one from the proposed method in Chapter VI.

In Fig. VII-3, the values of the distortions are represented in ascending order from blue, green, yellow, and up to red. The blue colors represent the lower distortions. The color from the right image in Fig. VII-3 illustrates the deformation of the point clouds that cannot be described accurately enough by the affine transformation. It is worth noting that there are some red colors near the elbow area, which stands for a much higher prediction error. This is a combination of human motion and acquisition distortion. Since the 3D point cloud is acquired using a SfM algorithm, there is no guarantee in the temporal consistency of the 3D points in terms of the number of points and color values carried by the points located in the same 3D position in different instants.

Although some of the motion can be accurately described by a single affine transformation, the overall distortion in terms of point-to-point distances is inferior to the one provided by the V-PCC method as summarized in Table VII-1.

Table VII-1 Comparison of the mean point-to-point distance and variation of the distortion between V-PCC and the proposed method.

Method	Mean point-to-point distance	The standard deviation of the distances
V-PCC	0.19	0.39
Proposed	0.56	0.68

Thus, integrating residuals on top of the affine-transformation-based prediction model is mandatory if we wish to obtain the same level of visual quality as the one proposed by the V-PCC solution. However, directly dealing with 3D residuals associated with unorganized point clouds is a highly difficult issue.

On account of the observations above and the fact that color attributes occupy a greater percentage of bit consumption (Fig. III-6), we have decided to use the same solution for coding geometry as in V-PCC and focus on the optimization of color prediction with residual coding.

7.3. The prediction structure

In contrast with the previously proposed approaches (*cf.* Chapter V and Chapter VI), the affine transformation is no longer applied to the geometry of the 3D points. Instead, the V-PCC geometry coding solution is used, so that the reconstructed geometry is identical to the one obtained from the V-PCC method. The affine transformation is applied to the geometry to determine the correspondence between the 3D points in successive point cloud frames. Once the correspondence is found, a color mapping procedure is determined, to predict the color from the previously encoded and the current, reconstructed point cloud frame. The main idea behind this process is to save the bits consumed by encoding the redundant color video, which can take up to around 80% of the total bandwidth, and use the saved bits to encode the residuals.

The proposed color prediction scheme is illustrated in Fig. VII-4. First, the original point cloud frames are compressed with a V-PCC encoder. Note that only the color from the first point cloud within each group of frames is compressed in the proposed scheme. Meanwhile, all the geometry information is encoded and used for motion estimation and compensation. The colors of the rest of the point cloud frames in the group are predicted from the reconstructed and motion compensated point clouds. Note that in the following examples, if not specified, the prediction process using frames 1000 to 1007 of *loot,* encoded in rate R05.

Let us denote by $PC0$ and $PC1$ the two successive point cloud frames to be compressed. They are first encoded by a V-PCC encoder and reconstructed as $PC0_{rec}$ (noted as Reconstructed point cloud#1 in Fig. VII-4) and $PC1_{rec}$ (one of the frames in Geometry_rec#all). The $PC0_{rec}$ contains both the geometry (Geomtry_rec#1) and color information (Attribute_rec#1). The rest of the reconstructed frames in the GOF contain only geometry information and are used for motion estimation and compensation. The affine transformation matrices between the body parts in consecutive point cloud frames are computed with the help of pre-clustered point cloud segments. The motion estimation and compensation method follows the same strategy as the one described in Chapter V and Chapter VI.

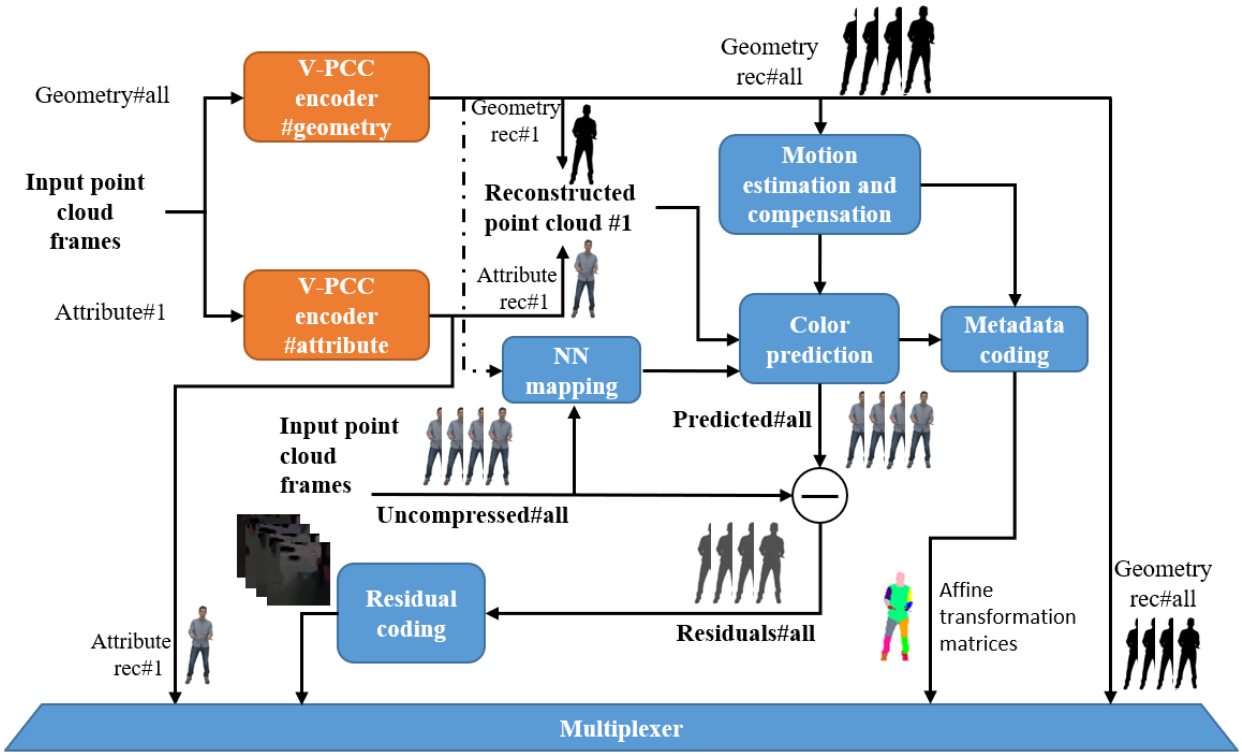


Fig. VII-4. The proposed color prediction scheme using affine-transformation-based motion estimation and compensation.

After the motion estimation and compensation procedure, a mapping between the points from $PC0_{rec}$ and $PC1_{rec}$ can be achieved. With the resulting correlation between the points in two consecutive point cloud frames, the colors of $PC1_{rec}$ can be predicted from the motion-compensated $PC0_{rec}$.

We propose to use 4 different modes for predicting the color to minimize the values of color residuals, including nearest neighbor search (NN), nearest neighbor search after affine transformation (NN_AT), averaged nearest neighbors (NN_avg), and averaged nearest neighbors after affine transformation (NN_AT_avg). Since the affine transformation is applied on each body part, it is likely that with this global transformation, some of the points can be estimated by the affine transformation while some can not.

The NN mode simply searches for the nearest neighbor of the query point in the reference point cloud and maps the color of the resulting point onto the query point. Note that the NN search takes advantage of the KD tree algorithm to speed up the process. This mode predicts better the points that have not moved.

The NN_AT mode is similar to NN but an affine transformation matrix is computed before the mapping process to get a possible better prediction. The computation of the affine transformation was explained in Chapter V-3.1. The query point is affine-transformed and then used for neighbor searching and color mapping. This mode predicts better the points that have anatomical motion.

The idea for NN_avg and NN_AT_avg modes are similar to the NN and NN_AT modes, with the difference that here the color is averaged among 10 nearest neighbor candidates after the color mapping. The value 10 is chosen here as a tradeoff between computational complexity and prediction accuracy. In this way, a better prediction with lower residual values can be achieved and further improve the compression.

To optimize the color prediction, we select the prediction mode with the lowest prediction error. The prediction error can be computed from the predicted color and the original uncompressed color using the mean square error (MSE), defined as:

$$MSE = \frac{1}{3}((R - R')^2 + (G - G')^2 + (B - B')^2) \quad \text{VII-(3)}$$

Note that (R, G, B) is the color from the original, uncompressed point cloud while (R', G', B') is the color from the predicted $PC1_{rec}$. The prediction mode with the lowest prediction error is then selected. Examples of the percentage of the selected prediction modes are illustrated in Table VII-2 and Table VII-3.

Table VII-2 The percentage of the selected prediction mode for points predicted in loot F1005

Prediction mode	**Percentage of each used prediction mode (among 770072 points)**
NN	44%
NN_AT	32%
NN_avg	20%
NN_AT_avg	4%

Table VII-3 The percentage of the selected prediction mode for points predicted in walking F1001

Prediction mode	**Percentage of each used prediction mode (among 614851 points)**
NN	18%
NN_AT	48%
NN_avg	28%
NN_AT_avg	6%

In these examples, the *walking* sequence contains more motion than *loot*. Consequently, we can observe that the affine-transformation-based prediction is preferable for content with higher motion amplitudes. In contrast, the nearest-neighbor-based prediction is more compatible for content with lower motion amplitudes.

An example of the resulting color residuals computed from the predicted frame 1001 and the V-PCC reconstructed frame 1001 of *loot* is illustrated in Fig. VII-5.

Fig. VII-5. The color residuals from loot frame 1001 (left) and redandblack frame 1452 (right).

For each sequence, the minimum values $(r_{min}, g_{min}, b_{min})$ within the color residuals are detected. Then they are compensated for all the points for easy visualization of residuals. The residuals show indications of where the motion happened for each sequence.

The number of bits consumed by V-PCC and the ones saved by skipping the encoding of partial attribute frames is illustrated in Table VII-4.

Table VII-4 The bits consumed from bypassing texture coding for partial frames in a GOF

	Geometry (Bytes)	**Texture (Bytes)**	**Metadata (Bytes)**	**Total (Bytes)**
V-PCC	93152	131217	27634	252003
Proposed	93152	54368	27634	175154

The texture bandwidth corresponding to the proposed method corresponds exclusively to the texture associated to the first frame of the sequence. It can be seen that there are 76849 bytes saved that can be used to encode the affine transformation matrices and the color residuals. In this example, the indexes and the resulting affine transformation matrices occupy 11561 bytes (which are not included in the Metadata field reported in Table VII-4).

Thus, we can use the remaining 65288 bytes to encode the color residuals, assuming that we can achieve similar or higher visual quality after adding the residuals back onto the predicted color of the reconstructed point cloud.

In this proposal, since the geometry information is coded using the V-PCC method, the residuals are composed of color information only. This makes it easier to compress the resulting residuals. The residual of each point can be computed from the distance (in the RGB color space) between the uncompressed color and the predicted color. Moreover, since the geometry information remains the same, it is possible to project the color residuals onto the same 2D locations so that the video codec can be again exploited.

7.4. Compression of the color residuals

The color residuals are expressed in the form of 3D vectors, which are computed from the V-PCC reconstructed colors and the predicted colors. They can be encoded by various entropy coding methods. Most of the time, the residuals are preferably compressed in a lossless manner. Many techniques such as Huffman/arithmetic coding or JPEG/MPEG solutions can be used to compress the residuals efficiently. In our proposal, both lossless methods such as Lempel–Ziv–Markov chain algorithm (LZMA) and Deflate, and lossy methods such as JPEG and HEVC video-based methods are considered. However, due to the acquisition and reconstruction noise from the input point clouds (geometry and color flickering), a pre-processing step is needed to reduce the huge amount of residuals.

7.4.1. Statistical filtering of the residuals

Although we have used 4 different modes to optimize the color prediction, the resulting residuals are still highly bandwidth consuming. For example, when we predict the color of frame 1001 from sequence *loot*, around 15% of the total points contain differences in color against the V-PCC reconstructed frame 1001. Their size is around 6 MBytes, which is almost 70 times the size of the encoded geometry bitstream. To reduce the number of residuals, we decided to code only the residuals with great amplitudes (with respect to the standard deviation value). In this way, only a relatively small number of residuals with bigger variations in color need to be transmitted. An example of the distribution for the resulting color residuals is illustrated in Fig. VII-6, corresponding to the prediction of *loot* frame 1005 from frame 1004.

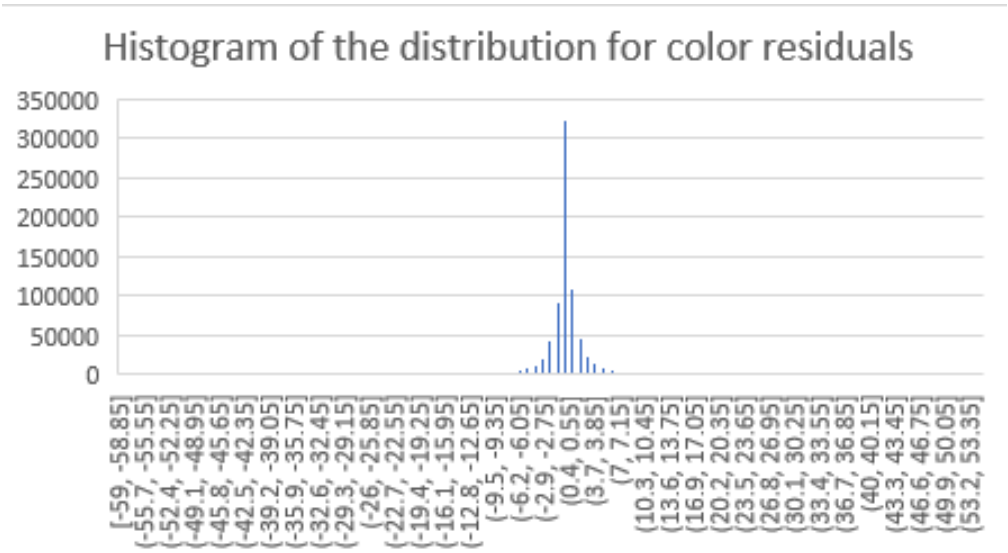


Fig. VII-6. The histogram of the distribution for color residuals.

The distribution of the resulting color residuals tends toward a normal distribution. Here, the confidence interval is set to $[-2\sigma, 2\sigma]$ (σ is the standard deviation value computed for each point). In this way, only around 5% of the residuals are transmitted. Meanwhile, additional bits are needed to indicate the indexes of the preserved delta colors.

7.4.2. Encoding with lossless compression

To compress the preserved residuals, we first considered the tool 7-zip[1] which is a file archiver with a high compression ratio. It is open-source software that supports compression in 7z format with LZMA, which also supports compression in ZIP and GZIP[2] formats with Deflate.

LZMA uses a dictionary compression algorithm (a variant of LZ77 (Ziv & Lempel, 1977) with huge dictionary sizes and special support for repeatedly used match distances), whose output is then encoded with a range encoder, using a complex model to make a probability prediction of each bit. The dictionary compressor finds matches using sophisticated dictionary data structures and produces a stream of literal symbols and phrase references, which is encoded one bit at a time by the range encoder. Various encodings are possible, and a dynamic programming algorithm is used to select an optimal one under certain approximations. Meanwhile, Deflate uses a combination of Lempel–Ziv–Storer–Szymanski (LZSS) (A. & G., 1982) and Huffman coding.

According to a compression benchmark[3], the 7z method offers optimal compression performances. This claim is confirmed by the results that we have obtained, which are summarized in Table VII-5.

[1] https://www.7-zip.org/

[2] https://www.gzip.org/

[3] https://peazip.github.io/peazip-compression-benchmark.html

Thus, the residuals are compressed using 7-zip since the compression ratio is our main objective. Moreover, modern algorithms as Google's Brotli[4] and Facebook's Zstandard[5] are becoming viable alternatives providing comparable compression ratios at higher speeds. However, they are not discussed here in the consideration of the complexity of integration.

Table VII-5 Comparison of the compression methods for the residuals of frame 1001 loot in terms of compression ratio.

Used archive method	Original file size (bytes)	Compressed file size (bytes)	Compression ratio
7z format with LZMA algorithm	780,590	145660	5.36:1
Gzip format with Deflate algorithm	780,590	168605	4.63:1

We can observe that the size of the residuals bitstream remains relatively large after lossless compression. Thus, the residuals computed from a single prediction between two frames take 1.5 times the space of the encoded entire geometry information (8 frames in total in the tested experiment) and exceed the number of bits saved from bypassing the texture coding. For this reason, in the following experiments, we focus on lossy color residuals compression, based on image and video coding technologies.

7.4.3. Encoding with lossy compression

7.4.3.1. Image-based encoder

We kept the geometry coding method as in V-PCC and encode the geometry information of all frames but only encode the texture of the first frame in the GOF. In this way, the encoded geometry information can be used to reconstruct a single point cloud frame with only geometry information and use that frame for building the correspondence between the uncompressed point cloud and the reconstructed one. This correspondence is used to map the uncompressed color onto the reconstructed geometry point cloud frame. This frame is then set as a ground truth frame for the later computation of color residuals. After that, the 3D color residuals are projected onto 2D images with the same mechanism as their corresponding 3D geometry as in V-PCC.

[4] https://github.com/google/brotli

[5] https://github.com/facebook/zstd

An image-based compression method is first retained. We have used a compact JPEG encoder, called TooJpeg[6], as an alternative to the complete implementation of libjpeg[7] or libjpeg-turbo[8]. The main principle is to use still image compression to achieve near-lossless quality with a better compression ratio.

First, a raw RGB image is saved as the uncompressed reference image. Then, different quality profiles are selected with the same color convention step from RGB to YUV422 as in the V-PCC method. The compressed file sizes and corresponding PSNR values are summarized in Table VII-6.

Table VII-6 The file size and PSNR for the compressed residual image frame 1001 of loot.

Quality (quality parameter defined in the TooJpeg where 100 represents the best quality)	**File size (bytes)**	**PSNR (dB) (against the uncompressed residual image)**
Uncompressed (100)	249856	infinite
90	90112	47.83
70	57344	42.73
50	45056	41.96
30	36864	40.68
0	24576	31.23

The PSNR values are computed using the SOFTPEDIA-PSNR[9] tool. This is the number of bits to be consumed for every single computed residual image. It is shown that even if the residual image is encoded with the lowest compression quality (i.e., 0 as shown in Table VII-6), the total number of bits required for compressing all the residual images will exceed the one we saved. Moreover, the residual information is lost during compression (Fig. VII-7).

[6] https://github.com/stbrumme/toojpeg

[7] https://github.com/thorfdbg/libjpeg

[8] https://github.com/libjpeg-turbo/libjpeg-turbo

[9] https://www.softpedia.com/get/Multimedia/Graphic/Graphic-Others/PSNR.shtml

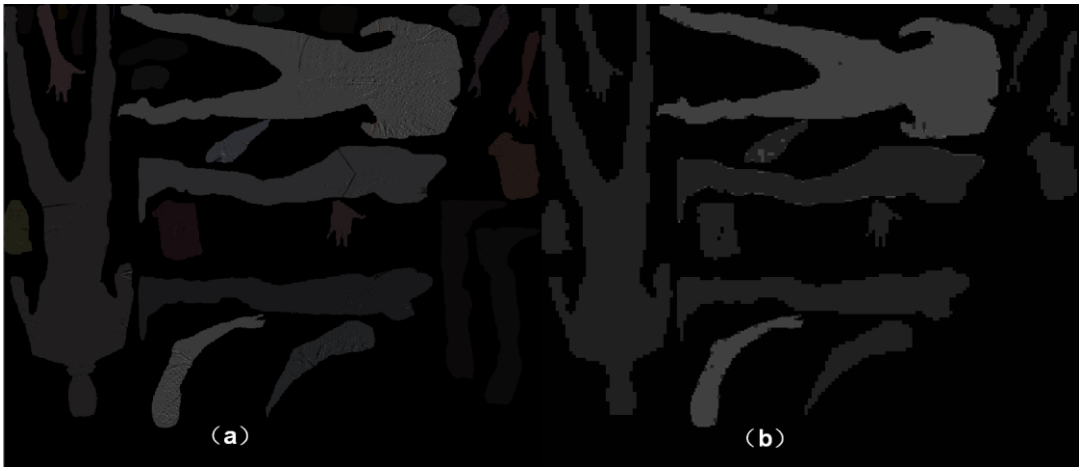


Fig. VII-7. (a) The uncompressed residual image and (b) the compressed residual image with a compression quality 0.

7.4.3.2. Video-based encoder

To achieve a higher compression ratio, we have decided to take advantage of the HEVC video codec, already used in the V-PCC test model. Instead of encoding the projected point cloud with the original color, we project the color residuals and HEVC-encode the projected color residuals as a video sequence. Since the computed delta color values can be negative, we add an offset per patch as normalization. The resulting projected residual image is illustrated in Fig. VII-8.

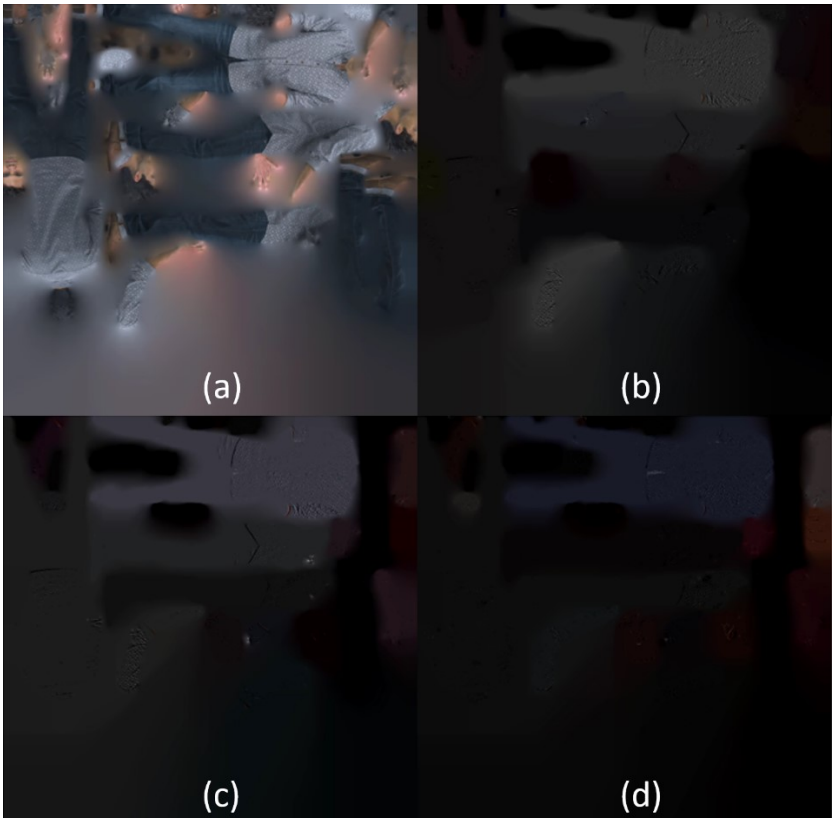


Fig. VII-8. (a) The projected point cloud of frame 1000 with original color and the projected residual image with computed delta color values of (b) frame 1001, (c) 1002, and (d) 1003.

In our proposal, the first frame within the GOF is encoded by V-PCC independently, the resulting residual images being encoded by a different V-PCC process with the same encoding parameters.

7.5. Experimental results

We have tested the proposed color prediction method on the four *8i* sequences (*longdress, loot, redandblack,* and *soldier*). For each sequence, the first 8 frames are selected as test materials to reduce the overall duration of the experiments. The experimental results are compared to the ones from V-PCC version 8.0 in INTER-lossy coding condition from rate R01 to R05 where R05 represents the highest coding rate.

We first experimented with the color prediction in a per-point precision, where each 3D point to be predicted is assigned with a mode flag to indicate the used mode for color prediction. In this way, the number of required resources to encode the flags in a lossless manner is huge as shown in Fig. VII-9.

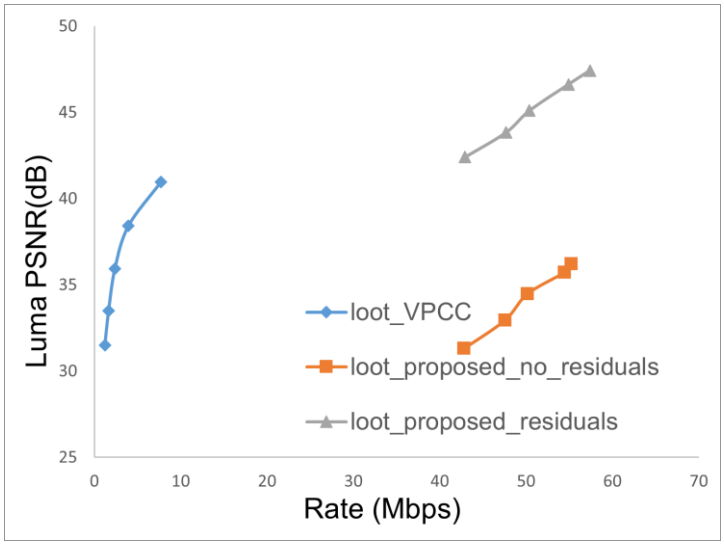


Fig. VII-9. The RD curves for the luminance component when coding loot with the V-PCC and the proposed method using a per-point flag of color prediction.

The 7zip compressed file of the flags alone can occupy up to 20 times of the bits consumed by encoding the entire point cloud sequence.

Thus, a per-patch flag is designed to reduce the number of flags (e.g., 778139 points in *loot frame 1001*) to a more compression-friendly level (e.g., 31 patches in *loot frame 1001*). This operation lowers the accuracy of color prediction and increases the residual values. However, for different sequences, the per-patch color prediction scheme lower the prediction accuracy (measured by the MSE values from the prediction and ground truth) ranging from 10-25% with the cost of an significant increase of the resources to encode the per-point prediction flags (as shown in Fig. VII-9). Thus, we decided to use the per-patch prediction mode to optimize the compression performance. By doing so, the resulting RD performances have been improved greatly with the help of efficient video compression as shown in Fig. VII-10.

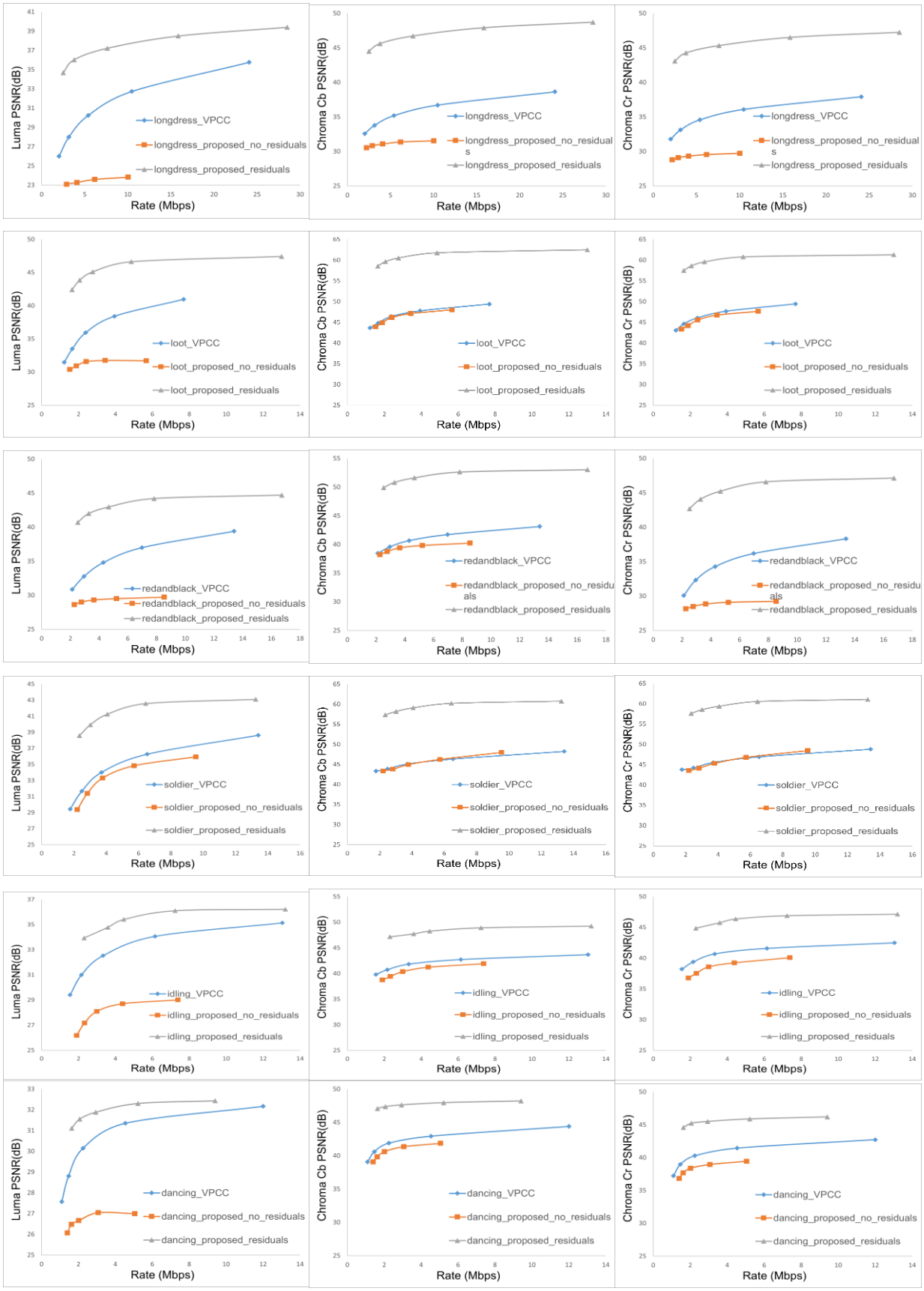


Fig. VII-10. The RD curves of the four 8i sequences encoded by the V-PCC method (blue), the proposed method without residuals (orange), and the proposed method with residuals (grey).

It is shown in Fig. VII-10 that all the tested sequences have significant gains in all Y, Cb, Cr channels. The grey lines demonstrate the RD performances of the reconstructed point cloud compensated with residuals. Since the *soldier* and *loot* sequence contain low-motion amplitudes, the resulting RD curves without residual compensation are already close to the V-PCC ones but with a lower rate when encoded in higher rate conditions. This proves again the huge potential of 3D prediction and the redundancies within the dynamic point clouds.

The composition of the compressed bitstream is shown in Table VII-7, and the values are averaged between the tested five rates for demonstration. It can be observed that additionally from the Geometry, metadata and Texture information in the V-PCC, the Skeleton information, Affine transformation matrices for motion estimation and the computed color residuals are embedded into the compressed bitstream. Moreover, the Texture information is composed of the original first frame encoded by the V-PCC and the mapped color residuals of the rest of the frames after prediction in the proposed method.

Table VII-7 Composition of the compressed bitstream of the proposed method.

Sequence	Geometry	metadata	Texture	Skeleton information	Affine transformation matrices	Texture Residuals(%of the total Texture)
dancing	24.07%	13.26%	45.00%	12.46%	5.22%	30.8% (69%)
idling	28.79%	12.06%	47.57%	8.27%	3.31%	33.6% (70.6%)
longdress	21.78%	8.25%	59.68%	7.65%	2.63%	41.2% (63.8%)
loot	34.78%	14.03%	36.66%	9.70%	4.83%	22.8% (55%)
redandblack	33.92%	15.77%	39.11%	7.93%	3.26%	25.8% (62.4%)
soldier	34.39%	17.05%	34.96%	10.39%	3.19%	11.4% (30.8%)

It is also shown in Fig. VII-11 that it is more beneficial to transmit the residuals which may bring more gains in visual quality than coding the texture video with a similar level of visual quality, especially in the conditions of low bitrate encoding.

Fig. VII-11. The visualization of (left) the uncompressed frame 1453 from redandblack, (middle) the V-PCC reconstructed frame in R02, and (right) the reconstructed frame using the proposed method in R01 condition.

The *redandblack* frame 1453 shown in Fig. VII-11 (a) is encoded by V-PCC at a rate of 2.95 Mbps resulting in a PSNR value of 32.8 dB in the luminance component while the one in Fig. VII-11 (b) is encoded by the proposed method at a rate of 2.51 Mbps resulting in a PSNR value of 40.7 dB in the luminance component. Here again, the proposed method provides much better visual quality at a similar rate for dynamic point clouds.

7.6. Conclusion

In this chapter, we proposed a solution to predict the color of the point cloud from previously V-PCC encoded and reconstructed point cloud using affine transformation. Different methods are used to compress the resulting color residuals in both the lossy and lossless manners. The experimental results obtained show that significant gains can be achieved by transmitting residuals instead of the original texture video in the V-PCC method, for all the tested sequences. Moreover, we show that solely 5% of the entire color residuals need to be encoded and transmitted. Let us note that it is possible to optimize the rate-distortion performance, which is part of our planned future work.

However, we need to highlight that several problems still need to be solved:

1) The dynamic point clouds contain acquisition noise which makes it hard to achieve accurate prediction. The acquisition noise can come from the photogrammetry-based reconstruction method where the 3D geometry location of a point is computed from a pair of images taken from multiple views. The noise can either come from the mismatching of the computed geometry information or the mapping error from the color of 2D images.

2) The complexity of our proposed method is quite high: 3 to 5 times slower for encoding and around 1.5 to 3 times slower for decoding, with respect to the standard V-PCC approach. This additional complexity is explained by the need to construct several KD trees for nearest neighbor search and compute affine transformations for different segments iteratively.

In our future studies, better acquisition technologies or preprocessing methods are needed to create dynamic point clouds with more temporal consistency in terms of geometrical locations and colors among the sampled points from consecutive frames. It is possible to down-sample (voxelize) the point cloud as in (Dorina Thanou et al., 2016) and achieve better predictions among sequential frames. However, this procedure lowers the resolution (density of the 3D points and corresponding colors) of the point clouds, which are devoted to providing high-fidelity immersive experiences. Moreover, DL methods can be considered in order to reduce the complexity of temporal prediction between consecutive frames.

CHAPTER VIII. CONCLUSION AND FUTURE WORK

Institut Polytechnique de Paris

91120 Palaiseau, France

8.1. Conclusion

In this thesis, we have addressed a highly challenging research topic, which concerns the compression of 3D point clouds. With the recent hardware and software advances of capturing systems, sparse and dense dynamic point clouds can be acquired with various features to satisfy the requirements of different use cases and applications. For example, the sparse point clouds with high-precision can provide accurate navigation for autonomous driving, the dense point clouds can offer an immersive experience with high-fidelity content.

However, with the increasing demand for point clouds with higher precision and density, compression becomes an essential, and highly challenging issue that conditions to a significant extent the deployment of the corresponding applications. The main difficulty when addressing this issue concerns the completely unstructured nature of the 3D point clouds, which lack entirely the topological information.

We have first proposed a comprehensive survey of the state-of-the-art 3D PCC methods (Chapter II). The highly diverse techniques proposed, exploit either 3D representation (octrees, KD-trees...), 1D predictive methods, clustering-based techniques, 3D/2D projection-based approaches, or DL-based methods. The various state-of-the-art methods have been extensively discussed and analyzed, with principles, advantages, limitations, and related compression performances. Our analysis highlights the complexity and difficulty of dealing with such huge amounts of unstructured 3D information that includes both geometry and photometric features.

Meanwhile, the MPEG consortium also identified this trend and, after extensive experimentations carried out over several years, issued two standards for representing compressed point clouds, based on the two most efficient approaches: 1) a 3D-to-2D projection-based one followed by video coding and 2) a purely geometric octree-based one. They are denoted by V-PCC and G-PCC, respectively.

Our research work has been entirely carried out within the framework of the MPEG 3D PCC development process and is specifically concerned with the optimization of the V-PCC approach. In Chapter III, we have proposed a detailed description of the current V-PCC and G-PCC MPEG technologies, along with an experimental, objective evaluation of the related compression performances. Our analysis, which highlights the current limitations, showed that a lot of room for improvement can be envisaged for the V-PCC approach, notably at the level of the inter-coding compression mode. For this reason, the rest of our work and our contributions have focused on the optimization of the V-PCC approach for compressing dynamic point clouds.

A first contribution has been introduced in Chapter IV. It concerns an RDO segmentation approach, jointly exploited with a by-part affine modeling process dedicated to motion compensation purposes. The experimental results obtained have shown promising performances, notable at low at very-low bitrate conditions. However, the integration of geometrical residuals remains a challenge.

In the following chapters, we notably attacked the optimization of the affine motion estimation and compensation process. By limiting ourselves to the uses case of humanoid-like 3D PCC compression, we have proposed a skeleton-based, motion-driven segmentation approach that can be exploited for motion compensation purposes. The experimental results obtained show improvement in the case of low motion sequences. However, obtaining better results requires the consideration of more advanced estimation techniques.

This issue has been notably addressed in Chapter VI, where an improved affine motion estimation approach is introduced. Here, a two-step estimation approach has been proposed. First, a rigid transform is estimated using an ICP algorithm. Then, a complete affine model is estimated, benefiting from this first estimation. Experimental results have shown that the proposed method is more preferable to compress point cloud sequences with continuous rigid motion at low bit rate conditions in random-access mode, resulting in a higher luminance PSNR value. Nevertheless, more studies need to be performed for accurate estimation of complex motion such as flying hair, clothes deformations, and varying facial expressions.

Finally, Chapter VII proposes a method that can make it possible to consider and exploit efficiently the residual errors, obtained after motion compensation, corresponding to the photometric attributes. Various prediction modes are also investigated and integrated within the proposed approach. The results obtained have shown significant improvement when compared with the baseline V-PCC approach.

It is expected that the compression of 3D point clouds will be evolving with the advance of acquisition technologies and computer science. It is exciting to witness the entry of a new era where we can finally digitize the world into 3D representation and learn more about it.

8.2. Future work

We have concentrated our work on dynamic point clouds with human shapes. However, there are various use cases where the point cloud contains different characteristics.

Our future work is expected to first extend the study to cover more types of the point cloud. G-PCC is a good starting point to study the LiDAR-captured point clouds. It is expected that the LiDAR sensors will be quite common on mobile devices so that various 3D objects can be captured and stored for personal use.

Then, the technologies for 3D meshes are expected to be exploited to help to guide the prediction or sampling of the point cloud. The point cloud has become popular among 3D content creators because it requires fewer artificial operations to obtain compared to mesh. However, it can not provide a continuous visualization as surface-like mesh when it is sparse while it requires massive computational resources to process and visualize the point cloud when it is dense. A good compromise point is expected to realize the balance between required resources and visual quality. To do so, a new type of file format or a novel compression method is required to combine the advantages of mesh and point cloud together.

Moreover, artificial intelligence technologies such as DL deserve further investigation for point clouds. The 2D solutions have already been used in numerous applications such as autonomous driving, object recognition, object detection/classification and so on. However, the lack of 3D information results in inaccurate predictions which can lead to dangerous circumstances. For example, when the luminance of the surrounding of a vehicle is not sufficient, the 2D image-based algorithm may not detect well what is actually near that vehicle. This results in a potential safety problem and can be avoided if 3D information is provided by LiDAR sensors which are much less affected in poor light conditions. With the help of DL-based detection and classification technologies, different compression algorithms and configurations can be specified for each type of point cloud. (e.g., human-shape point clouds, vehicle-shape point clouds, and planar point clouds (roads, sidewalks)).

REFERENCES

3DG. (2015). *Current Status on Point Cloud Compression*. ISO/IEC JTC1/SC29/WG11.

3DG. (2020). *Common Test Conditions for PCC* (p. 15). ISO/IEC JTC1/SC29/WG11.

A., S., & G., S. (1982). Data compression via textual substitution. *Journal of the ACM (JACM)*, *29*(4), 928–951. https://doi.org/10.1145/322344.322346

Ainala, K., Mekuria, R. N., Khathariya, B., Li, Z., Wang, Y.-K., & Joshi, R. (2016). An improved enhancement layer for octree based point cloud compression with plane projection approximation. In A. G. Tescher (Ed.), *Applications of Digital Image Processing {XXXIX}*. SPIE. https://doi.org/10.1117/12.2237753

Alexiou, E., Tung, K., & Ebrahimi, T. (2020). *Towards neural network approaches for point cloud compression*. https://doi.org/10.1117/12.2569115

Alsmeyer, G. (2011). Chebyshev's Inequality. In *International Encyclopedia of Statistical Science*. https://doi.org/10.1007/978-3-642-04898-2_167

Arthur, D., & Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, *07-09-Janu*, 1027–1035.

Bai, S., Kolter, J. Z., & Koltun, V. (2018). *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling*. http://arxiv.org/abs/1803.01271

Bentley, J. L. (1975a). Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*. https://doi.org/10.1145/361002.361007

Bentley, J. L. (1975b). Multidimensional Binary Search Trees Used for Associative Searching. *Communications of the ACM*, *18*(9), 509–517. https://doi.org/10.1145/361002.361007

Besl, P. J., & McKay, N. D. (1992a). A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. https://doi.org/10.1109/34.121791

Besl, P. J., & McKay, N. D. (1992b). Method for registration of 3-D shapes. In P. S. Schenker (Ed.), *Sensor Fusion IV: Control Paradigms and Data Structures* (Vol. 1611, pp. 586–606). https://doi.org/10.1117/12.57955

Bjontegaard, G. (2001). *Calculation of Average PSNR Differences between RD-Curves*. ITU-T VCEG-M33.

BlackBerry Limited and QNX Software Systems Limited. (2018). *Content provided to MPEG*. ISO/IEC JTC1/SC29/WG11.

Bogoslavskyi, I., & Stachniss, C. (2016). Fast range image-based segmentation of sparse 3D laser scans for online operation. *IEEE International Conference on Intelligent Robots and Systems*. https://doi.org/10.1109/IROS.2016.7759050

C. Tulvan, A. Gabrielli, M. P. (2016). *Datasets update on Point Cloud compression for cultural objects*. ISO/IEC JTC1/SC29/WG11. http://c3dc.fr/

Cao, C., Preda, M., & Zaharia, T. (n.d.). *What's new in Point Cloud Compression?* https://doi.org/10.33552/GJES.2020.04.000598

Cao, C., Preda, M., & Zaharia, T. (2019). 3D point cloud compression: A survey. *Proceedings - Web3D 2019: 24th International ACM Conference on 3D Web Technology*, 1–9. https://doi.org/10.1145/3329714.3338130

Cao, C., Preda, M., Zakharchenko, V., Jang, E. S., & Zaharia, T. (2021). Compression of Sparse and Dense Dynamic Point Clouds-Methods and Standards. *Proceedings of the IEEE*, *109*(9), 1537–1558. https://doi.org/10.1109/JPROC.2021.3085957

CAO, C., TULVAN, C., PREDA, M., & ZAHARIA, T. (2020). Skeleton-based motion estimation for Point Cloud Compression. *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, 1–6. https://doi.org/10.1109/MMSP48831.2020.9287165

Chen, Y., & Medioni, G. (1991). Object modeling by registration of multiple range images. *Proceedings - IEEE International Conference on Robotics and Automation*. https://doi.org/10.1109/robot.1991.132043

Cheung, G. K. M., Baker, S., & Kanade, T. (2003). Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *1*, I–I. https://doi.org/10.1109/cvpr.2003.1211340

Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., & Ranzuglia, G. (2008). MeshLab: An open-source mesh processing tool. *6th Eurographics Italian Chapter Conference 2008 - Proceedings*.

Cohen, R. A., Tian, D., & Vetro, A. (2016a). Attribute compression for sparse point clouds using graph transforms. *Proceedings - International Conference on Image Processing, ICIP*. https://doi.org/10.1109/ICIP.2016.7532583

Cohen, R. A., Tian, D., & Vetro, A. (2016b). Point Cloud Attribute Compression Using 3-D Intra Prediction and Shape-Adaptive Transforms. *Data Compression Conference Proceedings*. https://doi.org/10.1109/DCC.2016.67

Corsini, M., Cignoni, P., & Scopigno, R. (2012). Efficient and flexible sampling with blue noise properties of triangular meshes. *IEEE Transactions on Visualization and Computer Graphics*. https://doi.org/10.1109/TVCG.2012.34

Daribo, I., Furukawa, R., Sagawa, R., Kawasaki, H., Hiura, S., & Asada, N. (2012). Efficient rate-distortion compression of dynamic point cloud for grid-pattern-based 3D scanning systems. *3D Research*, *3*(1), 1–9. https://doi.org/10.1007/3DRes.01(2012)2

de Queiroz, R L, & Chou, P. A. (2016). Compression of 3D Point Clouds Using a Region-Adaptive Hierarchical Transform. *IEEE Transactions on Image Processing*, *25*(8), 3947–3956. https://doi.org/10.1109/TIP.2016.2575005

de Queiroz, R L, & Chou, P. A. (2017). Motion-Compensated Compression of Dynamic Voxelized Point Clouds. *IEEE Transactions on Image Processing*, *26*(8), 3886–3895. https://doi.org/10.1109/TIP.2017.2707807

De Queiroz, Ricardo L., & Chou, P. A. (2016). Compression of 3D Point Clouds Using a Region-Adaptive Hierarchical Transform. *IEEE Transactions on Image Processing*, *25*(8), 3947–3956. https://doi.org/10.1109/TIP.2016.2575005

De Queiroz, Ricardo L., & Chou, P. A. (2017a). Motion-Compensated Compression of Dynamic Voxelized Point Clouds. *IEEE Transactions on Image Processing*, *26*(8), 3886–3895. https://doi.org/10.1109/TIP.2017.2707807

De Queiroz, Ricardo L., & Chou, P. A. (2017b). Transform coding for point clouds using a Gaussian process model. *IEEE Transactions on Image Processing*, *26*(7), 3507–3517. https://doi.org/10.1109/TIP.2017.2699922

Deutsch, P. (1996). GZIP file format specication version 4.3. In *Aladdin Enterprises*.

Draco. (n.d.). https://github.com/google/draco

E. d'Eon, B. Harrison, T. Myers, P. A. C. (2017, January). *8i Voxelized Full Bodies – A Voxelized Point Cloud Dataset*.

Fan, Y, Huang, Y., & Peng, J. (2013). Point cloud compression based on hierarchical point clustering. *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 1–7. https://doi.org/10.1109/APSIPA.2013.6694334

Fan, Yuxue, Huang, Y., & Peng, J. (2013). Point cloud compression based on hierarchical point clustering. *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2013*. https://doi.org/10.1109/APSIPA.2013.6694334

Gandoin, P.-M., & Devillers, O. (2002). Progressive Lossless Compression of Arbitrary Simplicial Complexes. *ACM Trans. Graph.*, *21*(3), 372–379. https://doi.org/10.1145/566654.566591

Garcia, D C, & de Queiroz, R. L. (2017). Context-based octree coding for point-cloud video. *2017 IEEE International Conference on Image Processing (ICIP)*, 1412–1416. https://doi.org/10.1109/ICIP.2017.8296514

Garcia, D C, & de Queiroz, R. L. (2018). Intra-Frame Context-Based Octree Coding for Point-Cloud Geometry. *2018 25th IEEE International Conference on Image Processing (ICIP)*, 1807–1811. https://doi.org/10.1109/ICIP.2018.8451802

Garcia, Diogo C., & De Queiroz, R. L. (2018). Context-based octree coding for point-cloud video. *Proceedings - International Conference on Image Processing, ICIP*, *2017-Septe*, 1412–1416. https://doi.org/10.1109/ICIP.2017.8296514

Georgescu, B., Shimshoni, I., & Meer, P. (2003). Mean shift based clustering in high dimensions: A texture classification example. *Proceedings of the IEEE International Conference on Computer Vision*, *1*(Iccv), 456–463. https://doi.org/10.1109/iccv.2003.1238382

Gitl HEVC Analyzer. (n.d.). https://github.com/lheric/GitlHEVCAnalyzer

Golla, T., & Klein, R. (2015). Real-time point cloud compression. *IEEE International Conference on Intelligent Robots and Systems*, *2015-Decem*, 5087–5092. https://doi.org/10.1109/IROS.2015.7354093

Golub, G. H., & Reinsch, C. (1970). Singular value decomposition and least squares solutions. *Numerische Mathematik*. https://doi.org/10.1007/BF02163027

GTI-UPM. (n.d.). *3D models/point clouds/meshes*. https://mpegfs.int-evry.fr/mpegcontent/

Gu, S., Hou, J., Zeng, H., Yuan, H., & Ma, K. K. (2020). 3D Point Cloud Attribute Compression Using Geometry-Guided Sparse Representation. *IEEE Transactions on Image Processing*. https://doi.org/10.1109/TIP.2019.2936738

Guarda, A. F. R., Rodrigues, N. M. M., & Pereira, F. (2019a). Deep Learning-Based Point Cloud Coding: A Behavior and Performance Study. *Proceedings - European Workshop on Visual Information Processing, EUVIP*, *2019-Octob*, 34–39. https://doi.org/10.1109/EUVIP47703.2019.8946211

Guarda, A. F. R., Rodrigues, N. M. M., & Pereira, F. (2019b). Point Cloud Coding: Adopting a Deep Learning-based Approach. *2019 Picture Coding Symposium, PCS 2019*, 19–23. https://doi.org/10.1109/PCS48520.2019.8954537

Güler, R. A., Neverova, N., & Kokkinos, I. (2018). DensePose: Dense Human

Pose Estimation in the Wild. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2018.00762

Gumhold, S., Karni, Z., Isenburg, M., & Seidel, H. P. (2005). Predictive point-cloud compression. *ACM SIGGRAPH 2005 Sketches, SIGGRAPH 2005*, 137. https://doi.org/10.1145/1187112.1187277

H.-L. Guillaume, T. Doneux, A. Schenkel, G. L. (n.d.). *ULB Unicorn photogrammetric point cloud data*. ISO/IEC JTC1/SC29/WG11.

Hansen, E. H., Gobakken, T., & Næsset, E. (2015). Effects of pulse density on digital terrain models and canopy metrics using airborne laser scanning in a tropical rainforest. *Remote Sensing*. https://doi.org/10.3390/rs70708453

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., & Stuetzle, W. (1992). Surface reconstruction from unorganized points. *Computer Graphics (ACM)*. https://doi.org/10.1145/142920.134011

Huang, T., & Liu, Y. (2019). 3D point cloud geometry compression on deep learning. *MM 2019 - Proceedings of the 27th ACM International Conference on Multimedia*. https://doi.org/10.1145/3343031.3351061

Huang, Y, Peng, J., Kuo, C.-. J., & Gopi, M. (2008). A Generic Scheme for Progressive Point Cloud Coding. *IEEE Transactions on Visualization and Computer Graphics*, *14*(2), 440–453. https://doi.org/10.1109/TVCG.2007.70441

Huang, Y, Peng, J., Kuo, C., & Gopi, M. (2006). Octree-Based Progressive Geometry Coding of Point Clouds. *Eurographics Symposium on Point-Based Graphics (SPBG)*. https://doi.org/10.2312/SPBG/SPBG06/103-110

Huang, Yan, Peng, J., Kuo, C.-C. J., & Gopi, M. (2006). Octree-based Progressive Geometry Coding of Point Clouds. *Proceedings of the 3rd Eurographics / IEEE VGTC Conference on Point-Based Graphics*, 103–110. https://doi.org/10.2312/SPBG/SPBG06/103-110

Im, J. J., Leonessa, A., Kurdila, A., & Ryoo, Y. J. (2010). A real-time data compression for ground-based 3D LIDAR data using wavelets and compressive sensing. *SCIS and ISIS 2010 - Joint 5th International Conference on Soft Computing and Intelligent Systems and 11th International Symposium on Advanced Intelligent Systems*.

International Telecommunication Union. (2002). Parameter values for the HDTV standards for production and international programme exchange BT Series Broadcasting service. *Recommendation ITU-R BT.709-5*, *5*, 1–32. http://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.709-5-200204-I!!PDF-E.pdf

Isenburg, M. (2013). LASzip: Lossless compression of lidar data. *Photogrammetric Engineering and Remote Sensing*, *79*(2), 209–217. https://doi.org/10.14358/PERS.79.2.209

ISO/IEC. (2015). *ISO base media file format*.

J. Ricard , C. Guède, R. Doré, S. L. (n.d.). *CGI-based dynamic point cloud test content*. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG).

James and others, M. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, *1*(14), 281–297.

http://books.google.de/books?hl=de&lr=&id=IC4Ku_7dBFUC&oi=fnd&pg=PA281&dq=MacQueen+some+methods+for+classification&ots=nNTcK1IdoQ&sig=fHzdVcbvmYJ-lTNHu1HncmOFOkM#v=onepage&q=MacQueen some methods for classification&f=false

Kammerl, J, Blodow, N., Rusu, R. B., Gedikli, S., Beetz, M., & Steinbach, E. (2012). Real-time compression of point cloud streams. *2012 IEEE International Conference on Robotics and Automation*, 778–785. https://doi.org/10.1109/ICRA.2012.6224647

Kammerl, Julius, Blodow, N., Rusu, R. B., Gedikli, S., Beetz, M., & Steinbach, E. (2012). Real-time compression of point cloud streams. *Proceedings - IEEE International Conference on Robotics and Automation*, 778–785. https://doi.org/10.1109/ICRA.2012.6224647

Kathariya, B., Li, L., Li, Z., Alvarez, J., & Chen, J. (2018). Scalable Point Cloud Geometry Coding with Binary Tree Embedded Quadtree. *Proceedings - IEEE International Conference on Multimedia and Expo*, *2018-July*, 1–6. https://doi.org/10.1109/ICME.2018.8486481

Kitago, M., & Gopi, M. (2006). Efficient and Prioritized Point Subsampling for CSRBF Compression. *Eurographics Symposium on Point-Based Graphics*, 9. https://doi.org/10.2312/SPBG/SPBG06/121-128

Kohira, K., & Masuda, H. (2017). POINT-CLOUD COMPRESSION for VEHICLE-BASED MOBILE MAPPING SYSTEMS USING PORTABLE NETWORK GRAPHICS. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *4*(2W4), 99–106. https://doi.org/10.5194/isprs-annals-IV-2-W4-99-2017

Li, L., Li, Z., Liu, S., & Li, H. (2021). Occupancy-Map-Based Rate Distortion Optimization and Partition for Video-Based Point Cloud Compression. *IEEE Transactions on Circuits and Systems for Video Technology*, *31*(1), 326–338. https://doi.org/10.1109/TCSVT.2020.2966118

Li, L., Li, Z., Liu, S., & Li, H. (2019). Occupancy-Map-Based Rate Distortion Optimization for Video-Based Point Cloud Compression. *2019 IEEE International Conference on Image Processing (ICIP)*, 3167–3171. https://doi.org/10.1109/ICIP.2019.8803233

Li, L., Li, Z., Zakharchenko, V., Chen, J., & Li, H. (2020). Advanced 3D Motion Prediction for Video-Based Dynamic Point Cloud Compression. *IEEE Transactions on Image Processing*, *29*, 289–302. https://doi.org/10.1109/TIP.2019.2931621

Li, S., & Li, W. (2000). Shape-adaptive discrete wavelet transforms for arbitrary shaped visual object coding. *Circuits and Systems for Video Technology, IEEE Transactions On*, *10*, 725–743. https://doi.org/10.1109/76.856450

Lien, J.-M., Kurillo, G., & Bajcsy, R. (2009). Multi-camera tele-immersion system with real-time model driven data compression. *The Visual Computer*, *26*(1), 3–15. https://doi.org/10.1007/s00371-009-0367-8

Lien, J. M., Kurillo, G., & Bajcsy, R. (2010). Multi-camera tele-immersion system with real-time model driven data compression: A new model-based compression method for massive dynamic point data. *Visual Computer*, *26*(1), 3–15. https://doi.org/10.1007/s00371-009-0367-8

Lin, C. H., Kong, C., & Lucey, S. (2018). Learning efficient point cloud generation for dense 3D object reconstruction. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*.

Liu, X., Yan, M., & Bohg, J. (2019). Meteornet: Deep learning on dynamic 3D point cloud sequences. *Proceedings of the IEEE International Conference on Computer*

Vision. https://doi.org/10.1109/ICCV.2019.00934

Loop, C., Zhang, C., & Zhang, Z. (2013). Real-time High-resolution Sparse Voxelization with Application to Image-based Modeling. *Proceedings of the 5th High-Performance Graphics Conference*, 73–79. https://doi.org/10.1145/2492045.2492053

Lu, Q., & Lee, S. (2017). Image-Based Technologies for Constructing As-Is Building Information Models for Existing Buildings. *Journal of Computing in Civil Engineering*. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000652

Masuda, H., & He, J. (2015). TIN generation and point-cloud compression for vehicle-based mobile mapping systems. *Advanced Engineering Informatics*, *29*(4), 841–850. https://doi.org/10.1016/j.aei.2015.05.007

Mekuria, R., Blom, K., & Cesar, P. (2017). Design, Implementation, and Evaluation of a Point Cloud Codec for Tele-Immersive Video. *IEEE Transactions on Circuits and Systems for Video Technology*, *27*(4), 828–842. https://doi.org/10.1109/TCSVT.2016.2543039

Merkle, P., Smolic, A., Müller, K., & Wiegand, T. (2007). Multi-view video plus depth representation and coding. *Proceedings - International Conference on Image Processing, ICIP*, *1*, I--201. https://doi.org/10.1109/ICIP.2007.4378926

Merry, B., Marais, P., & Gain, J. (2006a). Compression of dense and regular point clouds. *ACM International Conference on Computer Graphics, Virtual Reality and Visualisation in Africa*, *2006*, 155–159. https://doi.org/10.1145/1108590.1108593

Merry, B., Marais, P., & Gain, J. (2006b). Compression of Dense and Regular Point Clouds. *Proceedings of the 4th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, 15–20. https://doi.org/10.1145/1108590.1108593

Mongus, D., & Žalik, B. (2011). Efficient method for lossless LIDAR data compression. *International Journal of Remote Sensing*, *32*(9), 2507–2518. https://doi.org/10.1080/01431161003698385

Morell, V., Orts, S., Cazorla, M., & Garcia-Rodriguez, J. (2014). Geometric 3D point cloud compression. *Pattern Recognition Letters*, *50*, 55–62. https://doi.org/10.1016/j.patrec.2014.05.016

Moreno, C., Chen, Y., & Li, M. (2017). *A Dynamic Compression Technique for Streaming Kinect-Based Point Cloud Data*.

MPEG. (n.d.). *V-PCC on the MPEG GitHub*. https://github.com/MPEGGroup/mpeg-pcc-tmc2

Navarrete, J., Viejo, D., & Cazorla, M. (2018). Compression and registration of 3D point clouds using GMMs. *Pattern Recognition Letters*. https://doi.org/10.1016/j.patrec.2018.03.017

Ochotta, T., & Saupe, D. (2004). Compression of point-based 3D models by shape-adaptive wavelet coding of multi-height fields. *Proc. Symposium on Point-Based Graphics*, 103–112. https://doi.org/10.2312/SPBG/SPBG04/103-112

Otepka, J., Ghuffar, S., Waldhauser, C., Hochreiter, R., & Pfeifer, N. (2013). Georeferenced Point Clouds: A Survey of Features and Point Cloud Management. *{ISPRS} International Journal of Geo-Information*, *2*(4), 1038–1065. https://doi.org/10.3390/ijgi2041038

Owoyemi, J., & Hashimoto, K. (2018). Spatiotemporal Learning of Dynamic Gestures from 3D Point Cloud Data. *Proceedings - IEEE International Conference on Robotics and Automation*. https://doi.org/10.1109/ICRA.2018.8460910

Pandey, G., McBride, J. R., & Eustice, R. M. (2011). Ford Campus vision and lidar data set. *International Journal of Robotics Research*. https://doi.org/10.1177/0278364911400640

Pauly, M., & Gross, M. (2001). Spectral processing of point-sampled geometry. *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001*, 379–386. https://doi.org/10.1145/383259.383301

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017, July). PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Quach, M., Valenzise, G., & Dufaux, F. (2019). Learning Convolutional Transforms for Lossy Point Cloud Geometry Compression. *Proceedings - International Conference on Image Processing, ICIP*. https://doi.org/10.1109/ICIP.2019.8803413

Quach, M., Valenzise, G., & Dufaux, F. (2020a). *Folding-based compression of point cloud attributes*. http://arxiv.org/abs/2002.04439

Quach, M., Valenzise, G., & Dufaux, F. (2020b). *Improved Deep Point Cloud Geometry Compression*. http://arxiv.org/abs/2006.09043

R. Cohen, H. Ochimizu, D. Tian, A. V. (2017). *Mobile Mapping System Point Cloud Data from Mitisubishi Electric*. ISO/IEC JTC1/SC29/WG11.

Raghuraman, S., Venkatraman, K., Wang, Z., Prabhakaran, B., & Guo, X. (2013). A 3D tele-immersion streaming approach using skeleton-based prediction. *MM 2013 - Proceedings of the 2013 ACM Multimedia Conference*. https://doi.org/10.1145/2502081.2502188

Rusu, R. B., & Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). *2011 IEEE International Conference on Robotics and Automation*, 1–4. https://doi.org/10.1109/ICRA.2011.5980567

Sander, P. V, Wood, Z. J., Gortler, S. J., & Hoppe, H. (2003). Multi-chart geometry images. *SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*.

Sayood, K. (2012). Introduction to Data Compression. In *Introduction to Data Compression*. https://doi.org/10.1016/C2010-0-69630-1

Schnabel, R., & Klein, R. (2006). Octree-based Point-Cloud Compression. *Spbg*. https://doi.org/10.2312/SPBG/SPBG06/111-120

Schwarz, S, Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P. A., Cohen, R. A., Krivokuća, M., Lasserre, S., Li, Z., Llach, J., Mammou, K., Mekuria, R., Nakagami, O., Siahaan, E., Tabatabai, A., Tourapis, A. M., & Zakharchenko, V. (2019). Emerging MPEG Standards for Point Cloud Compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *9*(1), 133–148. https://doi.org/10.1109/JETCAS.2018.2885981

Schwarz, Sebastian, Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P. A., Cohen, R. A., Krivokuca, M., Lasserre, S., Li, Z., Llach, J., Mammou, K., Mekuria, R., Nakagami, O., Siahaan, E., Tabatabai, A., Tourapis, A. M., & Zakharchenko, V. (2019). Emerging MPEG Standards for Point Cloud Compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *9*(1), 133–148. https://doi.org/10.1109/JETCAS.2018.2885981

Shao, Y., Zhang, Q., Li, G., Li, Z., & Li, L. (2018). Hybrid point cloud attribute compression using slice-based layered structure and block-based intra prediction. *MM 2018 - Proceedings of the 2018 ACM Multimedia Conference*.

https://doi.org/10.1145/3240508.3240696

Shao, Y., Zhang, Z., Li, Z., Fan, K., & Li, G. (2018). Attribute compression of 3D point clouds using Laplacian sparsity optimized graph transform. *2017 IEEE Visual Communications and Image Processing, VCIP 2017*. https://doi.org/10.1109/VCIP.2017.8305131

Simon, D. A., Hebert, M., & Kanade, T. (1994). Real-time 3-D pose estimation using a high-speed range sensor. *Proceedings - IEEE International Conference on Robotics and Automation*. https://doi.org/10.1109/robot.1994.350953

stanford 2 dataset. (n.d.). http://buildingparser.stanford.edu/dataset.html

Stanford Bunny. (n.d.). http://graphics.stanford.edu/data/3Dscanrep/

Sun, X., Ma, H., Sun, Y., & Liu, M. (2019). A Novel Point Cloud Compression Algorithm Based on Clustering. *IEEE Robotics and Automation Letters*, *4*(2), 2132–2139. https://doi.org/10.1109/LRA.2019.2900747

Taubin, G., & Rossignac, J. (1998). Geometric compression through topological surgery. *ACM Transactions on Graphics*, *17*(2), 84–115. https://doi.org/10.1145/274363.274365

Thanou, D, Chou, P. A., & Frossard, P. (2016). Graph-Based Compression of Dynamic 3D Point Cloud Sequences. *IEEE Transactions on Image Processing*, *25*(4), 1765–1778. https://doi.org/10.1109/TIP.2016.2529506

Thanou, D, Chou, P. A., & Frossard, P. (2015). Graph-based motion estimation and compensation for dynamic 3D point cloud compression. *2015 IEEE International Conference on Image Processing (ICIP)*, 3235–3239. https://doi.org/10.1109/ICIP.2015.7351401

Thanou, Dorina, Chou, P. A., & Frossard, P. (2016). Graph-based compression of dynamic 3D point cloud sequences. *IEEE Transactions on Image Processing*, *25*(4), 1765–1778. https://doi.org/10.1109/TIP.2016.2529506

Thanou, Dorina, Chou, P. A., & Frossard, P. (2015). Graph-based motion estimation and compensation for dynamic 3D point cloud compression. *Proceedings - International Conference on Image Processing, ICIP*, *2015-Decem*, 3235–3239. https://doi.org/10.1109/ICIP.2015.7351401

Tu, C., Takeuchi, E., Carballo, A., & Takeda, K. (2019a). Real-Time Streaming Point Cloud Compression for 3D LiDAR Sensor Using U-Net. *IEEE Access*. https://doi.org/10.1109/access.2019.2935253

Tu, C., Takeuchi, E., Carballo, A., & Takeda, K. (2019b). Point cloud compression for 3d lidar sensor using recurrent neural network with residual blocks. *Proceedings - IEEE International Conference on Robotics and Automation*, *2019-May*, 3274–3280. https://doi.org/10.1109/ICRA.2019.8794264

Tu, C., Takeuchi, E., Miyajima, C., & Takeda, K. (2016). Compressing continuous point cloud data using image compression methods. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 1712–1719. https://doi.org/10.1109/ITSC.2016.7795789

Tu, C., Takeuchi, E., Miyajima, C., & Takeda, K. (2017). Continuous point cloud data compression using SLAM based prediction. *IEEE Intelligent Vehicles Symposium, Proceedings*, 1744–1751. https://doi.org/10.1109/IVS.2017.7995959

VCEG, J. V. T. (JVT) of I. M. & I.-T., & Q.6), (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16. (n.d.). *Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC)*.

Viejo, D., & Cazorla, M. (2014). A robust and fast method for 6DoF motion estimation from generalized 3D data. *Autonomous Robots*. https://doi.org/10.1007/s10514-013-9354-z

Wang, J., Zhu, H., Ma, Z., Chen, T., Liu, H., & Shen, Q. (2019). *Learned Point Cloud Geometry Compression*. 1–13. http://arxiv.org/abs/1909.12037

Wang, L., Wang, L., Luo, Y., & Liu, M. (2017). Point-Cloud compression using data independent method - A 3D discrete cosine transform approach. *2017 IEEE International Conference on Information and Automation, ICIA 2017*. https://doi.org/10.1109/ICInfA.2017.8078873

Wang, Q., Tan, Y., & Mei, Z. (2020). Computational Methods of Acquisition and Processing of 3D Point Cloud Data for Construction Applications. *Archives of Computational Methods in Engineering*. https://doi.org/10.1007/s11831-019-09320-4

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., & Solomon, J. M. (2019). Dynamic graph Cnn for learning on point clouds. *ACM Transactions on Graphics*. https://doi.org/10.1145/3326362

Waschbüsch, M., Gross, M. H., Eberhard, F., Lamboray, E., & Würmlin, S. (2004). Progressive Compression of Point-Sampled Models. *SPBG*, 95–102.

Wendland, H. (1995). Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. *Advances in Computational Mathematics*, *4*(1), 389–396. https://doi.org/10.1007/BF02123482

Wiman, H., & Yuchu, Q. (2009a). Fast compression and access of LiDAR point clouds using wavelets. *2009 Joint Urban Remote Sensing Event*. https://doi.org/10.1109/URS.2009.5137589

Wiman, H., & Yuchu, Q. (2009b). Fast compression and access of LiDAR point clouds using wavelets. *2009 Joint Urban Remote Sensing Event*, 0–5. https://doi.org/10.1109/URS.2009.5137589

Xu, Y., Hu, W., Wang, S., Zhang, X., Wang, S., Ma, S., & Gao, W. (2018). Cluster-based point cloud coding with normal weighted graph fourier transform. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. https://doi.org/10.1109/ICASSP.2018.8462684

Yan, W., shao, Y., Liu, S., Li, T. H., Li, Z., & Li, G. (2019). *Deep AutoEncoder-based Lossy Geometry Compression for Point Clouds*. http://arxiv.org/abs/1905.03691

Yang, Y., Feng, C., Shen, Y., & Tian, D. (2018). FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2018.00029

Zhang, C, Florêncio, D., & Loop, C. (2014). Point cloud attribute compression with graph transform. *2014 IEEE International Conference on Image Processing (ICIP)*, 2066–2070. https://doi.org/10.1109/ICIP.2014.7025414

Zhang, Cha, & Florêncio, D. (2013). Analyzing the optimality of predictive transform coding using graph-based models. *IEEE Signal Processing Letters*, *20*(1), 106–109. https://doi.org/10.1109/LSP.2012.2230165

Zhang, K, Zhu, W., & Xu, Y. (2018). Hierarchical Segmentation Based Point Cloud Attribute Compression. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3131–3135. https://doi.org/10.1109/ICASSP.2018.8461644

Zhang, Ke, Zhu, W., & Xu, Y. (2018). Hierarchical Segmentation Based Point Cloud Attribute Compression. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2018-April*, 3131–3135. https://doi.org/10.1109/ICASSP.2018.8461644

Zhang, Q., Shao, Y., & Li, G. (2018). Point Clouds Attribute Compression Using Data-Adaptive Intra prediction. *VCIP 2018 - IEEE International Conference on Visual Communications and Image Processing*. https://doi.org/10.1109/VCIP.2018.8698681

Zhou, Y., & Tuzel, O. (2018). VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2018.00472

Ziv, J., & Lempel, A. (1977). A Universal Algorithm for Sequential Data Compression. *IEEE Transactions on Information Theory*, *23*(3), 337–343. https://doi.org/10.1109/TIT.1977.1055714

ANNEX

A: Objective Evaluation Metrics

A.1 Geometric Distortions

Let **A** and **B** denote the original and the compressed point cloud, respectively. Consider evaluating the compression errors, denoted as $e_{\mathbf{B,A}}$ in point cloud **B** relative to reference point cloud **A**. The steps to compute both point-to-point error (D1) and point-to-plane error (D2) for geometric errors are summarized in the following and illustrated in the below figure.

For each point b_i in point cloud **B**, i.e., the black point in the figure, identify a corresponding point a_j in point cloud **A**, i.e., the red point in the figure. Nearest neighbor is used to locate the corresponding point. In particular, a KD-tree search is used to perform the nearest neighbor search to reduce the computation complexity.

A.1.1 Computing D1

Determine an error vector $E(i,j)$ by connecting the identified point a_j in reference point cloud **A** to point b_i in point cloud **B.** The length of the error vector is the point-to-point error, i.e.,

$$e^{\mathrm{D1}}_{\mathbf{B,A}}(i) = \|E(i,j)\|_2^2 \qquad \text{(A-1)}$$

Based on the point-to-point distances $e^{\mathrm{D1}}_{\mathbf{B,A}}(i)$ for all points $i \in \mathbf{B}$, the point-to-point error (D1) for the whole point cloud, with N_B as the number of points in point cloud **B**, is defined as:

$$e^{\mathrm{D1}}_{\mathbf{B,A}} = \frac{1}{N_B}\sum_{\forall b_i \in \mathrm{B}} e^{\mathrm{D1}}_{\mathbf{B,A}}(i) \qquad \text{(A-2)}$$

For near-lossless geometry encoding, the maximum point-to-point error, denoted $\varepsilon^{\mathrm{D1}}_{\mathbf{B,A}}$, should be considered instead:

$$\varepsilon^{\mathrm{D1}}_{\mathbf{B,A}} = max_{b_i \in \mathrm{B}}\{e^{\mathrm{D1}}_{\mathbf{B,A}}(i)\} \qquad \text{(A-3)}$$

A.1.2 Computing D2

Project the error vector $E(i,j)$ along the normal direction N_j and get a new error vector $\hat{E}(i,j)$. In this way, the point-to-plane error is computed as,

$$e^{\mathrm{D2}}_{\mathbf{B,A}}(i) = \left\|\hat{E}(i,j)\right\|_2^2 = (E(i,j)\cdot N_j)^2 \qquad \text{(A-4)}$$

The point-to-plane error (D2) for the whole point cloud is then defined as,

$$e^{\mathrm{D2}}_{\mathbf{B,A}} = \frac{1}{N_B}\sum_{\forall b_i \in \mathrm{B}} e^{\mathrm{D2}}_{\mathbf{B,A}}(i) \qquad \text{(A-5)}$$

For near-lossless geometry encoding, the maximum plane-to-plane error, denoted $\varepsilon^{\mathrm{D2}}_{\mathbf{B,A}}$, should be considered instead:

$$\varepsilon^{\mathrm{D2}}_{\mathbf{B,A}} = max_{b_i \in \mathrm{B}}\{e^{\mathrm{D2}}_{\mathbf{B,A}}(i)\} \qquad \text{(A-6)}$$

Institut Polytechnique de Paris

91120 Palaiseau, France

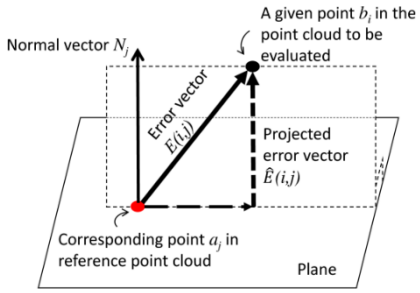


Illustration of point-to-point distance (D1) and point-to-plane distance (D2)

A.1.3 PSNR Calculation

For reporting distortion values, the PSNR is defined as the peak signal over the symmetric distortion, i.e., for geometry distortions D1 or D2 computed as:

$$PSNR = 10\log_{10}\left(\frac{3p^2}{max(e^{Dx}_{B,A}, e^{Dx}_{A,B})}\right), \tag{A-7}$$

where p is the peak constant value defined for each reference point, and MSE is the mean squared point-to-point (D1) or point-to-plane (D2) error. For dynamic content, the peak value is unchanged over the frames of a sequence.

A.2 Attribute Distortions

For lossy attribute coding, the attribute PSNR value is computed as:

$$PSNR = 10\log_{10}\left(\frac{p^2}{symmetricMSE}\right), \tag{A-8}$$

For color attributes, the MSE for each of the three color components is calculated. A conversion from RGB space to YUV space is conducted using ITU-R BT.709 (International Telecommunication Union, 2002), since YUV space correlates better with human perception. A symmetric computation of the distortion is utilized, in the same way as is done for geometric distortions. The maximum distortion between the two passes is selected as the final distortion. Since the color attributes for all test data have a bit depth of 8 bits per point, the peak value p for PSNR calculation is 255.

For reflectance attributes, the MSE for a single component is calculated. Since the reflectance attribute for all test data has a bit depth of 16 bits per point, the peak value p for PSNR calculation is 65535.

Titre: Compression d'objets 3D représentés par nuages de points

Mots clés: compression, représentations par nuages de points 3D, transformation spectrale, corrélation spatiale, estimation de mouvement, modèles prédictif

Résumé: Avec la croissance rapide du contenu multimédia, les objets 3D deviennent de plus en plus populaires. Ils sont généralement modélisés sous forme de maillages polygonaux complexes ou de nuages de points 3D denses, offrant des expériences immersives dans différentes applications multimédias industrielles et grand public. La représentation par nuages de points, plus facile à acquérir que les maillages, a suscité ces dernières année un intérêt croissant tant dans le monde académique que commercial.

Un nuage de points est par définition un ensemble de points définissant la géométrie de l'objet et les attributs associés (couleurs, textures, les propriétés des matériaux, etc.). Le nombre de points dans un nuage de points peut aller d'un millier, pour représenter des objets relativement simples, jusqu'à des milliards pour représenter de manière réaliste des scènes 3D complexes. Ces énormes quantités de données posent de grands défis liés à la transmission, au traitement et au stockage des nuages de points 3D.

Ces dernières années, de nombreux travaux ont été dédiés principalement à la compression de maillages, tandis qu'un nombre plus réduit de techniques s'attaquent à la problématique de compression de nuages de points 3D. L'état de l'art fait ressortir deux grandes familles approches principales: une première purement géométrique, fondée sur une décomposition en octree et une seconde hybride, exploitant à la fois la projection multi-vues de la géométrie et le codage vidéo. La première approche permet de préserver une information de géométrie 3D précise mais contient une faible cohérence temporelle. La seconde permet de supprimer efficacement la redondance temporelle mais est pénalisé par une diminution de la précision géométrique, liée au processus de projection 3D/2D. Ainsi, le compromis entre efficacité de compression et précision des objets reconstruit doit être optimisé.

Premièrement, une segmentation adaptative par octree a été proposée pour regrouper les points avec différentes amplitudes de mouvement dans des cubes 3D. Ensuite, une estimation de mouvement est appliquée à ces cubes en utilisant une transformation affine. Des gains en termes de performances de distorsion de débit (RD) ont été observés dans des séquences avec des amplitudes de mouvement plus faibles. Cependant, le coût de construction d'un octree pour le nuage de points dense reste élevé tandis que les structures d'octree résultantes contiennent une mauvaise cohérence temporelle pour les séquences avec des amplitudes de mouvement plus élevées. De plus, une méthode de prédiction adaptative davantage au mouvement humain est requise que l'approche basée sur les cubes.

Ainsi, une structure anatomique a ensuite été proposée pour modéliser le mouvement de manière intrinsèque. À l'aide d'outils d'estimation de pose 2D, le mouvement est estimé à partir de 14 segments anatomiques à l'aide d'une transformation affine. Il est montré qu'au lieu de coder les informations redondantes, il est plus intéressant de coder le mouvement estimé sous forme de matrices de transformation affines. La méthode proposée permet d'obtenir de meilleures performances de compression pour le contenu avec de faibles amplitudes de mouvement. Il peut grandement améliorer la qualité visuelle des nuages de points décodés dans des conditions de faible débit. Une telle caractéristique est bénéfique pour les applications VR/AR, où la bande passante du réseau est limitée et la fidélité du contenu est demandée.

De plus, nous avons proposé une nouvelle solution pour la prédiction des couleurs et discuté du codage des résidus de la prédiction. La même estimation de mouvement basée sur le squelette est utilisée sur les informations géométriques des nuages de points pour établir la correspondance entre des images consécutives. Une fois la cohérence établie, le processus de prédiction de couleur est ensuite poursuivi et les résidus de couleur sont calculés. Il est montré qu'au lieu de coder des informations de texture redondantes, il est plus intéressant de coder les résidus, ce qui a entraîné une meilleure performance RD. On observe que de grands gains en termes de performances RD ont été obtenus pour toutes les séquences testées.

Les différentes approches proposées ont permis d'améliorer les performances des modèles de test V-PCC. Toutefois, la compression temporelle de nuages de points 3D dynamiques reste une tâche complexe et difficile. Ainsi, en raison des limites de la technologie d'acquisition actuelle, les nuages acquis peuvent être bruyants à la fois dans les domaines de la géométrie et des attributs, ce qui rend difficile l'obtention d'une estimation précise du mouvement. On s'attend à ce que la compression des nuages de points 3D évolue avec l'avancée des technologies d'acquisition et de l'informatique. Dans les études futures, les technologies utilisées pour les maillages 3D peuvent être exploitées et adaptées pour fournir des informations de connectivité cohérentes dans le temps entre des nuages de points 3D dynamiques. De plus, les technologies d'intelligence artificielle telles que DL méritent une enquête plus approfondie sur les nuages de points et peuvent grandement aider à la compression d'un contenu aussi volumineux.

Institut Polytechnique de Paris

91120 Palaiseau, France

Title: 3D point cloud compression

Keywords: 3D geometry compression, spectral transform, predictive models, 3D point cloud representation, motion estimation.

Abstract: With the rapid growth of multimedia content, 3D objects are becoming more and more popular. Most of the time, they are modeled as complex polygonal meshes or dense point clouds, providing immersive experiences in different industrial and consumer multimedia applications. The point cloud, which is easier to acquire than mesh and is widely applicable, has raised many interests in both the academic and commercial worlds.

A point cloud is a set of points with different properties such as their geometrical locations and the associated attributes (e.g., color, material properties, etc.). The number of the points within a point cloud can range from a thousand, to constitute simple 3D objects, up to billions, to realistically represent complex 3D scenes. Such huge amounts of data bring great technological challenges in terms of transmission, processing, and storage of point clouds.

In recent years, numerous research works focused their efforts on the compression of meshes, while less was addressed for point clouds. We have identified two main approaches in the literature: a purely geometric one based on octree decomposition, and a hybrid one based on both geometry and video coding. The first approach can provide accurate 3D geometry information but contains weak temporal consistency. The second one can efficiently remove the temporal redundancy yet a decrease of geometrical precision can be observed after the projection. Thus, the tradeoff between compression efficiency and accurate prediction needs to be optimized.

We focused on exploring the temporal correlations between dynamic dense point clouds. We proposed different approaches to improve the compression performance of the MPEG (Moving Picture Experts Group) V-PCC (Video-based Point Cloud Compression) test model, which provides state-of-the-art compression on dynamic dense point clouds.

First, an octree-based adaptive segmentation is proposed to cluster the points with different motion amplitudes into 3D cubes. Then, motion estimation is applied to these cubes using affine transformation. Gains in terms of rate-distortion (RD) performance have been observed in sequences with relatively low motion amplitudes. However, the cost of building an octree for the dense point cloud remains expensive while the resulting octree structures contain poor temporal consistency for the sequences with higher motion amplitudes.

An anatomical structure is then proposed to model the motion of the point clouds representing humanoids more inherently. With the help of 2D pose estimation tools, the motion is estimated from 14 anatomical segments using affine transformation.

Moreover, we propose a novel solution for color prediction and discuss the residual coding from prediction. It is shown that instead of encoding redundant texture information, it is more valuable to code the residuals, which leads to a better RD performance.

Although our contributions have improved the performances of the V-PCC test models, the temporal compression of dynamic point clouds remains a highly challenging task. Due to the limitations of the current acquisition technology, the acquired point clouds can be noisy in both geometry and attribute domains, which makes it challenging to achieve accurate motion estimation. In future studies, the technologies used for 3D meshes may be exploited and adapted to provide temporal-consistent connectivity information between dynamic 3D point clouds.

Institut Polytechnique de Paris

91120 Palaiseau, France