DISSERTAÇÃO DE MESTRADO

PLENOPTIC POINT CLOUD CODING USING MPEG-BASED STRATEGIES

Davi Rabbouni de Carvalho Freitas

Brasília, Julho de 2021

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO

PLENOPTIC POINT CLOUD CODING USING MPEG-BASED STRATEGIES

Davi Rabbouni de Carvalho Freitas

Dissertação de Mestrado submetida ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Mestre em Engenharia Elétrica

Banca Examinadora

Prof. Ricardo Lopes de Queiroz, Ph.D., PPGEE / UnB
Orientador

Prof. Bruno Luiggi Macchiavello Espinoza, Ph.D., UnB
Examinador Externo

Prof. Bruno Zatt, Ph.D., UFPel
Examinador Externo

FICHA CATALOGRÁFICA

FREITAS, DAVI RABBOUNI DE CARVALHO
PLENOPTIC POINT CLOUD CODING USING MPEG-BASED STRATEGIES [Distrito Federal] 2021.
xvi, 60 p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2021).
Dissertação de Mestrado - Universidade de Brasília, Faculdade de Tecnologia.
Departamento de Engenharia Elétrica

1. point cloud	2. plenoptic
3. compression	4. MPEG standards
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

FREITAS, D.R.C (2021). *PLENOPTIC POINT CLOUD CODING USING MPEG-BASED STRATEGIES*. Dissertação de Mestrado, Publicação: PPGENE.DM-772/21, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 60 p.

CESSÃO DE DIREITOS

AUTOR: Davi Rabbouni de Carvalho Freitas
TÍTULO: PLENOPTIC POINT CLOUD CODING USING MPEG-BASED STRATEGIES.
GRAU: Mestre em Engenharia Elétrica ANO: 2021

Davi Rabbouni de Carvalho Freitas
Depto. de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

Acknowledgments

First of all, I would like to thank my advisor, Professor Ricardo Queiroz, whose insights and continuous guidance were critical for the development of this work. I would also like to thank our other colleagues from the DIVP group who were part of this research at different stages of its development: Professors Camilo Dorea, Diogo Garcia, Eduardo Peixoto, Gustavo Sandri and Renan Ferreira.

I would like to thank my family for their continuous support throughout this journey. In particular, I would like to thank my parents, who always guided me in moments of uncertainty and encouraged me with their examples of dedication and seriousness in what they do. To my grandmother, who, despite the distance, always kept me close with her thoughts and prayers.

Finally, I would also like to acknowledge my friends for their patience and support throughout these studies.

Davi Rabbouni de Carvalho Freitas

ABSTRACT

Plenoptic point clouds (PPC) are data structures that provide a higher degree of realism to regular point clouds by representing the light from different viewing directions. This is achieved by associating each point to multiple colors instead of a single one. In this work, we present two different approaches to efficiently compress the N_c plenoptic attributes of a PPC that are compliant with MPEG's existing coding strategies for point cloud compression (PCC). The first is a geometry-based method, consisting of a Karhunen-Loève transform over the color attributes followed by multiple attribute coders with intra-prediction capability. This compression scheme can be incorporated within the MPEG's geometry-based PCC (G-PCC) standard, using any of G-PCC's existing solutions for attribute coding. The second one is a video-based approach that is backwards compatible with MPEG's video-based PCC (V-PCC) codec and is based on a pixel-wise transform of the N_c plenoptic atlases with an N_c-point discrete cosine transform. Transformed atlases are encoded using a conventional video codec and are conveyed as an additional payload to the conventional V-PCC's bitstream. Compression performance assessment using PPCs of different spatial resolutions reveals competitive results, in comparison to the state-of-the-art, while providing plenoptic enhancement to both of MPEG's compression standards. Finally, we have also proposed a lossy intra-frame coder of the geometry of voxelized point clouds that is based on an alternative method of representing the point cloud coordinates. By representing the point cloud as an array of binary images, this approach provides competitive results in comparison to state-of-the-art intra coders while using existing processing methods for bi-level images to compress the information directly in 3-D space.

RESUMO

Nuvens de pontos plenópticas (PPC) são estruturas que fornecem um maior grau de realismo às nuvens de pontos convencionais, representando a luz por diferentes direções de visualização. Isto é alcançado associando cada ponto a várias cores ao invés de única só. Neste trabalho, apresentam-se duas abordagens diferentes para comprimir eficientemente os N_c atributos plenópticos de uma PPC que estão em conformidade com as estratégias de codificação existentes do MPEG para compressão de nuvens de pontos (PCC). A primeira é um método baseado em geometria, que consiste em uma transformação de Karhunen-Loève sobre os atributos de cor seguida por múltiplos codificadores de atributos com capacidade de predição intra-quadro. Este esquema de compressão pode ser incorporado dentro do padrão de PCC do MPEG baseado na geometria (G-PCC), utilizando qualquer uma das soluções existentes do G-PCC para codificação de atributos. O segundo é uma abordagem baseada em vídeo que é retrocompatível com o padrão de PCC baseado em vídeo do

MPEG (V-PCC) e é baseado em uma transformação a nível de pixel dos N_c atlas plenópticos com uma DCT de N_c pontos. Os N_c atlas são codificados por um codec de vídeo convencional e são transmitidos como informação adicional para o codec convencional do V-PCC. A avaliação do desempenho da compressão usando PPCs de diferentes resoluções revela resultados competitivos em comparação com o estado da arte, ao mesmo tempo em que introduz uma capacidade de incorporar suporte a sequências plenópticas aos dois padrões de compressão do MPEG. Finalmente, foi também proposto um codificador intra-quadro da geometria, com perdas, de nuvens de pontos que é baseado em um método alternativo de representação das coordenadas dessas estruturas. Ao representar a nuvem de pontos como um conjunto de imagens binárias, esta abordagem fornece resultados competitivos em comparação com os codificadores intra-quadro do estado da arte, enquanto que usa métodos de processamento existentes para imagens binárias para comprimir a informação diretamente no espaço 3-D.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Definitions

D1 metric	Point-to-point metric. In this work, we calculate the final D1 metric as the maximum between $D1_O$ and $D1_D$. Where the first measurement relates to the omission of correct points in the degraded version and the second one with the excess of incorrect points.
D2 metric	Point-to-plane metric. Accounts for the perceived surface distortions. It is the D1 metric projected along the normal direction of each evaluated point.
octree	A tree data structure in which each node has exactly zero or eight children.
pixel	Picture element.
L-depth pointcloud	A point cloud constrained within a grid of $N \times N \times N$ voxels, where $N = 2^L$
trisoup	Surface approximation method used in TMC13.
voxel	Volume element.

Acronyms and Symbols

2-D	Two-dimensions
3-D	Three-dimensions
AC	Alternating Current, represents coefficients with non-zero frequency
AR	Augmented Reality
bpov	Bits per occupied voxel
CfP	Call for Proposals for Point Cloud Compression
CIE	International Commission on Illumination (Commission Internationale de l'Éclairage)
CTC	Common Test Conditions
DC	Direct Current, represents coefficients with zero frequency
DCT	Discrete Cosine Transform
DCM	Direct Coding Mode
G-PCC	Geometry-based Point Cloud Compression
HDTV	High Definition Television
HEVC	High Efficiency Video Encoding
HVS	Human Visual System
KLT	Karhunen-Loève transform
LiDAR	Light Detection And Ranging sensor

LoD	Level of Detail
L-PCC	LIDAR Point Cloud Compression
MPEG	Moving Pictures Expert Group
MSE	Mean-Squared Error
MV-HEVC	Multiview High Efficiency Video Coding
NNI	Nearest Neighbor Interpolation
PCC	Point Cloud Compression
PPC	Plenoptic Point Cloud
PSNR	Peak Signal-to-Noise Ratio
RAHT	Region-Adaptive Hierarchical Transform
RD	Rate-distortion
RGB	Red, Blue and Green color space
SDTV	Standard Definition Television
SE	Structuring Element
S-PCC	Surface Point Cloud Compression
TMC13	Test Model Categories 1 and 3
TMC2	Test Model Category 2
V-PCC	Video-based Point Cloud Compression
V-PPCC	Video-based Plenoptic Point Cloud Compressor
VR	Virtual Reality
YUV	Luma, Chrominance blue, and Chrominance red color space, following the BT.601 HDTV standard

1 INTRODUCTION

The emergence of technologies that simplify the capture of 3-D objects have increased the demand for 3-D applications, such as augmented/virtual reality and 3-D telepresence [1]. In this context, point clouds are important data structures for VR/AR applications, as it provides a means of representing objects with complex geometry in a flexible manner. A point cloud is a data structure that represents objects and space in a 3-D coordinate system. It is common to voxelize its points by constraining them to a grid of $N \times N \times N$ voxels [2]. Although PCs have been favored over meshes due to the greater simplicity in its capture process, there is a huge amount of data required to represent these data structures. Thus, the development of efficient algorithms for storing and transmitting high-quality point clouds is paramount to the feasibility of 3-D applications involving them.

With this increasing interest in immersive content, the Moving Pictures Experts Group (MPEG) issued a Call for Proposals (CfP) for the compression of point clouds in 2017 [3]. With the support of multiple technology companies and research institutions, MPEG proposed two coding strategies for point cloud compression (PCC): the Video-based Point Cloud Compression (V-PCC) and the Geometry-based Point Cloud Compression (G-PCC). The V-PCC strategy converts the 3-D point clouds into 2-D projections and leverages existent video codecs in order to efficiently compress the point cloud's geometry and attributes. Thus, this approach makes it suitable for compressing dynamic point clouds, *i.e.*, moving objects consisting of several point cloud frames. On the other hand, the G-PCC approach compresses the point cloud content directly in the 3-D space, and it was developed specifically to compress static point clouds and dynamically acquired ones — those acquired with LiDAR sensors.

Although the use cases of interest for MPEG include immersive applications, both G-PCC and V-PCC current implementations in their reference softwares TMC13 and TMC2, respectively, favor the compression of point clouds where each voxel is associated to a single color. For AR/VR applications this is equivalent to representing the objects as completely diffuse sources of light, which is not a completely realistic representation in most cases. This is especially prominent when working with specular surfaces. Therefore, enabling a more realistic representation of the point cloud attributes from different viewpoints is essential to provide a more immersive experience. Thus, developing a method to efficiently compress point clouds with directional color information is essential, which is commonly done by representing the plenoptic function [4].

Although several efforts have been made to compress the representation of the plenoptic function for point clouds [5]–[11], existing solutions from the literature either do not provide full compatibility with MPEG's existing standards or are not competitive with the state-of-the-art. Therefore, we propose two solutions to compress plenoptic point clouds (PPC) — point clouds that represent the plenoptic function by associating each voxel to multiple colors instead of a single one — that are both efficient and compliant with TMC2 and TMC13 reference softwares. In addition

to that, we also introduce an alternative means of compressing the point cloud geometry with losses that is competitive with MPEG's strategies while providing a different manner of interpreting the point cloud content.

This work is organized as follows: in Chapter 2, a review of the basic concepts required to understand this work are presented. Then, Chapter 3 provides a overview of the literature concerning the topics covered in this research. After that, Chapter 4 describes the methodology and presents the experiments performed to our solution for encoding with losses the point cloud geometry. Subsequently, Chapter 5 presents the methods and its associated experiments that were carried in order to assess our proposals for encoding the plenoptic attributes of a point cloud. Finally, in Chapter 6 conclusions and suggestions for future work are presented.

1.1 PUBLICATIONS

Publications related to this manuscript are as follows:

- Freitas, D. R.; Peixoto, E.; De Queiroz, R. L. e Medeiros, E. "Lossy Point Cloud Geometry Compression Via Dyadic Decomposition." *2020 IEEE International Conference on Image Processing (ICIP)*, p.2731, 2020, Abu Dhabi.
- Freitas, D. R.; Peixoto, E.; De Queiroz, R. L. e Medeiros, E. "Estudo de Perdas para Codificação de Geometria de Nuvens de Pontos por Decomposição Diádica." *XXXVIII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT)*, 2020, Florianópolis.

Submitted publications under review:

- Garcia, D. C.; Dorea, C.; Ferreira, R. U.; Freitas, D. R.; De Queiroz, R. L.; Higa, R.; Seidel, I.; Testoni, V. "Differential Transform for Video-based Plenoptic Point Cloud Coding". Submitted to: *IEEE Transactions on Image Processing, 2020*

2 BACKGROUND

2.1 POINT CLOUDS

2.1.1 Definition

The term *three-dimensional* (3-D) computer graphics system is used to emphasize that the graphics are generated by an internal representation of a 3-D world, which is then rendered into a *two-dimensional* (2-D) image [12]. This rendering stage consists generally of a simulation of physical processes that occur in the real world, *e.g.* illumination. In order to provide a compact and organized representation of the 3-D scenes, these graphic systems are often represented by global coordinate systems, in which the basic geometric unit is a 3-D point. Thus, this point is typically represented by a three element tuple (x, y, z), corresponding to its x, y and z coordinates.

A *point cloud* (PC) is a volumetric structure produced by a 3-D surface scanning method over objects and typically represent only its surfaces, *i.e.* it does not contain data of the objects' internal features [13]. An example is shown in Figure 2.1. These scattered point sets are often used due to its ease of measure, encoding and visualization.

Figure 2.1: The *Longdress* point cloud from the 8i Voxelized Full Bodies Dataset [14].

The representation of each point in a PC consists of a 3-D location and a list of K attributes. For a PC of M points, the geometry information can be represented as

$$V = \{(x_m, y_m, z_m)\}, m = 1, ..., M, \tag{2.1}$$

while the attribute information can be expressed by a list of attributes

$$A = \{(a_{m,1}, a_{m,2}, ..., a_{m,K})\}, m = 1, ..., M. \tag{2.2}$$

The PC attributes often refer to the color information in the *RGB* space, where the $a_{m,1...3}$ values in Eq. 2.2 — for $K = 3$ — correspond to the (r_m, g_m, b_m) color for the m-th point. The surface normals and the reflectance of the object are other attributes that are often represented in these structures.

In order to provide a more efficient means of representation and processing, it is common to constrain the points to a grid of $N \times N \times N$ elements in a process known as voxelization. Each point in this procedure is quantized to a cube with dimensions $1 \times 1 \times 1$ that occupies an integer position in space. Each of these cubes is called a *volume element*, also known as a voxel. Generally, the value that is chosen for N is a power of 2. Hence, a PC within a grid of N voxels is commonly called a point cloud of depth-L, where $N = 2^L, L \in \mathbb{N}$. This quantization process is of general interest because of the similarity with how 2-D images are represented. However, one significant difference between voxels and its two-dimensional counterparts is that all pixels in a image are occupied, which is not true for voxelized point clouds.

2.1.2 Applications

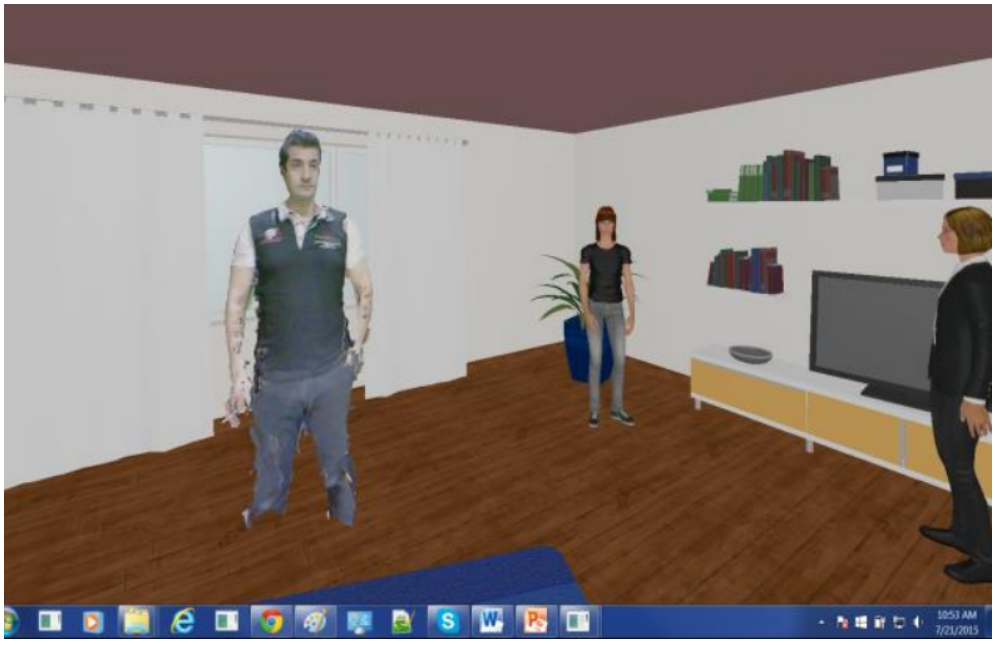


Figure 2.2: Example of an use case of point clouds: telepresence [15].

As it was mentioned before, PCs have been favored over other 3-D structures due to the greater simplicity of its capture process and ease of representation, which makes it suitable for applications of real-time rendering and capture of objects. In particular, some of the applications targeted for point cloud use include real-time 3-D telepresence (Fig. 2.2), *virtual reality* (VR), free viewpoint

sport replays broadcasting, cultural heritage and autonomous navigation [1].

In the light of this wide range of applications, MPEG proposed the classification of point clouds in three different categories, based on its temporal features: static, dynamic and dynamically acquired PCs [3]. Static point clouds, also known as PCs of category 1, are generally represented by the capture of a single frame of structures with no temporal variation, *e.g.* buildings and scenes. Dynamic (or category 2) PCs represent structures that contain temporal information - typically movement - and are represented through several frames. Lastly, dynamically acquired ones are also known as category 3 PCs and the temporal information is comprised in the changes of the sensor position itself in the process of capture of its surroundings. Hence, category 3 point clouds are often used for autonomous navigation and are captured by LiDAR technology. Examples from these three types of PCs are shown in Fig. 2.3.

Point clouds from categories 1 and 2 are typically captured by an array of stereo cameras. This method is referred as passive acquisition because the depth of the 3-D scene derives from the disparity between the multiple cameras. In addition to these, a variety of sensors can also be used in order to directly provide the depth information of the scene, such as infrared depth cameras. This method is also known as active acquisition. For point clouds of category 3, LiDAR sensors are often employed for mapping of outdoor environments, by measuring the reflections of the light emitted from the scanner.

2.2 VOLUME VISUALIZATION

Volume visualization is a field of study that concerns the extraction of information from volumetric data, *i.e.*, it is used to generate 2-D graphical representations from 3-D volumes [12]. As a means to capture this information, some areas of concern include the study of methods for volume data representation, modeling, manipulation and rendering. While many techniques have already been developed to render 3-D data, the selection of a specific one depends on the nature and the purpose of the application. For example, an application that requires maximum visual quality has to afford rendering techniques that are more computationally expensive. On the other hand, real-time applications - as is the case for this dissertation - cannot afford to benefit from such expensive methods, which results in rendering strategies that prioritize performance in compromise of visual fidelity.

2.2.1 Color Spaces

Colors spaces are multidimensional models that define an explicit range of colors and luminance values, where each individual color can be represented as points in a specific coordinate system. In addition to that, a multidimensional system means that each of the coordinates can be treated independently from the other ones. Typically, each one of these dimensions represent either a color channel or a component. Thus, the definition of a specific color space is essential in order to prop-

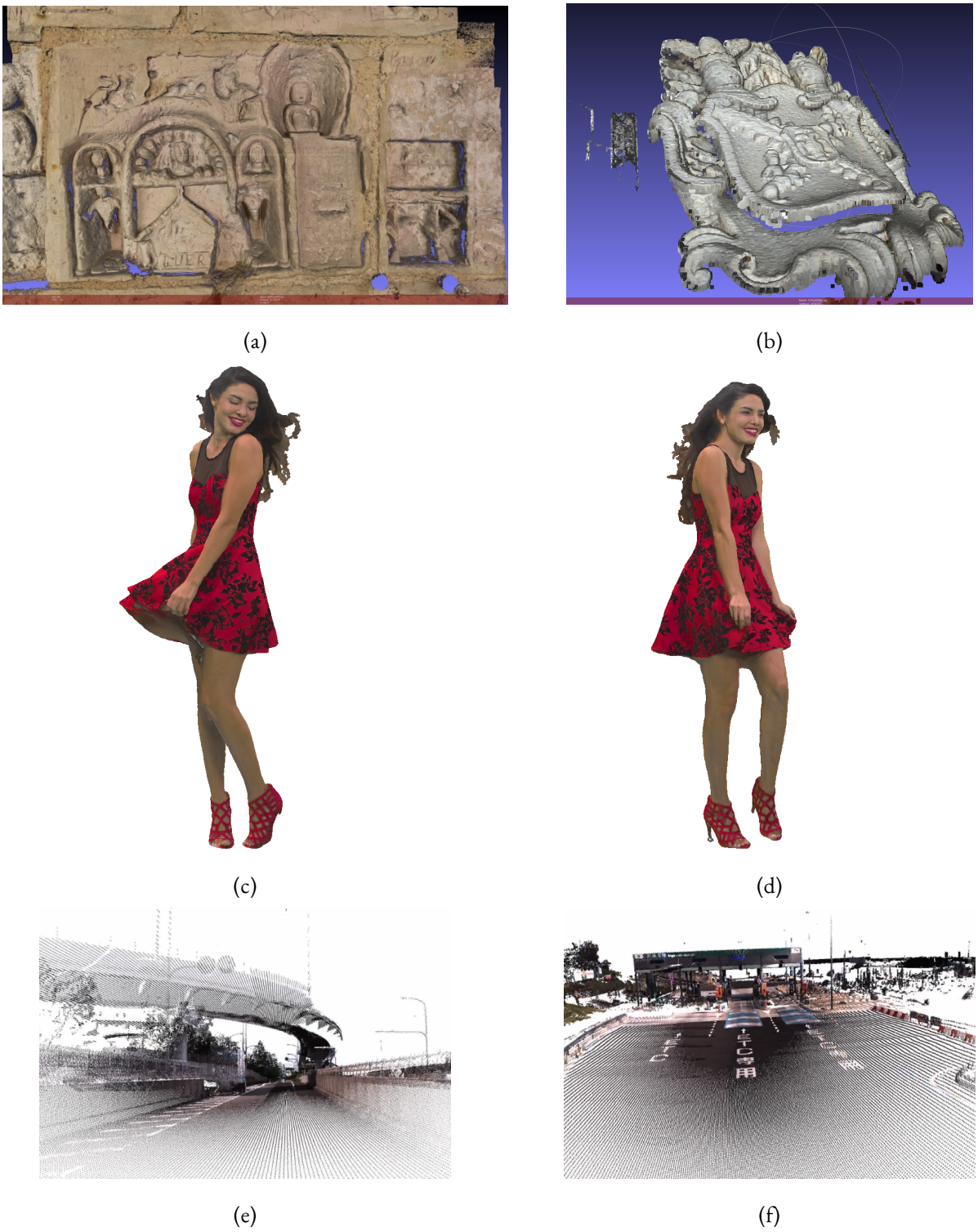


Figure 2.3: Examples of point clouds for a), b) Category 1 [16]; c), d) Category 2 [14]; e), f) Category 3 [17].

erly convey the color information for a given image (2-D) or point cloud (3-D) data.

In 1931, the Commission Internationale de l'Eclairage (CIE) conducted experiments in order to model a color space that aimed to encompass the colors perceived by the human visual system (HVS). Based on the trichromatic theory from the HVS, this experiment attempted to recreate the

different colors from the visible spectrum by using a combination of amounts of red, green and blue light. These results were summarized by defining standardized values for the red, green and blue colors based on wavelengths: 700.00nm, 546.1nm and 435.8nm, respectively. With that in mind, the RGB space defines colors by red, green and blue coordinates with the goal of mimicking the additive mix of these three primary colors. This means that the absence of these three colors replicates the absence of light, *i.e.* represents black, while the maximum amount for these three components represent white. The RGB values are often comprised in the range $[0, 255]$ for each dimension in systems with 8-bit depth integer representation.

The YUV color space is an alternative representation to the popular RGB space where the Y channel represents exclusively the luminance component and the U and V channels represent the lower frequency blue and red chrominance channels, respectively. When a gamma correction is applied to Y, this component is also called luma - represented by Y′. The importance of the conversion from RGB to YUV space lies on the human's eye higher sensitivity to contrast changes in brightness (luminance) in comparison to changes in color. Thus, separating the luminance from the chrominance channels allows the latter to be further compressed without resulting in significant perceptual loss in visual quality. It follows that the YUV color space is not only suited for image (and PC) compression, but it is also typically used for point cloud assessment of the color information. The conversion from the RGB to YUV space employed throughout this work follows the BT.601 standard adopted by SDTV [18] - as there's also the BT.709 equations which relate to HDTV [19] -, which in matrix form can be represented by:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 0.502 \\ 0.502 \end{bmatrix} \tag{2.3}$$

2.3 IMAGE PROCESSING

An essential design approach when working with point cloud data is whether its content is analyzed and processed directly in 3-D space (*i.e.*, Section 3.2.2) or if the methods are applied over 2-D structures, by taking some kind of PC's projections (Section 3.2.1) or representing it in some alternative way (Section 3.1.1.2), for example. While both of these approaches are utilized in this work, this Section presents a brief explanation of some of the processing techniques used to improve the perceived quality of the point cloud information presented in a 2-D format.

2.3.1 Downsampling and Upsampling

The downsampling of an input image is often applied in situations where computer resources are limited and/or transmissions savings are required by reducing the spatial resolution of the respective frame. Alternatively, the upsampling of an image is a commonly used method for im-

proving the perceived quality of the data by increasing its spatial resolution. Such resampling tasks generate newly unobserved locations with unknown values that have to be estimated. Thus, these values are typically estimated by using known data, which is referred as interpolation.

In this work, the nearest neighbor interpolation (NNI) is the method of choice for estimating these unknown values. The NNI was selected over other resampling methods (*e.g.*, linear and bicubic interpolations) because the image processing techniques are being applied over 2-D images that represent geometry, which means that each pixel has a value of 1 (if the voxel is occupied) or 0 (in case it is unoccupied). Thus, methods that use the average of multiple pixels in its computation are not optimal when working with binary data. In the case of the NNI, its interpolation algorithm only takes one pixel into consideration, which is the nearest point to the interpolated location. That is, consider that s is the scaling factor between the resampled and the original images $\mathbf{g}$ and $\mathbf{f}$ respectively, so that $s > 1$ implies upsampling and $s < 1$ represents downsampling. Therefore, the nearest neighbor algorithm is defined for each pixel of coordinates (x, y) by

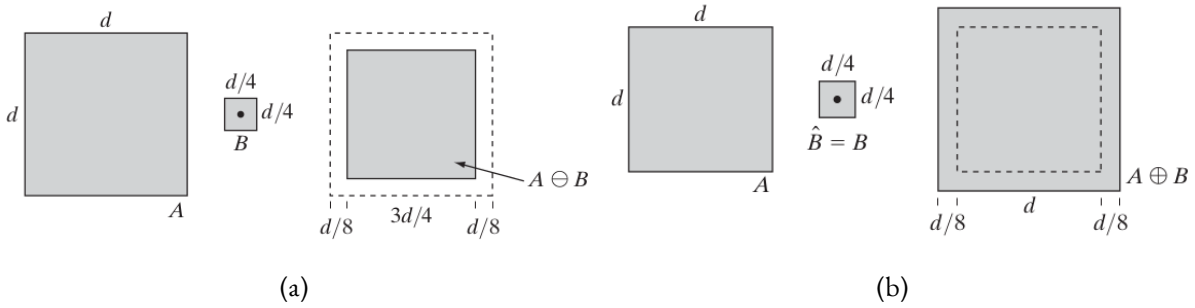
$$\mathbf{g}(x, y) = \mathbf{f}(\lceil \frac{x}{s} \rceil, \lceil \frac{y}{s} \rceil). \tag{2.4}$$

2.3.2 Morphological Transformations

Morphological transformations are techniques commonly used for image pre- and post-processing for changing the morphology of features in a given frame, hence its name. This is substantiated by probing an image for specified patterns according to small shapes called structuring elements (SE). Thus, these operations process an image relative to the ordering of its pixel values, which make them suitable for operations with binary images and, consequently, processing geometry information [20]. The two basic operations that provide the building blocks for achieving other morphological transformations are erosion and dilation.

Consider a SE B to be applied over an image A. The erosion operation of A by B is represented by $A \ominus B$ and can be seen as taking from A the pixels whose neighborhood fully contains B, otherwise they are removed. This is illustrated in Fig. 2.4-a). On the other hand, the dilation operation of A by B is represented by $A \oplus B$ by doing the opposite of the erosion. That is, it can be seen as taking the pixels where its neighborhood and B overlap by at least one pixel, *i.e.*, the union between each pixel's neighborhood and B. This is depicted in Fig. 2.4-b).

The opening of an image is A by a structuring element B is defined by $A \circ B = (A \ominus B) \oplus B$. In other words, it is the erosion of A by B followed by a dilation by B. A closing operation is defined by the dilation of A by B followed by a erosion by B, or $A \bullet B = (A \oplus B) \ominus B$.



(a) (b)

Figure 2.4: a) Erosion and b) Dilation operations for an image of dimensions $d \times d$ by a SE of dimensions $\frac{d}{4} \times \frac{d}{4}$ [20]

2.4 TRANSFORMS

As it was introduced in Section 2.2.1 with the usage of different color spaces like RGB and YUV, the pursuit to represent the information through different points of view is rather relevant in the field of signal processing. This is due to the fact that although the same information is present, looking at the data from different "vantage points" allows the detection of new correlations. That is, a better understanding of the characteristics of the signal allows the separation of the different components that constitute it and highlights the relationship between them.

In this context, the usage of transforms like the Fourier transform are often used in order to convert the information to another domain, which typifies the aforementioned change of way of viewing the data. In the scope of data compression, this allows the identification of the essential components of the signal in order to transmit the data without losses while providing transmission savings. For further savings, the data can also be compressed in a lossy manner, where the usage of transforms allows the detection of the least important components of the processed signal so that they can be discarded. This provides reduced costs to the transmission while presenting minimal degradation to the data conveyed.

2.4.1 Discrete Cosine Transform

The Discrete Cosine Transform (DCT) [21] is a widely used linear transform in the field signal processing due to its capability of concentrating a large part of the transform's energy in a few low-frequency coefficients, also known as energy compaction. Using cosines waveforms at increasing frequencies as orthogonal basis functions, it can be represented as a real matrix $\mathbf{M}$ $N \times N$ where

$$M_{i,j} = \begin{cases} \sqrt{\frac{1}{N}}, & \text{for } i = 0 \\ \sqrt{\frac{2}{N}} \cos\left(\frac{\pi(j+\frac{1}{2})i}{N}\right), & \text{for } i > 0. \end{cases} \tag{2.5}$$

For a vector signal $\mathbf{x}(n)$ in $\mathbb{R}^N$, the transformed signal $\mathbf{t}$(n) is obtained by $\mathbf{t}(n) = \mathbf{M}\mathbf{x}(n)$.

2.4.2 Karhunen-Loève Transform

The Karhunen-Loève Transform (KLT) [22] is a linear transform where the basis functions are taken from the statistical properties of the given signal. Thus, this adaptive transform maximizes the energy in as few coefficients as possible, *i.e.*, it produces optimal energy compaction. Consider an input signal $\mathbf{x}(n) = [X_n^1, X_n^2, X_n^3, \cdots, X_n^N]^T$. The KLT is defined as the transform that diagonalizes the autocovariance matrix of $\mathbf{x}(n)$ in discrete time. That is, a $N \times N$ KLT matrix $\mathbf{H}$ is an orthogonal matrix made of the eigenvectors of the covariance matrix $\mathbf{K}_{cc}$, which can be described as

$$K_{i,j} = \{E[(X^i - E[X^i])(X^j - E[X^j])]\} \tag{2.6}$$

Note that for zero mean processes, the Eq. 2.6 is the same as the autocorrelation matrix. However, the KLT is data dependent, which means that its transform matrix does not have a way of being pre-computed as the DCT, for example. Therefore, KLT algorithms have a higher complexity than other transform algorithms, as it requires more computational resources.

3 LITERATURE REVIEW

3.1 POINT CLOUD ENCODING

Although point clouds are easier to capture and more flexible to represent in comparison to other structures such as polygonal meshes [23], its significant data volume urges the development of efficient compression algorithms to convey both the geometry and the attributes. In the context of point cloud encoding, the geometry and the attributes are typically manipulated using separate methods.

As it was mentioned in Section 2.2, different manners of representing the same data can provide a greater emphasis on the information which is perceivable by the HVS. While this concept was introduced when talking about attributes of a PC, the same can be idea can be applied for the geometry of these structures. Ideally, it's desirable for a given representation to provide not only a means to fully reconstruct the transmitted data without loss of information (lossless compression), but to also enable the discard of information at with different levels of distortion while providing minimal degradation to the reconstructed data (lossy compression), which allows for further savings through the data transmission.

3.1.1 Geometry Encoding

3.1.1.1 Octrees

A great number of existing geometry encoder use the octree representation [24]–[26] as means to structure point clouds for its compression algorithms. Octrees are specific tree structures where each node has exactly zero or eight children. In general, the initial node of the octree (also called root) spans the entire point cloud space, which is recursively subdivided into eight child nodes. This is due to the fact that the node is partitioned into two subnodes for each dimension in the 3-D space, x, y and z. This recursive process is repeated until it reaches a leaf (where the node has no children) or when it reaches the voxel level, where each node has dimensions of $1 \times 1 \times 1$.

As an illustration, the Fig. 3.1(a) depicts the entire point cloud space represented as one node. Assuming its subdivision in one dimension, *i.e.*, x, Fig. 3.1(b) shows the two generated subnodes in that specific direction. Subsequently, Figs. 3.1(c) and 3.1(d) show the partition in directions y and z, generating the eight subnodes that constitute one depth level. As aforementioned, this process is recursive for each depth level, which is depicted in Fig. 3.1(e).

In terms of point cloud encoding, octrees can efficiently represent the point cloud space by traversing it from its root, signaling the occupied nodes with a single bit. Therefore, a bit 1 is assigned to a node if it bounds at least one occupied voxel among its children, and a bit 0 otherwise. The nodes that are signaled with a bit 0 - also known as leaves or empty nodes -, do not need to be

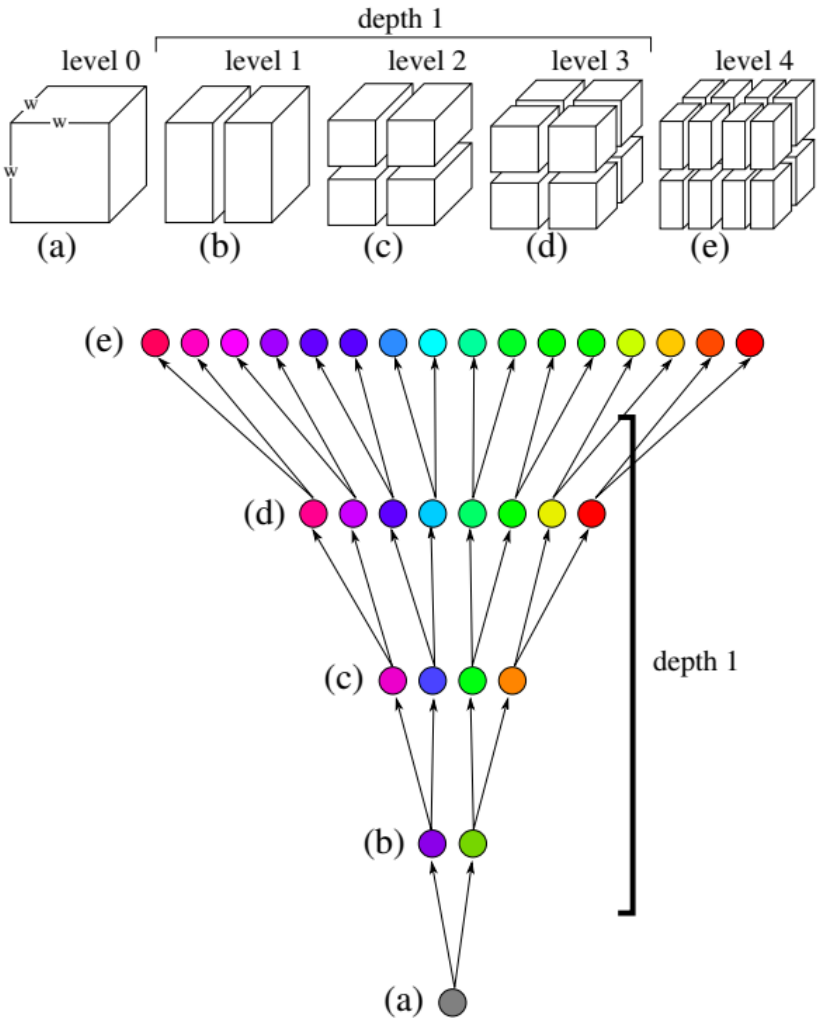


Figure 3.1: Octree scanning per direction [27].

further partitioned, as they do not contain additional information for the point cloud space. The bits signaled with a bit 1 are subdivided into its eight children, where the bits assignment process is repeated for each child. This encoding process is illustrated in Fig. 3.2. Note that each depth k of the octree can be represented by k bytes, where the n-th bit at the byte k indicates if the n-th node at depth k is occupied or not.

3.1.1.2 Alternative Representations

Although representing PCs by octrees is a popular approach, solutions that use different methods of representation have also been introduced, which includes modelling of a point cloud using 3-D space curves [28], encoding the geometry using binary tree partitioning [29] and representing the PC geometry using a reversible cellular automata block transform [30].

In 2020, an alternative representation for point cloud geometry using dyadic decomposition

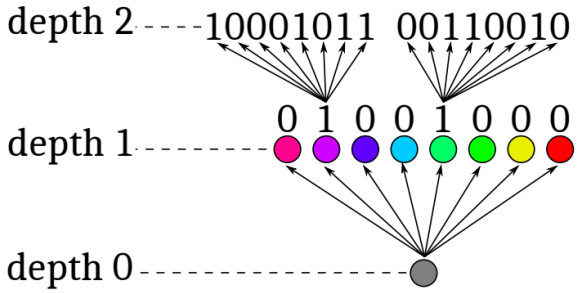


Figure 3.2: Example of geometry encoding of a PC using octree signaling [27].

was proposed [31]. In this algorithm, a point cloud of dimensions $N \times N \times N$ is represented as a 3-D boolean occupancy array $\mathbf{G}(x, y, z)$. The principle of this representation is to slice $\mathbf{G}$ recursively along a chosen axis of interest. At each step of this procedure, an $N \times N$ slice is generated by projecting all the occupied voxels in the region being processed throughout the chosen axis, which is also known as a silhouette image. In other words, the point cloud can be seen as a sequence of silhouette images in a 3-D cube.

The recursive part of this method consists of splitting an interval of slices in two smaller equal intervals in a binary tree traversal. Each node of this dyadic binary tree holds a silhouette image of the entire 3-D region it spans. The main advantage of this silhouette representation is that from any given silhouette image represented by a node, all blank pixels are empty because they were empty for all the slices that span the specific region through the given axis. Thus, it infers that all subnodes from that node will have at least these same unoccupied voxels. Fig. 3.3 illustrates an example where instead of transmitting the entire $\mathbf{A}$ and $\mathbf{B}$ images, only the marked symbols from $\mathbf{Y}$, $\mathbf{Y_A}$ and $\mathbf{Y_B}$ are conveyed. These three images are defined by:

$$\mathbf{Y} = \mathbf{A} + \mathbf{B} \tag{3.1}$$

$$\mathbf{Y_A} = \mathbf{Y} \oplus \mathbf{A} \tag{3.2}$$

$$\mathbf{Y_B} = \mathbf{Y} \oplus \mathbf{B} \tag{3.3}$$

where the symbols + and $\oplus$ stand for the boolean operations of OR and XOR, respectively.

Then, the algorithm uses both the similarities between a given node and its parent to mask part of the information that is conveyed, as well as it uses information of the neighbor nodes for building contexts to a binary arithmetic coder [32]. Note that the parent node can also be seen as the $\mathbf{Y}$ image from Eq. 3.1, while $\mathbf{A}$ and $\mathbf{B}$ can be interpreted as the child nodes.

Finally, the dyadic decomposition algorithm also proposed another encoding approach for the slices close to the leaves of the tree. With the addition of a parameter $P \in [1, N]$, the algorithm follows with the dyadic slicing until the processing region comprises a interval of $k <= P$ slices. From this point on, the algorithm also tests another encoding method, called single mode encoding, which consists of transmitting all the k remaining slices as leaves of the tree instead of the

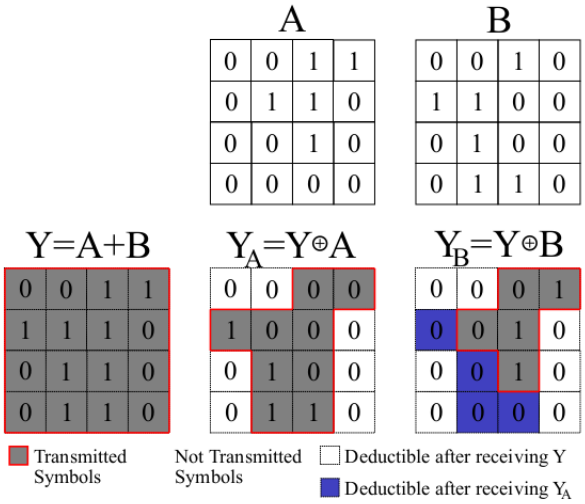


Figure 3.3: Example of a boolean composition of binary images A and B [32].

recursive dyadic partitioning, using the silhouette for this interval as a mask for transmitting each leaf. Then, the algorithm chooses whoever method - between the dyadic decomposition and single mode encoding - that yields the lowest bitrate.

3.1.2 Attribute Encoding

3.1.2.1 Region-Adaptive Hierarchical Transform

The Region-Adaptive Hierarchical (or Haar) Transform is a hierarchical orthogonal sub-band transform [25], [33]. Considered an adaptive variation of the Haar transform, it takes advantage of the point cloud data's sparsity in order to compress attributes, such as color or reflectance. Its basic principle is to follow a backwards direction for the traversal of the octree, *i.e.* from the leaves to the root, to combine the voxels in each of the (x, y, z) dimensions into larger cubes until it spans the entire point cloud space (the root).

The algorithm work as illustrated in Fig. 3.4 for a $2 \times 2 \times 2$ node. While occupied voxels are assigned with weight $\omega = 1$, unoccupied ones are signaled with $\omega = 0$. Therefore, the weight of a given node represents the number of leaf voxels in its sub-tree. The attribute $A^k_{x,y,z}$ from a voxel at a k-th level is obtained by applying a linear transformation [25] over the attributes from the voxels of the same branch on the $k+1$-th level for each dimension. Considering it for one dimension, *e.g.* x, these attributes are combined through

$$\begin{bmatrix} A^k_{x,y,z} \\ H^k_{x,y,z} \end{bmatrix} = T_{(\omega_1,\omega_2)} \begin{bmatrix} A^{k+1}_{2x,y,z} \\ A^{k+1}_{2x+1,y,z} \end{bmatrix}, \tag{3.4}$$

where

$$T_{(\omega_1,\omega_2)} = \frac{1}{\sqrt{\omega_1 + \sqrt{\omega_2}}} \begin{bmatrix} \sqrt{\omega_1} & \omega_2 \\ -\sqrt{\omega_2} & \sqrt{\omega_1} \end{bmatrix}. \tag{3.5}$$

In other words, the attribute for a voxel at level k is given by a weighted average of the attributes $A^{k+1}_{2x,y,z}$ and $A^{k+1}_{2x+1,y,z}$ at level $k+1$ with weights ω_1 and ω_2, respectively. This generates one low-pass ($A^{k}_{x,y,z}$) and one high-pass ($H^{k}_{x,y,z}$) coefficients. While the low-pass coefficient is passed to the following level of the octree with its updated weight, the high-pass one is quantized and encoded. Note from Eqs. (3.4) and (3.5) that if either of the higher level voxels are unoccupied, the other one is passed to the next level with its weight unchanged. Finally, the low-pass coefficient of the tree root $A^{0}_{0,0,0}$ is also encoded alongside the high-pass coefficients generated throughout the recursive process.

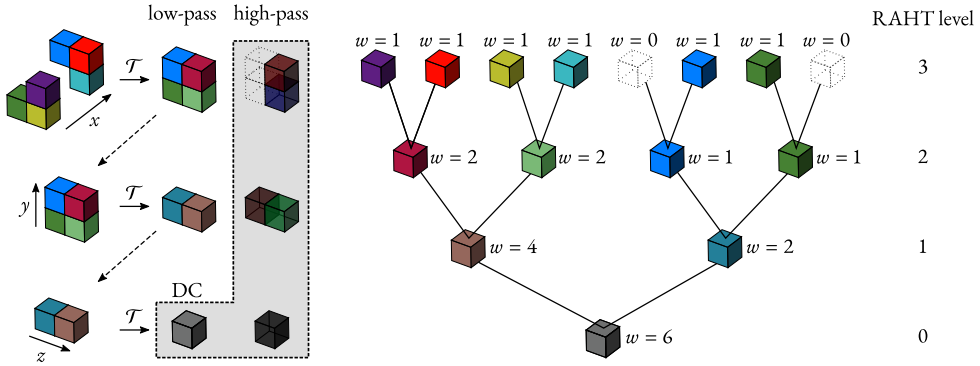


Figure 3.4: RAHT procedure for a $2 \times 2 \times 2$ node [34].

3.1.2.2 Predict/Lifting

The Predict Transform is a *Level of Details* (LoD)-based coder that re-organizes the geometry information of the PC by level sets of refinement (R) in the LoD generation, which is based on point-to-point Euclidean distance thresholds [35]. This is shown in Fig. 3.5. It is considered a predictive scheme due to the attribute values of a given refinement level (R_ℓ) being obtained by the prediction of the attribute values of the k-nearest neighbors in the LoD for the previous level ($\text{LoD}_{\ell-1}$), where $\text{LoD}_\ell = \bigcup_{i=0}^{\ell} R_i$. The Predict Transform consists of two different schemes, known as forward and inverse. The main operator in the forward scheme is the split, where it takes L_ℓ - which is the set of attributes associated with LoD_ℓ - as an input and returns low-resolution $L_{\ell-1}$ and high-resolution $H_{\ell-1}$ outputs. In the inverse scheme, the merge operator does the opposite (hence its name): L_ℓ and H_ℓ are taken as inputs and $L_{\ell+1}$ is returned as output. Then, a weighted interpolation-based prediction of the color information is performed for the points in R_ℓ in order to predict H_ℓ. Subsequently, the residuals D_ℓ are quantized and arithmetically encoded in the order established by the LoD algorithm.

Further optimization of this method can be achieved with the Lifting Transform, which introduces an operator called update. Note from Fig. 3.6 that the lifting transform is built on top of

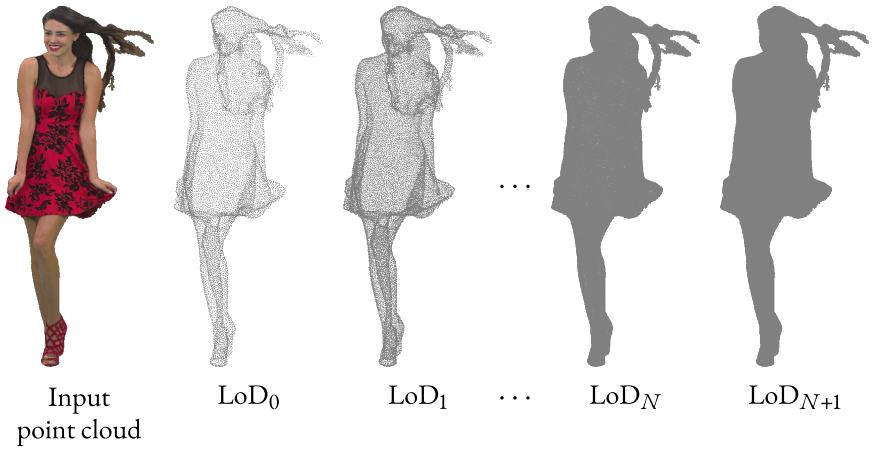


Figure 3.5: LoDs for the *Redandblack* sequence [34], [36].

the Predict Transform framework. As a result, the scheme which involves both transforms is often known as predlift. The update operator of the Lifting Transform consists of an inverse traversal of the order defined by the LoD structure, in which the attribute values of points in the LoD$_{\ell-1}$ are updated by the residual of the points in the LoD$_\ell$. Each of these points is associated with a weight value, which are used in order to make the points in lower LoDs more influential [37]. In addition to that, the lifting scheme uses the influence weights of each point to scale its residual in the quantization step, using a technique called adaptive quantization [37].

3.2 POINT CLOUD COMPRESSION STANDARDS

With the increasing accessibility of 3-D capturing technologies and the emergence of device capable of capturing point clouds - like Microsoft Kinect -, The MPEG 3-D graphics coding group (MPEG-3DG) first started discussions back in 2013 for possible use cases of point clouds like immersive telepresence [38]. At the time, existing MPEG standards for 3-D graphics were not suitable for applications that required low latency, and the increased interest from the industry on PCs prompted discussions about the compression of the huge amount of data required to represent these structures. In 2016, MPEG collected requirements [39] and, in the following year, started the point cloud compression (PCC) research process with a CfP [3].

First, three categories of point cloud content were identified based on the nature of acquisition and its temporal features, which was explained in Section 2.1.2. Subsequently, several compression technologies were proposed for each of these three categories, and one compression strategy was chosen for each type of dataset. The compression strategy for the static point clouds of category 1 was named Surface Point Cloud Compression (S-PCC); the dynamic content from category 2 was assigned as Video-based Point Cloud Compression; and finally, the dynamically acquired PCs from category 3 were studied under the name of LiDAR Point Cloud Compression (L-PCC) [40]. Later on, the similarities between the S-PCC and L-PCC proposals resulted in the merging of both strategies under the name Geometry-based Point Cloud Compression (G-PCC).

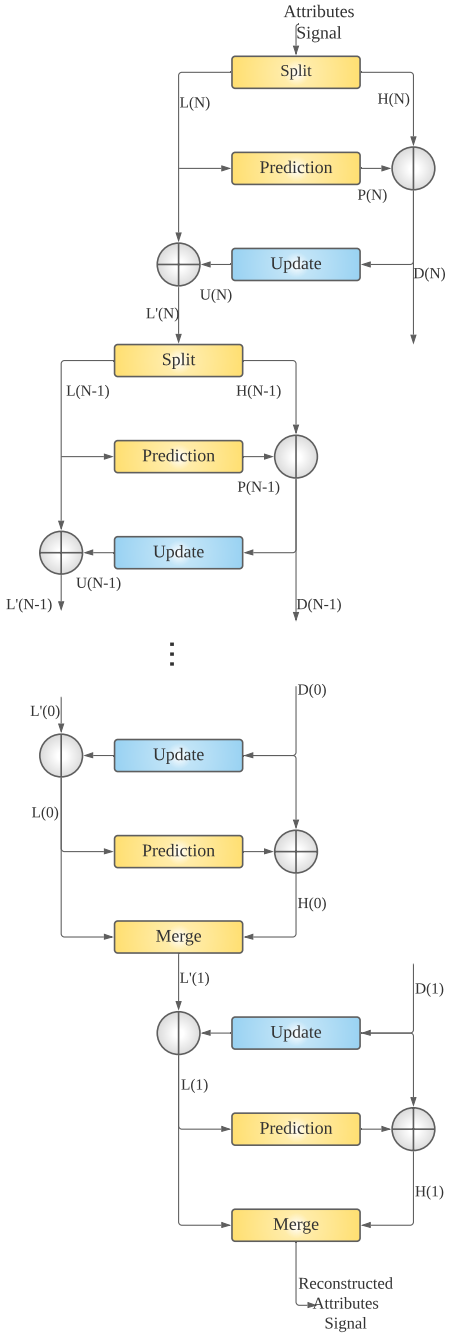


Figure 3.6: Forward (top) and Inverse (bottom) Predict/Lifting scheme.

In order to provide benchmark performance and a framework for experiments and comparisons of proposals, MPEG developed two reference software Test Model Category 2 (TMC2) and Test Model Categories 1 and 3 (TMC13) for the V-PCC and G-PCC compression standards, respectively [41].

3.2.1 Video-based Point Cloud Compression

The main idea behind the proposal for the V-PCC standard is to convert the point cloud data from 3-D to 2-D, in order to code the information with 2-D video encoders. As the compression of 2-D videos is a well-researched field, a projection-based solution could take advantage of the high performance and widespread adoption of existing video codecs - such as HEVC [42], VVC [43] and AVI [44] - for compressing both the geometry and the attribute information for dynamic point clouds. The proposal that was accepted as basis for the TMC2 software divides the point cloud by 3-D surface segments, known as 3-D patches. Then, each 3-D patch is projected into a 2-D patch, which is packed under a specific criteria into a set of 2-D mosaic-like frames for compression with standard video coders [45].

The 3-D patches are independently projected into two dimensions in order to reduce projection issues, such as self-occlusions and hidden surfaces [46]. The collection of patches projected in mosaic-like images, also known as atlas, can be compared to a capturing different parts of the PC using multiple virtual cameras. Each atlas representing a point cloud frame contains up to three associated images to be compressed by a conventional video codec: the occupancy map, which is a binary image that indicates which pixels from the atlas should be used; a geometry image with the depth information and a set of attribute images, which can represent the texture or the material, for example. This is illustrated in Fig. 3.7.

(a) Occupancy map (b) Geometry (a) Attribute

Figure 3.7: Atlases that compose V-PCC's main payload [34].

3.2.1.1 Coding Architecture

The main components of the V-PCC codec architecture consist of the patch generation and packing, atlas generation, image padding and video compression. In order to generate the 3-D patches, first the normal vector at every point has to be estimated [47]. A segmentation of the point cloud is then performed by quantizing the normals in the six orthographic projection directions $(\pm x, \pm y, \pm z)$, generating an initial clustering. Subsequently, this segmentation is refined by smoothing the clusters based on the nearest neighbors of a given point in addition to its normal vector. From these refined clusters, the patches are segmented with a connected components

algorithm [46]. This process is depicted in Fig. 3.8.

(a) (b)

Figure 3.8: a) Initial and b) Refined segmentation of points from a clustered point cloud.

The patch packing process is where the extracted patches are projected into a 2-D atlas with dimensions $W \times H$. The order in which the patches are fitted in the grid is based on their sizes and their location is determined by an iterative process. Using eight different orientations — where the patch can both be mirrored and be rotated in the four possible directions —, the first location that guarantees an overlap-free insertion is selected. In addition to that, the packing process also aims to be temporally consistent by trying to maintain the placement for similar patches in consecutive frames, in order to improve the compression efficiency in the video compression.

One of the main concerns in the atlas generation procedure is to properly handle the case where multiple points are projected in the same pixel. The adopted strategy consists of projecting each patch onto two layers, known as near and far layers, which comprises the points with the lowest and highest depth relative to the projection plane, respectively [46]. Finally, in order to improve the compression efficiency for video coding, a process named padding is applied to the atlases to generate piece-wise smooth images. This procedure works by filling the empty space between the patches using a number of possible algorithms, *e.g.* iteratively filling with the average value of non-empty neighbors. The complete pipeline for the TMC2 scheme is illustrated in Fig. 3.9.

3.2.2 Geometry-based Point Cloud Compression

Originated from the merge of the L-PCC and S-PCC (see Section 3.2), the G-PCC targets the compression of point clouds of categories 1 and 3 by encoding its content directly in 3-D space. In contrast to V-PCC, which assumes that the input point cloud data is sparse, G-PCC makes no assumptions about the density of the data structure, which allows the compression of both static and dynamically acquired data. However, the G-PCC standard is currently only defined for intra-frame prediction, as it does not use any temporal prediction tools.

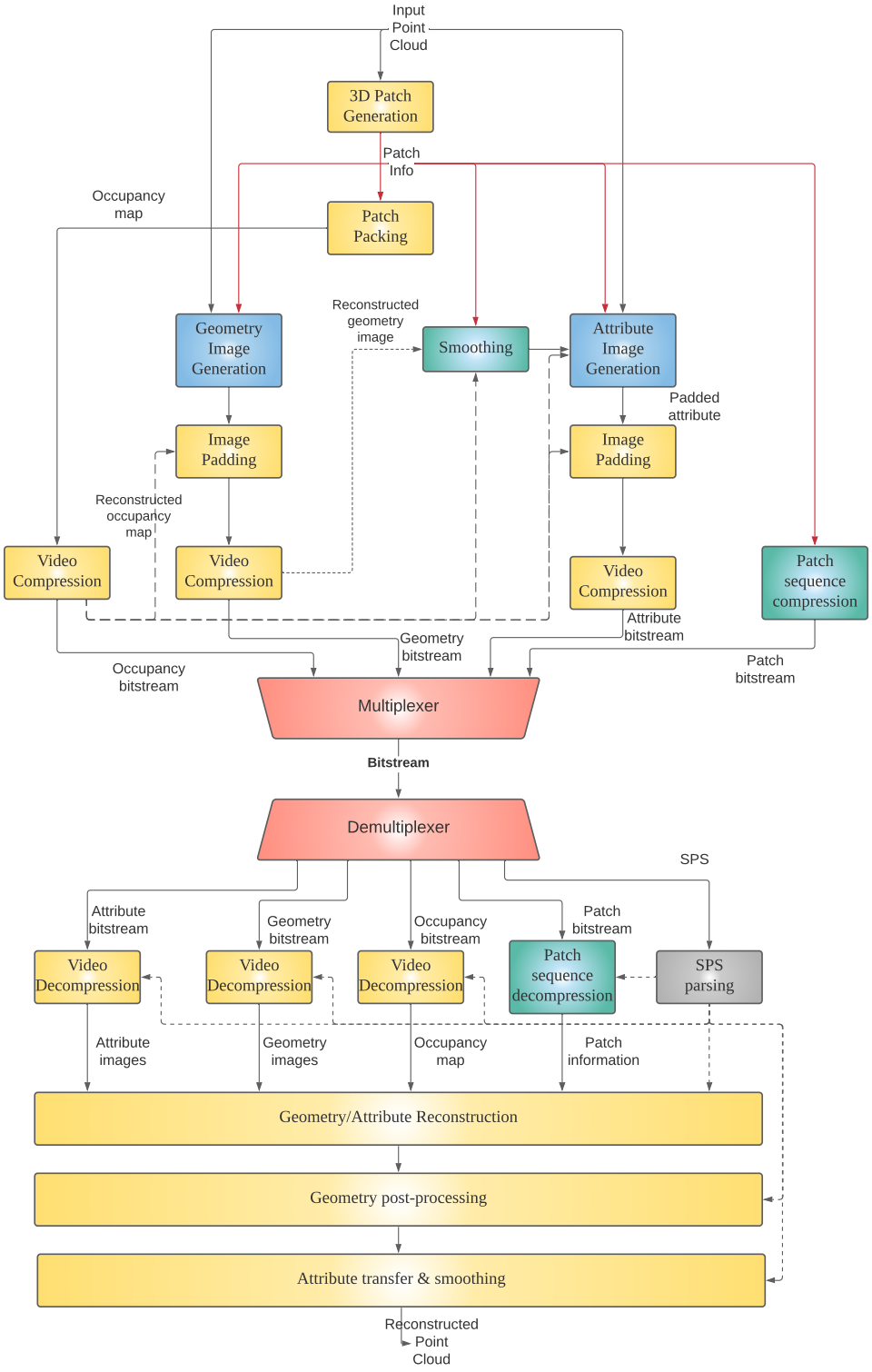


Figure 3.9: Encoder (top) and decoder (bottom) pipelines for the TMC2 [46].

3.2.2.1 Coding Architecture

The first step of the TMC13 coding scheme, which is depicted in Fig. 3.10, is a pre-processing procedure that consists of a coordinate transformation followed by voxelization. This is due to the fact that the input point cloud data can contain both integer or floating point values for its geometry representation. Then, the second step is based on using the octree or trisoup scheme for geometry analysis of the point cloud. The octree — which is explained in Section 3.1.1.1 — is compressed with an entropy coder with two different methods, depending on the neighboring octets. While the points in denser areas are entropy coded taking the correlation with neighboring octets in consideration [36], the isolated points are coded using the Direct Coding Mode (DCM), since there are no points to correlate with. In DCM, the points' coordinates are encoded without performing any compression. For the lossless compression of geometry, all the octets of the octree are used. Alternatively, with lossy compression the geometry is represented by pruning the octree from the root to an arbitrary level. This pruning procedure can optionally be combined with a method known as triangle soup (trisoup), which consists of approximating the object surface with triangles with no connectivity information among them [36]. In the reconstruction process, the pruned tree is refined by intersecting the mesh surface and the quantization grid, estimating previously lost voxels.

Following the encoding of the geometry, there is an attribute transfer procedure for the reconstructed geometry prior to the encoding of the attributes. TMC13 currently supports three transforms for attribute encoding: RAHT, Predict Transform and Lifting Transform, which were explained in detail in Section 3.1.2. However, the RAHT implementation currently in the TMC13 software model differs from the one previously explained by adding an intra-frame prediction step [48]. This modified version, referred here as U-RAHT (as this version is commonly referred as "upside down RAHT"), works by computing the high-pass and low-pass coefficients in the reverse order. In other words, U-RAHT computes the coefficients in a top-down order, going from the root of the octree to the leaves, in contrast to the bottom-up approach present in the initial version of RAHT. While this reversal of the traversal order does not affect the compression performance, it allows the introduction of a inter-depth upsampling for computing a local prediction to the node.

The U-RAHT algorithm takes advantage of the spatial correlation among neighbors by predicting the attributes at the next level of the octree. After transforming both the predicted and the original attributes, only the residual ACs and the DC have to be encoded. In other words, the algorithm works by computing a weighted average of the attribute $A^k_{x,y,z}$ at level k of the octree with its neighbors, generating the estimations $\hat{A}^k_{2x,y,z}$ and $\hat{A}^k_{2x+1,y,z}$ at the next level. The weights in this upsampling process are determined by the distance from each node to the upsampled sub-node. Subsequently, both the upsampled attributes $\hat{A}^k_{2x,y,z}$ and $\hat{A}^k_{2x+1,y,z}$ and the original ones $A^k_{2x,y,z}$ and $A^k_{2x+1,y,z}$ are transformed with (3.4), producing the estimated and original coefficients, respectively. The residual coefficients are then quantized and entropy encoded, which is illustrated in Fig. REF. This intra prediction scheme show reported gains of up to around 30% for color and 16% for reflectance over the RAHT scheme without prediction [48].

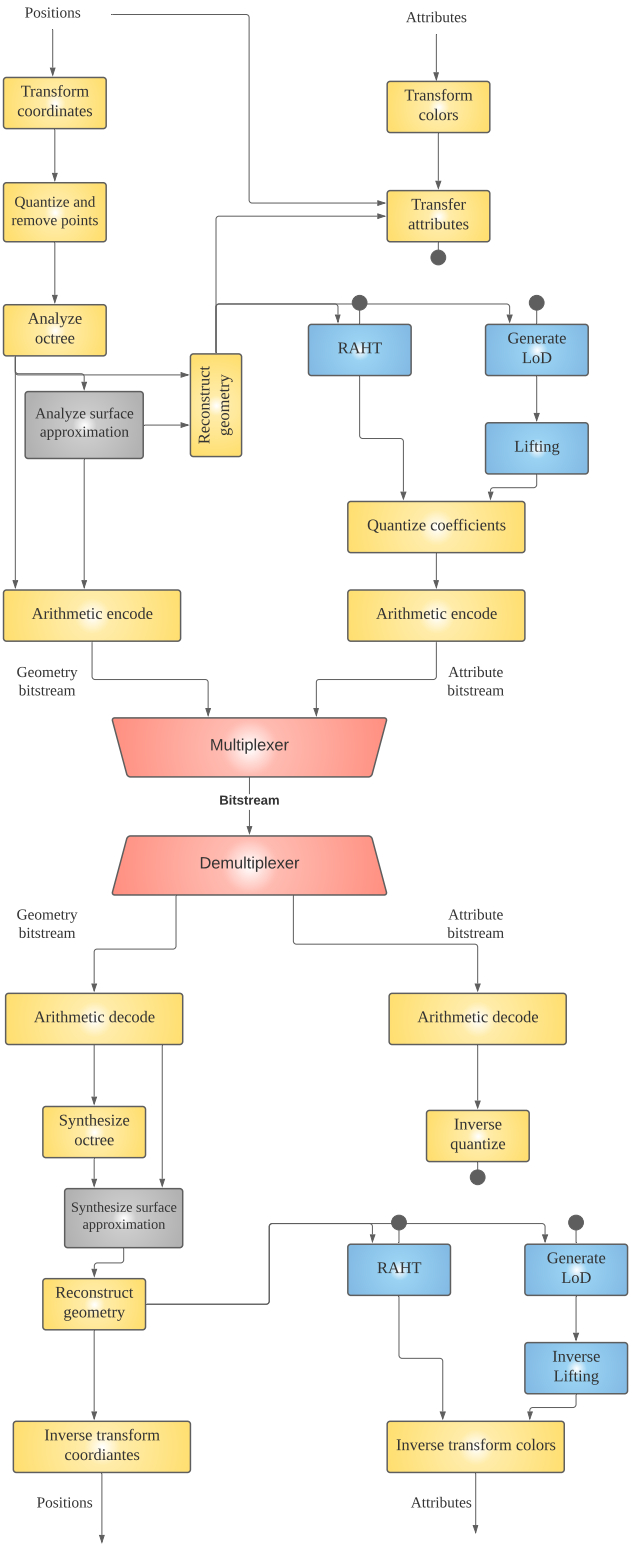


Figure 3.10: Overview of the G-PCC encoder (top) and decoder (bottom) pipelines [36].

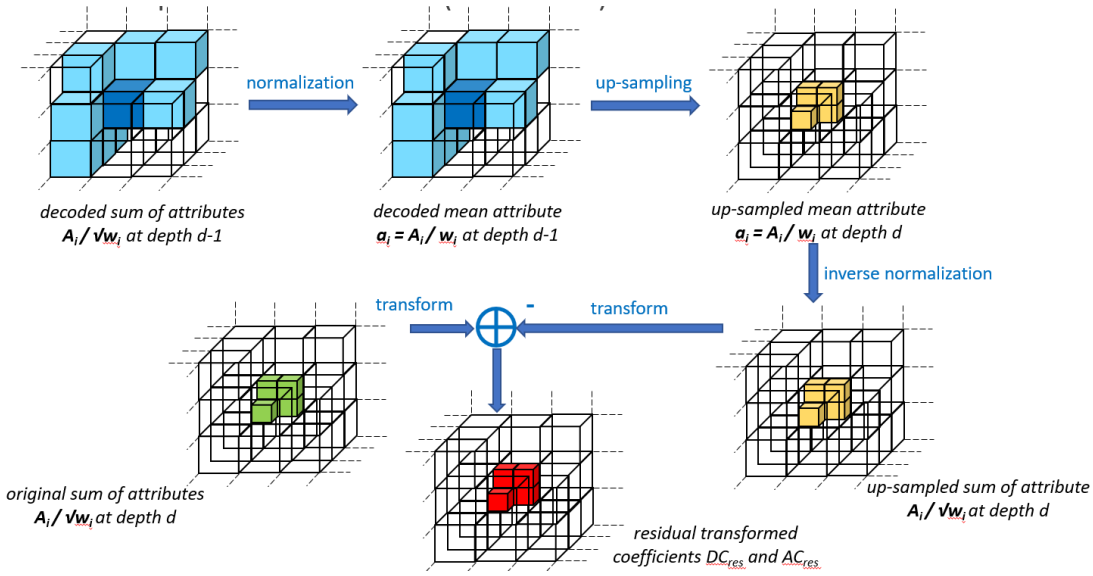


Figure 3.11: Illustration of the intra-frame predictive RAHT [36].

3.3 PLENOPTIC FUNCTION

3.3.1 Definition

The plenoptic function is a concept idealized in the computer graphics field that aims to completely represent the intensity of light at any given point by a 7-dimensional function

$$P(x, y, z, \theta, \phi, \lambda, t), \tag{3.6}$$

where (x, y, z) are space coordinates, (θ, ϕ) indicate the viewing direction, λ is the light wavelength and t is the time [4]. In other words, the plenoptic function can be seen as the space where all possible light rays pass through a point at a specific time.

A Plenoptic Point Cloud can represent such a function by discretizing these parameters. The coordinates (x, y, z) are discretized into voxel positions, the wavelength λ is discretized into RGB color components and time t is discretized with the capture of successive frames. The azimuth and elevation angles (θ, ϕ) are naturally sampled by a finite number of camera rigs in the capture process, and the intermediate values for these parameters can be interpolated by the renderer. Hence, a dynamic point cloud with N_c color attributes per voxel — where N_c represents the number of camera rigs — can represent the plenoptic function and it is referred here as a PPC.

3.3.2 Plenoptic Point Clouds

Plenoptic point clouds can be produced by capturing the scene information with an array of cameras with depth scanning capability or from light-field cameras. The representation of a PPC by using N_c color attributes per voxel introduced in Section 3.3.1 is a sampled representation derived

from viewing a voxel as a light source which emits light as a continuous function in all directions, as is illustrated in Fig. 3.12. In particular, the representation of this function in this work can be described as a non-uniform sampling of the directions, as each of the N_c attributes corresponds to the direction of the cameras employed to capture the data. Therefore, plenoptic point clouds are represented in this work by depicting each voxel with coordinates (x, y, z) that contains a different color value (λ) for each of the N_c cameras (θ, ϕ) at a given point cloud frame (t). Note that this selected representation only reflects one way of conveying the plenoptic information, which can be represented in different manners [8], [49]. One setback of our selected approach is that many cameras can be occluded for a given voxel, which results in a variable number of attribute samples. In order to convey a fixed number of samples per voxel, the occluded views were interpolated for the point clouds included in this dissertation.

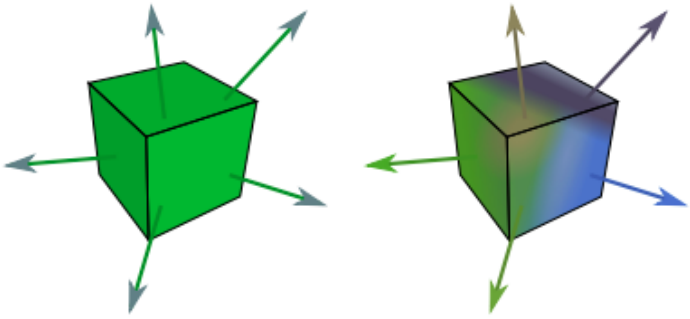


Figure 3.12: Illustration of a plenoptic voxel with directional color information [27].

In comparison to typical point clouds where each voxel is associated to a single color, PPCs enables a more realistic representation of the object from different viewpoints, which is essential to provide a more immersive experience in AR and VR applications. This is due to the fact that single-color point clouds are equivalent to a representation of completely diffuse sources of light, which are also known as lambertian surfaces. These differences are especially prominent when attempting to represent specular surfaces, as it is depicted by the changes in lighting for different viewing directions in Fig. 3.13. Note the changes in lighting at the metallic embellishments of the model's dress, as well as her forehead.

Point cloud compression is an active research field [41] and the representation of the plenoptic function has been the subject of MPEG and JPEG's ongoing standardization activities [50], [51].

Sandri et al. proposed the compression of PPCs by several approaches using the RAHT coder [5], [6]. These approaches address the problem of encoding plenoptic point clouds by taking into account the similarity between different cameras, where an intermediate transform is applied to reduce the redundancy among them. These transforms are either 2-D, by projecting the viewing angle of each camera in a 2-D map, or 1-D, by rastering the cameras to a 1-D vector. The results have shown that the method that achieved the best performance was the one using the 1-D KLT as the intermediate transform over the voxel colors. Following the transform, its coefficients are viewed as

Figure 3.13: Rendering of the plenoptic sequence *Thaidancer* for different views.

the attributes of the point cloud, and are then encoded using RAHT. This scheme is also known as RAHT-KLT.

Later on, Krivokuća et al. [7] proposed subdividing the PPC into clusters of specular and diffuse components as a pre-processing step prior to the scheme proposed by Sandri et al. This is done in order to reduce the high standard deviations produced by taking into account the entire point cloud as input data when computing the KLT vectors for the RAHT-KLT method. Results show that this pre-processing step provides marginal gains over RAHT-KLT for point clouds with highly specular regions, while presenting slightly worse performance for point clouds with few specular regions.

Zhang et al. model the PPC with a different approach, where the color function over (θ, ϕ) is continuously represented and compressed using spherical functions [8]. Therefore, the view synthesis technique is built into their encoding scheme.

Naik et al. proposed a solution to handle PPC data with MPEG's V-PCC standard [9], by adding a number of texture video streams that represent each view of the plenoptic point cloud. However, processing each of these textures independently results in significant redundancy among the views conveyed, producing not only a high bit rate, as it also requires a large amount of memory for processing. Afterwards, they proposed optimizations to this scheme by discarding some of the views based on the voxels' specularity as a pre-processing step [10]. While this optimization resulted in reduced memory consumption and bitrate savings, it also decreased the quality of the reconstructed structures. Therefore, the rate-distortion results of this optimization did not provide significant gains over the original results.

Finally, Li et al. proposed a video-based solution using V-PCC by compressing the multiple attributes via the MV-HEVC [11]. In order to efficiently compress the unoccupied pixels padded in the texture frames, this work also proposes two methods to minimize their bit rate cost. The first method optimizes the V-PCC framework to filter out the isolated unoccupied pixels using a block-

based group padding algorithm. The second one introduces a occupancy-map-based rate distortion optimization scheme over the default encoder of MV-HEVC. While this proposed scheme is not fully compatible with V-PCC due to the usage of the MV-HEVC, results proved this scheme to be a competitive video-based codec.

Therefore, existing solutions from the literature either do not provide full compatibility with MPEG's existing standards [5], [7], [8], [11] or are not competitive with the state-of-the-art, as is the case with [9], [10].

4 ALTERNATIVE MEANS OF REPRESENTING LOSSY POINT CLOUD GEOMETRY

The production of internationally accepted standards like the MPEG-2 and HEVC [42] by MPEG is essential for the development of common grounds for data exchange and for eventual improvements over existing coding solutions in a standardized manner. However, the research for the development of alternative representations that are competitive with the standardized state-of-the-art is also critical to the advancement of the prevailing technologies.

Thus, in this Chapter we propose to encode the point cloud geometry in a lossy manner using an alternative representation to the popular octree encoding introduced in Section 3.1.1.1. This is achieved by adapting the lossless Peixoto's algorithm [31] (Section 3.1.1.2) — that provides up to 8% bitrate savings in comparison to the TMC13 software's lossless mode — to support the introduction of losses, thereby achieving even lower bitrates that correspond to the operation points from TMC13's lossy mode. This solution is presented in Section 4.1.

4.1 LOSING INFORMATION TO IMPROVE COMPRESSION

In Section 3.1.1.2, an algorithm to losslessly represent the geometry information of a point cloud using a recursive dyadic decomposition approach is introduced. By representing a point cloud as a $N \times N \times N$ cube that can be sliced into N bi-level images, this algorithm allows the usage of well-known techniques for coding these types of images while working directly in the 3-D space. Here, we propose to introduce lossy mechanisms on the dyadic decomposition encoder to achieve lower bitrates without presenting significant geometry degradation.

In the lossless algorithm, a parameter $P \in [1, N]$ is added in order to efficiently encode the slices that are closer to the leaves of the dyadic binary tree. It works by following with the dyadic slicing until the processing region comprises a interval of $k <= P$ slices. From this point on, the algorithm also tests the single mode encoding method in order to determine the most efficient approach. For the lossy approach, we perform only the single mode encoding using a fixed value of $P = 64$ slices. That is, we proceed with the dyadic decomposition with lossless encoding of the silhouettes until and including depth $\log_2 N - 5$. The algorithm then stops the dyadic slicing of the tree and jumps to the leaf images that will then be compressed in a lossy fashion. As a result, the lossy compression techniques have to be implemented only in the leaves for this method of encoding. Fig. 4.1 shows an schematic representation of the tree construction. The specific choice of the parameter $P = 64$ was empirically determined and we regard further tuning of this parameter as a future work.

In order to compress the geometry in the single mode encoding, two lossy techniques were implemented. First, a *downsampling* technique is achieved by applying the non-adaptive, nearest

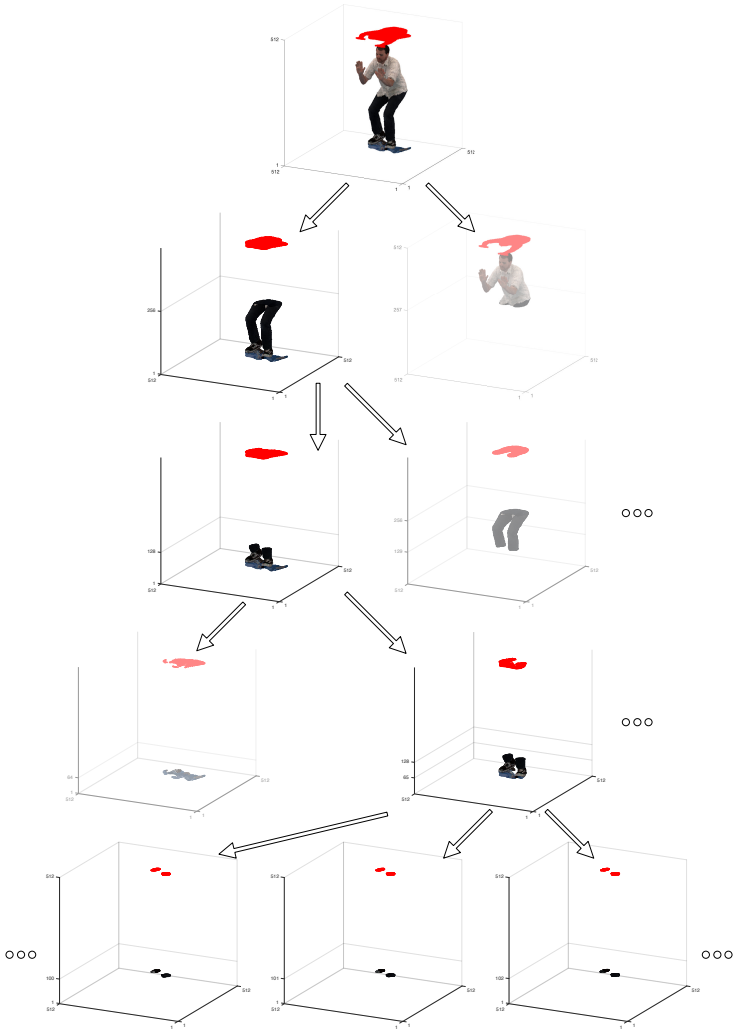


Figure 4.1: Diagram showing the dyadic decomposition and fixed single mode encoding with $P = 64$.

neighbor interpolation method (Section 2.3.1) to each single slice. Second, we sought to skip a certain number of consecutive slices, interpolating the intervening ones on the decoder side. For the rest of this work, this second technique will be referred simply as *step*, where a step value of k means that $k - 1$ successive slices were skipped. These techniques can be combined to achieve different bitrates. Fig. 4.2 illustrates the lossy techniques combinations. Note that the *step* technique can also be viewed as a form of downsampling in the direction of the axis that slices the point cloud into N bi-level images.

We propose a set of reconstruction techniques to improve the quality of the reconstructed voxels without transmitting additional data. To recover some of the occupied pixels lost in the downsampling, a morphological operation of closing (Section 2.3.2) is performed on the upsampled image at the decoder side. Then, the parent silhouette Y_S is used as a mask to constraint the region in which occupied voxels can be recovered. For the step technique, four different algorithms for reconstructing the discarded images were analyzed according to a rate-distortion approach.

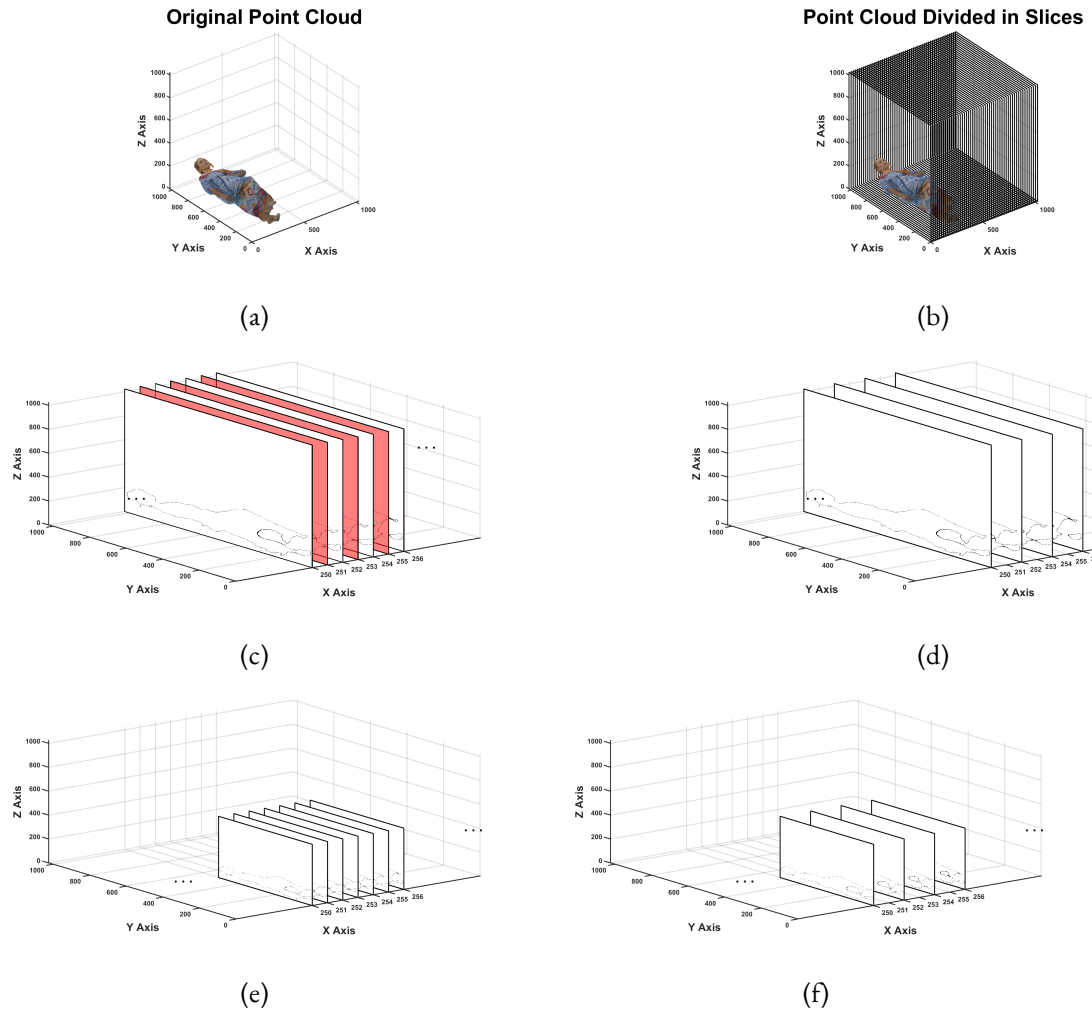


Figure 4.2: Example of application of the lossy techniques with the single mode encoding: a) Original point cloud; b) PC viewed as a 3-D cube and divided in slices; c), d) Application of the *step* technique for *step* = 2; e) Downsampling with resizing factor of 0.5; f) Combination of downsampling and *step*.

1. *3-D Linear*: this method is based on the application of a 3-D linear interpolation to compute the values of each point of interest;

2. *Morphologic*: a basic interpolation algorithm between the non skipped frames (referred here as *antecedent* and *subsequent* frames) is performed to find the best fit between them - referred here as t. Once this value t is found, the intermediate slices are generated by translating the antecedent frame by fractions of t. As an example, suppose the best value $t = 1$ between two frames. For a step value of 4, the three intermediate frames are generated by translating the antecedent frame by values by $\frac{1}{4}$, $\frac{2}{4}$ and $\frac{3}{4}$. In addition, operations of morphological closing and the use of Y_S as a mask are performed to further enhance the interpolated slices.

3. *Repeat*: this method consists of replacing the discarded frames by the “nearest" transmitted image. As an example, consider a case where the number of discarded consecutive slices is 5 (which implies a *step* value of 6). In the reconstruction process, these 5 frames would be replaced in the decoder side by: *Antecedent*, *Antecedent*, *Subsequent*, *Subsequent* and *Subse-*

quent.

4. *Merge*: the *merge* reconstruction method is based on replacing the discarded images by the *OR* binary operation between the *Antecedent* and *Subsequent* frames.

Note that these methods provide the same bitrate, as they consist of a post-processing step to reconstruct the already conveyed frames in the decoder side.

4.2 EXPERIMENTAL RESULTS

In the scope of the point cloud's geometry, this section presents the experiments involving our scheme for alternatively encoding the geometry in a lossy manner, which was explained in Section 4.1.

In order to assess our solutions, a collection of point cloud sequences from popular datasets were used. As we largely compare our proposals with solutions that revolve around the MPEG's standards for point cloud compression, we prioritize PC sequences from MPEG's common test conditions (CTC) [52]. For the experiments concerning geometry encoding in Section 4.2, we used multiple frames from the four-full body sequences of the 8i Labs' Voxelized Full Bodies dataset (8iVFBv2): *Longdress*, *Loot*, *RedandBlack* and *Soldier* [14]. Fig. 4.3 shows the projections of a single frame of each of the mentioned sequences. These four sequences which are present in G-PCC's CTC are depth-10 PCs that contain up to 300 frames that range from 700, 000 to 1, 100, 000 voxels. In addition to these, we also tested our geometry solution separately using multiple frames from the *Man 2* sequence [25], which is a depth-9 point cloud that contains around 200 frames in the 200, 000 voxels range. This information is detailed in Table 4.1.

Table 4.1: Characteristics of the point clouds used for the geometry evaluation

Sequence	Depth	Voxels	Frames
Longdress	10	850,000	300
Loot	10	800,000	300
RedandBlack	10	750,000	300
Soldier	10	1,090,000	300
Man 2	9	200,000	200

The evaluation present in the following experiments is based on rate-distortion performance, where the rate is reported as bits per occupied voxel of the input point cloud (bpov) and the distortion is be measured in terms of peak signal to noise ratio (PSNR) by two different metrics [52]: the point-to-point error, or D1 metric, is obtained by calculating the Mean Squared Error (MSE) between the reconstructed points and their nearest neighbors in the reference point cloud. The second metric is the point-to-plane error, or D2 metric, which is computed by taking into account the surface planes instead of the nearest neighbors [53].

Figure 4.3: Projections of the point clouds sequences used in experiments. From left to right, top to bottom: *Man 2*, *Boxer*, *Longdress*, *Loot*, *RedandBlack*, *Soldier* and *Thaidancer*.

4.2.1 RD Evaluation for Reconstruction Methods

Our first experiment consisted on evaluating the RD performance of our four reconstruction methods for the lossy *step* technique introduced in Section 4.1: *3-D Linear*, *Morphologic*, *Repeat* and *Merge*. Fig. 4.4 illustrates the results for the frame 1300 of *Longdress* and the frame 1200 of the *Loot* sequence for five different operation points, which correspond to *step* values of 2, 3, 4, 5 and 6. Table 4.2 presents the results with the four sequences in terms of BD-PSNR [54] for the aforementioned methods, where the *3-D Linear* method is used as anchor.

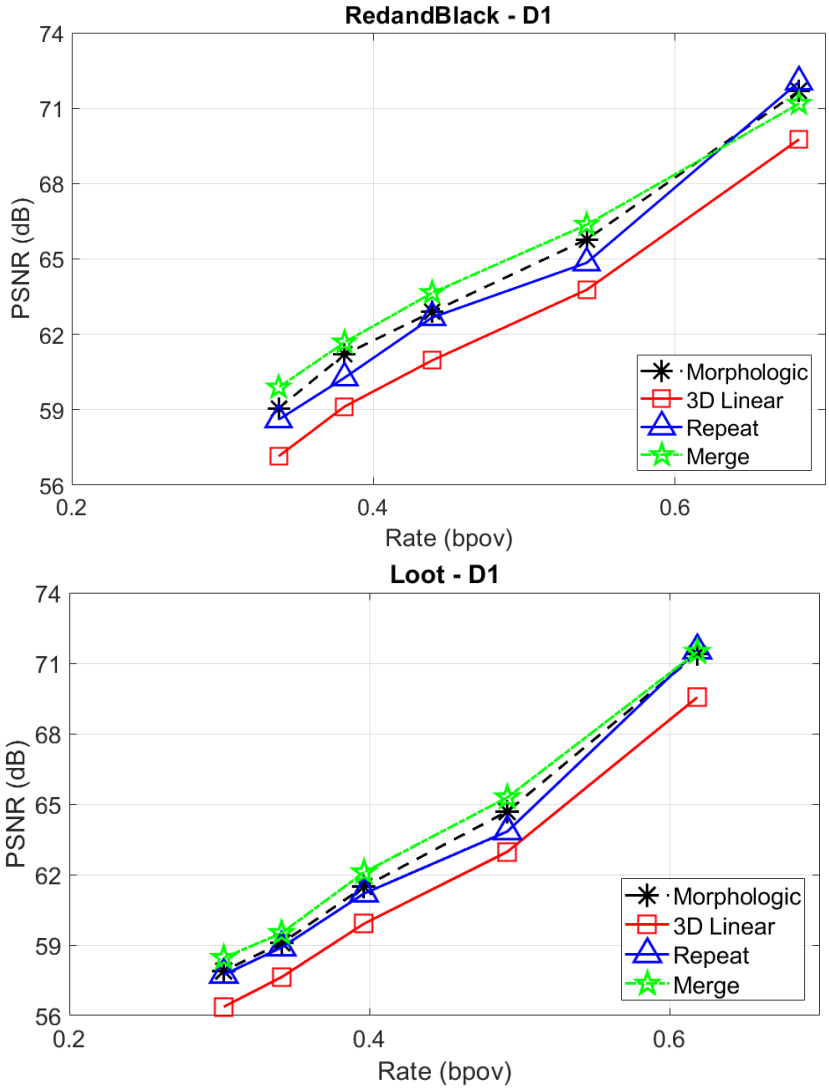


Figure 4.4: RD performance comparison for the proposed reconstruction methods for (left) Frame 1550 of RedandBlack; (right) Frame 1200 of Loot.

Results from Table 4.2 show that the *merge* algorithm presented the superior results over the other three reconstruction methods, with an average of 2.35dB over the *3-D Linear* and 0.57dB over the *merge* algorithms. However, the D1 metric is dependent of the MSE, which means that its value is susceptible to the number of points of a point cloud. Hence, a PC sequence with a significantly larger number of points can present a higher PSNR value even though it producer a

Table 4.2: BD-PSNR (in dB) comparisons between the four proposed reconstruction methods. Results from the *3-D Linear* reconstruction method were used as reference.

Sequence	***D1***		
	Morphologic	**Merge**	**Repeat**
LongDress	1.76	2.50	1.50
Loot	1.64	2.15	1.21
RedAndBlack	1.98	2.49	1.43
Soldier	1.73	2.26	1.30
Average	1.78	2.35	1.36

higher absolute error — and thus, a greater distortion. Therefore, Table 4.3 presents the percentage of decoded points generated by each reconstruction method. These values were obtained by taking the ratio between the average number of points generated by the five compression rates tested and the original number of points in each sequence. One can notice that the *merge* method produces an average increase of 57% in the number of points, which supports the suggestion about its superior RD performance. On the other hand, the *3-D Linear* method presented a 57% average decrease relaive to the original number of voxels. While a significantly smaller number of points is not desirable due to the reduction of the PC's density and increased information loss, a considerably larger amount is not desirable also, as it introduces additional complexity to the PC while adding information that was previously non-existent. Therefore, the *morphologic* reconstruction method was chosen as the post-processing method for the step technique, as it presented the most suitable balance between the number of decoded points and the associated PSNR values.

Table 4.3: Comparisons regarding number of decoded points (in % relative to the original number of points) between the four proposed reconstruction methods.

Sequence	***Percentage of decoded points (%)***			
	Morphologic	**Merge**	**Repeat**	**3-D Linear**
LongDress	102	157	100	44
Loot	100	157	100	44
RedAndBlack	104	158	100	43
Soldier	100	158	100	42
Average	101	157	100	43

4.2.2 Selection of the Best Coding Parameters

The combination of different values of the downsampling and step lossy techniques can provide distinct levels of distortion and bitrate on the point cloud. As a result, it is necessary to obtain a definite set of parameters to provide a RD optimized curve. Taking that into account, 18 different combinations of values were tested: for downsampling, values of 1, 1.33, 1.6, 2, 3.2 and 4 were analyzed; for step, values of 1, 2 and 4 were used.

In an attempt to avoid bias in our results with the four sequences of the 8i Labs' dataset, this test was performed over ten randomly selected frames of the point cloud *Man 2*, resulting in the values observed in Fig. 4.5. Each marker type in the figure corresponds to a specific downsampling value, while the different number near each point indicates the step value it represents. The chosen combinations are: *(downsampling,step)* = $[(1, 2), (1.33, 2), (2, 2), (3.2, 2), (3.2, 4)]$, where we have chosen operation points that better cover the possible bitrates.

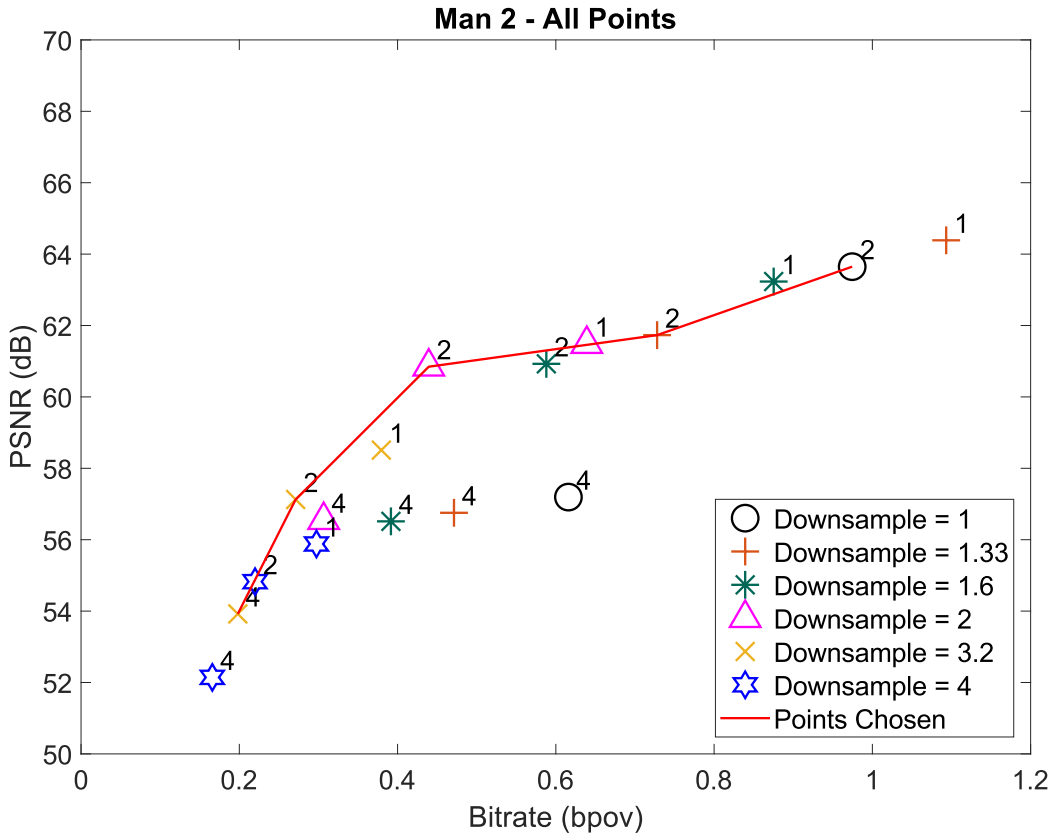


Figure 4.5: Operation points for sequence *Man2*. The values near each point indicates the step value it corresponds to.

4.2.3 Post-processing Reconstruction Improvements

In order to measure quantitatively the improvements provided from our proposed post-processing techniques — the reconstruction for the downsampling technique presented in Section 4.1 and the *morphologic* method for the step —, we ran a small experiment comparing them to a simple reconstruction, which is presented in Fig. 4.6. Results show that the proposed algorithms improved significantly the reconstruction quality, with a BD-PSNR gain of 1.89dB.

4.2.4 Comparison to the MPEG Standard

Finally, we evaluate our proposal against state-of-the-art solutions by comparing to version 7 of the TMC13 software from G-PCC's standard. For ease of identification, we refer to TMC13's lossy

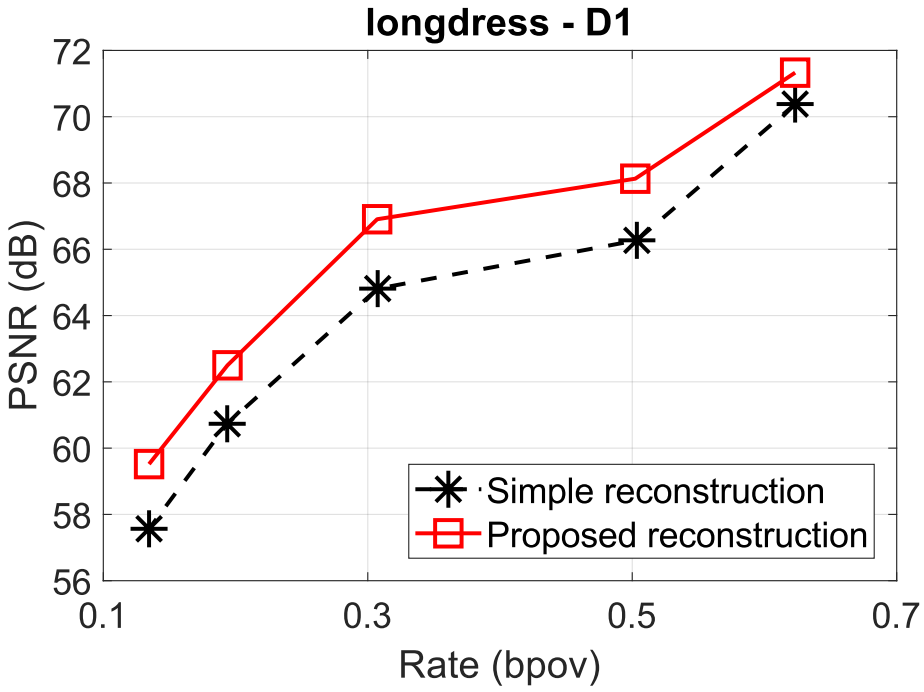


Figure 4.6: Comparison of proposed reconstruction methods.

encoding of geometry by using direct geometry quantization as *TMC13 Octree* and using triangle surface approximation as *TMC13 Trisoup* (see Section 3.2.2). In order to do so, we computed the RD performance for D1 and D2 metrics over the first 100 frames for the four 8i sequences, which is shown in Figs. 4.7 and 4.8, respectively. In order to better assess the proposed method we have divided the bitrate range in two ranges: a low range, up to 0.3 bpov, and a high range, from 0.3 bpov. In the figures, this is shown as a vertical dotted line.

The experimental results are summarized in Table 4.4. The results were obtained by taking the average of the BD-PSNR values [54] - using TMC13 Octree as anchor - for both D1 and D2 metrics over the first 100 frames of each sequence. The BD-PSNR over the whole bitrate interval is also shown.

Table 4.4: BD-PSNR comparisons of lossy geometry compression with state-of-the-art algorithms. All values are in dB. Results of the MPEG G-PCC TMC13 v7.0 with direct geometry quantization method are used as the benchmark.

Sequence	*D1*						*D2*					
	Proposed			**TMC 13 - Trisoup**			**Proposed**			**TMC 13 - Trisoup**		
	Low	High	Total	Low	High	Total	Low	High	Total	Low	High	Total
Longdress	2.10	**2.49**	2.10	4.28	0.99	3.07	1.58	**-0.31**	0.11	2.51	-1.27	1.48
Loot	0.34	**1.81**	0.88	4.47	0.81	3.37	1.67	**0.42**	0.57	2.78	-1.45	1.74
RedandBlack	1.85	**2.53**	2.04	4.05	0.99	2.93	1.95	**0.08**	0.39	2.86	-0.96	1.71
Soldier	2.06	**2.67**	2.21	4.39	1.03	3.26	2.11	**0.55**	0.70	2.86	-1.09	1.84
Average	1.59	**2.38**	1.81	4.30	0.96	3.14	1.83	**0.19**	0.44	2.75	-1.19	1.60

We can observe that both TMC13 Trisoup and our proposal perform better in the D1 met-

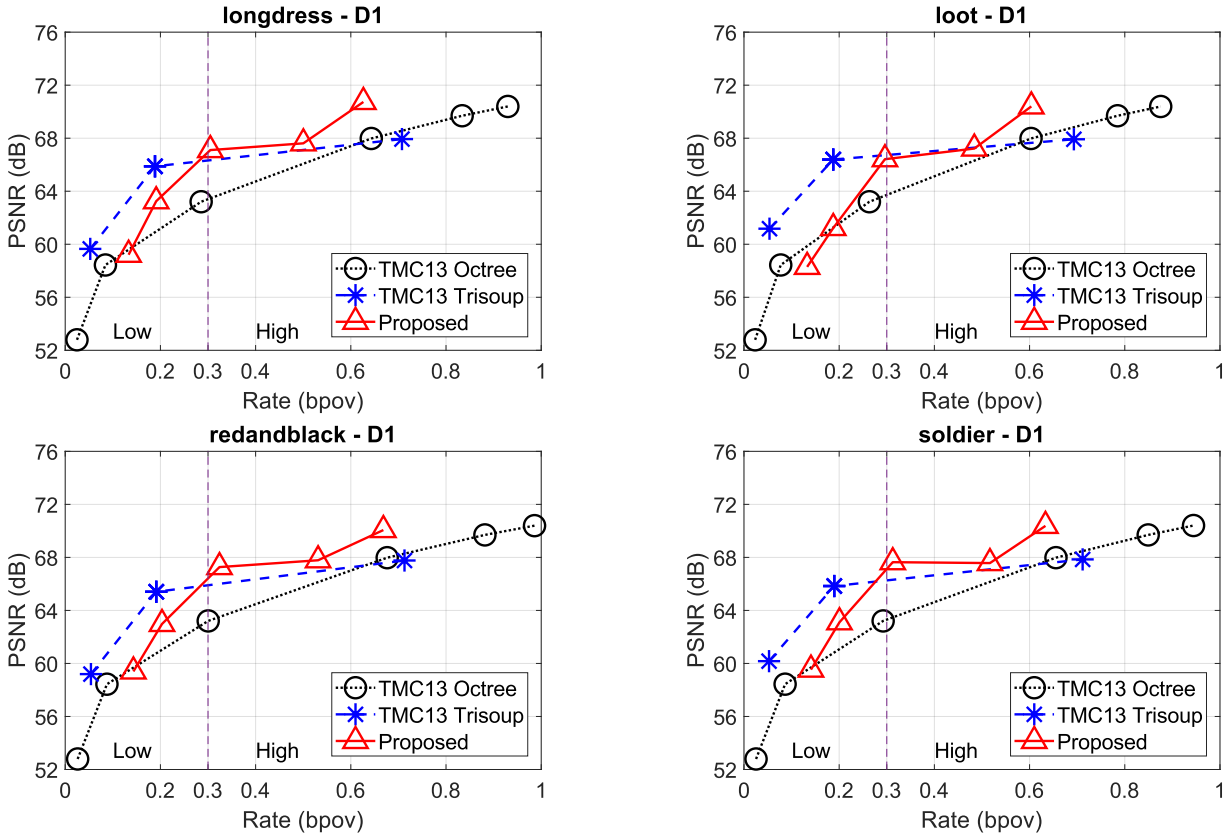


Figure 4.7: RD performance results for D1 metrics for the sequences, from left to right, top to bottom: *Longdress*, *Loot*, *RedandBlack* and *Soldier*.

rics than the TMC13 Octree, with our proposal presenting better results than both algorithms for higher bitrates (an average of 2.4dB advantage over TMC13 Octree and 1.4dB over TMC13 Trisoup), while TMC13 Trisoup performs better for lower bitrates. For the d2 metrics, both algorithms again present better results than TMC13 Octree. However, TMC13 Trisoup performs worse for higher bitrates (average of −1.19dB), while our solution performs only slightly better with an average of 0.19dB.

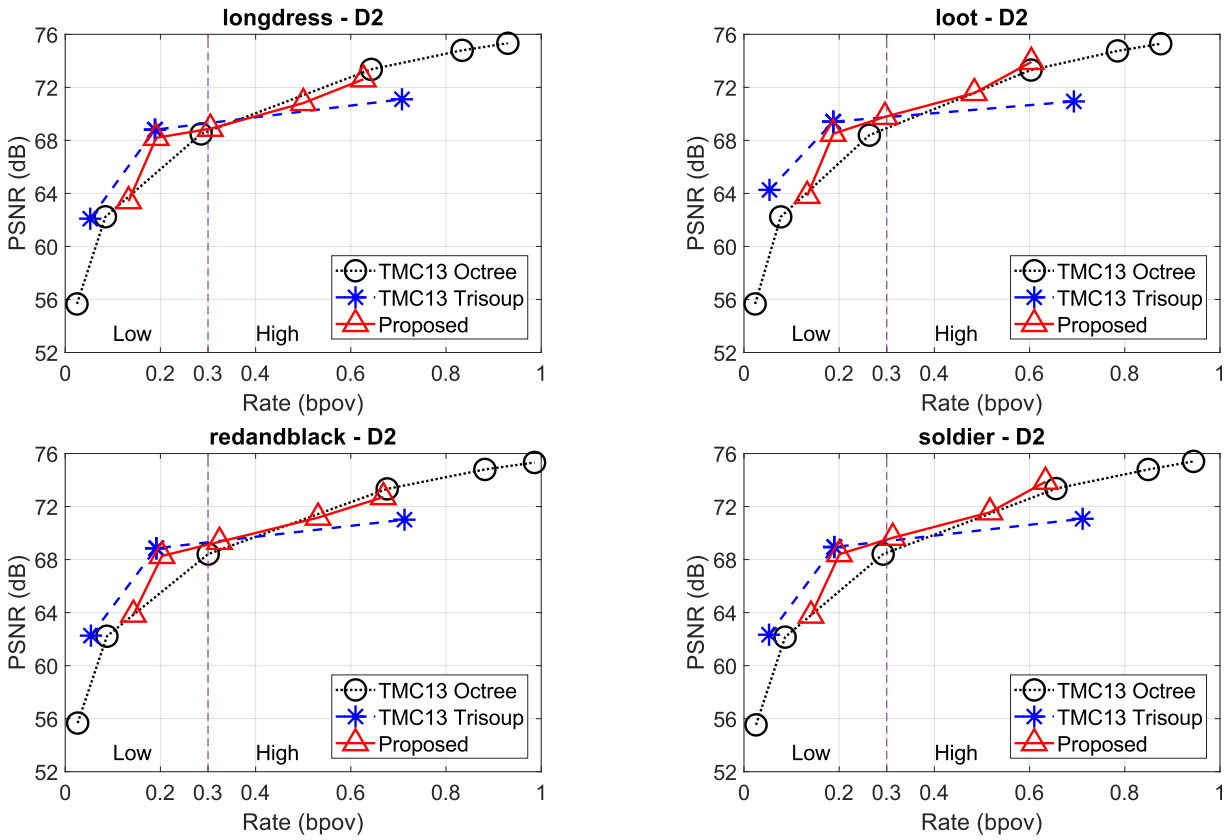


Figure 4.8: RD performance results for D2 metrics for the sequences, from left to right, top to bottom: *Longdress*, *Loot*, *RedandBlack* and *Soldier*.

5 PLENOPTIC ENHANCEMENT INTO MPEG'S STANDARDS

In this Section, we seek to enhance MPEG's both geometry- and video-based solutions for point cloud compression — V-PCC and G-PCC — with plenoptic capability. Table 5.1 summarizes the solutions for plenoptic point cloud compression presented in Section 3.3.2 relative to their compatibility with MPEG's current coding strategies. Although the solution from Naik et al. [9], [10] is V-PCC compliant, the results are not as competitive as the ones presented by Li et al. [11]. However, while the solution from Li et al. is a competitive video-based codec that was the state-of-the-art up until now, we consider it to be only partially compatible with TMC2 due to encoding each atlas from V-PCC (Section 3.2.1) with a multiview HEVC instead of the single version one.

Table 5.1: Proposed solutions for PPC compression found in the literature according to its compatibility with MPEG's standards

Proposal	Compatibility	
	TMC13	TMC2
Sandri et al [6]	✗	✗
Krivokuća et al. [7]	✗	✗
Zhang et al. [8]	✗	✗
Naik et al. [10]	✗	✓
Li et al. [11]	✗	✓[1]

[1] Partial compatibility

Therefore, we propose two solutions for efficient compression of plenoptic point clouds while taking into account the different coding approaches from MPEG's both current PCC strategies. In Section 5.1 we present our geometry-based solution based on Sandri et al.'s work [6], taking a KLT over the color attributes followed by multiple attribute coders with intra prediction capability that are compliant with TMC13's current implementation. In Section 5.2 we introduce our video-based solution, which consists of sending a number N_c of pixel-wise transformed differential atlases to be encoded by a video encoder along with TMC2's main payload. These N_c atlases correspond to the N_c attribute colors per point that are generated by the multiple camera views and are considered a secondary payload, which makes it backwards compatible with the existing TMC2's implementation.

5.1 ADDING PLENOPTIC ENHANCEMENT INTO G-PCC

For our geometry-based solution of compressing the plenoptic textures, we build over the framework proposed by Sandri et al. [6] for encoding these attributes directly in the 3-D space. Consider a PPC where the n-th occupied voxel has a set of N_c RGB color values associated to the different

camera viewpoints used in the capture process. Assume the RGB values are converted to YUV space, yielding plenoptic color vectors $[Y_n^1, U_n^1, V_n^1, \cdots, Y_n^{N_c}, U_n^{N_c}, V_n^{N_c}]$. This is due to the fact that the color variation is most significant in the luminance component (Y), which enables the spatial decimation of the chrominance components (U and V) for lower bitrates. As in [6], we apply a linear transform over each of the color components Y, U and V in order to take advantage of the correlation between the N_c plenoptic views. Consider C as one of the color channels and let

$$\mathbf{c}(n) = [C_n^1, C_n^2, C_n^3, \cdots, C_n^{N_c}]^T \tag{5.1}$$

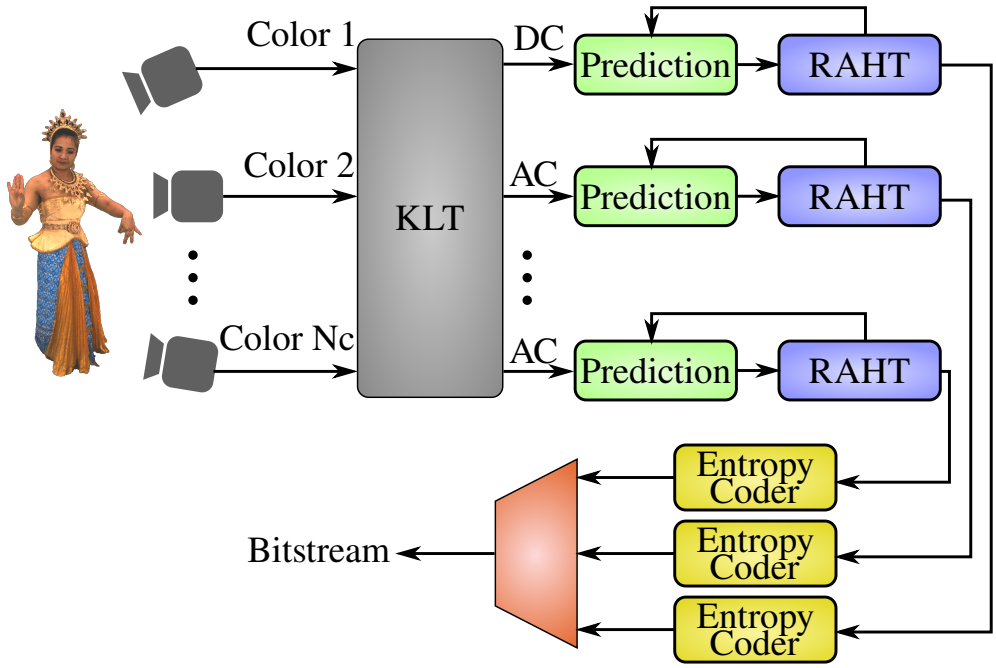
represent the color component for the n-th voxel as viewed by the N_c camera rigs. For this approach, the KLT is applied over the $\mathbf{c}(n)$ vector signal as

$$\mathbf{s}(n) = \mathbf{H}_c\mathbf{c}(n), \tag{5.2}$$

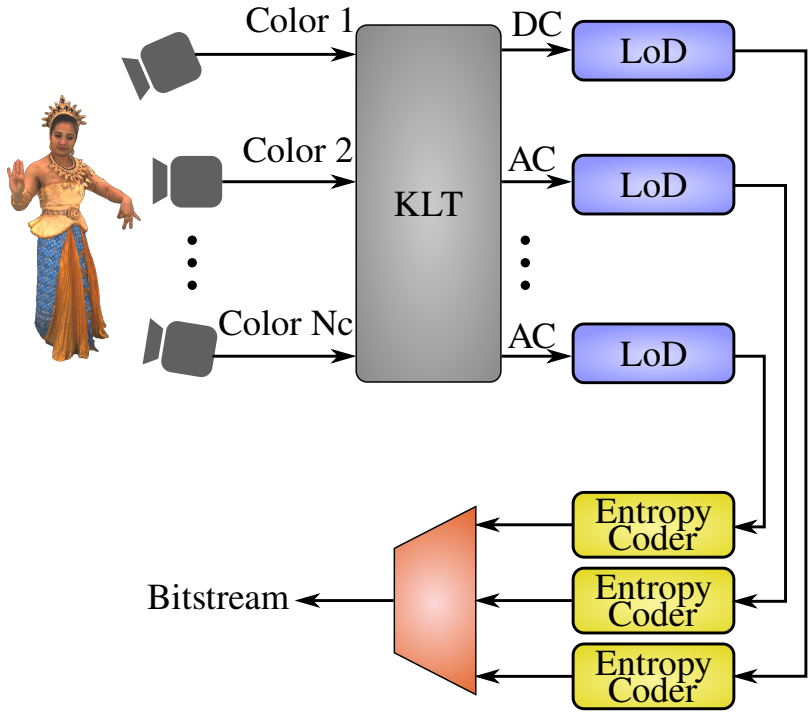
where $\mathbf{s}(n) = [S_n^1, S_n^2, \cdots, S_n^{N_c}]^T$ is the vector with the transformed coefficients for the n-th voxel. $\mathbf{H}_c$ is the $N_c \times N_c$ KLT matrix of the color channel C, i.e. an orthogonal matrix made of the eigenvectors of the covariance matrix $\mathbf{K}_{cc}$ as seen in Eq. 2.6. In this case, the entries for $\mathbf{K}_{cc}$ are the color signals from Eq. 5.1, such that $K_{ij} = \{E[(C_n^i - E[C_n^i])(C_n^j - E[C_n^j])]\}$. Each of the transformed S_n^k is an attribute of the n-th transformed voxel and the set of $\{S_n^k\}$ is a PC to be transformed and encoded. The set $\{S_n^1\}$ is often called the DC coefficient PC. The others ($\{S_n^{k>1}\}$) are referred as AC PCs. These AC coefficients may contain negative values and, because of that, they are made positive by adding an offset.

Subsequently, we propose to encode the $\{S_n^k\}$, which are still redundant signals, using state-of-the-art attribute coding schemes with intra-frame prediction capability. Thus, in order to be compliant with the G-PCC standard we encode the transformed attributes using TMC13's LoD- and RAHT-based coding schemes, which were described in detail in Sections 3.1.2.2 and 3.2.2, respectively.

In essence, we run N_c attribute coders in parallel as depicted in Figs. 5.1- a) and 5.1- b). Figure 5.1- a) illustrates the encoder based on RAHT, wherein the N_c KLT output signals are fed to the multiple encoders. This approach is referred here as Proposed Method 1. Note that it is important, however, that we use intra-frame-prediction within RAHT [48], [55] in order to further remove redundancy and to improve compression by exploiting the correlation between neighboring voxels. We have tested the use of intra-frame-prediction in both the DC and AC channels and the results consistently pointed to a superior performance when using the prediction, which is described in [48]. A very similar approach based on LoD is illustrated in Fig. 5.1- b) and is referred as Proposed Method 2. In Figs. 5.1- a) and 5.1- b), we used the G-PCC entropy coder. Also note that the plenoptic information generated by these two schemes is sent as an additional payload to the conventional information that is conveyed by the TMC13 model, which makes our solution backwards compatible with G-PCC's current implementation.



(a)



(b)

Figure 5.1: Compression schemes for the a) Intra-frame predicted RAHT (proposed method 1) and b) LoD (proposed method 2).

5.2 ADDING PLENOPTIC ENHANCEMENT INTO V-PCC

The framework from where our video-based solution for plenoptic point clouds is based on consists of representing the plenoptic information as extra color attributes projected as atlases to

be encoded with a standard video codec [41]. Our proposal extends this representation by applying a differential transform to these extra atlases created for each color. As was seen in our geometry-based solution in Section 5.1, we seek to comply our solution to the existing V-PCC decoders, where the encoder generates two payloads: the main payload is composed of the depth, occupancy and texture atlas (Section 3.2.1) that can be decoded by any conventional decoder. The second payload is the plenoptic enhancement one, which is supplemental to the regular V-PCC's encoded data and contains the information of all the N_c plenoptic colors for each voxel. A "plenoptic" decoder would then reconstruct a PPC by adding this plenoptic information to the single-color reconstructed PC. This coder, which we refer to as video-based plenoptic point cloud compressor (V-PPCC) is illustrated in Fig. 5.2. Note that we work directly with the pixels of the plenoptic texture atlases, because not all voxels of the point cloud are mapped to pixels in the projection procedure. This enables us to use locally reconstructed images in order to perform our experiments. Also, we work with the texture atlases already in the YUV color space in order to take advantage of the correlation between cameras, which is mentioned in detail in Section 5.1.

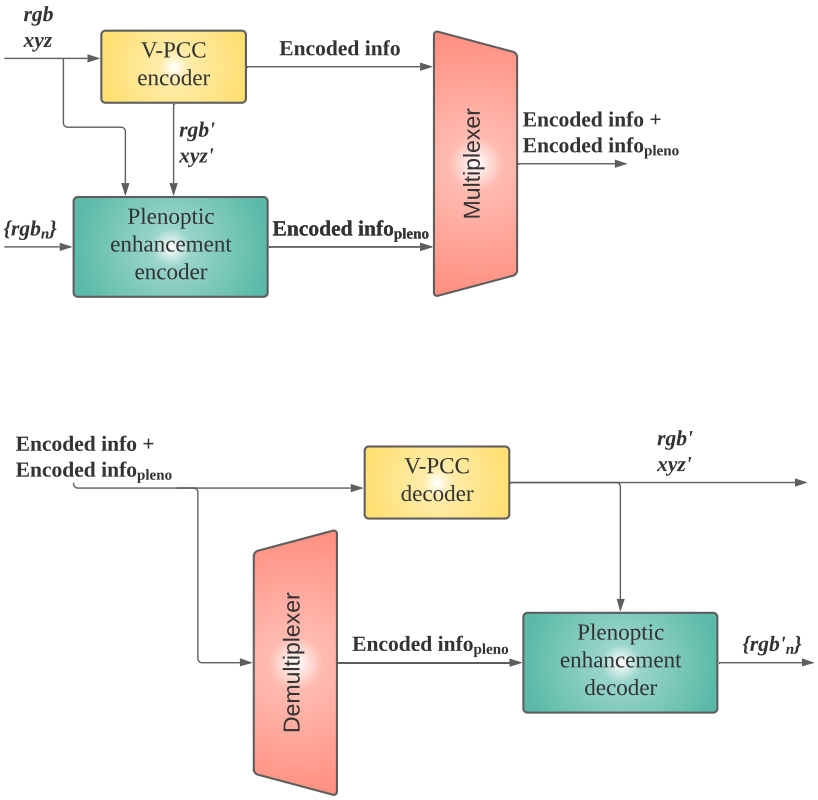


Figure 5.2: Proposed plenoptic V-PCC codec (V-PPCC)

The main idea of our solution lies on whether the PPC to be encoded comes with a main RGB channel. It is generally assumed that, given the existence of this main texture channel, it is conveyed to the decoder as it should be included in the main payload. In the case that this main RGB channel is not present, the encoder may create it from the N_c camera colors or use the DC atlas as the main texture for the main payload. In the former situation, the plenoptic enhancement payload

carries the data for the N_c plenoptic colors and is presented in Section 5.2.1. For the latter, it carries information about the other $N_c - 1$ texture atlases and is presented in Section 5.2.2.

5.2.1 Differential Coding Scheme

Our differential encoder complies with the current TMC2 implementation by compressing both the geometry and the attribute information from the point cloud through the conventional V-PCC scheme, which is illustrated in Fig. 5.3. Consider that i and j are the pixel coordinates of a given atlas. The conventional V-PCC scheme first generates the occupancy $Occ(i, j)$ and the geometry atlas $Geom(i, j)$ in order to map the 3-D to 2-D projections necessary for transferring the point cloud's main color information to its respective main texture atlas. After the conversion to the YUV color space, we refer to this texture atlas as $YUV(i, j)$. After encoding these three atlases in the main payload, a local reconstruction of $YUV(i, j)$ is computed, which is referred here as $YUV'(i, j)$. For the plenoptic enhancement payload, the same procedure in the texture atlas generation is repeated for each of the N_c plenoptic colors, resulting in a set of $\{YUV_n(i, j)\}$ plenoptic atlases. Our differential encoder then computes a set of differential signals $\{E_n(i, j)\}$, where $E_n(i, j) = YUV_n(i, j) - YUV'(i, j)$. Subsequently, we transform each color channel of $\{E_n(i, j)\}$ with a pixel-wise DCT, generating N_c images of coefficients $\{DCT_n(i, j)\}$. These coefficient images are then scaled into an integer M-bit representation by adding an offset of 2^{M-1} to all coefficients and rounding them to the nearest integer. In the case of a HEVC with 8-bits, the coefficient images are encoded with a scaling of $M = 8$ and merged with the conventional V-PCC stream.

Fig. 5.4 depicts the the reverse process that is applied in the decoder. The streams corresponding to the main payload and the plenoptic enhancement are separated, the transformed plenoptic atlases are de-scaled, inverse transformed and added with the decoded $YUV'(i, j)$, generating the decoded set of $\{YUV'_n(i, j)\}$ atlases. Finally, the plenoptic point cloud can be reconstructed with the decoded atlases from the main payload in addition to the decoded plenoptic images, which follow the same reconstruction scheme that is applied to the main texture atlas.

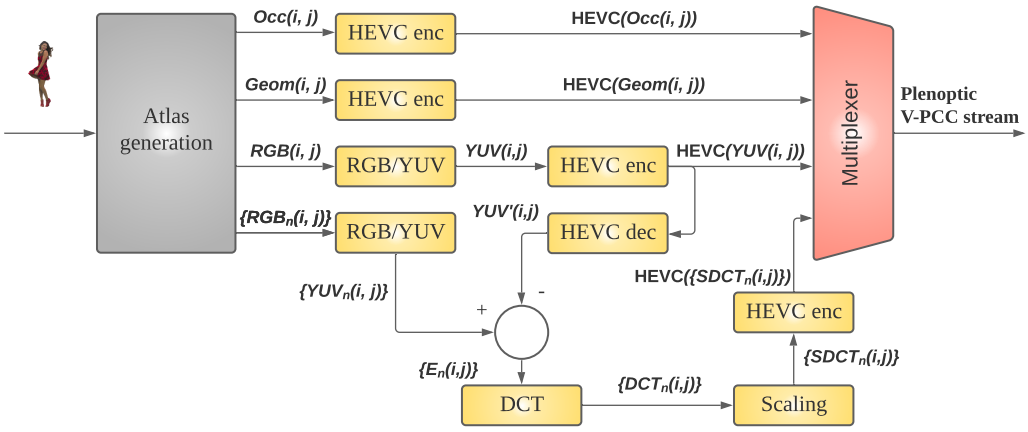


Figure 5.3: Differential encoder for plenoptic enhancement.

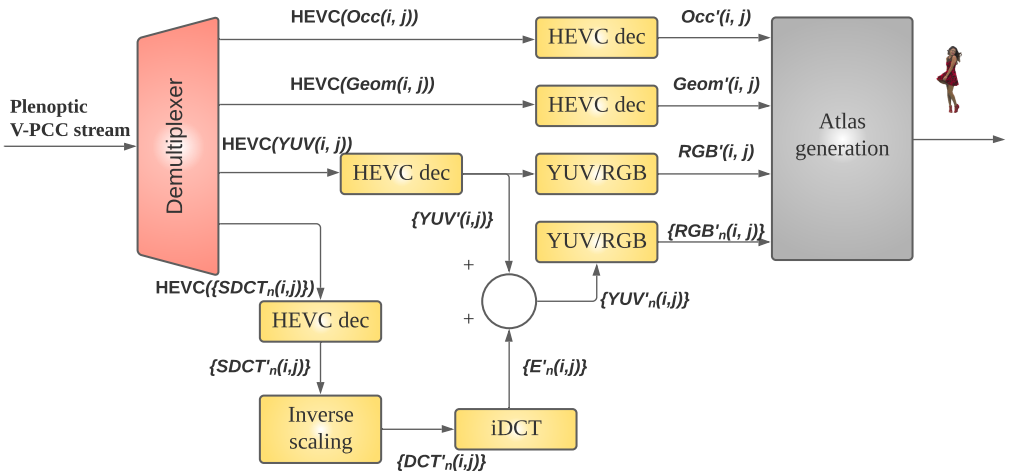


Figure 5.4: Differential decoder for plenoptic enhancement.

5.2.2 Non-Differential Coding Scheme

Our non-differential solution is adopted for the cases where the PPC that is being encoded does not contain a main color attribute set. In this proposal, the DC coefficient image resulting from the DCT takes the place as the main color attribute to be encoded. The DC coefficient image is multiplied by $\frac{1}{\sqrt{N_c}}$ and rounded to the nearest integer. In contrast to our differential solution, the remaining AC coefficient atlases are not subjected to any differentiation, hence its name. Instead, they only scaled by adding an offset of 2^{M-1} to all coeficients and rounding them to the nearest integer. All scaled coefficient images are then encoded using HEVC. This scheme is depicted in Fig. 5.5.

The inverse process happens at the decoder side, which is shown in Fig. 5.6. Following the stream separation, the DC coefficient image that replaces the main payload texture is decoded with HEVC along with the rest of the main payload and the $N_c - 1$ AC coefficient atlases. Subsequently, the DC and the ACs atlases are reintegrated, de-scaled and the inverse transformed, generating the decoded plenoptic colors for reconstruction.

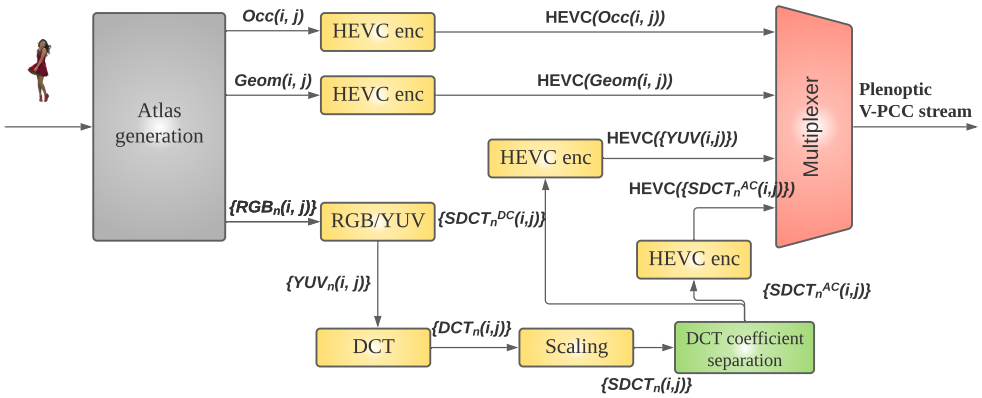


Figure 5.5: Non-differential encoder for plenoptic enhancement.

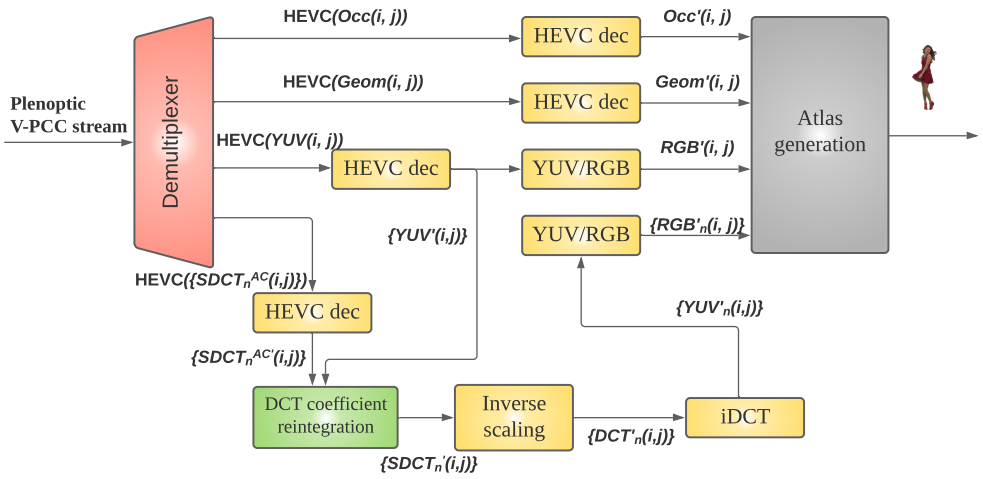


Figure 5.6: Non-differential decoder for plenoptic enhancement.

5.3 EXPERIMENTAL RESULTS

Within the framework of the PC's attributes, we present in Sections 5.3.1 and 5.3.2 our compression schemes for enhancing MPEG's G-PCC and V-PCC with plenoptic capability, which were devised in Sections 5.1 and 5.2, respectively.

Although a CTC for plenoptic point clouds has not been established, discussions regarding them have been initiated [56]. Hence, we evaluate our plenoptic enhancement solutions in Sections 5.3.1 and 5.3.2 with the Voxelized Surface Light Field dataset from 8i Labs (8iVSLF) [57]. This dataset consists of one dynamic point cloud sequence of 300 frames and six single-frame point clouds, where the extra plenoptic attributes were obtained from a set of 12 or 13 camera rigs. Four of these sequences are plenoptic depth-12 versions of the sequences present in 8iVFBv2 dataset: *Longdress*, *Loot*, *RedandBlack* and *Soldier*. In addition to these, the two other PCs are *Boxer* and *Thaidancer*, whose projections were previously shown in Fig. 4.3. Downsampled versions of the six PPCs from this dataset have also been generated which, among other advantages, provide reduced processing time and reduced memory consumption [58]. Further details are shown in Table 5.2.

Table 5.2: Characteristics of the (plenoptic) 8iVSLF dataset

Sequence	Voxels		N_c
	Depth 12	Depth 10	
Boxer	3493085	995099	13
Longdress	3096122	912518	12
Loot	3017285	869565	13
Redandblack	2770567	839315	12
Soldier	4001754	1193515	13
Thaidancer	3130215	283206	13

5.3.1 Adding Plenoptic Enhancement into G-PCC

For assessing our plenoptic enhancement solution for the G-PCC standard, we carried the experiments with version 10.0 of the TMC13 reference software [36]. Both proposed LoD- and RAHT-based coding schemes were compared to the works of Sandri et al. [6] and Li et al. (MV-HEVC) [11]. The distortion was computed by considering the PSNR in between original and reconstructed Y channels for the N_c cameras either concatenated as a single signal or taking the average across all cameras, in accordance to [11]. We use the former method except when comparing to MV-HEVC, in which case we use the latter. The experiments were ran with the six depth-12 plenoptic point cloud sequences from the 8iVSLF dataset. In addition to that, tests for our proposed methods and the method in [6] were also performed with the depth-10 versions of these sequences.

5.3.1.1 LoD vs RAHT-based Solutions

Initially, we compare our solutions where our attribute coders with intra-frame prediction capability are based on TMC13's RAHT- and LoD-based schemes, which are also referred as proposed methods 1 and 2, respectively. As it was discussed in Section 5.2, plenoptic point clouds usually come with a main RGB channel in addition to the N_c plenoptic colors from the different camera viewpoints. In the case of the 8iVSLF dataset, every PPC comes with such information. Therefore, we take advantage of this extra information to assess the performance of both coders in two different scenarios, which are referred here as *Main* and *Plenoptic*: in the former, we apply these coders over the non-transformed main color information. The latter consists of evaluating the plenoptic results by comparing both our compression schemes. In order to adequately compare the plenoptic scenario with the one where we the main color, both the DC and ACs coefficients being encoded in our proposed solution are scaled to fit an 8-bit representation. Results are presented in Table 5.3 with comparison results for the six depth-10 8iVSLF sequences.

Table 5.3: BD-PSNR (in dB) for the Y component of depth-10 PCs, comparing Proposed Method 1 over 2.

PC	Main	Plenoptic	DC	AC 1	AC2
Boxer	−0.02	0.27	0.02	0.29	0.28
Longdress	0.10	0.72	0.26	0.26	0.33
Loot	−0.30	0.24	0.02	0.28	0.32
RedandBlack	0.07	0.29	0.05	0.17	0.21
Soldier	0.02	0.34	−0.07	0.17	0.20
Thaidancer	0.04	0.81	0.37	0.21	0.22
Average	−0.02	0.45	0.11	0.23	0.26

BD-PSNR results with the main color show an even comparison between coders. However, plenoptic results for the proposal 1 present an average BD-PSNR value of 0.45 dB over the proposal 2. In order to investigate where the gains for the RAHT-based scheme come from in the plenoptic

scenario, we broke down the results of the KLT-transformed color vectors by the RD performance of the individual DC and AC coefficient PCs. Some of the results are shown in Table 5.3, in which the DC yields an average BD-PSNR gain of 0.11 dB for the proposal 1 over the proposed method 2, and the first and second ACs yield higher gains of 0.23 dB and 0.26 dB, respectively. We believe that the prediction model in predictive RAHT performs better than LoD's scheme for the transformed color information due to the selection of neighboring voxels. The LoD generation defines the order in which the colors are encoded. This order establishes which attribute values are available as references for prediction, which is based on the *k*-nearest-neighbors algorithm and uses point-to-point Euclidean distance thresholds (see Section 3.1.2.2). Therefore, the encoding order for this approach may not be optimal when using the transformed attributes. Hence, the lower correlation between neighbors for the AC-coefficient PCs may produce larger residuals in method 2 in comparison to method 1. This may lead to worse coding performance over the scaled plenoptic colors. The usage of unscaled coefficients for both methods further accentuates these differences, which is shown in section 5.3.1.2.

5.3.1.2 Proposal Evaluation

Table 5.4 presents the BD-PSNR comparing our proposals 1 and 2 and MV-HEVC, using [6] as an anchor. The BD metrics were computed using the points within the bitrate range that comprises RAHT's quantization stepsizes from 12 to 300. The proposed method 1 significantly outperforms the others for most point clouds. Fig. 5.7 presents RD curves comparing the four methods for the depth-12 *Thaidancer* PC. Results show greater rate-distortion gains for proposal 1 in comparison to MV-HEVC for medium and high-bitrate cases.

Table 5.4: BD-PSNR(in dB) comparisons of the Y component for different methods, using [6] as reference.

Sequence	**Depth 12**			**Depth 10**	
	MV-HEVC	**Method 1**	**Method 2**	**Method 1**	**Method 2**
Boxer	0.55	**1.78**	0.69	**1.83**	0.95
Longdress	3.40	**3.42**	2.17	**2.69**	1.79
Loot	1.88	**2.17**	1.11	**1.82**	0.99
Redandblack	1.66	**2.73**	1.66	**2.36**	1.58
Soldier	2.13	**2.53**	1.54	**1.99**	1.14
Thaidancer	**3.37**	3.33	1.86	**2.46**	1.27

Table 5.4 also presents the BD-PSNR results for both our proposed solutions compared to [6] for the depth-10 PPCs. As is the case with the depth-12 sequences, method 1 offers substantial coding gains in comparison to the method in [6]. Fig. 5.8 presents the RD performance for the *Thaidancer* sequence, which exhibits the same pattern found with the other PPCs tested. Note that the depth-10 results do not show curves for the MV-HEVC solution. This is due to the fact that the MV-HEVC results are only available for depth-12 PPCs.

In addition to that, we also subjectively compare a specific viewing direction of the *Thaidancer*

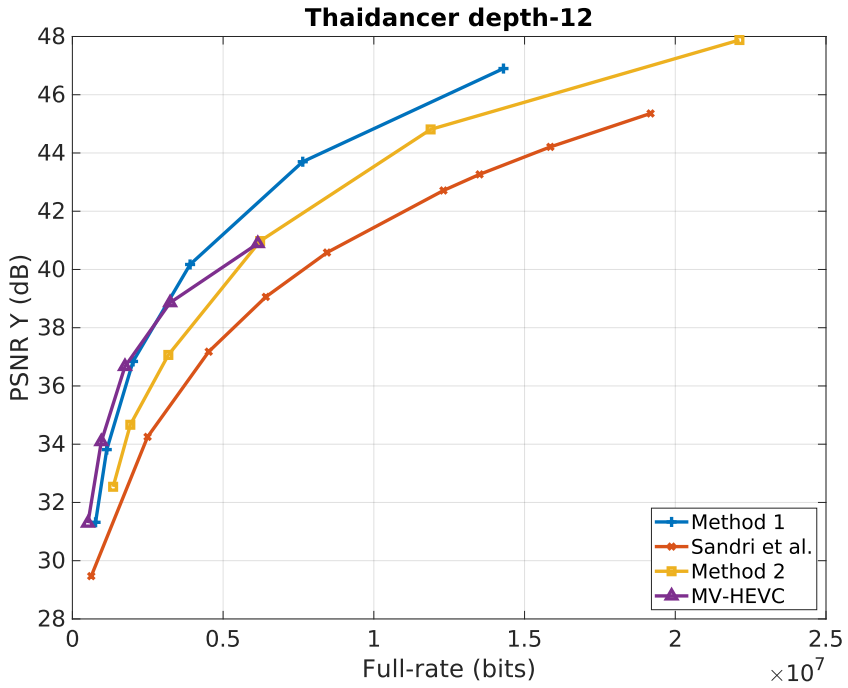


Figure 5.7: Rate-distortion performance over the depth-12 *Thaidancer* for the four compression schemes considered.

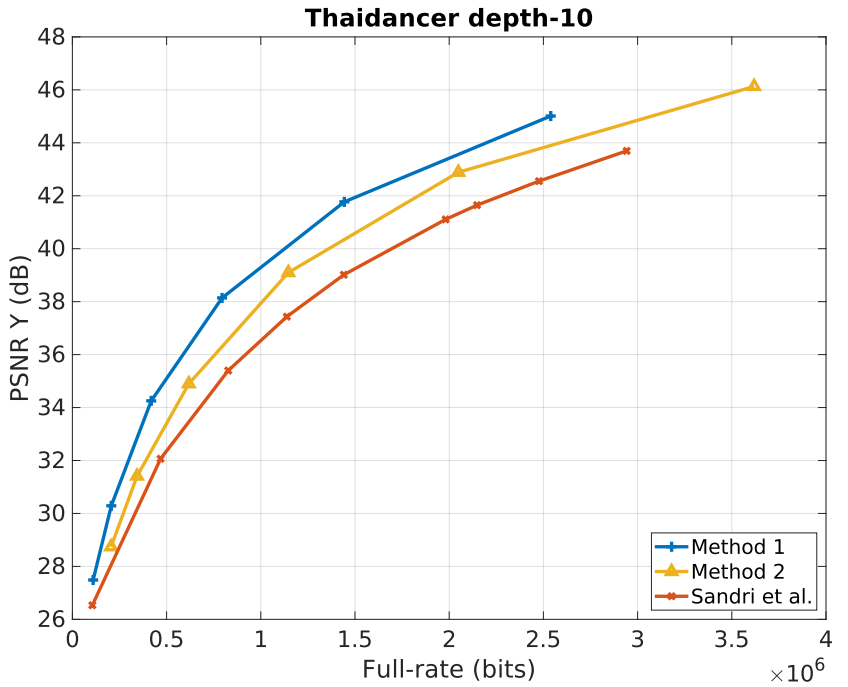
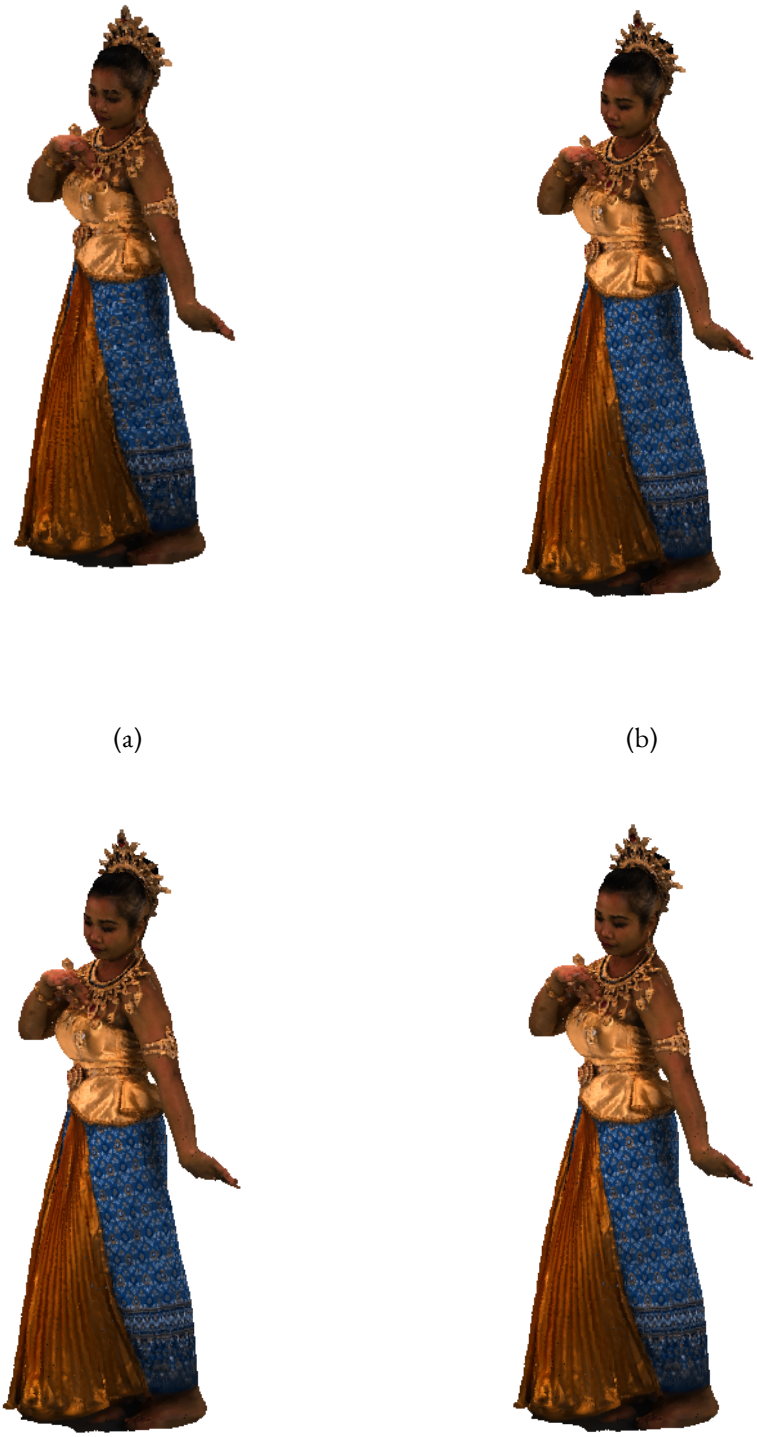


Figure 5.8: Rate-distortion performance over the depth-10 *Thaidancer* for the four compression schemes considered.

sequence in Fig. 5.9 between the original point cloud, the solution from [6] and our two proposed methods. Note that a visual comparison between our Proposed Method 1 and the solution from [6] presents finer details in the golden cloth for the former solution while also presenting a lower bit rate.



(a) (b)

(c) (d)

Figure 5.9: Subjective comparison of the plenoptic depth-10 *Thaidancer* PC. From left to right, top to bottom: (a) Original point cloud; (b) Sandri et al. [6], where the plenoptic data occupies 827273 bits with a PSNR of 35.4 dB in comparison to the original point cloud; (c) Proposed Method 1 (RAHT-based), where the plenoptic textures occupy 794137 bits with a PSNR of 38.1 dB; (d) Proposed Method 2 (LoD-based), with a plenoptic payload size of 1072105 bits and a PSNR of 39.09 dB.

5.3.2 Adding Plenoptic Enhancement into V-PCC

The experiments with the proposed differential and non-differential schemes were carried with version 9.0 of the TMC2 reference software from V-PCC's standard. Our proposals were compared to the current solution available for plenoptic encoding with V-PCC, which consists of encoding the extra colors as multiple point cloud attributes, and to MV-HEVC solution once again. As performed in Section 5.3.1, the distortion in comparisons with MV-HEVC was computed by considering the PSNR between the original and reconstructed Y channels for the N_c cameras. For the other cases, however, the distortion was calculated as a weighted average of the Y, U and V channels:

$$\text{PSNR} = \frac{6\,\text{PSNRY} + \text{PSNRU} + \text{PSNRV}}{8}. \tag{5.3}$$

5.3.2.1 Plenoptic Differential Proposal

In the scenario where the main RGB channel is available as well as the N_c plenoptic attributes (see Section 5.2.1), its rate and distortion values have to be taken into account when evaluating the coding performances of our solutions. Therefore, in this scenario we calculate both the rate and the PSNR considering $N_c + 1$ attributes.

Table 5.5 presents the BD-Rate values for the differential coding solution over encoding with the conventional V-PCC, where the bitrate takes into account the $N_c + 1$ cameras and the PSNR was computed as seen in Eq. 5.3. Results show that the proposed solution presents significant gains over the conventional V-PCC solution. Fig. 5.10 illustrates the RD performance of both methods for the *Longdress* and *RedandBlack* sequences, which exhibit the same pattern found with the other PPCs tested.

Table 5.5: YUV-PSNR BD-Rate for plenoptic differential coding over conventional V-PCC.

Dataset	BD-Rate
Boxer	-72.8%
Longdress	-86.7%
Loot	-67.2%
Redandblack	-81.1%
Soldier	-74.7%
Thaidancer	-82.4%

Table 5.6 presents the BD-Rate values for the plenoptic differential solution and MV-HEVC, using the conventional V-PCC as anchor. Results show that, on average, our proposed differential coding scheme offers greater bitrate savings for all PPCs, with the exception of *Thaidancer*, which presents a difference of less than 1%. Fig. 5.11 presents the RD performance of the three solutions for *Longdress* and *RedandBlack*.

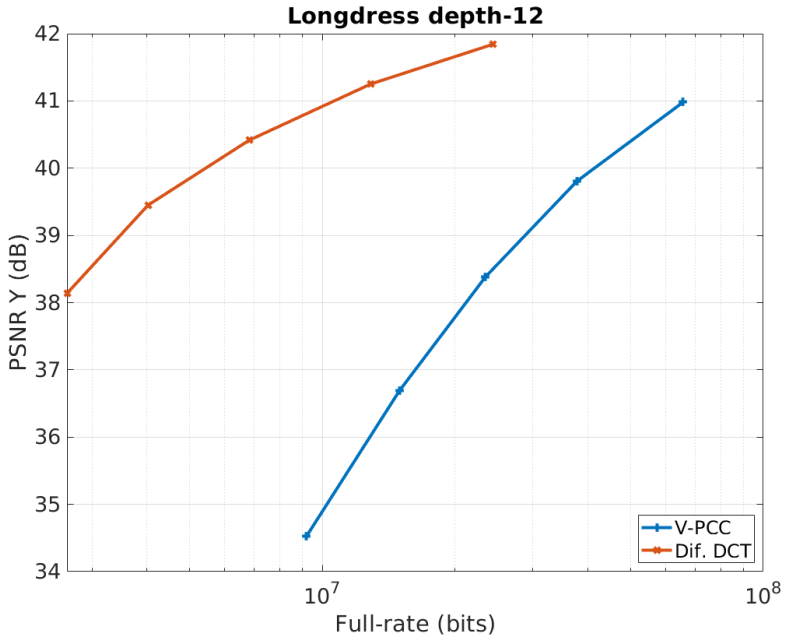


Figure 5.10: RD performance for *Longdress* (top) and *RedandBlack* (bottom) for the plenoptic enhancement differential codec ("Dif. DCT") and conventional V-PCC ("V-PCC").

Table 5.6: YUV-PSNR BD-Rate for Li *et al.* and plenoptic differential coding over conventional V-PCC.

Dataset	MV-HEVC	Differential coding
Boxer	-57.1%	**-74.5%**
Longdress	-81.3%	**-85.5%**
Loot	-61.5%	**-67.9%**
Redandblack	-71.3%	**-80.0%**
Soldier	-68.7%	**-74.7%**
Thaidancer	-79.8%	**-80.9%**

Finally, Fig. 5.12 contains the results comparing the proposed method with the KLT instead of the DCT for optimal energy compaction. Note that the KLT offers little gains over the DCT while being much more expensive in terms of complexity.

5.3.3 Plenoptic Non-Differential Proposal

On the other hand, in the scenario where that main RGB channel is not available (see Section 5.2.2), only the N_c plenoptic attributes are used for computing the bitrate and distortion values. Table 5.7 presents the BD-Rate for the non-differential coding solution over encoding with the conventional V-PCC. Results show once more that the proposed non-differential solution offers significant gains over the V-PCC. Fig. 5.13 shows the RD performance of both methods for the *Longdress* and *RedandBlack* sequences, which exhibit the same pattern found with the other PPCs tested.

Table 5.8 presents the BD-Rate values for the plenoptic non-differential coding scheme and the

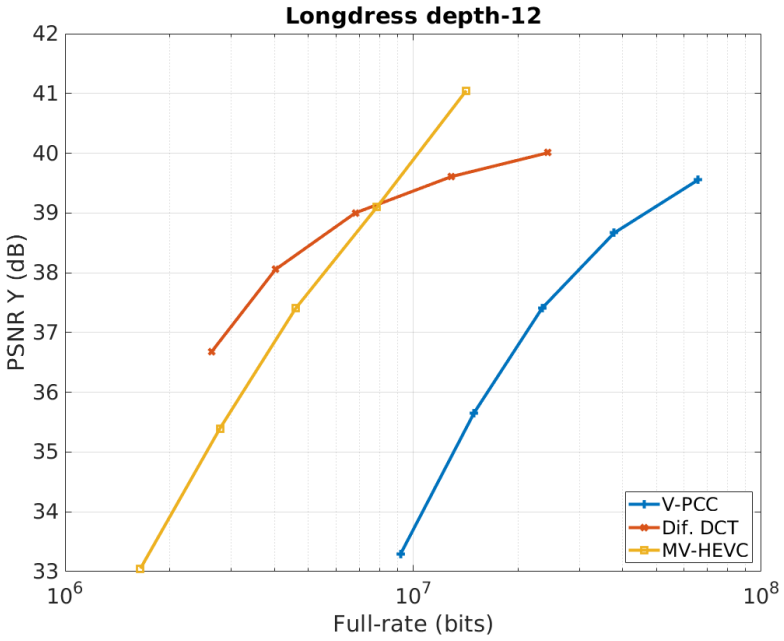


Figure 5.11: RD performance for *Longdress* (top) and *RedandBlack* (bottom) for the plenoptic enhancement differential codec ("Dif. DCT"), conventional V-PCC ("V-PCC") and the method of Li et al. ("MV-HEVC").

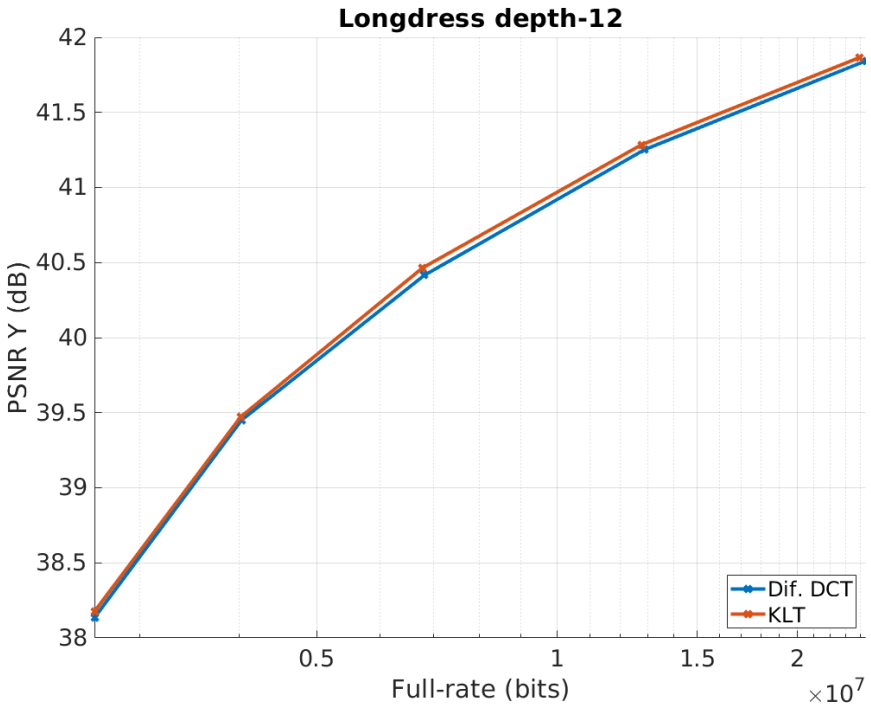


Figure 5.12: RD performance for *Longdress* for the plenoptic enhancement differential codec using the DCT ("Dif. DCT") and the KLT ("Dif. KLT").

MV-HEVC solution, with the conventional V-PCC being used as anchor. These results consider the PSNR of Y channel for the distortion and the N_c cameras for computing the rate. Results show that the proposed method presents greater rate reduction for all PPCs. Fig. 5.14 presents the RD performance for the three solutions for *Longdress*. Results for the other PPCs share a similar

Table 5.7: YUV-PSNR BD-Rate for plenoptic non-differential coding over conventional V-PCC.

Dataset	BD-Rate
Boxer	-77.4%
Longdress	-88.9%
Loot	-75.4%
Redandblack	-84.0%
Soldier	-81.1%
Thaidancer	-86.6%

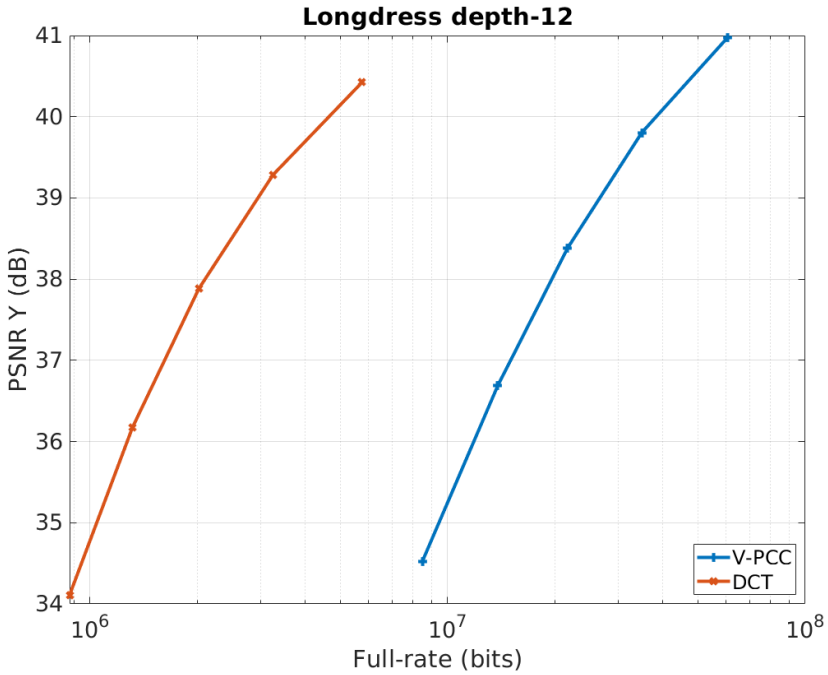


Figure 5.13: RD performance for *Longdress* (top) and *RedandBlack* (bottom) for the plenoptic enhancement non-differential codec ("DCT") and conventional V-PCC ("V-PCC").

pattern.

Finally, we present in Fig. 5.15 a subjective comparison between the decoded versions of the *Longdress* sequence, with the conventional V-PCC, plenoptic differential and non-differential coding, respectively. All methods provide similar results objetively, but with our solutions providing these results at a much smaller bit rate.

Table 5.8: Y-PSNR BD-Rate for Li *et al.* and plenoptic non-differential coding over conventional V-PCC.

Dataset	MV-HEVC	Non-differential coding
Boxer	-61.8%	-78.7%
Longdress	-87.8%	-88.7%
Loot	-66.7%	-76.2%
Redandblack	-77.6%	-84.0%
Soldier	-74.2%	-81.2%
Thaidancer	-85.6%	-86.3%

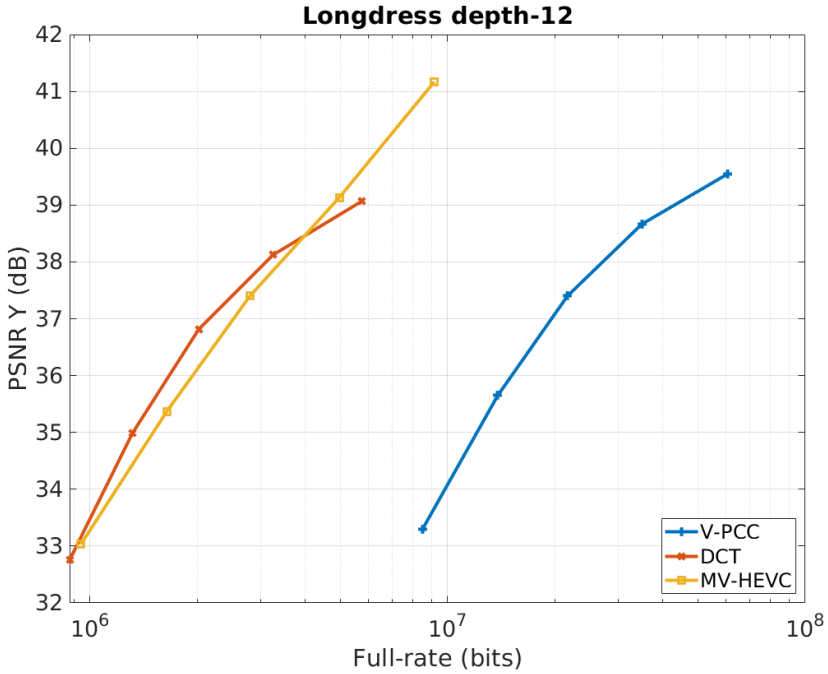


Figure 5.14: RD performance for *Longdress* (top) and *RedandBlack* (bottom) for the plenoptic enhancement non-differential codec ("DCT"), conventional V-PCC ("V-PCC") and the method of Li et al. ("MV-HEVC").

Figure 5.15: Decoded views (cameras 1, 2 and 3) of the plenoptic *Longdress* PC. From top to bottom: original point cloud; conventional V-PCC, where the plenoptic data occupies 8488000 bits, or 9193136 bits when considering the default RGB value triplet, in both cases with a PSNR of 34.5 dB with respect to the original point cloud; plenoptic differential coding, where the plenoptic data occupies 968080 bits when considering RGB, with a PSNR of 34.3 dB; plenoptic non-differential coding, where the plenoptic data occupies 876160 bits, with a PSNR of 34.1 dB.

6 CONCLUSIONS

This research work presented two different solutions for encoding plenoptic point clouds that sought to be efficient but also compliant with the design philosophy of MPEG's existing standards for point cloud compression. For the TMC13 reference software of G-PCC, we proposed a solution consisting of a Karhunen-Loève transform over the plenoptic color attributes followed by multiple attribute coders with intra prediction capability. For the TMC2 reference software of V-PCC, we proposed a video-based solution that is backwards compatible with TMC2 and consists of applying a DCT to the plenoptic color-channel atlases that are encoded by an image/video encoder along with the conventional payload conveyed by this scheme. Moreover, we also looked to highlight the relevance of the development of alternative solutions to the current ones established by MPEG's compression strategies by introducing an algorithm for lossy encoding of the point cloud geometry based on an alternative representation of the point cloud coordinates. By viewing the PC as a 3-D cube that can be partitioned into slices, this representation allows the usage of existing processing methods for bi-level images while still working directly in the 3-D space.

Results show that both our geometry- and video-based approaches provide not only substantial coding gains in comparison to competing approaches but also provide compatibility with G-PCC's and V-PCC's current implementations, respectively. Therefore, we consider these proposed codecs as the state-of-art for geometry- and video-based compression of plenoptic point clouds.

Finally, our codec for lossy geometry also presented competitive results in comparison to the TMC13's existing solutions for lossy geometry compression, in particular for medium to high bitrates, *i.e.*, > 0.3 bits per voxel.

6.1 FUTURE WORK

For our solutions of plenoptic enhancement to the MPEG's standards, future work may focus on accounting for camera occlusions for some of the plenoptic views. Properly dealing with these occlusions might result in even greater rate reductions while minimizing additional reconstruction distortions. On the other hand, future work with the lossy geometry codec may consist of improving the proposed reconstruction algorithms, especially for lower bitrates, as well as expanding the codec operating points of the codec to even lower bitrates.

REFERENCES

[1] C. Tulvan, R. Mekuria, Z. Li, and S. Laserre, "Use Cases for Point Cloud Compression (PCC)," ISO/IEC MPEG JTC 1/SC 29/WG 11, Geneva, CH, Tech. Rep. N16331, Jun. 2016.

[2] R. L. de Queiroz and P. A. Chou, "Motion-Compensated Compression of Dynamic Voxelized Point Clouds," *IEEE Transactions on Image Processing*, vol. 26, no. 8, pp. 3886–3895, 2017.

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

[4] M. Landy and J. A. Movshon, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*, 1991, pp. 3–20.

[5] G. Sandri, R. De Queiroz, and P. A. Chou, "Compression of plenoptic point clouds using the region-adaptive hierarchical transform," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 1153–1157.

[6] G. Sandri, R. L. de Queiroz, and P. A. Chou, "Compression of Plenoptic Point Clouds," *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1419–1427, 2019.

[7] M. Krivokuća and C. Guillemot, "Colour compression of plenoptic point clouds using raht-klt with prior colour clustering and specular/diffuse component separation," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 1978–1982.

[8] X. Zhang, P. A. Chou, M. Sun, M. Tang, S. Wang, S. Ma, and W. Gao, "A Framework for Surface Light Field Compression," in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2595–2599.

[9] D. Naik and S. Schwarz, "[V-PCC] CE2.15 report on Attribute Coding (SLF)," iSO/IEC JTC1/SC29/WG11 MPEG, input document M49123, Jul. 2019.

[10] D. Naik, S. Schwarz, V. K. M. Vadakita, and K. Roimela, "Surface Lightfield Support in Video-based Point Cloud Coding," in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, 2020, pp. 1–6.

[11] L. Li, Z. Li, S. Liu, and H. Li, "Video-Based Compression for Plenoptic Point Clouds," in *2020 Data Compression Conference (DCC)*, 2020, pp. 378–378.

[12] A. B. Tucker, Ed., *Computer science handbook*, 2nd ed. Chapman & Hall/CRC, 2004.

[13] C. K. Chua, C. H. Wong, and W. Y. Yeong, *Standards, quality control and measurement sciences in 3D printing and additive manufacturing*. Academic Press, 2017.

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

[15] R. Mekuria, K. Blom, and P. Cesar, "Design, Implementation, and Evaluation of a Point Cloud Codec for Tele-Immersive Video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 4, p. 828–842, 2017.

[16] C. Tulvan and M. Preda, "Point cloud compression for cultural objects," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Document m37240, Oct. 2015.

[17] R. Cohen, H. Ochimizu, D. Tian, and A. Vetro, "Mobile Mapping System Point Cloud Data from Mitsubishi Electric," ISO/IEC MPEG JTC1/SC29/WG11, Hobart, AU, Input Contribution m40495, Apr. 2017.

[18] ITU-R BT.601-7, "Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios," International Telecommunication Union, Recommendation, May 2011.

[19] ITU-R BT.709-6, "Parameter values for the hdtv standards for production and international programme exchange," International Telecommunication Union, Recommendation, May 2015.

[20] R. Gonzalez, *Digital image processing*, 3rd ed. Upper Saddle River, N.J: Prentice Hall, 2008.

[21] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," *IEEE Transactions on Computers*, vol. C-23, no. 1, pp. 90–93, 1974.

[22] K. Fukunaga, *Introduction to statistical pattern recognition*. Academic Press, 2009.

[23] S. Gebhardt, E. Payzer, L. Salemann, A. Fettinger, E. Rotenberg, and C. Seher, "Polygons, point-clouds and voxels: A comparison of high-fidelity terrain representations," in *Simulation Interoperability Workshop and Special Workshop on Reuse of Environmental Data for Simulation—Processes, Standards, and Lessons Learned*, 2009, p. 9.

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

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

[26] R. L. de Queiroz, D. C. Garcia, P. A. Chou, and D. A. Florencio, "Distance-based probability model for octree coding," *IEEE Signal Processing Letters*, vol. 25, 2018.

[27] G. L. Sandri, "Compression of Point Cloud Attributes," Ph.D. dissertation, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília–DF, Brazil, 2019.

[28] I. Daribo, R. Furukawa, R. Sagawa, and H. Kawasaki, "Adaptive arithmetic coding for point cloud compression," in *2012 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, Oct 2012, pp. 1–4.

[29] W. Zhu, Y. Xu, L. Li, and Z. Li, "Lossless point cloud geometry compression via binary tree partition and intra prediction," in *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*, Oct 2017, pp. 1–6.

[30] S. Milani, "Fast point cloud compression via reversible cellular automata block transform," in *2017 IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 4013–4017.

[31] E. Peixoto, "Intra-Frame Compression of Point Cloud Geometry using Dyadic Decomposition," *IEEE Signal Processing Letters*, pp. 1–1, 2020.

[32] R. Rosário and E. Peixoto, "Intra-Frame Compression of Point Cloud Geometry using Boolean Decomposition," in *2019 IEEE Visual Communications and Image Processing (VCIP)*, Dec 2019, pp. 1–4.

[33] G. P. Sandri, P. A. Chou, M. Krivokuca, and R. L. de Queiroz, "Integer alternative for the region-adaptive hierarchical transform," *IEEE Signal Processing Letters*, vol. 26, no. 9, pp. 1369–1372, sep 2019.

[34] T. M. Borges, "Fractional Super-Resolution of Voxelized Point Clouds," Master's thesis, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília–DF, Brazil, 2020.

[35] 3DG, "PCC Test Model Category 3 v0," ISO/IEC MPEG JTC1/SC29/WG11, Macau, China, Approved WG 11 document w17249, Oct. 2017.

[36] "G-PCC Codec Description v10," iSO/IEC JTC1/SC29/WG11 MPEG, document N19331, Jun. 2020.

[37] K. Mammou, A. Tourapis, J. Kim, F. Robinet, V. Valentin, and Y. Su, "Lifting Scheme for Lossy Attribute Encoding in TMC1," ISO/IEC MPEG JTC1/SC29/WG11, San Diego, US, Input contribution m42640, Apr. 2018.

[38] P. Fechteler, R. Mekuria, P. Cesar, D. Monaghan, N. E. O'Connor, P. Daras, D. Alexiadis, T. Zahariadis, A. Hilsmann, P. Eisert, S. V. Broeck, C. Stevens, J. Wall, M. Sanna, D. A. Mauro, and F. Kuijk, "A framework for realistic 3d tele-immersion," in *Proceedings of the 6th International Conference on Computer Vision / Computer Graphics Collaboration Techniques and Applications - MIRAGE '13*. ACM Press, 2013.

[39] R. Mekuria, C. Tulvan, and Z. Li, "Requirements for point cloud compression," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, MPEG Requierements w16330, Feb. 2016.

[40] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai, "An overview of ongoing point cloud compression standardization activities: video-based (V-PCC) and geometry-based (G-PCC)," *APSIPA Transactions on Signal and Information Processing*, vol. 9, 2020.

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

[42] G. Sullivan and J.-R. Ohm, "Meeting report of the 13th meeting of the joint collaborative team on video coding (jct-vc)," ITU-T SG16 WP3 and ISO/IEC MPEG JTC1/SC29/WG11, Incheon, KR, Report JCTVC-M1000, Apr. 2013.

[43] Wikipedia contributors. (2021) Versatile Video Coding. Wikipedia, The Free Encyclopedia. Accessed: 22-05-2021. [Online]. Available: https://en.wikipedia.org/wiki/Versatile_Video_Coding

[44] ——. (2021) Audio Video Interleave. Wikipedia, The Free Encyclopedia. Accessed: 22-05-2021. [Online]. Available: https://en.wikipedia.org/wiki/Audio_Video_Interleave

[45] M. 3DG, "V-PCC Test Model v8," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Approved WG 11 document N18884, Oct. 2019.

[46] ——, "V-PCC Codec Description," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Approved WG 11 document N18892, Oct. 2019.

[47] H. Hoppe, T. Derose, T. Duchamp, J. Mcdonald, and W. Stuetzle, "Surface reconstruction from unorganized points," *Proceedings of the 19th annual conference on Computer graphics and interactive techniques - SIGGRAPH 92*, 1992.

[48] S. Lasserre and D. Flynn, "[G-PCC] On an improvement of RAHT to exploit attribute correlation," iSO/IEC JTC1/SC29/WG11 MPEG, input document M47378, Jul. 2019.

[49] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," 2020.

[50] M. Domański, O. Stankiewicz, K. Wegner, and T. Grajek, "Immersive visual media — MPEG-I: 360 video, virtual navigation and beyond," in *2017 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2017, pp. 1–9.

[51] T. Ebrahimi, S. Foessel, F. Pereira, and P. Schelkens, "JPEG Pleno: Toward an Efficient Representation of Visual Reality," *IEEE MultiMedia*, vol. 23, no. 4, pp. 14–20, 2016.

[52] 3DG, "Common test conditions for point cloud compression," ISO/IEC MPEG JTC1/SC29/WG11, Gothenburg, SE, Approved WG 11 document N18883, July 2019.

[53] D. Tian, H. Ochimizu, C. Feng, R. Cohen, and A. Vetro, "Geometric distortion metrics for point cloud compression," in *2017 IEEE International Conference on Image Processing (ICIP)*, Sep. 2017, pp. 3460–3464.

[54] G. Bjøntegaard, "Improvements of the BD-PSNR Model," VCEG-AI11, ITU-T SG16/Q6, Berlin, Germany, Tech. Rep., Jul. 2008.

[55] A. L. Souto and R. L. de Queiroz, "On predictive raht for dynamic point cloud coding," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 2701–2705.

[56] C. Dorea, D. C. Garcia, R. U. Ferreira, D. Freitas, R. Higa, R. L. de Queiroz, I. Seidel, and V. Testoni, "Discussion on Common Test Conditions for plenoptic V-PCC," iSO/IEC JTC1/SC29 Joint WG11/WG7 (MPEG/JPEG), input document WG7M55146, Oct. 2020.

[57] M. Krivokuća, P. A. Chou, and P. Savill, "8i Voxelized Surface Light Field (8iVSLF) Dataset," ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG), Ljubljana, input document m42914, July 2018.

[58] C. Dorea, D. C. Garcia, R. U. Ferreira, D. Freitas, R. Higa, R. L. de Queiroz, I. Seidel, and V. Testoni, "Downsampled 8iVSLF Dataset," iSO/IEC JTC1/SC29 Joint WG11/WG7 (MPEG/JPEG), input document WG7M55148, Oct. 2020.