DISSERTAÇÃO DE MESTRADO

ON PREDICTIVE RAHT FOR DYNAMIC POINT CLOUD COMPRESSION

André Luis Souto Ferreira

Brasília, Maio de 2021

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO

ON PREDICTIVE RAHT FOR DYNAMIC POINT CLOUD COMPRESSION

André Luis Souto Ferreira

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, PPGEE / UnB
Orientador

Profa. Mylène Christine Queiroz de Farias, PPGEE / UnB
Examinador Interno

Prof. Eduardo Antônio Barros da Silva, UFRJ
Examinador Externo

FICHA CATALOGRÁFICA

SOUTO, ANDRÉ LUIS
ON PREDICTIVE RAHT FOR DYNAMIC POINT CLOUD COMPRESSION [Distrito Federal] 2021.
xvi, 64 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. raht
3. inter-frame prediction	4. compression
I. ENE/FT/UnB	II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

SOUTO, A. L. (2021). *ON PREDICTIVE RAHT FOR DYNAMIC POINT CLOUD COMPRESSION*. Dissertação de Mestrado, Publicação: PPGEE-Jun/21, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 64 p.

CESSÃO DE DIREITOS
AUTOR: André Luis Souto Ferreira
TÍTULO: ON PREDICTIVE RAHT FOR DYNAMIC POINT CLOUD COMPRESSION.
GRAU: Mestre em Engenharia Elétrica ANO: 2021

André Luis Souto Ferreira
Depto. de Engenharia Elétrica (ENE) - FT
Universidade de Brasília (UnB)
Campus Darcy Ribeiro
CEP 70919-970 - Brasília - DF - Brasil

Acknowledgments

I would like to thank my parents, Gilmar Ferreira and Patricia Souto, for opening all the doors that allowed me to become a student at the University of Brasilia. It would not be possible without them. I would also like to thank my sister, Jéssica Souto, for her unconditional support during this thesis and my girlfriend, Véronique Lemaitre, for all her support and patience during my M.Sc. studies. I would like to thank Profa. Mylène, for helping me a lot when I needed to finish earlier the course of Image Processing. My thesis defense would not be possible without her help. Finally, I would like to thank Prof. Queiroz for his priceless expertise and guidance during the thesis.

André Luis Souto Ferreira

Work partially supported by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Finance Code 001.

ABSTRACT

The increase in 3D applications made necessary the research and development of standards for point cloud compression. Since point clouds represent a significant amount of data, compression standards are essential to efficiently transmit and store such data [1], [2]. For this reason, the Moving Pictures Expert Group (MPEG) started the standardization activities of point cloud compression algorithms resulting in two standards: the Geometry-based Point Cloud Compression (G-PCC) [3] and the Video-based Point Cloud Compression (V-PCC) [4]. G-PCC was designed to address static point clouds, those consisting of objects and scenes, and dynamically acquired point clouds, typically obtained by LiDAR technology. In contrast, V-PCC was addressed to dynamic point clouds, those consisting of several temporal frames similar to a video sequence.

In the compression of dynamic point clouds, algorithms to estimate and compensate motion play an essential role. They allow temporal redundancies among successive frames to be further explored, hence, significantly reducing the number of bits required to store and transmit the dynamic scenes. Although motion estimation algorithms have been studied, those algorithms for points clouds are still very complex and demand plenty of computational power, making them unsuitable for practical time-constrained applications. Therefore, an efficient motion estimation solution for point clouds is still an open research problem.

Based on that, the work presented in this dissertation focuses on exploring the use of a simple inter-frame prediction alongside the region-adaptive hierarchical (or Haar) transform (RAHT) [5]. Our goal is to improve RAHT's attribute compression performance of dynamic point clouds using a low-complexity inter-frame prediction algorithm. We devise simple schemes combining the latest version of RAHT with an intra-frame predictive step added [6] with a low-complexity inter-frame prediction to improve the compression performance of dynamic point clouds using RAHT. As previously mentioned, inter-frame prediction algorithms based on motion estimation are still very complex for point clouds. For this reason, we use an inter-frame prediction based on the spatial proximity of neighboring voxels between successive frames. The nearest-neighbor inter-frame prediction simply matches each voxel in the current point cloud frame to its nearest voxel in the immediately previous frame. Since it is a straightforward algorithm, it can be efficiently implemented for time-constrained applications.

Finally, we devised two adaptive approaches that combine the nearest-neighbor prediction alongside the intra-frame predictive RAHT. The first designed approach is referred to as fragment-based multiple decision, and the second is referred to as level-based multiple decision. Both schemes are capable of outperforming the use of only the intra-frame prediction alongside RAHT in the compression of dynamic point clouds. The fragment-based algorithm is capable of slightly outperforming the use of only the intra-frame prediction with average Bjontegaard delta (BD) [7] PSNR-Y gains of 0.44 dB and bitrate savings of 10.57%. The level-based scheme achieves more substantial gains over the use of only the intra-frame prediction with average BD PSNR-Y gains of 0.97 dB and bitrate savings of 21.73%.

RESUMO

O aumento no número de aplicações 3D fez necessária a pesquisa e o desenvolvimento de padrões para compressão de nuvem de pontos. Visto que nuvens de pontos representam uma quantidade significativa de dados, padrões de compressão são essenciais para transmissão e armazenamento eficientes desses formatos [1], [2]. Por esse motivo, o *Moving Pictures Expert Group* (MPEG) iniciou atividades de padronização de technologias para compressão de nuvens de pontos resultando em dois padrões: o *Geometry-based Point Cloud Compression* (G-PCC) [3] e o *Video-based Point Cloud Compression* (V-PCC) [4]. G-PCC foi desenvolvido para compressão de nuvens de pontos estáticas, aquelas que representam objetos e cenas, e nuvens de pontos adquiridas dinamicamente, obtidas por technologia LiDAR. Por outro lado, V-PCC foi direcionado para compressão de nuvens de pontos dinâmicas, aquelas representadas por diversos quadros temporais semelhantes a sequências de vídeo.

Na compressão de nuvens de pontos dinâmicas, os algoritmos para estimar e compensar movimento desempenham um papel essencial. Eles permitem que redundâncias temporais entre quadros sucessivos sejam exploradas, reduzindo significativamente o número de bits necessários para armazenar e transmitir as cenas dinâmicas. Embora técnicas de estimação de movimento já tenham sido estudadas, esses algoritmos para nuvens de pontos ainda são muito complexos e exigem muito poder computacional, tornando-os inadequados para aplicações práticas com restrições de tempo. Portanto, uma solução de estimação de movimento eficiente para nuvens de pontos ainda é um problema de pesquisa em aberto.

Com base nisso, o trabalho apresentado nesta dissertação se concentra em explorar o uso de uma predição inter-quadros simples ao lado da *region-adaptive hierarchical (or Haar) transform* (RAHT) [5]. Nosso objetivo é melhorar o desempenho de compressão de atributos da RAHT para nuvens de pontos dinâmicas usando um algoritmo de predição inter-quadros de baixa complexidade. Desenvolvemos esquemas simples combinando a última versão da transformada RAHT com uma etapa preditiva intra-quadros adicionada [6] a uma predição inter-quadros de baixa complexidade para melhorar o desempenho da compressão de nuvens de pontos dinâmicas usando a RAHT. Como mencionado anteriormente, os algoritmos de predição inter-quadros baseados em estimação de movimento ainda são muito complexos para nuvens de pontos. Por esse motivo, usamos uma predição inter-quadros com base na proximidade espacial de voxels vizinhos entre quadros sucessivos. A predição inter-quadros do vizinho mais próximo combina cada voxel no quadro de nuvem de pontos atual com seu voxel mais próximo no quadro imediatamente anterior. Por ser um algoritmo simples, ele pode ser implementado de forma eficiente para aplicações com restrições de tempo.

Finalmente, desenvolvemos duas abordagens adaptativas que combinam a predição inter-quadros do vizinho mais próximo ao lado da RAHT com predição intra-quadros. A primeira abordagem

desenvolvida é definida como *fragment-based multiple decision* e a segunda como *level-based multiple decision*. Ambos os esquemas são capazes de superar o uso apenas da predição intra-quadros ao lado da RAHT para compressão de nuvens de pontos dinâmicas. O algoritmo *fragment-based* tem um desempenho ligeiramente melhor se comparado ao uso apenas da predição intra-quadros com ganhos *Bjontegaard delta* (BD) [7] PSNR-Y médios de 0,44 dB e economia média de taxa de bits de 10,57%. O esquema *level-based* foi capaz de atingir ganhos mais substanciais sobre o uso apenas da predição intra-quadros com ganhos BD PSNR-Y médios de 0,97 dB e economia média de taxa de bits de 21,73%.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

Symbols and Acronyms

A	List of attributes associated with the occupied voxels from G.
AC	Coefficients with non-zero frequency.
BD	Bjontegaard Delta.
bpov	Bits per occupied voxel.
DC	Coefficients with zero frequency.
DCM	Direct Coding Mode.
DCT	Discrete Cosine Transform.
G	The geometry of a point cloud, a list of 3D positions, i.e., occupied voxels.
GFT	Graph Fourier Transform.
G-PCC	Geometry-based Point Cloud Compression.
GT	Graph Transform.
HEVC	High Efficiency Video Coding.
I-RAHT	MPEG's RAHT with intra-frame predictive step added.
LoD	Level of Detail.
L-PCC	LiDAR Point Cloud Compression.
MPEG	Moving Pictures Expert Group.
PSNR	Peak Signal-to-Noise Ratio.
PSNR-Y	Peak Signal-to-Noise Ratio for the luminance component.
RA-GFT	Region-Adaptive Graph Fourier Transform.
RAHT	Region-Adaptive Hierarchical (or Haar) Transform.
RLGR	Adaptive run-length Golomb-Rice.
S-PCC	Surface Point Cloud Compression.
TMC13	Test Model Categories 1 and 3 .
TMC2	Test Model Category 2.
3D	Three-dimensions.
2D	Two-dimensions.
V-PCC	Video-based Point Cloud Compression.

1 INTRODUCTION

Due to the increase in 3D applications, such as augmented/virtual reality, 3D telepresence systems, and autonomous navigation, point cloud data have enjoyed a growth in popularity in the past few years [1], [2]. In a nutshell, point clouds consist of a common way to represent 3D images, and they aim to represent the external surface of objects or people. Point clouds can be defined as a set of points in the 3D space, each point having proper geometry and proper attributes. Millions of points may represent such 3D data. Therefore, a significant challenge is to store and efficiently transmit high-quality point clouds. For this reason, efficient compression algorithms for both geometry and attributes are essential for practical 3D applications involving point clouds.

Based on this emerging problem, the Moving Pictures Expert Group (MPEG) issued a call for proposals in 2017 to standardize technologies for compression of point clouds [18]. Based on its responses, initial proposals were adopted for the standardization activities [1]. They evolved over the years with several contributions from technology companies, academic researchers, and research institutions. Due to those efforts, MPEG was able to standardize two technologies for point cloud compression: the Geometry-based Point Cloud Compression (G-PCC) [3] and the Video-based Point Cloud Compression (V-PCC) [4]. The idea behind G-PCC is to work directly in the 3D data, using octrees and transform coding techniques to compress geometry and attributes, respectively. G-PCC was developed for static point clouds, those consisting of only one frame, and dynamically acquired point clouds, those typically obtained by LiDAR technology. On the other hand, V-PCC converts the 3D point clouds into 2D projections and leverages existent video codecs to compress both geometry and attributes. V-PCC was developed for dynamic point clouds, those consisting of several temporal frames, similar to a video sequence.

In the scope of dynamic point clouds, motion estimation and compensation algorithms play a crucial role in compressing such 3D sequences. With them, temporal redundancies among different frames can be further explored, significantly reducing the number of bits needed to represent the dynamic scenes. Several efforts have been made to develop a suitable algorithm for motion estimation of point clouds [16], [17], [19]–[22]. However, for now, those algorithms for points clouds are still very complex and demand plenty of computational power, making them unsuitable for practical applications.

With that in mind, this work focuses on exploring the use of inter-frame prediction alongside the region-adaptive hierarchical (or Haar) transform (RAHT) [5], to improve RAHT's attribute compression performance of dynamic point clouds. The main idea is to develop a simple scheme combining MPEG's intra-frame predictive version of RAHT [6] with a low complexity inter-frame prediction, to improve the compression performance of dynamic point clouds using RAHT. As mentioned before, inter-frame prediction algorithms based on motion estimation are still very complex. Therefore, we opted here to use a simple inter-frame prediction based on the spatial proximity of neighboring voxels between frames. This nearest-neighbor inter-frame prediction matches each voxel in the current point cloud frame to its nearest voxel in the immediately previous frame. This algorithm can be efficiently implemented for time-constrained applications using k-nearest neighbor search, as in the C++ implementation of the Fast Library for Approximate Nearest Neighbors [23].

For testing purposes, two datasets with popular point cloud sequences were used [10], [13]. Performance evaluations compare rate-distortion curves, where the distortion is measured in peak signal-to-noise ratio for the luminance component (PSNR-Y), and the rate is measured in bits per occupied voxel (bpov) in the compressed bitstream.

This manuscript is organized as follows: Chapter 2 presents a review of the basic concepts necessary to understand this work. Chapter 3 describes the research developed here. Chapter 4 shows the datasets used and the results obtained. Finally, Chapter 5 presents a conclusion and suggestions for future work.

1.1 PUBLICATIONS

Publications related to this dissertation:

"On Predictive RAHT for Dynamic Point Cloud Coding". André L. Souto and Ricardo L. de Queiroz. In: *IEEE International Conference on Image Processing (ICIP), October 2020* [24];

"Transformada RAHT com Predição Inter-Quadros para Compressão de Nuvem de Pontos". André L. Souto and Ricardo L. de Queiroz. In: *Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT), November 2020* [25];

"A 3D Motion Vector Database for Dynamic Point Clouds". André L. Souto, Ricardo L. de Queiroz and Camilo Dorea. In: *ArXiv e-prints, August 2020, arXiv: 2008.08438 [eess.IV]* [22];

"Multi-Resolution Intra-Predictive Coding of 3D Point Cloud Attributes". Eduardo Pavez, André L. Souto, Ricardo L. de Queiroz and Antonio Ortega. In: *IEEE International Conference on Image Processing (ICIP), September 2021* [27].

Submitted work currently under review:

"Motion-Compensated Predictive RAHT for Dynamic Point Clouds". André L. Souto, Ricardo L. de Queiroz and Camilo Dorea. Submitted to: *IEEE Transactions on Image Processing 2020* [26].

The submitted paper entitled "Motion-Compensated Predictive RAHT for Dynamic Point Clouds" [26] contains part of the work presented here and an extension of it. We went beyond the use of the nearest-neighbor inter-frame prediction and studied the use of a motion estimation algorithm alongside RAHT. We were able to significantly improve the compression performance of dynamic point clouds using RAHT for all tested point cloud sequences under different motion conditions. However, since the motion estimation algorithm used demanded plenty of computational power, we decided not to include its results in this manuscript. Here, we aim to present only straightforward algorithms that can be efficiently implemented for time-constrained applications and are based on the very simple nearest-neighbor inter-frame prediction.

2 LITERATURE REVIEW

2.1 POINT CLOUDS

A point cloud consists of a set of individual points in the space. Each point contains a 3D location (x, y, and z coordinates) and a list of attributes. The 3D locations are referred to as the point cloud's geometry, and the attributes may depend on its purposes. For instance, they can be colors, usually used to represent objects and people. Alternatively, the point cloud attribute may be reflectance, traditionally employed in autonomous driving applications. Other common attributes are surface normal of each point, and motion vectors [1], [2]. For an N-point cloud, the geometry information can be represented as:

$$\boldsymbol{G} = \{(x_i, y_i, z_i)\}, i = 1, ..., N, \tag{2.1}$$

and the attribute information can be referred to as a list of D attributes for each point

$$\boldsymbol{A} = \{(a_{i,1}, ..., a_{i,D})\}, i = 1, ..., N. \tag{2.2}$$

According to MPEG, point clouds can be classified into three different categories: static (or category 1), dynamic (or category 2), and dynamically acquired (or category 3) [18]. The static point clouds are represented by a single frame, as illustrated in Fig. 2.1, and typically represent objects and scenes. The dynamic ones are represented by several temporal frames, similar to a video sequence, as illustrated in Fig. 2.2, and typically represent people in motion. Finally, the dynamically acquired ones, depicted in Fig. 2.3, are commonly obtained by LiDAR technology and are typically used in autonomous driving applications. The static and dynamic point clouds typically have colors as attributes, while the dynamically acquired ones commonly have reflectance.

Point clouds can be captured using an array of stereo cameras such as the ones used by 8i Labs, illustrated in Fig. 2.4. In this case, the disparity between cameras provides information about the depth of the scene and is used to obtain the geometry information. Microsoft Kinect cameras may also be used in the acquisition process. They use infrared light to generate a depth map of the scene and are also capable of acquiring RGB information, therefore, outputting RGB-D data [28]. For applications such as autonomous navigation that need outdoor areas to be mapped, LiDAR sensors are typically used. They can provide a 3D representation of outdoor spaces by targeting an object with a laser and measuring the time for the reflected light to return to the receiver.

For efficient processing, point clouds are voxelized, each point is constrained to a small cube of dimensions $1 \times 1 \times 1$ with integer coordinates, referred to as voxels. The space where the point cloud resides is divided in a regular grid representing the voxels position and points are assigned to voxels depending on their geometry, that is, if a point falls within the region delimited by a voxel it is

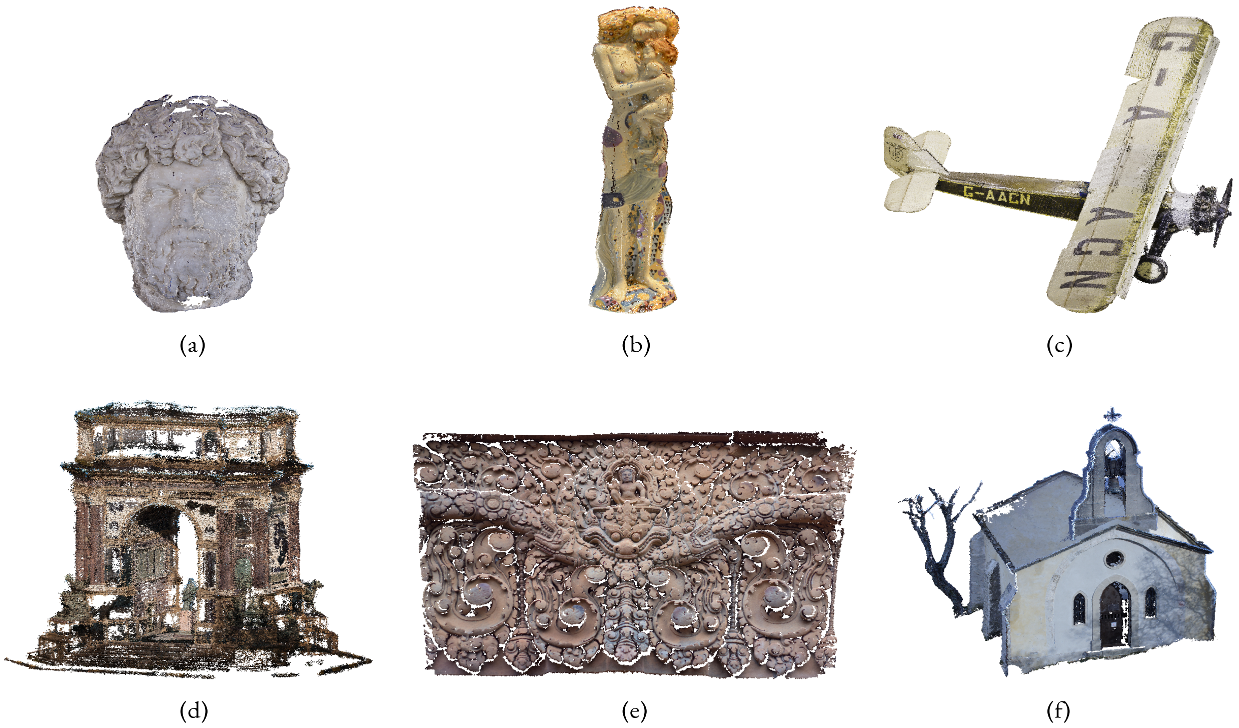


Figure 2.1: Examples of static point clouds [8], [9].

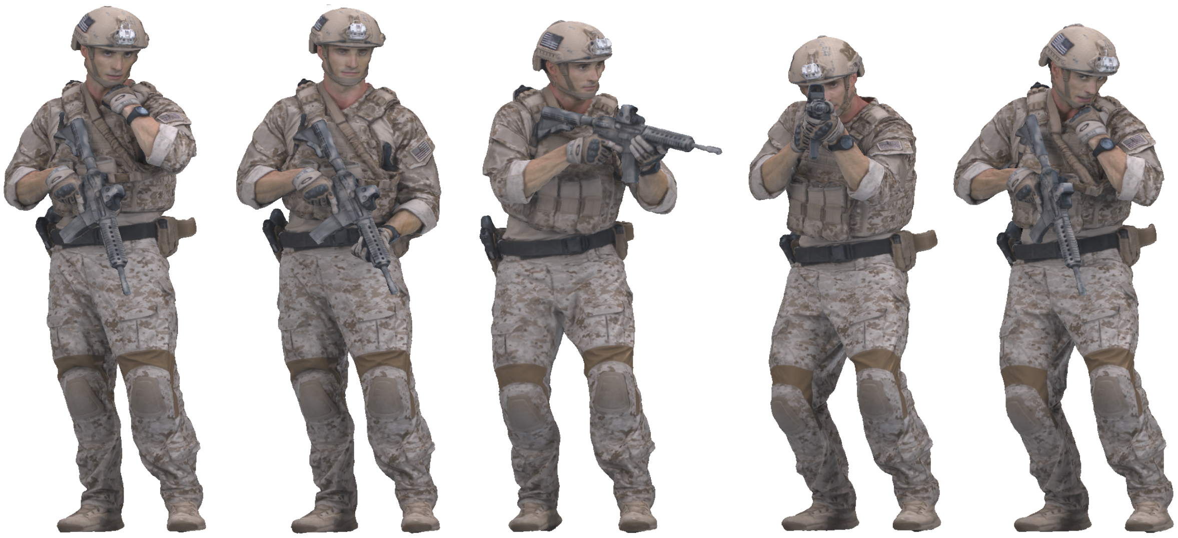


Figure 2.2: Examples of dynamic point clouds [10].

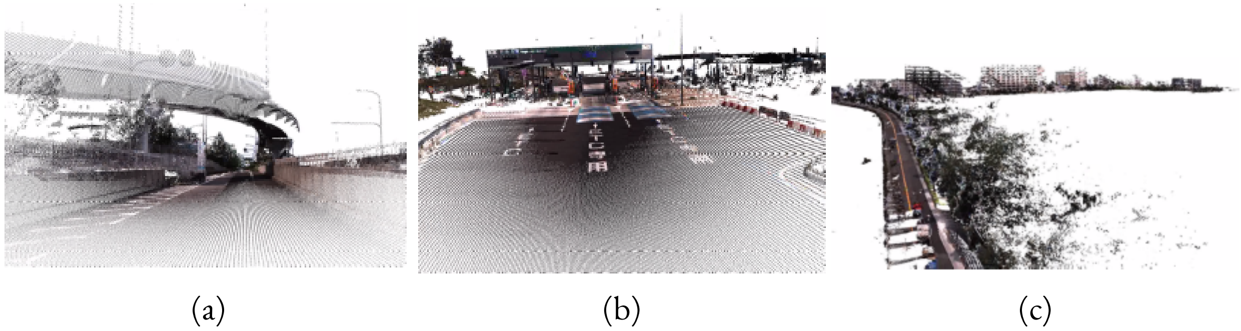


Figure 2.3: Examples of dynamically acquired point clouds [11].

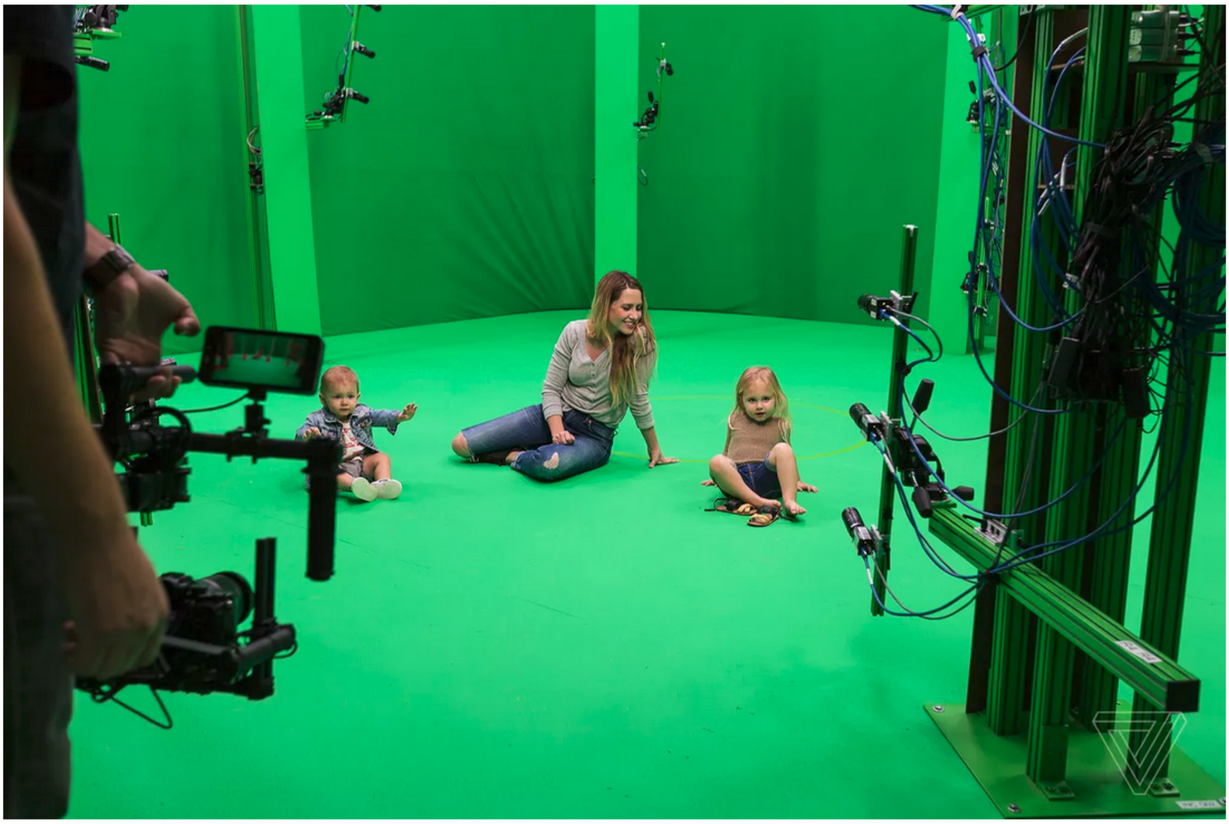
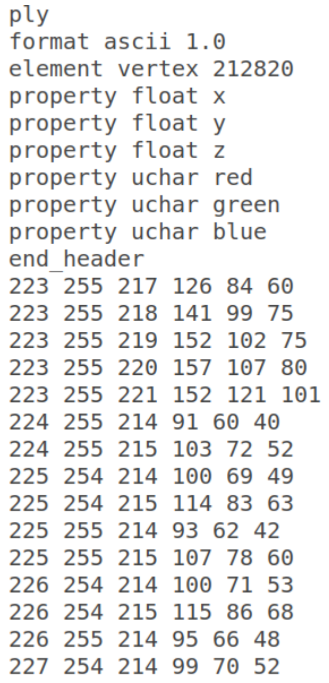


Figure 2.4: Array of cameras for point cloud acquisition at 8i [12].

assigned to that voxel. A voxel in 3D is analogous to a pixel in 2D. In contrast with images, in which all pixels are occupied, in point clouds, not all voxels contain points. A voxel containing one point is said to be occupied, while the empty ones are defined as empty voxels. Note that the datasets used in this dissertation, from Microsoft Research Labs [13] and from 8i Labs [10], consisted of point clouds that were previously voxelized.

Point clouds are commonly stored in the “Polygon File Format”, also known as the “Stanford Triangle Format”. This file format has extension PLY and may be available either in binary format or in ASCII format. An example of a PLY file in ASCII format is illustrated in Fig. 2.5 for a frame of the point cloud sequence “Ricardo” [13]. The first ten lines correspond to the header of the file, indicating the number of points and the point cloud attributes. After the header, at each row, the information of each point’s location and its attributes are shown. In the case of geometry, the first three values at each line represent x, y, and z coordinates. While in the case of colors, the last three values represent one channel each (from 0 to 255) of the RGB color system.

The point clouds used in this work have bit-depths 9 and 10. The bit-depth represents the number of bits needed to represent each point of the geometry. In terms of size, the point clouds used here have from 200,000 to around 1,000,000 points. Each one of these points has geometry and color information. Therefore, similar to images and videos, point clouds may be represented by a large amount of data. For this reason, to store and transmit such 3D data efficiently, proper



```
ply
format ascii 1.0
element vertex 212820
property float x
property float y
property float z
property uchar red
property uchar green
property uchar blue
end_header
223 255 217 126 84 60
223 255 218 141 99 75
223 255 219 152 102 75
223 255 220 157 107 80
223 255 221 152 121 101
224 255 214 91 60 40
224 255 215 103 72 52
225 254 214 100 69 49
225 254 215 114 83 63
225 255 214 93 62 42
225 255 215 107 78 60
226 254 214 100 71 53
226 254 215 115 86 68
226 255 214 95 66 48
227 254 214 99 70 52
```

Figure 2.5: Example of the first 25 lines of a PLY file from point cloud "Ricardo" [13].

compression algorithms of both geometry and attributes are needed.

2.2 POINT CLOUD COMPRESSION

As explained before, point clouds in their raw format require a considerable amount of data to be manipulated. Therefore, compression is essential for practical applications. In order to compress point clouds, the geometry and the attributes are typically processed in a separate manner.

In 2013, MPEG first considered the use of point clouds for immersive telepresence applications and conducted subsequent discussions on how to compress this type of data properly [29], [30]. With increased use cases of point clouds made available and presented [31], and seeing the current need to develop efficient point cloud compression technologies, MPEG issued a Call for Proposals in 2017. In the Call for Proposals, MPEG proposed exploring the three different point clouds using different compression strategies. The Surface Point Cloud Compression (S-PCC) addressed category 1 containing static point clouds. Category 2, with dynamic point clouds, was assigned to the Video-based Point Cloud Compression (V-PCC). Finally, category 3, comprising of dynamically acquired point clouds, was assigned to the LIDAR Point Cloud Compression (L-PCC). Later on, S-PCC and L-PCC were merged into the G-PCC after noticing similarities between the compression approaches used [1], [2].

Afterwards, MPEG issued two reference softwares: the Test Model Category 2 (TMC2) for V-PCC and the Test Model Categories 1 and 3 (TMC13) for G-PCC. TMC2 and TMC13 have been subjected to efforts from several research institutions worldwide, turning into powerful point cloud compression technologies representing the current state-of-the-art. In Sections 2.2.1 and 2.2.2, an overview of the techniques that have been developed for geometry and attribute compression are presented. In Sections 2.2.3 and 2.2.4, brief descriptions of the standards V-PCC and G-PCC are presented.

2.2.1 Geometry Compression

Popular examples of geometry encoder are the compression algorithms based on octrees [32], [33]. Such tree structures are often used in 3D graphics, and recently, they have also become popular in geometry compression algorithms as the field of point cloud compression emerged. Octree-based algorithms leverage the octree structure to compress the spatial location of points, and they have been shown to be very efficient [34]. The octree can be defined as a tree data structure in which each node has eight children (with the leaves as exceptions). Octrees consist of the 3D extension of quadtrees, in which they are used to partition the 3D space by recursively subdividing it into eight nodes. Typically, the octree starts as the root node corresponding to the entire point cloud space. The node is then subdivided in the directions x, y, and z. At each direction, each node is partitioned into two subnodes. This partitioning step is repeated until reaching the voxel level, where each node has dimensions $1 \times 1 \times 1$. The octree is illustrated in Fig. 2.6. Fig. 2.6-(a) shows the entire space as a single node, further subdivided in one direction (x, y, or z), let's say x direction, generating two subnodes. That is, in Fig. 2.6-(b), the entire space is subdivided into two nodes. Figs. 2.6-(c) and 2.6-(d) show the further subdivision in y direction and in z direction, respectively. In Fig. 2.6-(d), one can see the eight nodes after the subdivisions in each direction x, y, and z. Finally, in Fig. 2.6-(e), the tree is partitioned in the x direction again. As mentioned before, this procedure is repeated until voxel level.

In the scope of point cloud compression, octrees can be efficiently used to encode the geometry information, as mentioned before. The compression can be done by traversing the octree from the roots until the leaves and signaling the occupied nodes. For each node in the tree, a bit 1 is sent to signal that this node bounds at least one occupied voxel. On the other hand, if the node does not have any occupied voxels, a bit 0 is sent, and this node is referred to as an empty node. With this information, the point cloud geometry can be properly encoded. Therefore, the subdivisions referred to as empty nodes can be ignored for the next level, which means that the nodes signaled with a bit 0 will not be further divided because they do not carry any points of the cloud. This procedure is illustrated in Fig. 2.7. In this example, in level 1 of the octree, only two nodes are referred to as occupied nodes. Therefore, they are the only ones that are further divided into eight subnodes. In level 2, seven nodes are signaled as occupied. Hence, the sequence 010010001000101100110010 is enough to encode the positions associated with the sample point cloud shown in Fig. 2.7.

This octree-based representation of the point cloud geometry allows the coder to have a progressive functionality. Since the octree can be decoded by level, the application can stop the decoding at any chosen level [35], [36]. Octree-based compression methods can also be combined with entropy

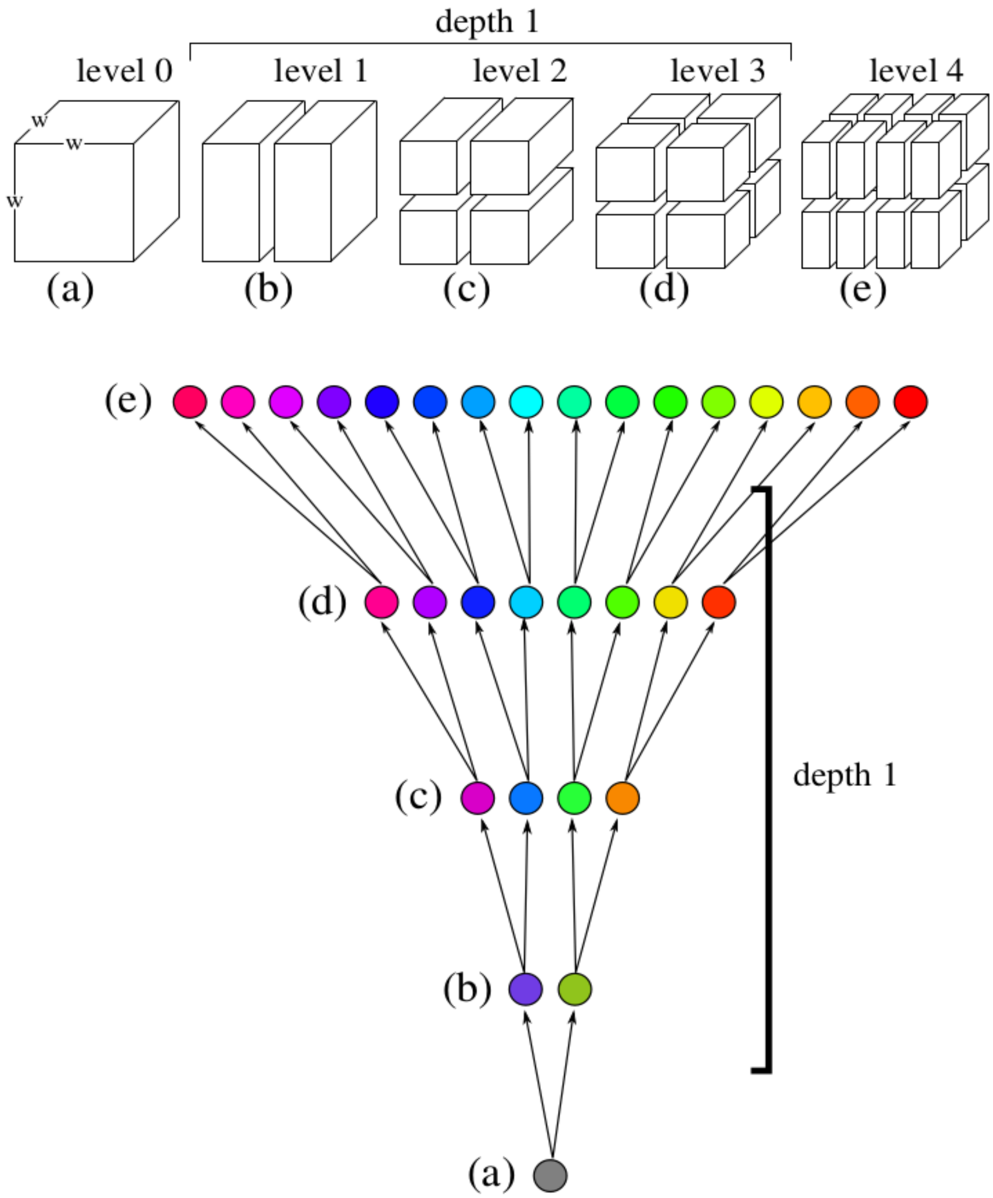


Figure 2.6: Illustration of the octree subdivision per direction [14].

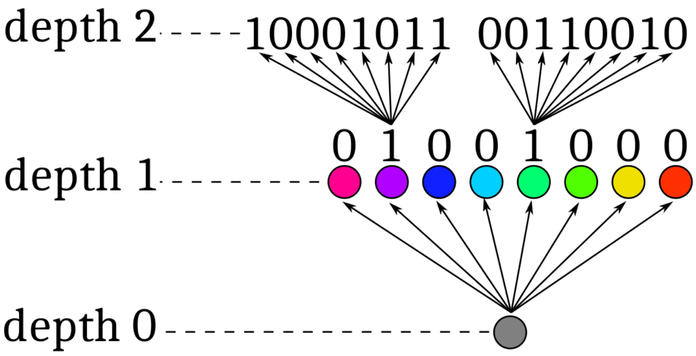


Figure 2.7: Example of geometry encoding of a sample point cloud using octree [14].

encoders to further exploit intra-frame and inter-frame redundancies by using contexts [36], [37].

2.2.2 Attribute Compression

The attributes of a point cloud are typically encoded separately from its geometry. For this reason, their ordering has to be preserved between encoder and decoder. That is necessary to associate the attributes to their corresponding 3D location correctly. Commonly, the geometry is encoded first and then the attributes. This is done because some transform coding techniques use the geometry information to compress the attributes better. So far, attribute compression of point clouds has been widely explored, some methods such as the discrete cosine transform (DCT) [38] and the graph transform (GT) [39] were used in the initial efforts. Currently, some of the popular methods are: the region-adaptive hierarchical transform (RAHT) [5], the predicting and lifting transforms [40], [41], and the recently introduced region-adaptive graph Fourier transform (RA-GFT) [15].

One of the most successful early attempts in attribute compression was the GT [39]. It was later replaced by RAHT, a transform with better performance and also smaller complexity. Recently, RAHT was outperformed by RA-GFT, which has a better performance and may have similar complexity. Therefore, RA-GFT currently represents the state-of-the-art of non-predictive transform coding. The current version of G-PCC incorporates an intra-frame predictive version of RAHT [6], which is still superior to RA-GFT. This predictive RAHT represents the state-of-the-art in non-video-based attribute compression. However, it leverages an intra-frame prediction step to achieve such outstanding performance. Also, G-PCC includes two other transform techniques: the predicting and the lifting transforms. The work presented in this manuscript was based on RAHT and its intra-frame predictive version, hence, the two approaches will be explained in detail in the Sections 2.3 and 2.3.1. This Section will briefly explain the other techniques here.

2.2.2.1 Predicting/Lifting Transform

The predicting transform is a predictive scheme that depends on a Level of Detail (LoD) representation. The LoD, illustrated in Fig. 2.8, distributes the input points into sets of refinement levels (R) using a criterion based on the Euclidean distance. The attributes of each point are encoded using a prediction determined by the LoD order, in which the attribute values of a refinement level R_ℓ are always predicted using the attribute values of its k-nearest neighbors in the previous LoD ($\text{LoD}_{\ell-1}$).

The predicting transform [40] is implemented using the operators split and merge. While the lifting transform is built on top of the predicting transform, introducing an update operator and an adaptive quantization scheme. The forward and inverse predicting/lifting scheme is depicted in Fig. 2.9. Let L_ℓ be the set of attributes associated with LoD_ℓ, and H_ℓ the set of attributes associated with R_ℓ. The split operator takes L_ℓ as an input and returns the low-resolution samples $L_{\ell-1}$ and the high-resolution samples $H_{\ell-1}$. The merge operator takes L_ℓ and H_ℓ and returns $L_{\ell+1}$.

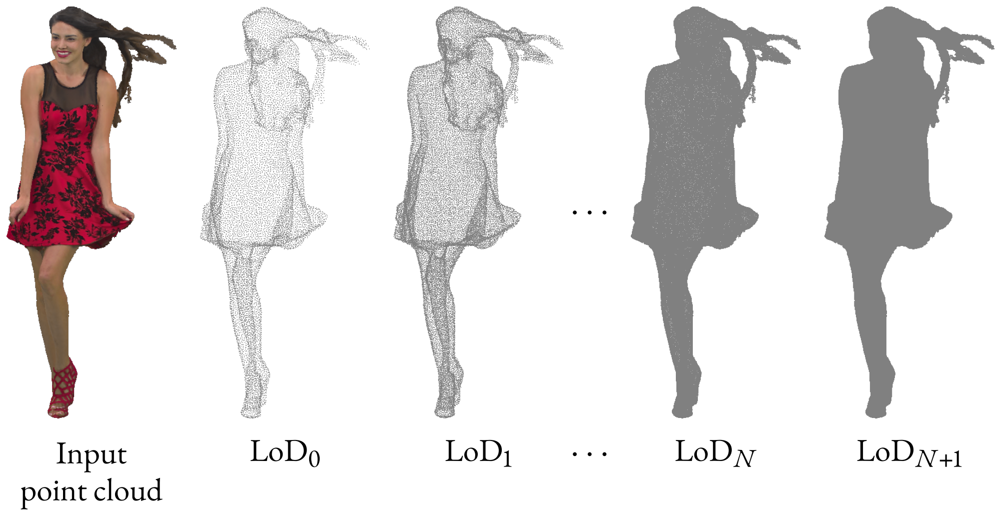


Figure 2.8: LoDs of point cloud “Redandblack” [9], [10].

The prediction operator is used to estimate H_ℓ, by using an interpolation based on the inverse distance weighted average of the k-nearest neighbors of R_ℓ in $\text{LoD}_{\ell-1}$, and then, the residues D_ℓ are computed.

In the case of the lifting transform [41], an update operator and an adaptive quantization strategy are introduced. In the prediction scheme, each point is associated with an influence weight based on its impact on the encoding process. Since the points in lower LoDs are used more frequently, they are associated with higher weights. The introduced update operator determines U_ℓ based on the residual D_ℓ, then, L_ℓ is updated based on U_ℓ, generating L'_ℓ.

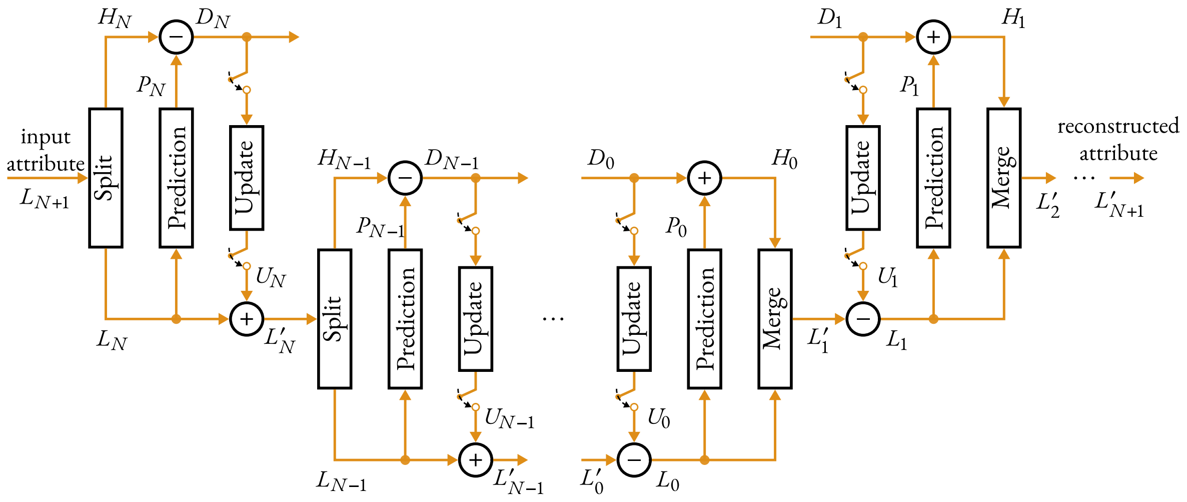


Figure 2.9: Forward and inverse predicting/lifting scheme [9].

Both predicting and lifting transforms are currently part of G-PCC for attribute compression of point clouds of categories 1 and 3. According to [3], predicting is commonly used for point clouds of category 1, while lifting is typically employed for category 3.

2.2.2.2 Region-Adaptive Graph Fourier Transform

The RA-GFT is a recently developed technique introduced in the IEEE International Conference on Image Processing 2020. It is a multi-resolution transform that was inspired by RAHT. The idea behind RA-GFT is to traverse the octree backwards, from the leaves to the root, combining spatially localized blocks at each level. Each block corresponds to a cube bounding a different number of occupied voxels. The blocks are combined based on a block transform that produces one low-pass coefficient and several high-pass coefficients. The low-pass coefficient is promoted to the upper level of the octree, while the high-pass coefficients are quantized and entropy encoded. This procedure is repeated for all blocks at each level of the octree. When at the tree root, the low-pass coefficient generated is also quantized and entropy encoded alongside all the high-pass coefficients.

Within each block, a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \boldsymbol{W}, \boldsymbol{Q})$ is first constructed by adding edges if the distance between a pair of point coordinates is below a fixed threshold, while edge weights are the reciprocal of the distance, as in [39]. In the graph $\mathcal{G}$, $\mathcal{V}$ denotes the vertex set, $\mathcal{E}$ represents the edge set, $\boldsymbol{W}$ is the edge weight matrix, and $\boldsymbol{Q}$ is a diagonal matrix with diagonal entries containing the number of descendants of each node. Also, $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is the combinatorial graph Laplacian matrix, where $\boldsymbol{D} = diag(d_i)$, with $d_i = \sum_j w_{ij}$, is the degree matrix. Since each block in the tree may contain a different number of internal points, the block transform should incorporate the importance of each node based on the number of points it bounds. For this reason, a proposed graph Fourier transform (GFT) is used, which is given by the eigenvectors of the $\boldsymbol{Q}$-normalized graph Laplacian,

$$\boldsymbol{L_Q} = \boldsymbol{Q}^{-\frac{1}{2}} \boldsymbol{L} \boldsymbol{Q}^{-\frac{1}{2}} = \boldsymbol{\Phi} \boldsymbol{\Lambda} \boldsymbol{\Phi}^{\intercal}, \tag{2.3}$$

where $\boldsymbol{\Phi}$ is the matrix of eigenvectors, $\boldsymbol{\Lambda}$ is the matrix of eigenvalues, and $\boldsymbol{\Phi}^{\intercal}$ is defined to be the block transform of RA-GFT with inverse $\boldsymbol{\Phi}$.

In Fig. 2.10, RA-GFT applied to a hypothetical point cloud with nine occupied voxels and three levels ℓ is illustrated. The circles represent the low-pass coefficients, and the squares represent the RA-GFT coefficients (low-pass coefficient of root node and all high-pass coefficients) to be quantized and entropy encoded. Consider the subtree of the attributes a_1, a_2, and a_3 at the highest resolution level ($\ell = L = 2$). The attributes are first transformed by $\boldsymbol{\Phi}^{\intercal}_{1,1}$, generating a low-pass $\hat{a}^1_{1,0}$ coefficient and two high-pass $\hat{a}^1_{1,1}$ and $\hat{a}^1_{1,2}$ coefficients. Then, $\hat{a}^1_{1,0}$ is promoted to level $\ell = 1$ and $\hat{a}^1_{1,1}$ and $\hat{a}^1_{1,2}$ are quantized and entropy encoded. The procedure is analogous for the other two subtrees at level $\ell = 2$. At level $\ell = 1$, the promoted low-pass coefficients $\hat{a}^1_{1,0}$, $\hat{a}^1_{2,0}$ and $\hat{a}^1_{3,0}$ are then transformed by $\boldsymbol{\Phi}^{\intercal}_{0,1}$, generating the tree root low-pass and high-pass coefficients that are quantized and entropy encoded.

RA-GFT was implemented with different block sizes b_L (2, 4, 8, and 16) at the level of highest resolution (leaves of the octree), but with the same block sizes $b_\ell = 2$ for all other resolutions. For point clouds with colors as attributes, RA-GFT outperforms RAHT by up to 0.5 dB for $b_L = 2$. Coding performance improves as b_L increases, achieving up to 2.5 dB over RAHT for $b_L =$

16. Nevertheless, RA-GFT does not perform very well for point clouds of category 3, due to the sparsity of the points of such LiDAR point clouds.

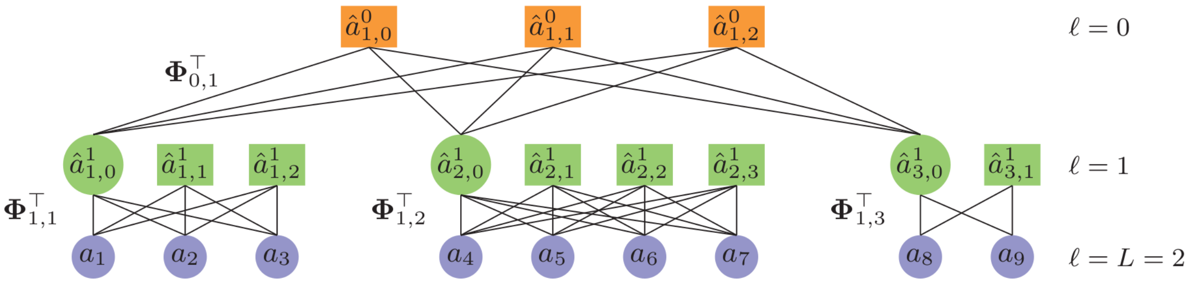


Figure 2.10: RA-GFT applied to a sample point cloud. Circles represent low-pass coefficients that are further transformed, while squares contain RA-GFT coefficients [15].

An alternative for the dedicated techniques for both geometry and attribute compression in the 3D space is the use of state-of-the-art video compression algorithms, such as the High-Efficiency Video Coding (HEVC). In this case, the idea is to convert the 3D point clouds into 2D projections and compress both the geometry and the attributes at the same time [4]. Note that this is a lossy compression algorithm where both geometry and color information are degraded. The advantage of leveraging existent video codecs is the current availability of dedicated hardware for video compression [42]. This technique is successfully implemented in the V-PCC standard and will be explained in the next Section.

2.2.3 Video-based Point Cloud Compression

The V-PCC is a standard technology for point cloud compression that has been a research topic for some years. V-PCC was the subject of several efforts from researchers in industry and academia, and now, it is a very powerful technology for dynamic point cloud compression, representing the state-of-the-art of such point cloud category [4]. The main elements of V-PCC codec architecture are patch generation, patch packing, image padding, and video compression. Note that all the parts mentioned above will be explained here based on the reference software TMC2. An overview of V-PCC architecture can be seen in Fig. 2.11.

To generate the patches, initially, the normal of each point in the cloud is computed. Then, given the six orthographic projection directions ($\pm x$, $\pm y$, $\pm z$), each point is associated with a projection direction based on its normal. Since multiple points may be projected onto the same pixel, TMC2 uses maps to store the overlapped points.

The generated patches are then packed into 2D images. This packing procedure is performed in an iterative way that guarantees an overlap-free insertion into the 2D image. TMC2 tries to insert similar patches of different frames at similar locations to generate a temporally consistent packing in order to improve compression efficiency.

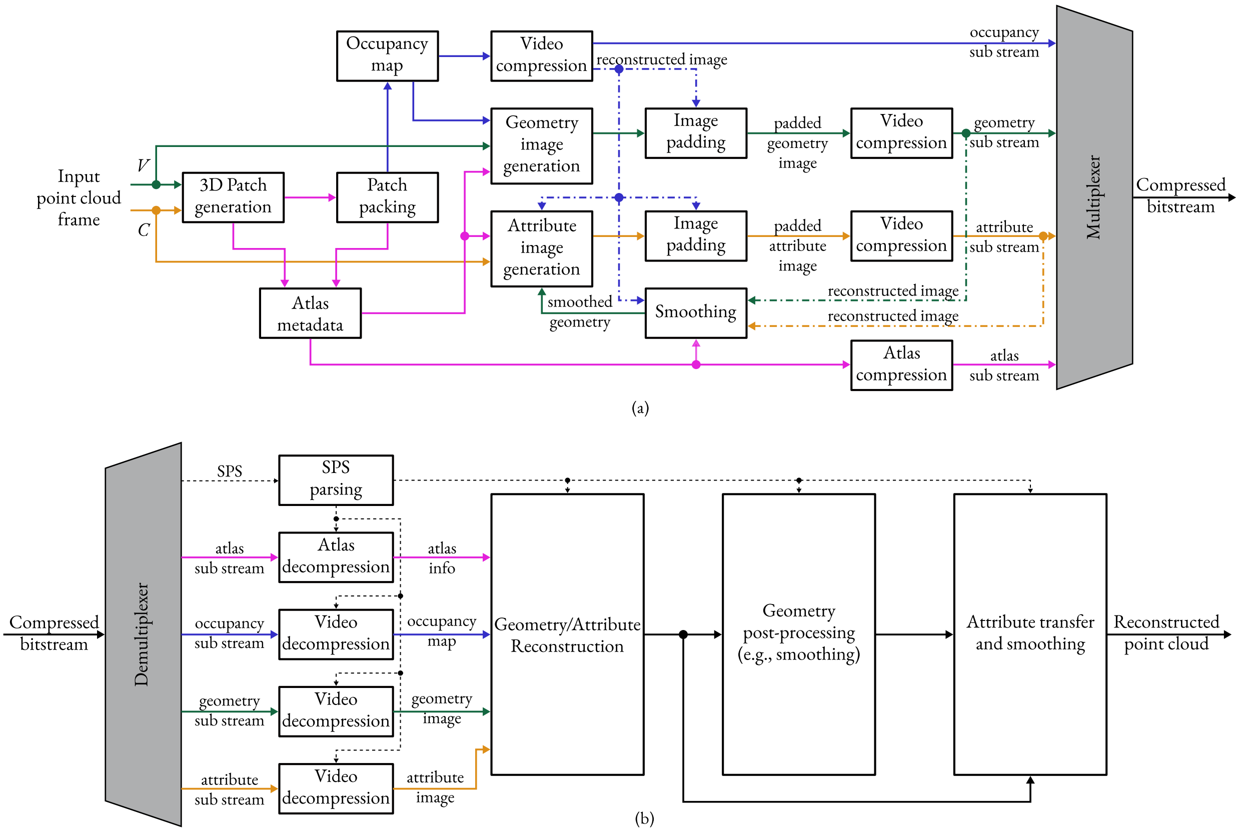
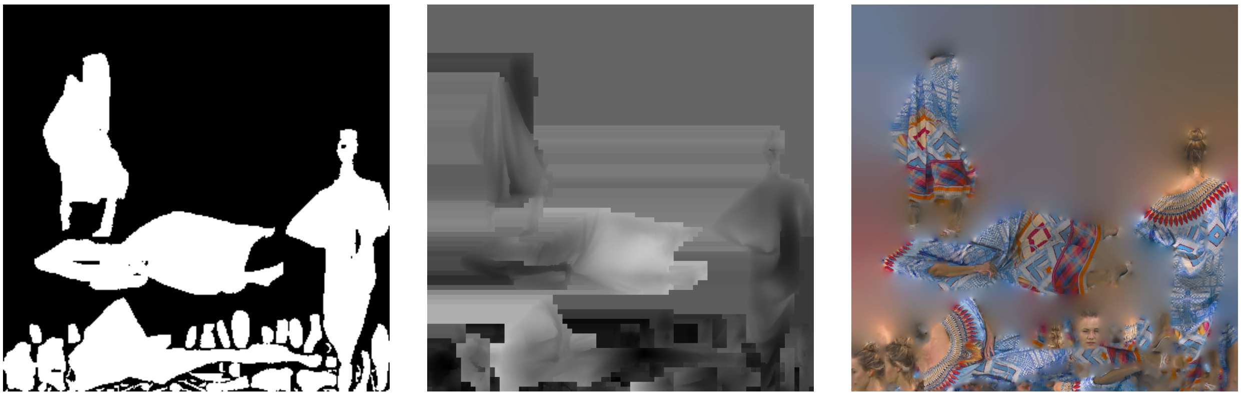


Figure 2.11: Overview of TMC2 architecture [9].

The patch generation and patch packing steps produce an atlas of three 2D images: a 3D patch occupancy map, a 3D patch geometry image, and a 3D patch attribute image. The 3D patch occupancy map signals whether a pixel corresponds to a valid 3D projected point. The occupancy map is necessary since some pixels of the 2D images may stay empty after packing. The 3D patch geometry image contains the depth information of each point regarding its projection plane. Finally, the 3D patch attribute image includes the attribute information for each projected point. Fig. 2.12 illustrates packed patches of occupancy map, geometry, and attributes.



(a) Occupancy map

(b) Geometry

(a) Attribute

Figure 2.12: Example of patch packing [9].

To generate piecewise-smooth images for higher video compression efficiency, the geometry and attribute images are padded. That is, the empty spaces between patches are filled in a process called dilation. Each geometry and attribute images of a specific frame are padded independently of the other frames. Then, the generated atlas information is compressed using a video codec. In TMC2, the video codec used is the HEVC. However, this is not mandatory, and any appropriate image or video codec could be used.

2.2.4 Geometry-based Point Cloud Compression

As explained before, V-PCC is a projection-based scheme that leverages video codecs to compress point clouds. Hence, V-PCC does not encode the point cloud content directly in the 3D space. In the opposite direction of V-PCC, G-PCC works directly on the 3D space using data structures and transforms [3]. Another difference from V-PCC is that G-PCC was designed for static and dynamically acquired point clouds as a consequence of it being originated from the merging of L-PCC and S-PCC. Therefore, G-PCC is limited to intra-frame and does not use any temporal prediction tools.

Similar to what was done for V-PCC, the main codec principles of G-PCC are explained in this Section based on the reference software TMC13. G-PCC encodes the geometry and attributes separately, nevertheless, since attribute coding depends on the geometry, the positions of the points are always encoded and decoded first.

That said, TMC13 starts by processing the geometry information. Since G-PCC is agnostic to the input data coordinate representation, the first step is a coordinate transformation, followed by a voxelization pre-processing step. Then, the second step performed by TMC13 is the geometry analysis using an octree or trisoup scheme. The octree scheme was explained in Section 2.2.1. Nevertheless, TMC13 entropy encodes the octets considering the correlation with neighboring octets. In the case of points isolated, there is no neighborhood to exploit correlation, hence, a Direct Coding Mode (DCM) can be used [43]. In DCM, the coordinates of the isolated points are directly encoded without any compression. Alternatively, the geometry can be encoded using the trisoup scheme. In this approach, the geometry is represented as a pruned octree, that is, an octree from the root until an arbitrary level where the nodes are larger than a voxel. The object surface is then approximated by a series of triangles with no connectivity information among them. Finally, the resulting data structure is arithmetically encoded.

After encoding the geometry, TMC13 encodes the attribute information. When encoding a point cloud with colors as attributes, the first step is to convert them from RGB color space to YUV previously to compression. In the case of reflectance, the color space conversion step is not performed. Then, one of the three transforming tools explained in Section 2.2.2 can be used: RAHT, predicting transform, or lifting transform. Typically, RAHT and lifting are used for colors, and predicting is used for reflectance. Finally, following the transform coding step, the generated coefficients are quantized and arithmetically encoded.

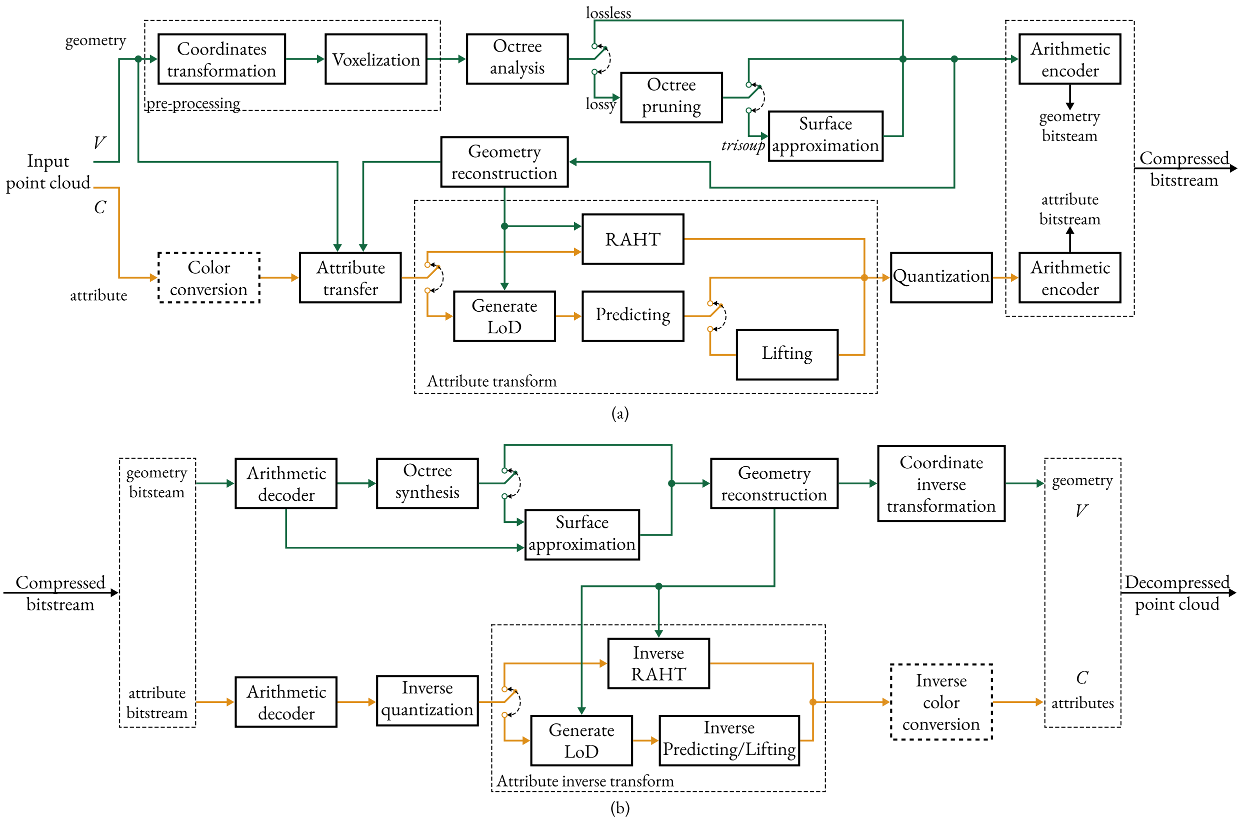


Figure 2.13: Overview of TMC13 architecture [9].

2.3 REGION-ADAPTIVE HIERARCHICAL TRANSFORM

The region-adaptive hierarchical (or Haar) transform (RAHT) [5] is a hierarchical orthogonal sub-band transform, that resembles an adaptive variation of a Haar wavelet transform. RAHT was developed to compress point cloud attributes, such as colors of point clouds representing objects and people, and reflectance of point clouds used in autonomous driving applications. RAHT was developed to be an efficient transform in both compression and time performances, being able to efficiently compress attributes of point clouds in real-time [5].

The basic idea behind RAHT is to use the color attributes associated with a node in a lower level of the octree to predict the colors of the nodes in the next level [5]. The transform follows the octree scan backwards, from voxels to the entire point cloud space. At each step, voxels are recombined along each dimension (x, y, z) into larger ones until reaching the root. For a given level ℓ, let $C^{\ell}_{x,y,z}$ be the average voxel color at position (x, y, z) in the space. $C^{\ell}_{x,y,z}$ is obtained by transforming $C^{\ell+1}_{2x,y,z}$ and $C^{\ell+1}_{2x+1,y,z}$ along dimension x. Only occupied voxels are subject to transformation, that is, if one of the voxels in the pair $C^{\ell+1}_{2x,y,z}$ and $C^{\ell+1}_{2x+1,y,z}$ is unocupied, let's say $C^{\ell+1}_{2x+1,y,z}$, the other one is promoted to the next level: $C^{\ell}_{x,y,z} = C^{\ell+1}_{2x,y,z}$. This grouping process is repeated for each direction, for each level ℓ, until the root of the octree. Each transformation of nodes generates high-pass and low-pass coefficients. The high-pass coefficients are quantized and entropy encoded while the low-pass ones are passed to the next level of the octree. RAHT's core transformation [5] is defined as

$$\begin{bmatrix} C^{\ell}_{x,y,z} \\ H^{\ell}_{x,y,z} \end{bmatrix} = T_{(w_1,w_2)} \begin{bmatrix} C^{\ell+1}_{2x,y,z} \\ C^{\ell+1}_{2x+1,y,z} \end{bmatrix}, \tag{2.4}$$

where

$$T_{(w_1,w_2)} = \frac{1}{\sqrt{w_1+w_2}} \begin{bmatrix} \sqrt{w_1} & \sqrt{w_2} \\ -\sqrt{w_2} & \sqrt{w_1} \end{bmatrix}, \tag{2.5}$$

and where $w_1 = w^{\ell+1}_{2x,y,z}$ and $w_2 = w^{\ell+1}_{2x+1,y,z}$ are the number of leaf voxels, or weights, represented by the nodes to be combined [5]. According to (2.5), the transform matrix $T_{(w_1,w_2)}$ changes at all times, adapting to the weights of each node [5].

In the last stage of the transform, when at the tree root, the two remaining voxels represented by $C^1_{0,0,0}$ and $C^1_{0,0,1}$ are grouped and transformed into two final coefficients: $C^0_{0,0,0}$ and $H^0_{0,0,0}$. The low-pass coefficient of the tree root $C^0_{0,0,0}$ (or DC) is also entropy encoded alongside all the high-pass coefficients generated during the transform steps. In Fig. 2.14, RAHT applied to a $2 \times 2 \times 2$ node is illustrated.

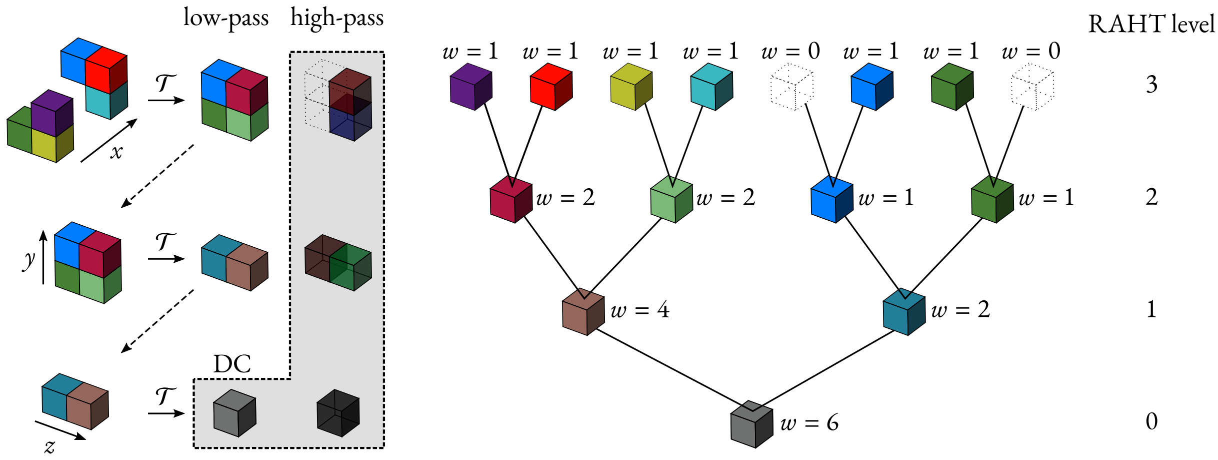


Figure 2.14: RAHT process applied to a $2 \times 2 \times 2$ node with four occupied sub-nodes [9].

In [44], the compression performance of RAHT was further improved. The RAHT coefficients were reordered in ascending order according to their associated depth (depth of the octree where they were generated), and encoded using adaptive run-length Golomb-Rice (RLGR) [45]. The depth-sorted case yields longer runs of zeros, and may lead to faster adaptation and better compression with the RLGR algorithm.

Due to its time efficiency and superior compression performance relative to GT and DCT, as shown in Fig. 2.15, RAHT was initially adopted as part of the G-PCC test model. However, in the latest G-PCC versions, the original RAHT has been modified. Despite the fact that the core of the transform remains the same, the current version consists of a fixed-point implementation [46] with an intra-frame prediction step added. The intra-frame prediction step is explained in Section 2.3.1 and it was introduced due to significant performance improvement over the original RAHT [6].

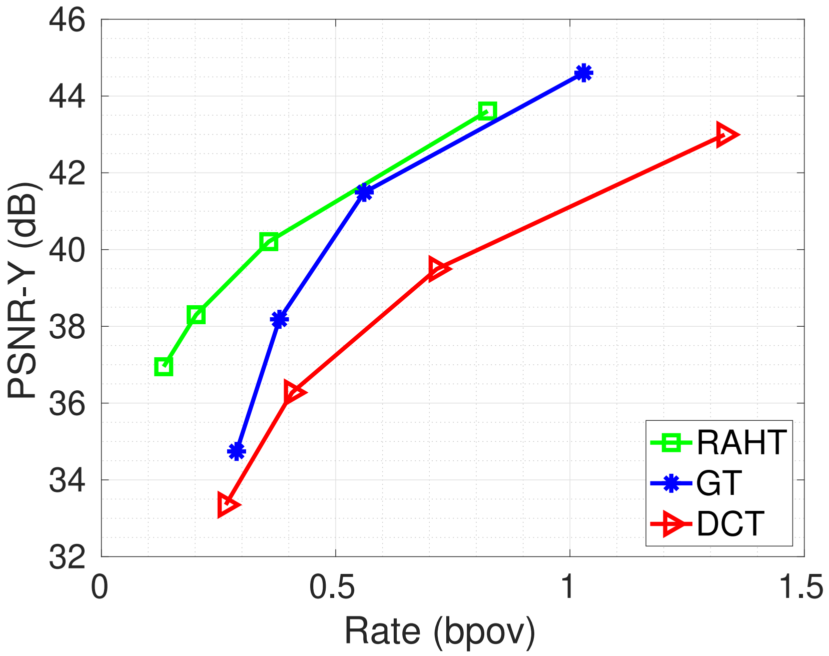


Figure 2.15: Performance of RAHT compared to GT and DCT for point cloud “Ricardo”.

2.3.1 Intra-Frame Predictive RAHT

The latest G-PCC [3] includes a version of RAHT with an intra-frame prediction step added [6]. This predictive version, commonly referred as “Upside down RAHT” and referred here as “I-RAHT”, computes the high-pass and low-pass coefficients in a different manner. In opposition to the original RAHT, which computes the coefficients in a bottom-up approach, i.e., from the leaves of the octree until the root, the current version of RAHT adopted a top-down approach, i.e., the computation goes from the root of the octree to the leaves. This change in the order of coefficient computation does not affect the compression performance of the transform, however, the top-down approach allowed the addition of an intra-frame prediction step to explore spatial correlation among neighboring nodes.

The idea behind I-RAHT, is to predict the attributes of the next level of the octree and transform them and the original attributes, in order to encode only the residues of coefficients. Consider a node at level ℓ of the octree where its attributes are represented by $C^{\ell}_{x,y,z}$. Let $C^{\ell+1}_{2x,y,z}$ and $C^{\ell+1}_{2x+1,y,z}$ be the attributes of the two occupied sub-nodes (out of eight) at level $\ell+1$ and $\hat{C}^{\ell+1}_{2x,y,z}$ and $\hat{C}^{\ell+1}_{2x+1,y,z}$ be the estimations of $C^{\ell+1}_{2x,y,z}$ and $C^{\ell+1}_{2x+1,y,z}$, respectively. The intra-frame prediction performs an upsampling of the node $C^{\ell}_{x,y,z}$ with its occupied neighboring nodes to estimate $\hat{C}^{\ell+1}_{2x,y,z}$ and $\hat{C}^{\ell+1}_{2x+1,y,z}$. The upsampling, illustrated in Fig. 2.16, simply consists of a weighted average of

$C^{\ell}_{x,y,z}$ with its neighbors, where the weights are determined by the inverse of the distances of the node to the upsampled sub-node. Then, both estimated (i.e. $\hat{C}^{\ell+1}_{2x,y,z}$, $\hat{C}^{\ell+1}_{2x+1,y,z}$) and original (i.e. $C^{\ell+1}_{2x,y,z}$, $C^{\ell+1}_{2x+1,y,z}$) sub-node attributes are subject to RAHT transform in (2.4), thus, generating original and estimated coefficients. The differences in between the original and estimated coefficients are taken in order to obtain residual coefficients which are quantized and entropy encoded. This procedure is repeated for all nodes at each level ℓ of the octree. Note that, in this intra-frame predictive approach, the differences of the coefficients of the original and of the predicted attributes are quantized and entropy encoded instead of the coefficients of the attribute differences. An illustration of I-RAHT process for one node is depicted in Fig. 2.17.

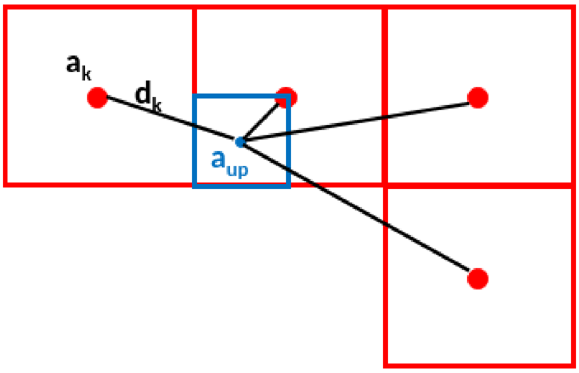


Figure 2.16: Upsampling of the next level of the octree used in I-RAHT [6].

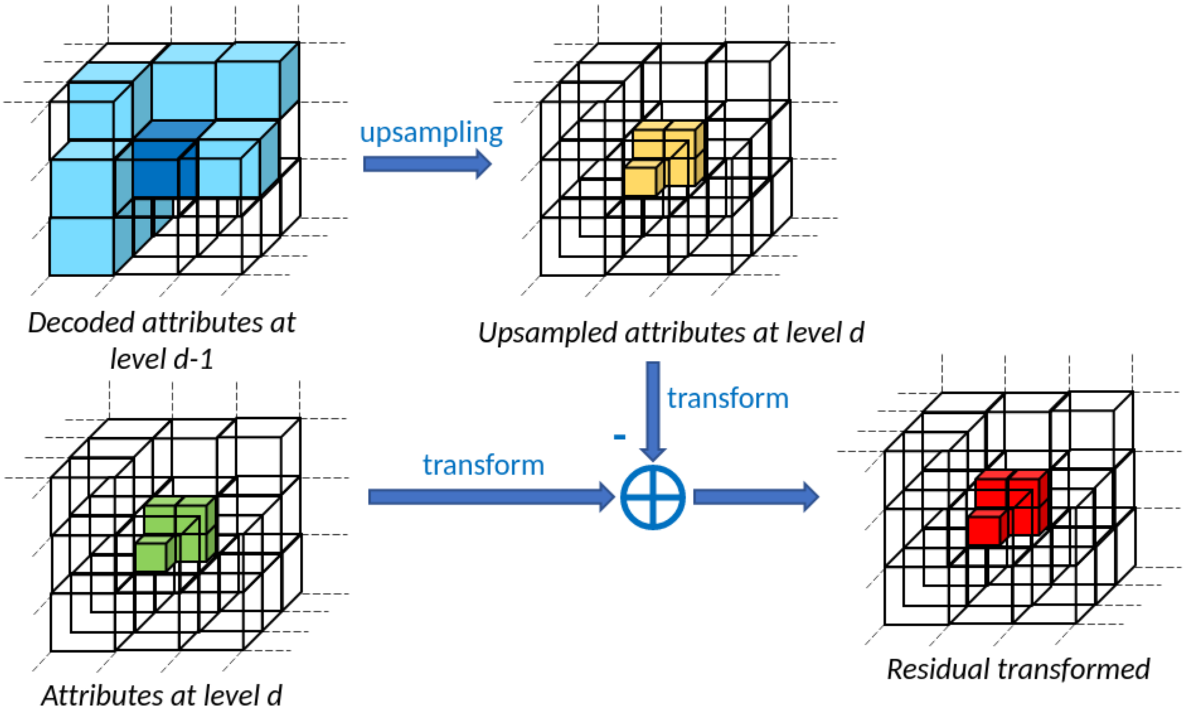


Figure 2.17: Illustration of I-RAHT process for one node.

Fig. 2.18 shows the performance of I-RAHT relative to RAHT for point cloud "Ricardo". It is possible to see that, due to its intra-frame prediction, I-RAHT can achieve significant gains over RAHT. According to [6], the average reported gains of I-RAHT over RAHT is around 30% for point clouds with colors, and around 15% for point clouds with reflectance. Due to its massive gains over the original RAHT, I-RAHT was adopted in the lastest versions of G-PCC, being the current version of RAHT present in the point cloud compression standard.

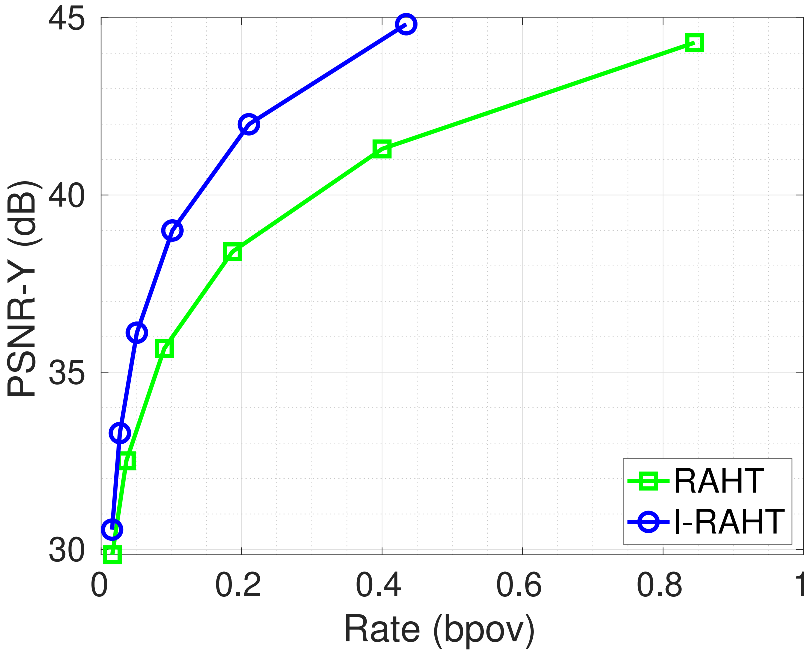


Figure 2.18: Performance of I-RAHT compared to RAHT for point cloud "Ricardo".

2.4 INTER-FRAME PREDICTION OF POINT CLOUDS

In a video sequence, several successive frames may be very similar, with only a few objects moving from one frame to the next. For this reason, when compressing videos, algorithms that exploit temporal redundancies have been successfully used to improve compression performance. In that sense, inter-frame prediction techniques play an important role in compression of dynamic sequences. Commonly used in video codecs, motion estimation algorithms consist of a type of inter-frame prediction that aims to predict the motion of objects in between frames and represent such motion as motion vectors to be encoded. In the decoder side, the motion vectors obtained are used to compensate an adjacent frame to obtain a prediction of a target frame. By the use of such motion estimation techniques, a significant amount of storage space may be saved.

In the scope of point clouds, motion estimation algorithms have the same goal. They aim to exploit temporal redundancies over different frames with significant object motion through the use of motion vectors. At the decoder side, the information can be predicted by motion compensating the adjacent frame with decoded motion vectors. However, unlike video sequences, successive point cloud frames may have different geometries, that is, a different number of voxels in different 3D locations. For this reason, algorithms for accurate motion estimation and compensation of point clouds are not trivial.

Such motion estimation algorithms for point clouds have been object of different studies [16], [17], [19]–[22]. In [16], a graph-based approach is proposed where features are used to find correspondences in temporally successive point clouds. Spectral graph wavelets computed on the color and on the geometry signals of the point cloud are used as the feature descriptors. Then, these wavelet coefficients are used to compute point-to-point correspondences between graphs of different frames, and only a subset of best matching correspondences are used to define a sparse set of motion vectors. The obtained sparse set of motion vectors describe the temporal correlation in the point cloud sequence. Finally, a dense motion field, that completely maps two successive frames, is interpolated based on the sparse set of motion vectors obtained. This approach is illustrated in Fig. 2.19.

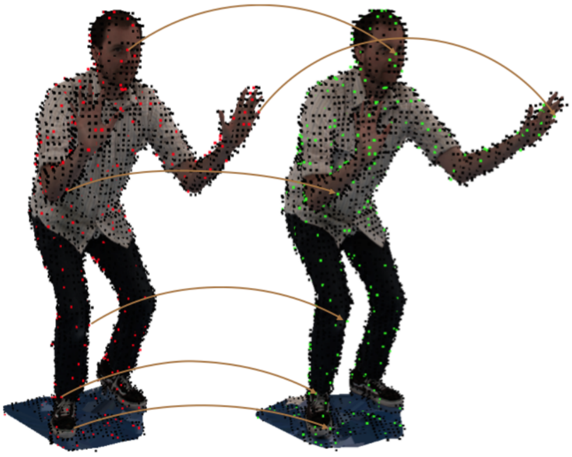


Figure 2.19: Illustration of the sparse set of correspondences in between point clouds using the graph-based motion estimation technique [16].

In [19], a block-based motion estimation approach is used. The algorithm divides the point clouds into blocks of dimensions $16 \times 16 \times 16$, and computes the motion vectors using a rigid transform $\boldsymbol{T}$. Note that, all the blocks have the same dimensions, however, they may have a different number of occupied voxels inside. The rigid transform is a 4×4 matrix, that consists of a 3×3 rotation matrix $\boldsymbol{R}$ and a 3×1 translation matrix $\boldsymbol{t}$. However, $\boldsymbol{T}$ of a certain block is only computed if the iterative closest points algorithm for such block converges to below a certain threshold.

In [17], [20]–[22], a block-based approach is also used. But in this case, the target frame is partitioned into several blocks of size $M \times M \times M$, where $M = 8$ or $M = 16$. For each block, a best block match is searched for, in a search space of dimensions $S \times S \times S$, in the adjacent frame. The best match is computed based on a metric of similarity in between the compared blocks. The metric take into consideration the colors of the points within each block, and their relative location inside each block. The motion vector is obtained by the location differences of the target block and its adjacent match. This block-based approach is capable of achieving good predicted frames, however, the comparison of blocks represents a very time consuming step. The efforts made in [20], [21] focused on reducing the time needed for the computation of the best block match, however, the accuracy of the predictions was also reduced. The block-based approach is illustrated in Fig. 2.20.

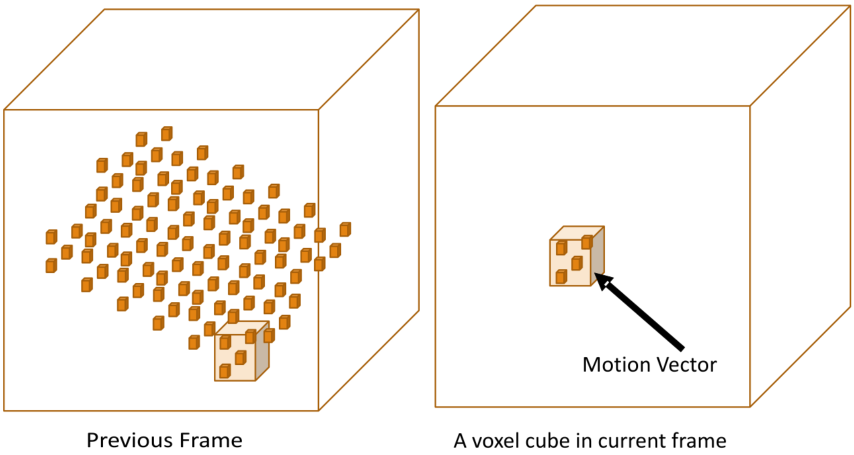
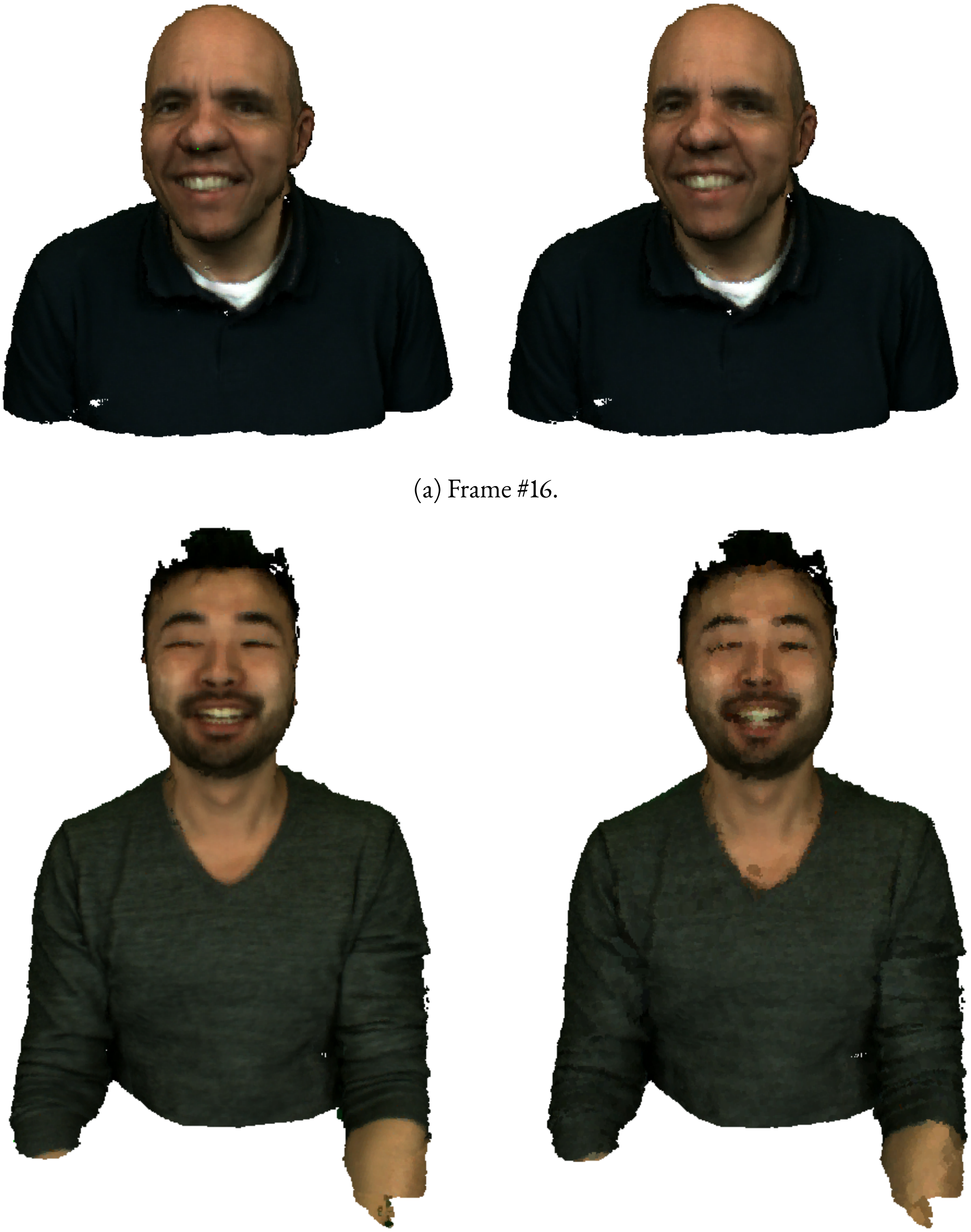


Figure 2.20: Illustration of the block-based motion estimation technique [17].

Even though the efforts aforementioned have been made in order to find a suitable algorithm for motion estimation of point clouds, the algorithms are still very complex methods and demand plenty of computational power. Due to the differences in the geometry of successive point clouds, it is far from trivial to develop suitable motion estimation algorithms for point clouds. Therefore, the current techniques for motion estimation developed for point cloud compression are not suitable for time-constrained applications. For this reason, the compression of point clouds leveraging motion estimation approaches is still an open issue for further research exploration.

With that in mind, an alternative for algorithms based on motion estimation is the use of a simple inter-frame prediction based on the nearest-neighbors. Consider a point cloud frame to be compressed $\mathcal{F}(t)$, its immediately previous frame $\mathcal{F}(t-1)$ and its inter-frame prediction $\hat{\mathcal{F}}(t) = \Omega(\mathcal{F}(t))$ (where $\Omega(\cdot)$ represents the inter-frame prediction used). The basic idea behind the nearest-neighbor inter-frame prediction is to match each voxel in $\mathcal{F}(t)$ with the nearest voxel in $\mathcal{F}(t-1)$, e.g., with the voxel in $\mathcal{F}(t-1)$ that presents the lowest euclidean distance. Hence, $\hat{\mathcal{F}}(t)$ will have the same geometry as $\mathcal{F}(t)$, but with the attributes from $\mathcal{F}(t-1)$. This inter-frame predictive algorithm can be efficiently implemented for time-constrained applications using k-nearest neighbor search from the Fast Library for Approximate Nearest Neighbors [23].

Since the nearest-neighbor method is a very simple inter-frame prediction, it does not yield good prediction results in all cases. When there is low motion in between $\mathcal{F}(t)$ and $\mathcal{F}(t-1)$, the nearest-neighbor inter-frame prediction performs well, generating accurate predicted frames. However, when there is fast motion in between $\mathcal{F}(t)$ and $\mathcal{F}(t-1)$, the nearest-neighbor inter-frame prediction performs badly, generating poor frame predictions. This can be seen in Fig. 2.21, for point clouds “Ricardo” with low motion and “David” with intense motion.



(a) Frame #16.

(b) Frame #206.

Figure 2.21: Results of the nearest-neighbor inter-frame prediction for point cloud sequences (a) "Ricardo" and (b) "David". The original frame on the left and the predicted frame on the right.

3 METHODOLOGY

In this Chapter, we explore the idea of using the nearest-neighbor inter-frame prediction alongside RAHT to leverage residual encoding to improve RAHT's compression performance of dynamic point clouds. However, as explained in Section 2.3.1, the current version of RAHT, with the intra-frame predictive step added (I-RAHT), is capable of achieving significant gains of up to 30% over the original RAHT. With that in mind, we decided here to use I-RAHT instead of the original RAHT. Therefore, we explore possible combinations of the inter-frame and the intra-frame predictions alongside RAHT to improve its attribute compression performance of dynamic point clouds. In order to assess the performance of our predictive schemes, we compare them to I-RAHT since it represents the best predictive RAHT that currently exists.

Initially, in Section 3.1, we perform a theoretical experiment to foresee the potential gains of using residual encoding with different prediction qualities. Then, two alternative approaches are explored in this work: encode the attribute residues and encode the coefficient residues. In Section 3.2, we explore the use of attribute residues in different compression schemes combining the nearest-neighbor inter-frame prediction with I-RAHT. Attribute residues consist of taking the differences, in the attribute domain, of two point cloud frames to obtain color residues. Then, the residues are subjected to the transform, and the generated coefficients are entropy encoded. In Section 3.3, we explore the use of coefficient residues as an alternative. Considering the intra-frame predictive step in I-RAHT that exploits attribute correlation, we expect to obtain slightly different rate-distortion results. Coefficient residues consist of subjecting the attributes of two point cloud frames to the transform and subtracting their generated coefficients to obtain residues of coefficients. That is, the residues are computed in the coefficient domain and then entropy encoded.

Throughout this Chapter, we use the symbols $\mathcal{F}(t)$ to denote the attributes of a point cloud frame to be compressed, $\mathcal{F}(t-1)$ to denote the immediately previous frame, and $\hat{\mathcal{F}}(t)$ to represent the prediction of $\mathcal{F}(t)$. The nearest-neighbor inter-frame prediction is denoted as $\Omega(\cdot)$, while RAHT and I-RAHT are represented by $\Gamma(\cdot)$ and $\Upsilon(\cdot)$, respectively.

3.1 THEORETICAL EXPERIMENT

To assess the benefits of introducing a predictive step to RAHT, we start with a theoretical experiment. This experiment serves as a starting point to determine how accurate a predictive approach has to be to promote improvements in dynamic point cloud compression using RAHT. Hence, it provides information on what kind of compression gains should be expected from the use of the nearest-neighbor inter-frame prediction.

In our experiment, we used the frame #1065 of the point cloud sequence "Longdress" [10] as the frame to be compressed $\mathcal{F}(t)$. By adding Gaussian noise with variance σ to $\mathcal{F}(t)$, we obtain the prediction $\hat{\mathcal{F}}(t)$. Then, the differences between the frames are taken $\mathcal{G}(t) = \mathcal{F}(t) - \hat{\mathcal{F}}(t)$

generating residues that are transformed with RAHT $\Gamma(\mathcal{G}(t))$, and then entropy encoded. This experiment was repeated for values of σ equal to $2, 4, 6,$ and 8. The idea is to simulate a frame $\hat{\mathcal{F}}(t)$ predicted with different distortion levels and determine how accurate the prediction has to be to generate compression improvements. Higher values of σ generate predictions with a higher level of distortions, while lower values generate more accurate predictions. The rate-distortion curves obtained are presented in Fig. 3.1.

From Fig. 3.1, one can easily see that significant gains may be achieved by encoding $\mathcal{G}(t)$ instead of $\mathcal{F}(t)$, if the predicted frame $\hat{\mathcal{F}}(t)$ achieves a certain degree of accuracy. In this example, it seems to be advantageous to use inter- or intra-frame predictive approaches alongside RAHT for $\sigma < 4$. This value of σ translates to an estimated frame $\hat{\mathcal{F}}(t)$ with mean squared error relative to $\mathcal{F}(t)$ of 26. Equivalently, the quality of $\hat{\mathcal{F}}(t)$ relative to $\mathcal{F}(t)$ has to be around 34-35 dB PSNR to promote compression gains. Otherwise, with poor prediction, encoding the residues $\mathcal{G}(t)$ seems to be disadvantageous if compared to encoding $\mathcal{F}(t)$. We are aware that encoding noise is different than encoding real predicted frames. However, as mentioned before, this experiment served only as a starting point to determine what kind of results we would obtain from the use of the nearest-neighbor inter-frame prediction and set a distortion threshold used in the following sections.

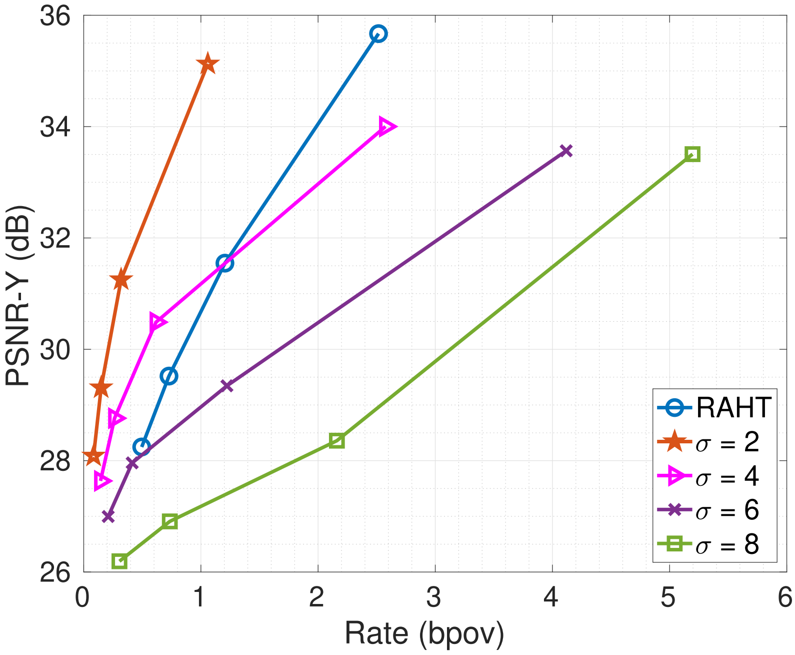


Figure 3.1: Theoretical experiment illustrating potential rate-distortion gains of the use of prediction alongside RAHT at various noise variances σ. The experiment shown was performed for the frame #1065 of point cloud sequence “Longdress”.

3.2 ATTRIBUTE RESIDUES

In this Section we explore leveraging the use of attribute residues to improve compression performance. As briefly mentioned before, the idea behind encoding attribute residues is to transform the differences of the attributes instead of the attributes themselves. Consider $\mathcal{G}(t)$ such that $\mathcal{G}(t) = \mathcal{F}(t) - \hat{\mathcal{F}}(t)$ and $\hat{\mathcal{F}}(t) = \Omega(\mathcal{F}(t-1))$. Leveraging attribute residues consist of encoding $\mathcal{C}(t) = \Upsilon(\mathcal{G}(t))$ instead of $\mathcal{C}(t) = \Upsilon(\mathcal{F}(t))$. Differently from the theoretical experiment, here we use I-RAHT to transform the attributes and the nearest-neighbor inter-frame prediction to generate $\hat{\mathcal{F}}(t)$. The described procedure is depicted in Fig. 3.2. As seen previously in the theoretical experiment, it may be advantageous to encode residues with proper attribute prediction. On the other hand, it may be better to encode the attributes themselves with poor attribute prediction.

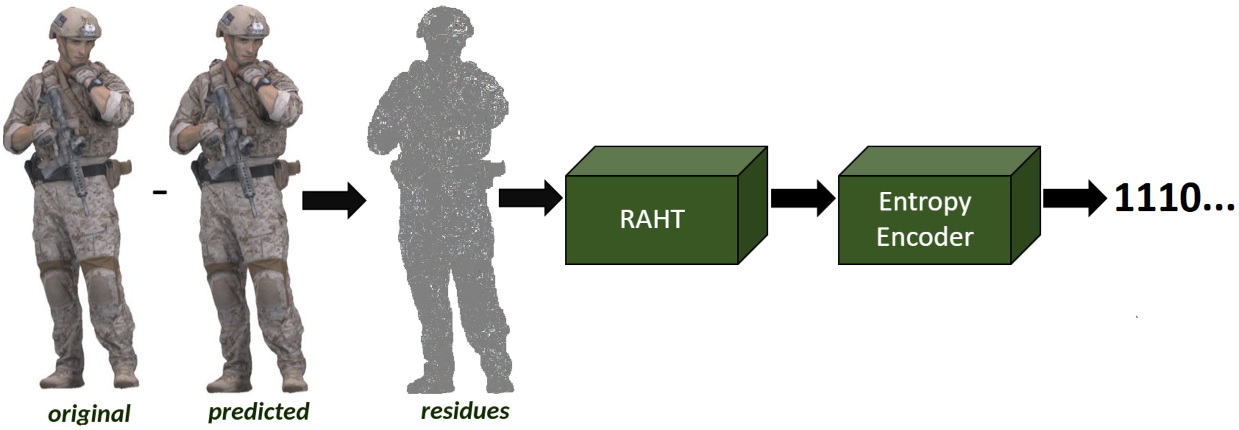
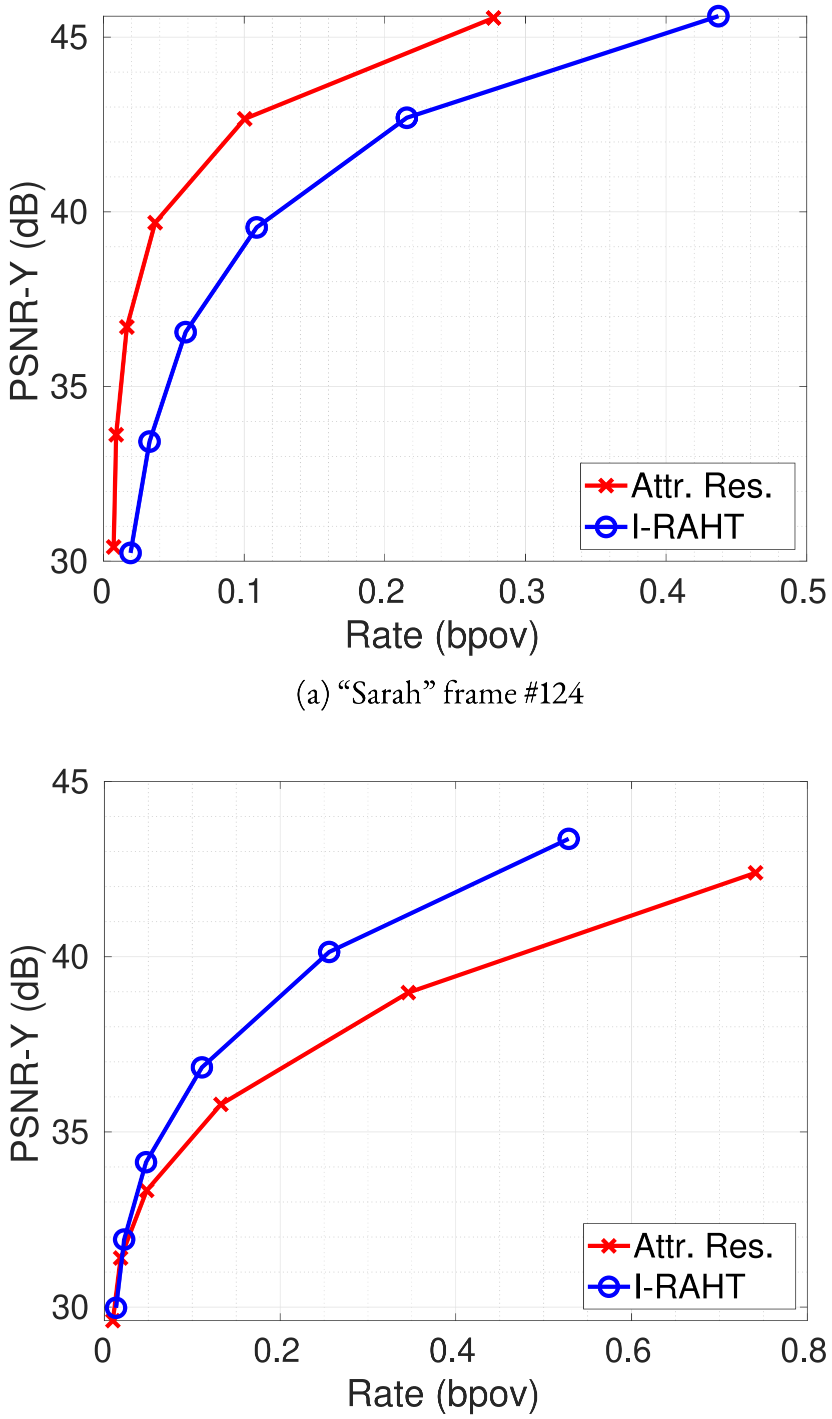


Figure 3.2: Illustration of the idea behind leveraging attribute residues

The compression performance of the combination of the inter- and the intra-frame predictions against I-RAHT is illustrated in Fig. 3.3. Table 3.1 shows the Bjontegaard delta (BD) [7] PSNR-Y gains and bitrate savings of the attribute residues approach relative to I-RAHT. We picked two consecutive frames from the point cloud sequences "Sarah" [13] and "Loot" [10]. The particular pair of frames from the sequence "Sarah" has very little motion, while the pair of frames from the sequence "Loot" contains more intense motion. Hence, we expect the simple inter-frame prediction based on the nearest-neighbors to be fairly accurate for "Sarah" and poor for "Loot". Therefore, based on that and the theoretical experiment, the combination of the inter- and the intra-frame predictions should outperform I-RAHT for "Sarah" and be outperformed for "Loot". That is precisely what the results illustrate in Fig. 3.3. The correspondence among point clouds using simple proximity will lead to good approximations and residual encoding improvement if there is not much local motion between frames. If there is, it may be better to use only the intra-frame prediction.

A solution based on motion estimation algorithms might be the first choice to improve inter-frame prediction reliability on regions with more intense object motion. However, as explained before, a motion estimation solution for point cloud frames suitable for practical applications remains elusive. Current motion estimation approaches for point clouds are too computationally expensive for time-constrained applications. With that in mind, we opted here to use adaptive strategies based on the nearest-neighbor inter-frame prediction algorithm.



(a) “Sarah” frame #124

(b) “Loot” frame #1073

Figure 3.3: Results of the attribute residues approach (Attr. Res.) compared to I-RAHT for point clouds “Sarah” and “Loot”.

Table 3.1: Average BD PSNR-Y gains and average bitrate savings of the attribute residues approach relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Sarah	4.22	-65.49
Loot	-1.26	47.91

3.2.1 Proposal 1 - Fragment-based Single Decision

To overcome the weakness of the last approach in front of regions of the point cloud frames with intense motion, we devised two adaptive methods. They are referred to as the "fragment-based single decision" and the "fragment-based multiple decision". These adaptive methods segment the point clouds into several fragments to only carry inter-frame prediction in parts where the nearest-neighbor inter-frame prediction is sufficiently accurate.

The fragment-based single decision starts by segmenting the morton-code-ordered point clouds $\mathcal{F}(t)$ and $\hat{\mathcal{F}}(t)$ in fixed fragments of 1500 voxels $\{\mathcal{F}_k(t)\}$ and $\{\hat{\mathcal{F}}_k(t)\}$. Each fragment $\mathcal{F}_k(t)$ and $\hat{\mathcal{F}}_k(t)$ can be seen as a small point cloud containing 1500 voxels. Then, the distortion (in mean squared error) of each fragment $\hat{\mathcal{F}}_k(t)$ in $\{\hat{\mathcal{F}}_k(t)\}$ is checked. If the distortion is below a threshold λ, then $\hat{\mathcal{F}}_k(t)$ is deemed reliable, and only a control bit 1 is entropy encoded. Otherwise, $\hat{\mathcal{F}}_k(t)$ is deemed unreliable and the attributes are transformed $\mathcal{C}(t) = \Upsilon(\mathcal{F}_k(t))$. Then $\mathcal{C}(t)$ and a control bit 0 are entropy encoded. Hence, in this adaptive method, the inter-frame prediction is only used in regions which can be well predicted by $\mathcal{F}(t-1)$. Otherwise, we use only the intra-frame prediction. This procedure is illustrated as a flowchart in Fig. 3.4.

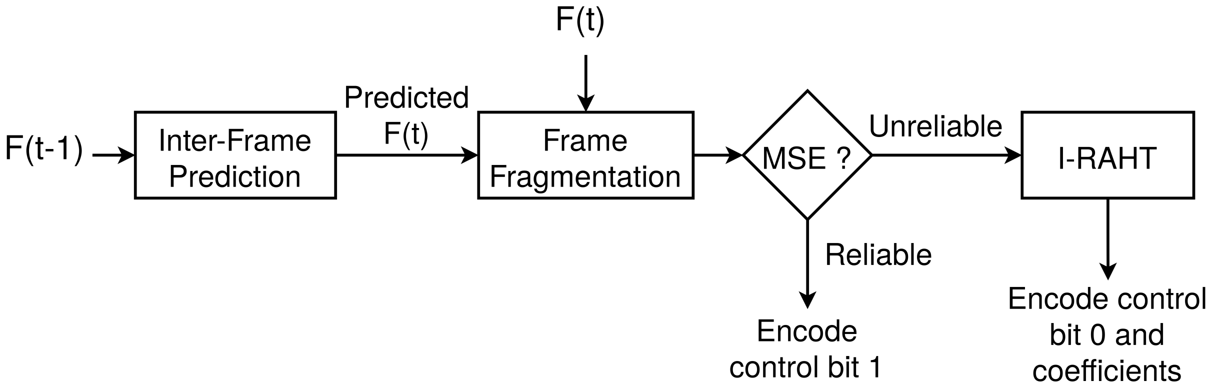
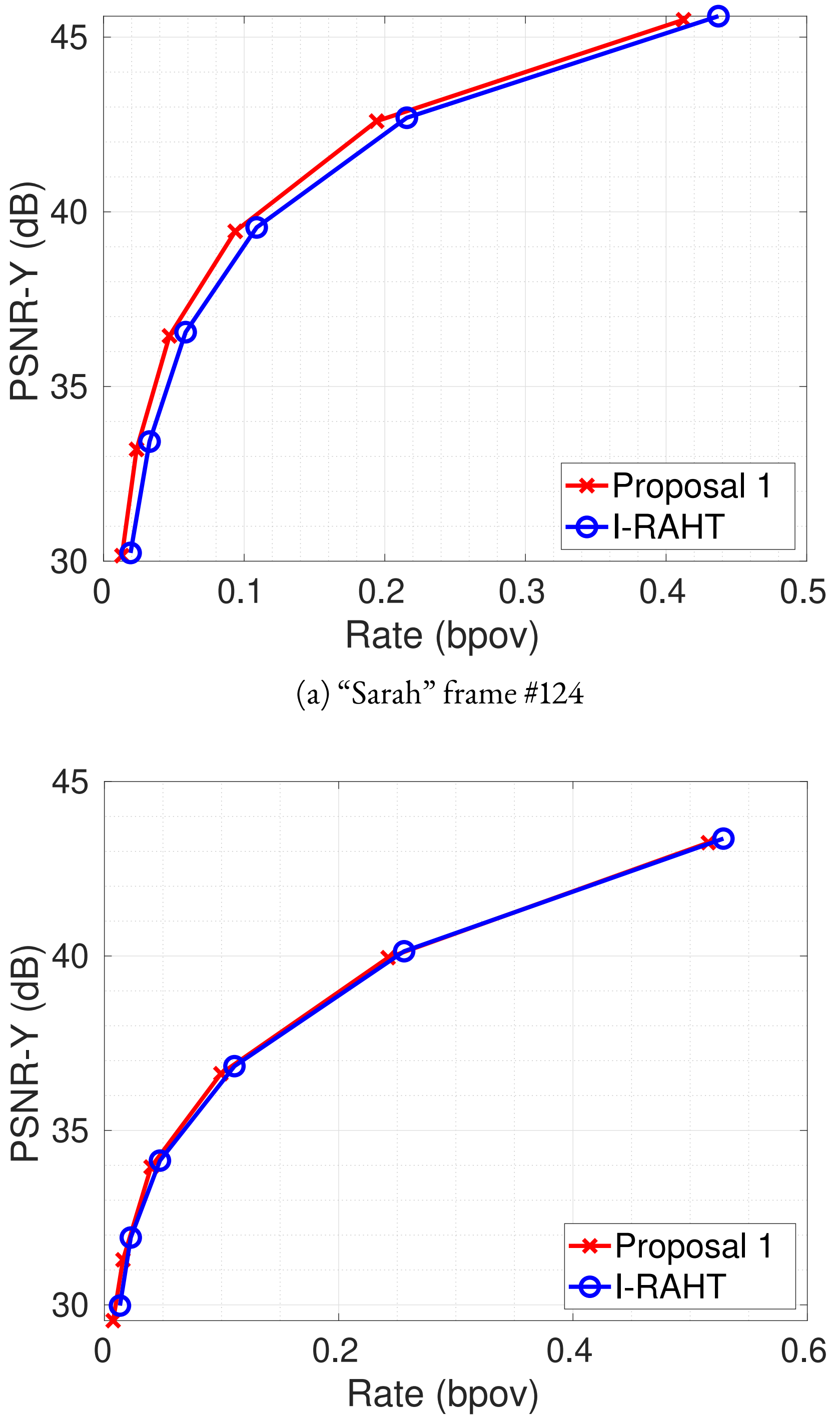


Figure 3.4: Flowchart describing the fragment-based single decision scheme.

In our implementation, λ was empirically defined as the distortion (in mean squared error) of $\mathcal{F}(t-1)$ locally decoded. The use of the distortion measure of $\mathcal{F}(t-1)$ was verified to represent a good approximation of the quality expected from each fragment to be considered reliable. The control bits inform the decoder how each fragment should be decoded.

Rate-distortion curves are presented in Fig. 3.5 for the same point clouds as in Fig. 3.3. Table 3.2 shows the average BD PSNR-Y and bitrate savings of the proposed scheme relative to I-RAHT. For "Loot", instead of the large negative gains over the use of only the intra-frame prediction, the proposed predictive approach now slightly outperforms it. However, compared to the results in Section 3.2, the performance improvement of the combination of the inter- and the intra-frame predictions for "Sarah" has significantly decreased. In summary, the adaptive method's performance is very similar to the one of I-RAHT in the worst-case scenario. While in the best-case scenario, the gains are smaller than in the non-adaptive case.



(a) “Sarah” frame #124

(b) “Loot” frame #1073

Figure 3.5: Results of the fragment-based single decision (Proposal 1) compared to I-RAHT for point clouds “Sarah” and “Loot”.

Table 3.2: Average BD PSNR-Y gains and average bitrate savings of the fragment-based single decision (Proposal 1) relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Sarah	0.77	-16.40
Loot	0.23	-7.64

3.2.2 Proposal 2 - Fragment-Based Multiple Decision

To guarantee significant gains when the inter-frame prediction is reliable and ensure a similar performance when unreliable, we devised a second adaptive approach. The fragment-based multiple decision starts by checking the overall distortion of $\hat{\mathcal{F}}(t)$, if it is below a threshold ρ the proposed scheme performs as follows: the point clouds $\mathcal{F}(t)$ and $\hat{\mathcal{F}}(t)$ are segmented in fixed fragments of 1500 voxels $\{\mathcal{F}_k(t)\}$ and $\{\hat{\mathcal{F}}_k(t)\}$. Then, the attribute differences are computed $\{\mathcal{G}_k(t)\} = \{\mathcal{F}_k(t)\} - \{\hat{\mathcal{F}}_k(t)\}$. The distortion of each fragment $\hat{\mathcal{F}}_k(t)$ in $\{\hat{\mathcal{F}}_k(t)\}$ is checked. If the distortion is below a threshold λ, $\hat{\mathcal{F}}_k(t)$ is deemed reliable and only a control bit 1 is entropy encoded. Otherwise, $\hat{\mathcal{F}}_k(t)$ is deemed unreliable and the attribute residues are transformed $\mathcal{C}(t) = \Upsilon(\mathcal{G}_k(t))$. Then $\mathcal{C}(t)$ and a control bit 0 are entropy encoded.

Otherwise, if the distortion of $\hat{\mathcal{F}}(t)$ is equal or above ρ the proposed scheme performs as follows: $\mathcal{F}(t)$ and $\hat{\mathcal{F}}(t)$ are segmented in fixed fragments of 1500 voxels $\{\mathcal{F}_k(t)\}$ and $\{\hat{\mathcal{F}}_k(t)\}$. Then, the distortion of each fragment $\hat{\mathcal{F}}_k(t)$ in $\{\hat{\mathcal{F}}_k(t)\}$ is checked. If the distortion is below λ, $\hat{\mathcal{F}}_k(t)$ is deemed reliable and only a control bit 1 is entropy encoded. Otherwise, $\hat{\mathcal{F}}_k(t)$ is deemed unreliable and the attributes themselves are transformed $\mathcal{C}(t) = \Upsilon(\mathcal{F}_k(t))$. Then $\mathcal{C}(t)$ and a control bit 0 are entropy encoded. This method merges both Sections 3.2 and 3.2.1, and is depicted in Fig. 3.6. In our implementation, ρ was defined as 26 based on the theoretical experiment, and λ was empirically defined as the distortion (in mean squared error) of $\mathcal{F}(t-1)$ locally decoded.

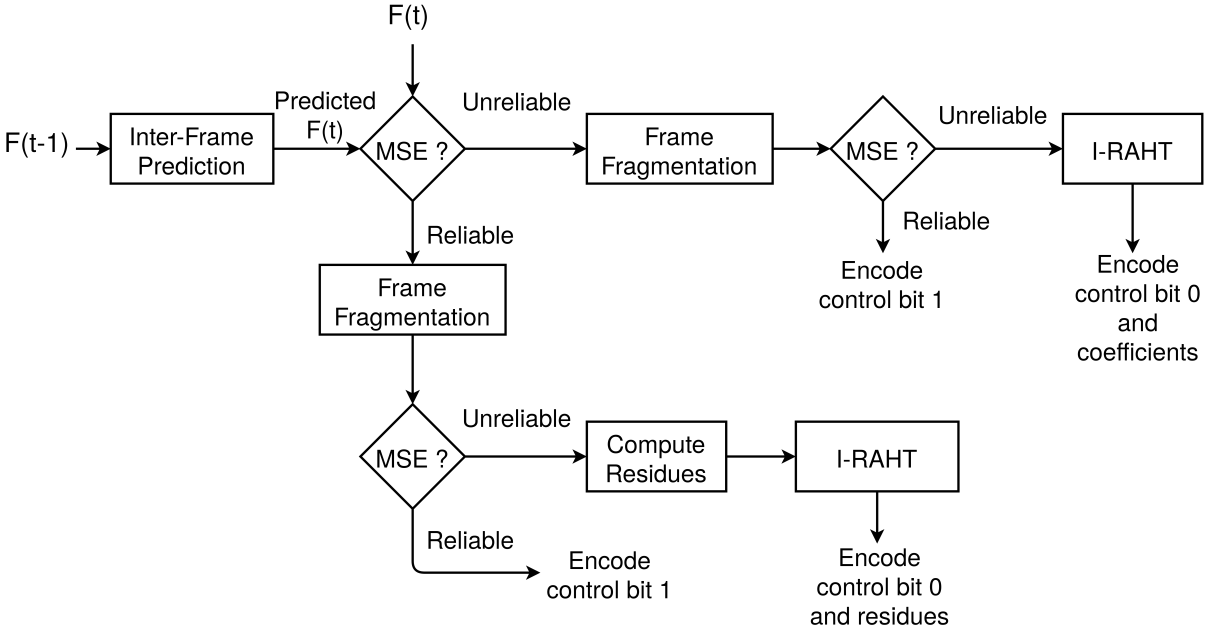
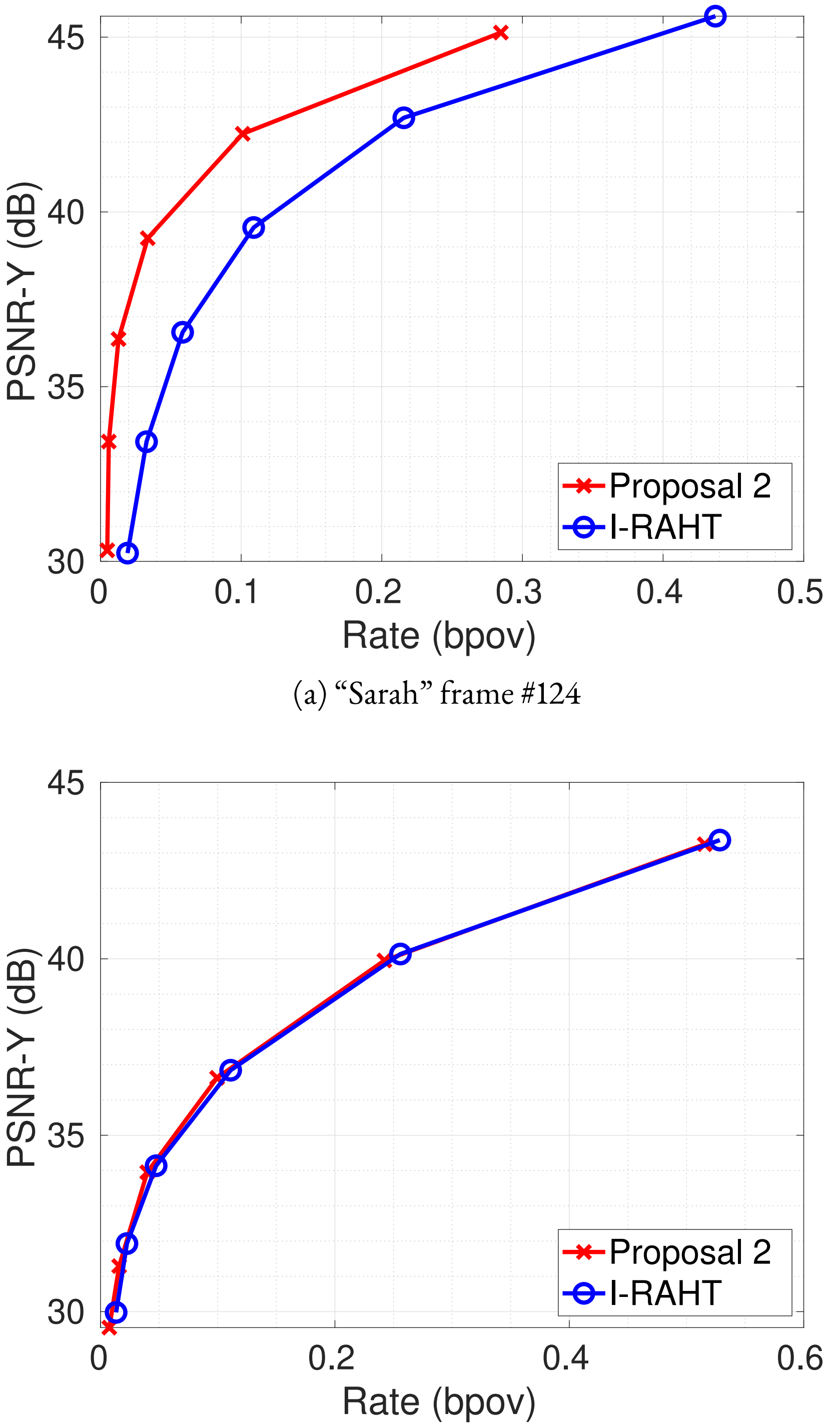


Figure 3.6: Flowchart describing the fragment-based multiple decision scheme.

The results obtained are presented in Fig. 3.7 and in Table 3.3 for point clouds "Sarah" and "Loot". For "Loot", the performance of the devised approach remains the same as in Section 3.2.1. For "Sarah", significant gains similar to Section 3.2 are obtained. Therefore, in the best scenario, the gains of this adaptive method are as high as in the non-adaptive case. While in the worst scenario,



(a) “Sarah” frame #124

(b) “Loot” frame #1073

Figure 3.7: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point clouds “Sarah” and “Loot”.

Table 3.3: Average BD PSNR-Y gains and average bitrate savings of the fragment-based multiple decision (Proposal 2) relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Sarah	3.87	-69.29
Loot	0.23	-7.64

its performance is very similar to the one of I-RAHT.

3.3 COEFFICIENT RESIDUES

In Sections 3.2 to 3.2.2, we explored leveraging attribute residues to improve RAHT's compression performance of dynamic point clouds. We achieved significant gains over the use of only the intra-frame prediction for frames with low motion. At the same time, ensuring a competitive performance for frames with intense motion. An alternative to that is leveraging coefficient residues instead, which is explored in the remainder of this Chapter. Due to the intra-frame prediction that exploits attribute correlation among neighboring nodes, we expect the results in the next Sections to be somewhat different from those presented in Sections 3.2 to 3.2.2.

The idea behind encoding coefficient residues is to encode the differences of the coefficients instead of the coefficients themselves. That is, subject both the current and predicted frames to I-RAHT generating the coefficients $\mathcal{C}(t) = \Upsilon(\mathcal{F}(t))$ and the predicted coefficients $\hat{\mathcal{C}}(t) = \Upsilon(\hat{\mathcal{F}}(t))$, respectively. Then, compute the differences $\mathcal{U}(t) = \mathcal{C}(t) - \hat{\mathcal{C}}(t)$. Finally, entropy encode $\mathcal{U}(t)$ instead of $\mathcal{C}(t)$. This procedure is depicted in Fig. 3.8.

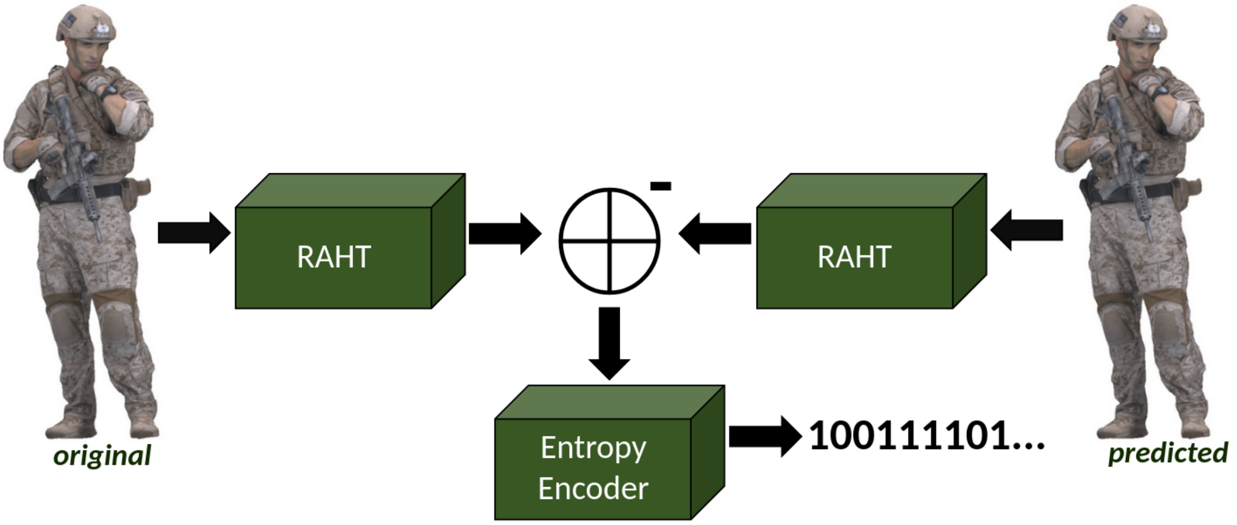
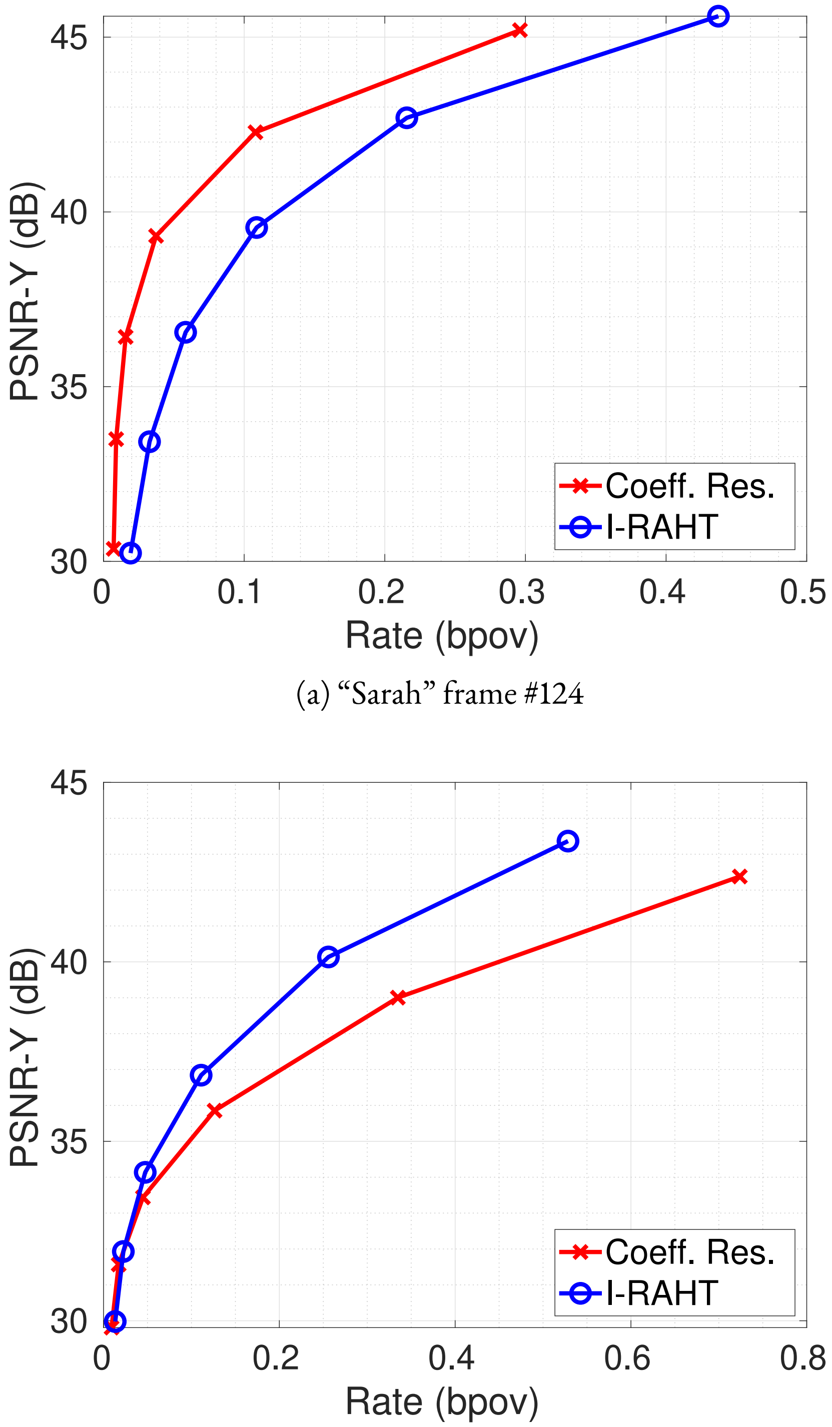


Figure 3.8: Illustration of the idea behind leveraging coefficient residues.

The compression performance against I-RAHT is illustrated in Fig. 3.9 and in Table 3.4. We used the same frames of the point cloud sequences "Sarah" and "Loot" as in Section 3.2. The results obtained are similar to the ones obtained in Fig. 3.3: for "Sarah", they are slightly worse than the attribute residues approach but slightly better for "Loot". Therefore, we can derive the same conclusions as before: the use of the simple inter-frame prediction can provide significant gains over the use of only the intra-frame prediction when there is not much motion between frames. In the case of intense motion between frames, it may be better to use only the intra-frame prediction. Hence, to improve the performance of our method, we explored adaptive schemes to use the inter-frame prediction only in regions of the point cloud where it is reliable.



(a) “Sarah” frame #124

(b) “Loot” frame #1073

Figure 3.9: Results of the coefficient residues approach (Coeff. Res.) compared to I-RAHT for point clouds “Sarah” and “Loot”.

Table 3.4: Average BD PSNR-Y gains and average bitrate savings of the coefficient residues approach relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Sarah	3.68	-62.81
Loot	-1.03	36.77

3.3.1 Proposal 3 - Block-Based Multiple Decision

Similarly to Section 3.2.1, to improve the performance of the method described in Section 3.3, we devise an adaptive scheme. Instead of partitioning into fragments as in Sections 3.2.1 and 3.2.2, we decided to break the point clouds into blocks of fixed dimensions, and to use the inter-frame prediction only when accurate enough.

Initially, the overall distortion of $\hat{\mathcal{F}}(t)$ is checked, if it is below a threshold ρ the scheme performs as follows: the point clouds $\mathcal{F}(t)$ and $\hat{\mathcal{F}}(t)$ are partitioned into blocks of dimensions $16 \times 16 \times 16$, $\{\mathcal{F}_k(t)\}$ and $\{\hat{\mathcal{F}}_k(t)\}$. Then, the distortion of each block $\hat{\mathcal{F}}_k(t)$ in $\{\hat{\mathcal{F}}_k(t)\}$ is checked. If the distortion is below a threshold λ, $\hat{\mathcal{F}}_k(t)$ is deemed reliable, and only a control bit 1 is entropy encoded. Otherwise, $\hat{\mathcal{F}}_k(t)$ is deemed unreliable, and both $\mathcal{F}_k(t)$ and $\hat{\mathcal{F}}_k(t)$ are transformed $\mathcal{C}(t) = \Upsilon(\mathcal{F}_k(t))$ and $\hat{\mathcal{C}}(t) = \Upsilon(\hat{\mathcal{F}}_k(t))$. Then, $\mathcal{U}(t) = \mathcal{C}(t) - \hat{\mathcal{C}}(t)$ and a control bit 0 are entropy encoded.

On the other hand, if the distortion of $\hat{\mathcal{F}}(t)$ is equal or above ρ, the method proceeds as follows: $\mathcal{F}(t)$ and $\hat{\mathcal{F}}(t)$ are partitioned into blocks of dimensions $16 \times 16 \times 16$, $\{\mathcal{F}_k(t)\}$ and $\{\hat{\mathcal{F}}_k(t)\}$. Then, the distortion of each block $\hat{\mathcal{F}}_k(t)$ in $\{\hat{\mathcal{F}}_k(t)\}$ is checked. If the distortion is below λ, $\hat{\mathcal{F}}_k(t)$ is deemed reliable and only a control bit 1 is entropy encoded. Otherwise, $\hat{\mathcal{F}}_k(t)$ is deemed unreliable and the original attributes are transformed $\mathcal{C}(t) = \Upsilon(\mathcal{F}_k(t))$. Then $\mathcal{C}(t)$ and a control bit 0 are entropy encoded. A flowchart describing the proposed scheme is shown in Fig. 3.10. In our implementation, ρ was defined as 26 based on the theoretical experiment, and λ was empirically defined as the distortion of $\mathcal{F}(t-1)$ locally decoded.

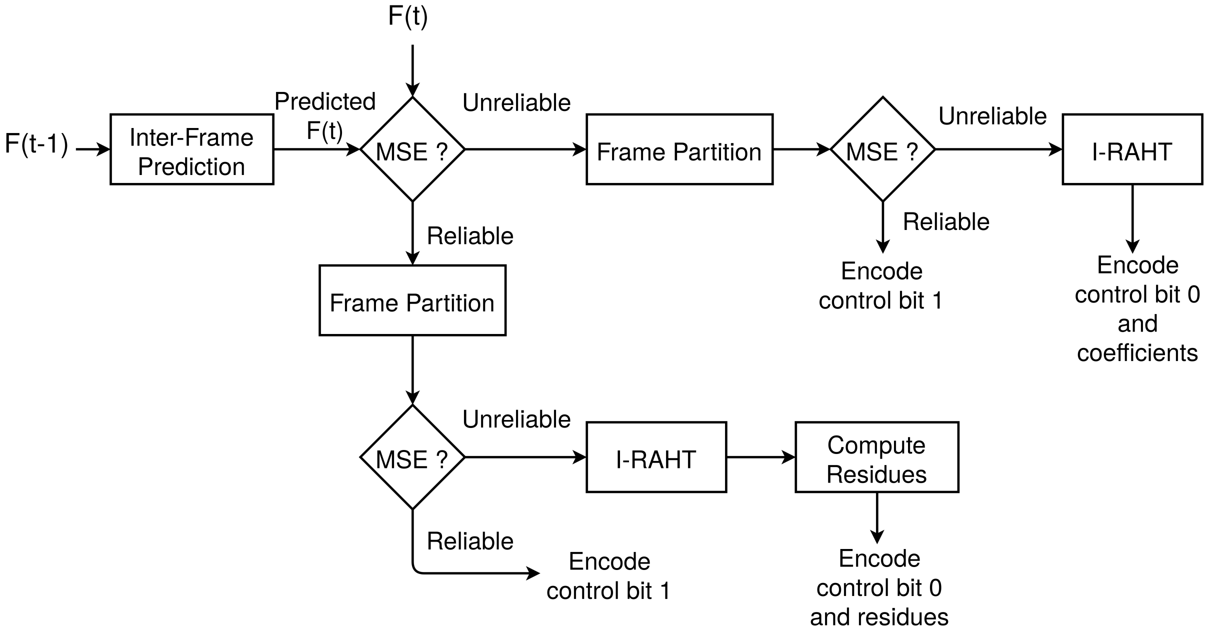
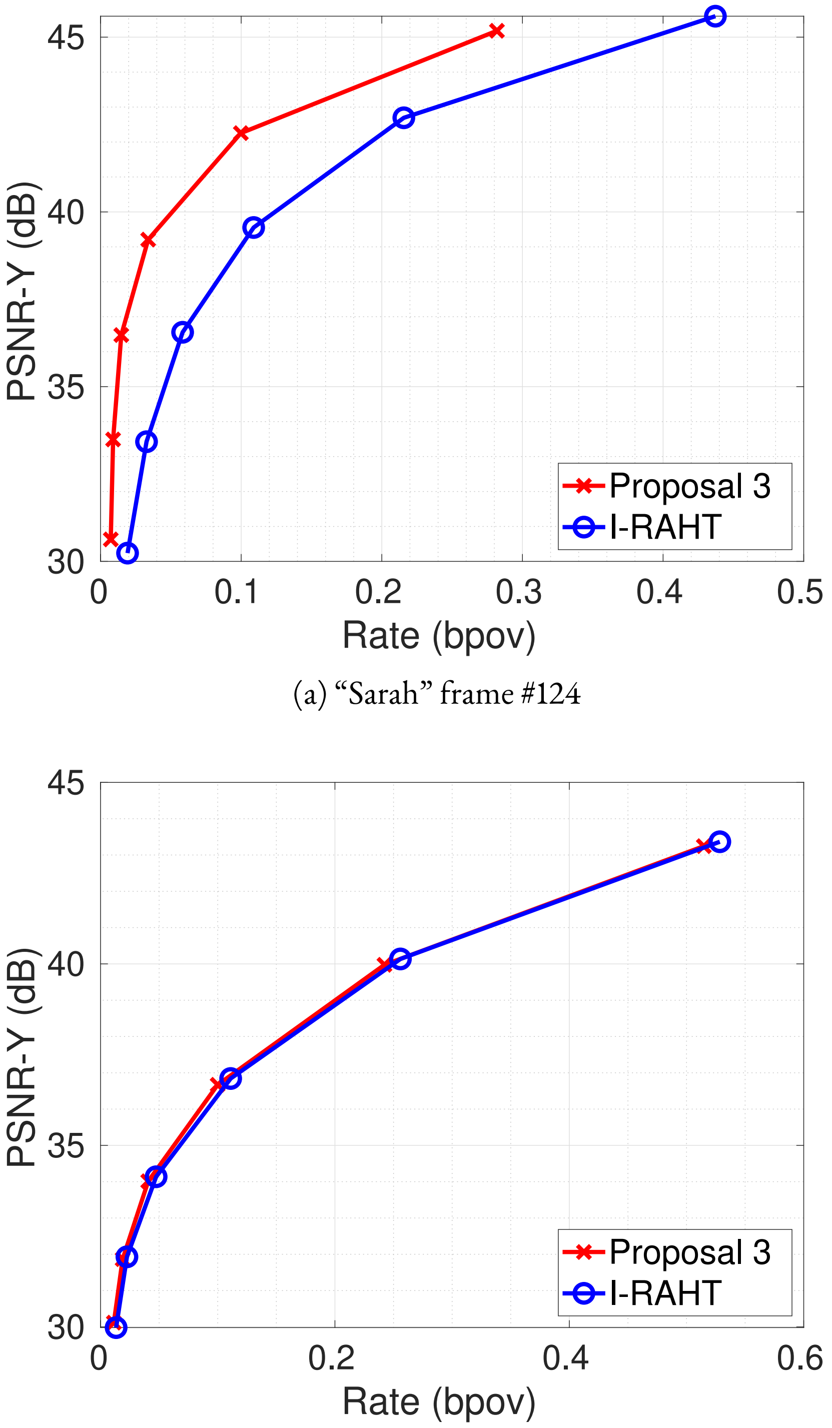


Figure 3.10: Flowchart describing the block-based multiple decision scheme.

The results obtained are presented in Fig. 3.11 and in Table 3.5. For "Sarah", one can see significant gains over the use of only the intra-frame prediction. For "Loot", a competitive performance regarding the use of only the intra-frame prediction is illustrated. We can see an improvement in



(a) “Sarah” frame #124

(b) “Loot” frame #1073

Figure 3.11: Results of the block-based multiple decision (Proposal 3) compared to I-RAHT for point clouds “Sarah” and “Loot”.

Table 3.5: Average BD PSNR-Y gains and average bitrate savings of the block-based multiple decision (Proposal 3) relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Sarah	3.92	-64.54
Loot	0.25	-7.00

performance for point clouds with intense motion while keeping the significant gains for point clouds with low motion. The results are very similar to those obtained in Section 3.2.2, with a subtle increase in BD PSNR-Y and a slight decrease in BD bitrate.

3.3.2 Proposal 4 - Level-Based Multiple Decision

In an attempt to improve the performance of our last adaptive method, we decided to explore the coefficients per level of the octree. The coefficients of I-RAHT are ordered in a top-down manner from the root of the octree to the leaves. In Fig. 3.12, the magnitudes of the serialized coefficients $\mathcal{C}(t)$ of Y-channel for one frame of point cloud "Loot" are shown. Note that the coefficients present a decaying pattern, with the higher magnitude coefficients in the beginning, e.g., in the first levels of the octree. One can also observe that the coefficients tend to decay until they become minimal in magnitude. Hence, we can conclude that, when taking $\mathcal{U}(t)$ for the entire set of residues, compression improvement is expected for the first residues, but not for the many very small coefficients. For those very small coefficients to be predicted appropriately, one would need a very accurate inter-frame prediction, which is not the case of the nearest-neighbor method used here.

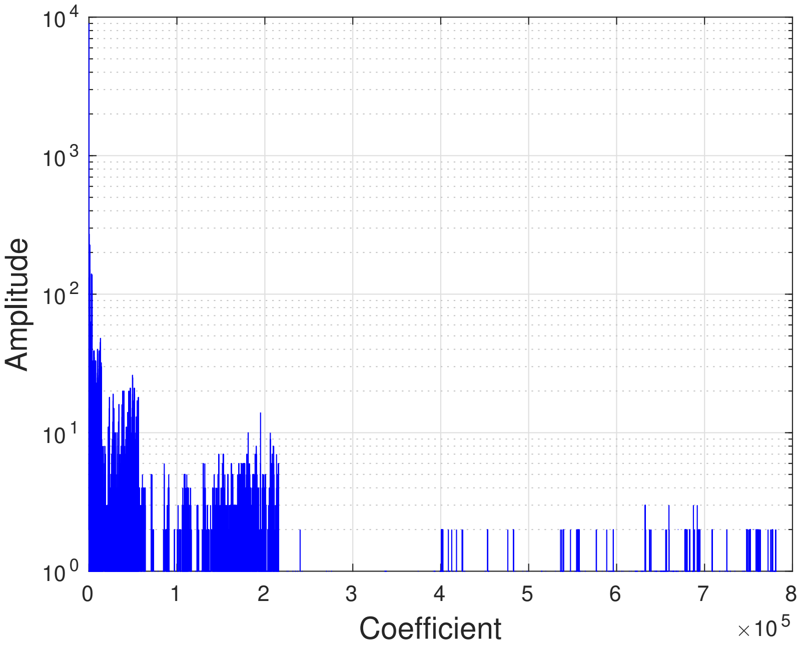


Figure 3.12: Coefficient magnitudes of Y-channel for one frame of point cloud "Loot".

Based on the decaying pattern illustrated in Fig. 3.12, we decided to only encode $\mathcal{U}(t)$ for a fraction of $\mathcal{C}(t)$ where high magnitude values are predominant. For the others, $\mathcal{C}(t)$ is directly encoded. In our tests, the optimal number of residues $\mathcal{U}(t)$ to be encoded varied depending on the inter-frame prediction quality. Let there be N coefficients in $\mathcal{C}(t)$ and in $\hat{\mathcal{C}}(t)$ such that we encode $\mathcal{U}(t)$ only for the first α levels of the octree which consists of a fraction of $n_\alpha < N$ coefficients (i.e., $\{\mathcal{U}_v(t)\}, v = 1, ..., n_\alpha$). When $\hat{\mathcal{F}}(t)$ is accurate, $\mathcal{C}(t)$ and $\hat{\mathcal{C}}(t)$ are very similar and we better encode $\mathcal{U}(t)$ for $\alpha = \alpha_1$. On the other hand, if $\mathcal{C}(t)$ and $\hat{\mathcal{C}}(t)$ differ considerably, we encode $\mathcal{U}(t)$ for $\alpha = \alpha_2 < \alpha_1$.

The level-based multiple decision algorithm initially checks the overall distortion of $\hat{\mathcal{F}}(t)$, if it is below a threshold ρ, the proposed scheme performs as follows: $\mathcal{F}(t)$ and $\hat{\mathcal{F}}(t)$ are partitioned into blocks of dimension $16 \times 16 \times 16$, $\{\mathcal{F}_k(t)\}$ and $\{\hat{\mathcal{F}}_k(t)\}$. Then, the distortion of each block $\hat{\mathcal{F}}_k(t)$ in $\{\hat{\mathcal{F}}_k(t)\}$ is checked. If the distortion is below a threshold λ, $\hat{\mathcal{F}}_k(t)$ is deemed reliable and only a control bit 1 is entropy encoded. Otherwise, $\hat{\mathcal{F}}_k(t)$ is deemed unreliable and the original and predicted attributes are transformed $\mathcal{C}(t) = \Upsilon(\mathcal{F}_k(t))$ and $\hat{\mathcal{C}}(t) = \Upsilon(\hat{\mathcal{F}}_k(t))$. Then, the residues are computed for the first α_1 levels of the octree $\{\mathcal{U}_v(t)\}, v = 1, ..., n_{\alpha_1}$. Finally, $\{\mathcal{U}_v(t)\}$ and $\{\mathcal{C}_r(t)\}, r = n_{\alpha_1} + 1, ..., N$ are entropy encoded with a control bit 0.

On the other hand, if the overall distortion of $\hat{\mathcal{F}}(t)$ is equal or above ρ, the proposed scheme performs as follows: $\mathcal{F}(t)$ and $\hat{\mathcal{F}}(t)$ are partitioned into blocks of dimension $16 \times 16 \times 16$, $\{\mathcal{F}_k(t)\}$ and $\{\hat{\mathcal{F}}_k(t)\}$. Then, the distortion of each block $\hat{\mathcal{F}}_k(t)$ in $\{\hat{\mathcal{F}}_k(t)\}$ is checked. If the distortion is below λ, $\hat{\mathcal{F}}_k(t)$ is deemed reliable and only a control bit 1 is entropy encoded. Otherwise, $\hat{\mathcal{F}}_k(t)$ is deemed unreliable and the original and predicted attributes are transformed $\mathcal{C}(t) = \Upsilon(\mathcal{F}_k(t))$ and $\hat{\mathcal{C}}(t) = \Upsilon(\hat{\mathcal{F}}_k(t))$. Then, the residues are computed for the first α_2 levels of the octree $\{\mathcal{U}_v(t)\}, v = 1, ..., n_{\alpha_2}$. Finally, $\{\mathcal{U}_v(t)\}$ and $\{\mathcal{C}_r(t)\}, r = n_{\alpha_2} + 1, ..., N$ are entropy encoded with a control bit 0. This method is illustrated in the flowchart of Fig. 3.13. In our implementation, ρ was defined as 26 based on the theoretical experiment, and λ was empirically defined as the distortion (in mean squared error) of $\mathcal{F}(t-1)$ locally decoded. Also, $\alpha_1 = L - 1$ and $\alpha_2 = L - 4$, where L represents the bit-depth of the point cloud.

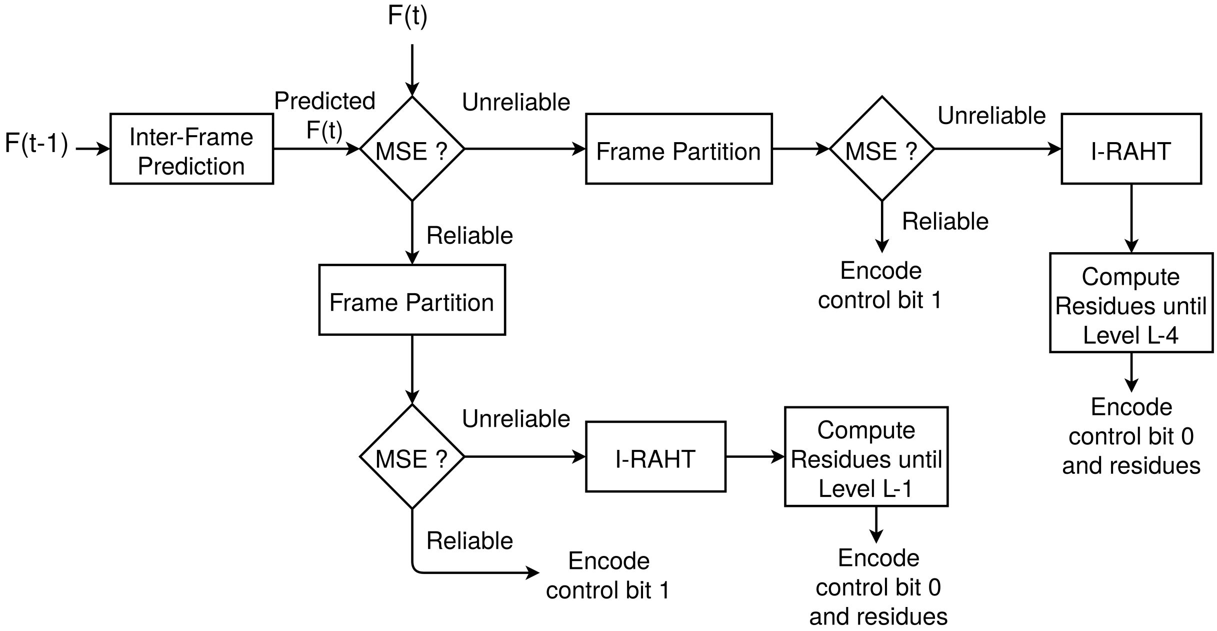
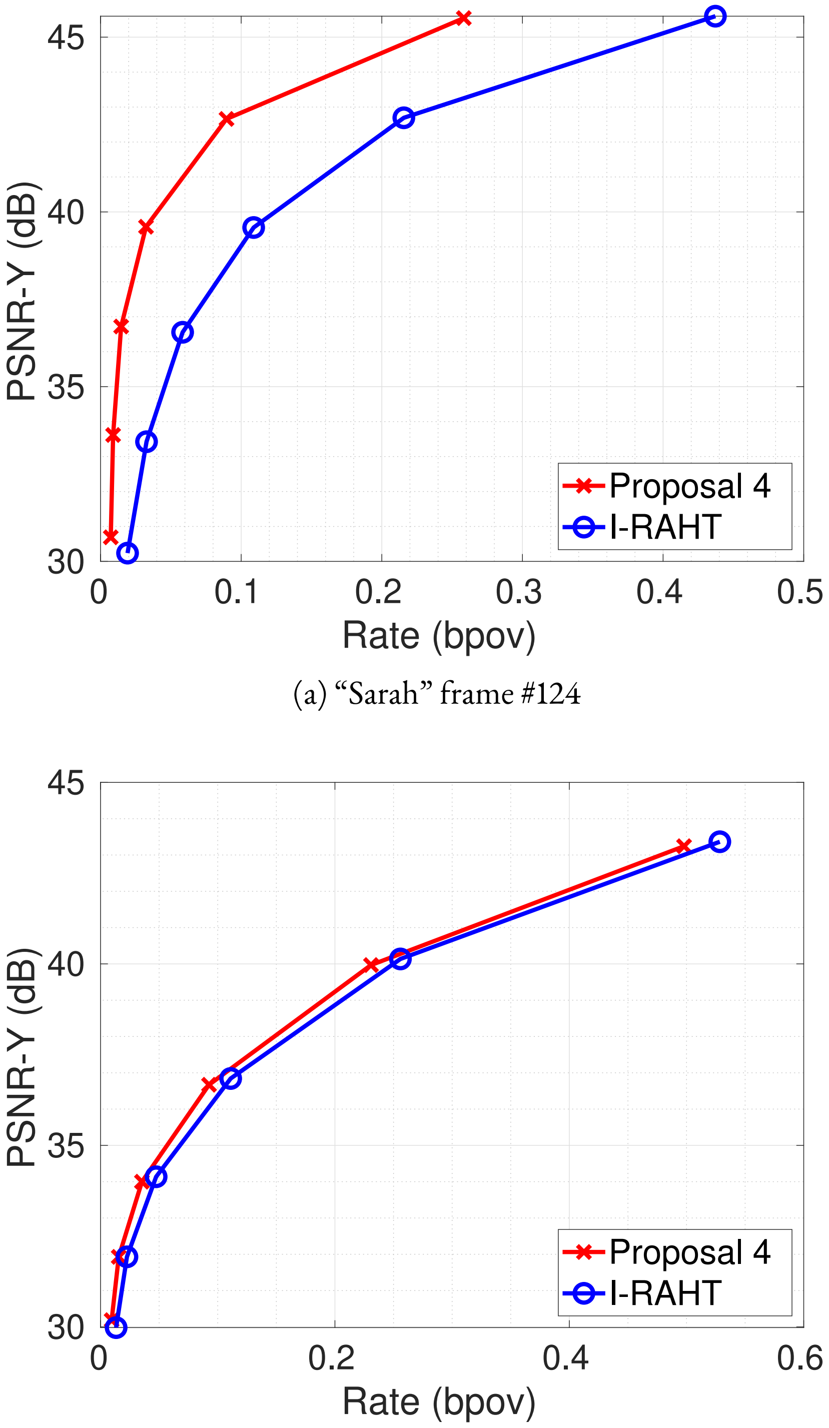


Figure 3.13: Flowchart describing the level-based multiple decision scheme.

The results of the aforementioned approach are shown in Fig. 3.14 and in Table 3.6 for point



(a) “Sarah” frame #124

(b) “Loot” frame #1073

Figure 3.14: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point clouds “Sarah” and “Loot”.

Table 3.6: Average BD PSNR-Y gains and average bitrate savings of the level-based multiple decision (Proposal 4) relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Sarah	4.59	-68.06
Loot	0.57	-15.82

clouds “Sarah” and “Loot”. One can see that for “Sarah”, significant gains over the use of only the intra-frame prediction are achieved. For “Loot”, small but consistent gains are obtained. The present adaptive scheme generated better compression results for both point clouds than all the previously devised adaptive methods.

4 RESULTS

This section presents the results obtained for the two final adaptive schemes that were devised in Sections 3.2.2 and 3.3.2: the fragment-based multiple decision (proposal 2) and the level-based multiple decision (proposal 4), respectively. The former is an adaptive combination of the nearest-neighbor inter-frame prediction with I-RAHT, segmenting the point clouds into fragments and encoding the attribute residues when advantageous. While the latter is also an adaptive combination of the nearest-neighbor inter-frame prediction alongside I-RAHT, partitioning the point clouds into small cubes and encoding the coefficient residues only for certain octree levels.

To test our approaches, we used two datasets with popular point cloud sequences: five frontal upper body sequences from Microsoft Research Labs [13] ("Andrew", "Ricardo", "Sarah", "Phil", "David") with around 200,000 to 400,000 voxels, and four full-body sequences from 8i Labs [10] ("Soldier", "Longdress", "Loot", "Redandblack") with around 700,000 to 1,100,000 voxels. The point cloud sequences from Microsoft have voxels at bit-depth 9, and the ones from 8i have voxels at bit-depth 10. The information of each sequence used is shown in Table 4.1. In Fig. 4.1, projections of one frame of each sequence are illustrated.

Table 4.1: Source, bit-depth, number of voxels, and frames available for each point cloud sequence used.

Point Cloud Sequence	Source	Bit-depth	Voxels	Frames
Andrew	Microsoft	9	$\approx 270,000$	1-317
David	Microsoft	9	$\approx 330,000$	1-215
Phil	Microsoft	9	$\approx 370,000$	1-244
Ricardo	Microsoft	9	$\approx 210,000$	1-215
Sarah	Microsoft	9	$\approx 300,00$	1-206
Longdress	8i Labs	10	$\approx 850,000$	1052-1350
Loot	8i Labs	10	$\approx 800,000$	1001-1299
Redandblack	8i Labs	10	$\approx 750,000$	1451-1749
Soldier	8i Labs	10	$\approx 1,090,000$	537-835

From each point cloud sequence, 20 frames were used and averaged to assess the compression performance of our devised schemes. Ten frames were skipped between two selected frames, covering a variety of inter-frame motions from all sequences. The immediately previous frame was always subjected to the nearest-neighbor inter-frame prediction to estimate the current frame to be compressed.

As mentioned before, in our tests, performance evaluations compare rate-distortion curves, where distortion is measured in peak signal-to-noise ratio for the luminance component, and the rate is measured in bits per occupied voxel in the compressed bitstream. The quantization steps range from 8 to 256, increased by a factor of 2 at each point in the rate-distortion curves. The devised schemes are first compared to the use of only the intra-frame prediction, in Sections 4.1 and 4.2. In Section 4.3 the two devised schemes are compared to each other.

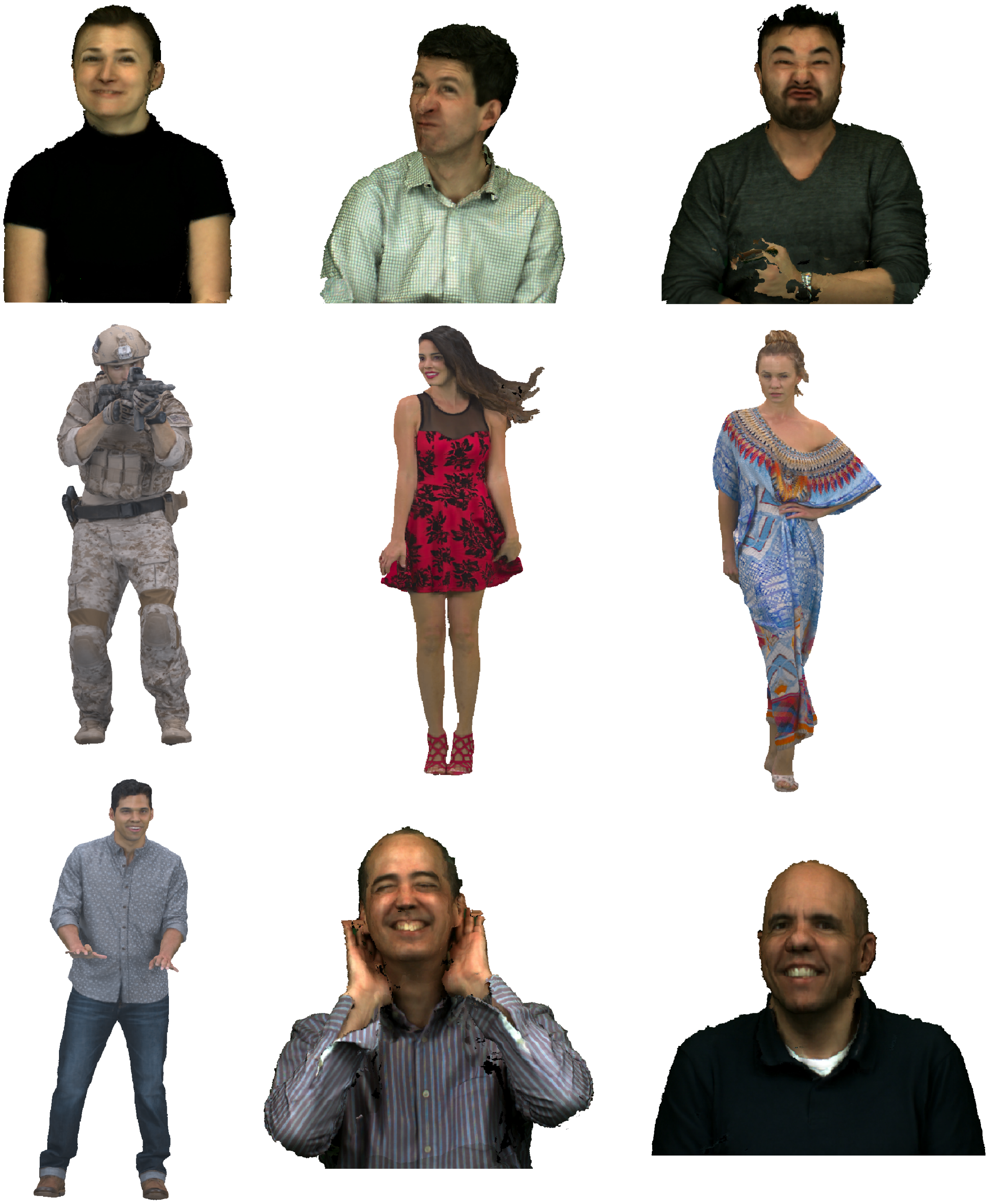


Figure 4.1: Projections of point cloud sequences used. From left to right, top to bottom: “Sarah”, “Andrew”, “David”, “Soldier”, “Redandblack”, “Longdress”, “Loot”, “Phil”, and “Ricardo”.

4.1 PROPOSAL 2 - FRAGMENT-BASED MULTIPLE DECISION

In Figs. 4.2 to 4.10, the rate-distortion performances of the fragment-based multiple decision scheme compared to I-RAHT are depicted. In Table 4.2, average BD PSNR-Y gains and average bitrate savings for each point cloud sequence are illustrated.

For the point cloud sequences "Andrew", "David", "Phil", "Loot", "Redandblack", and "Longdress" (Figs. 4.2, 4.3, 4.6, 4.8, 4.9, and 4.10) small gains over the use of only the intra-frame prediction can be observed. The PSNR gains ranged from 0.01 dB in the case of "Longdress" to 0.27 dB in the case of "Loot". While the bitrate savings ranged from 0.49% in the case of "Longdress" to 8.14% in the case of "Loot". An explanation for that is the fact that these sequences represent hard sequences to be predicted by our nearest-neighbor inter-frame prediction. More specifically, "David", "Loot", "Redandblack" and "Longdress" are represented by frames predominantly with intense motion, which makes a very hard environment for our inter-frame prediction. Also, "Andrew" and "Phil", similarly to "Redandblack" and "Longdress", are sequences with very small details on the clothes of the depicted people, this fact may also significantly contribute to a worse inter-frame accuracy.

For the point cloud sequences "Ricardo", "Sarah", and "Soldier" (Figs. 4.4, 4.5, and 4.7) more interesting gains can be observed. The PSNR gains ranged from 0.61 dB in the case of "Ricardo" to 1.26 dB in the case of "Sarah". While the bitrate savings ranged from 13.95% in the case of "Ricardo" to 27.76% in the case of "Soldier". The superior performance can be explained by the fact that these sequences do not predominantly consist of frames with intense motion, having several frames with low or no motion at all. The very high gains observed for "Sarah" throughout Section 3 were diluted by averaging the performance of several different frames with different motions, thus, with different compression performances. For the sequence "Soldier", one would expect a very hard environment for the nearest-neighbor inter-frame prediction to perform well due to the tiny details on the outfit of the person depicted. However, the successful performance in compressing this sequence can be explained by the fact that "Soldier" is mainly composed of frames with low motion.

According to the Table 4.2, the best performance in terms of bitrate is achieved for "Soldier" and in terms of PSNR is achieved for "Sarah". The worst performance in terms of bitrate and in terms of PSNR are achieved for "Longdress". The average gains in PSNR and the average bitrate savings for the fragment-based algorithm relative to I-RAHT are 0.44 dB and 10.57%, respectively. In conclusion, our fragment-based algorithm can achieve interesting gains over the use of only the intra-frame prediction for sequences composed of several low motion frames. At the same time, it guarantees a competitive performance relative to I-RAHT for point cloud sequences consisting mainly or entirely of frames with intense motion.

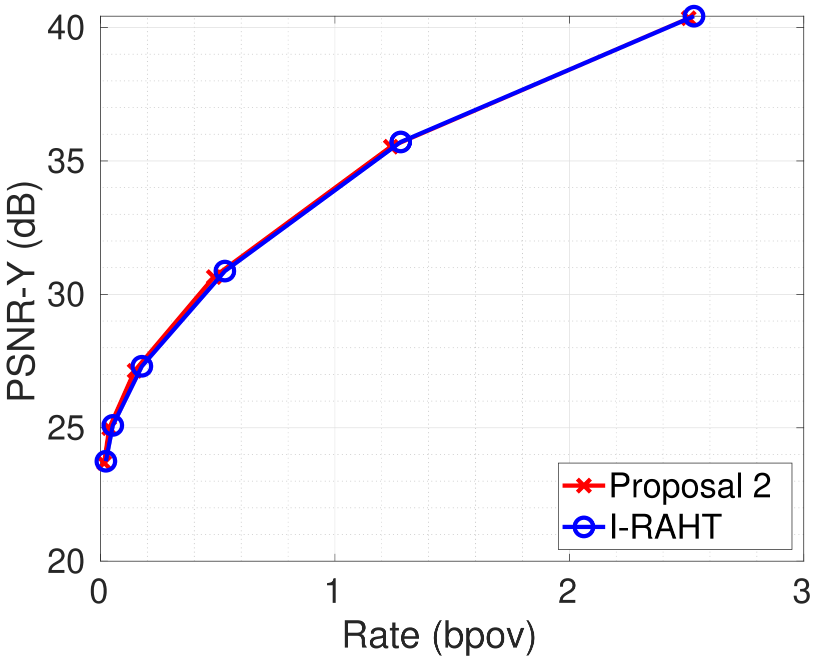


Figure 4.2: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud "Andrew".

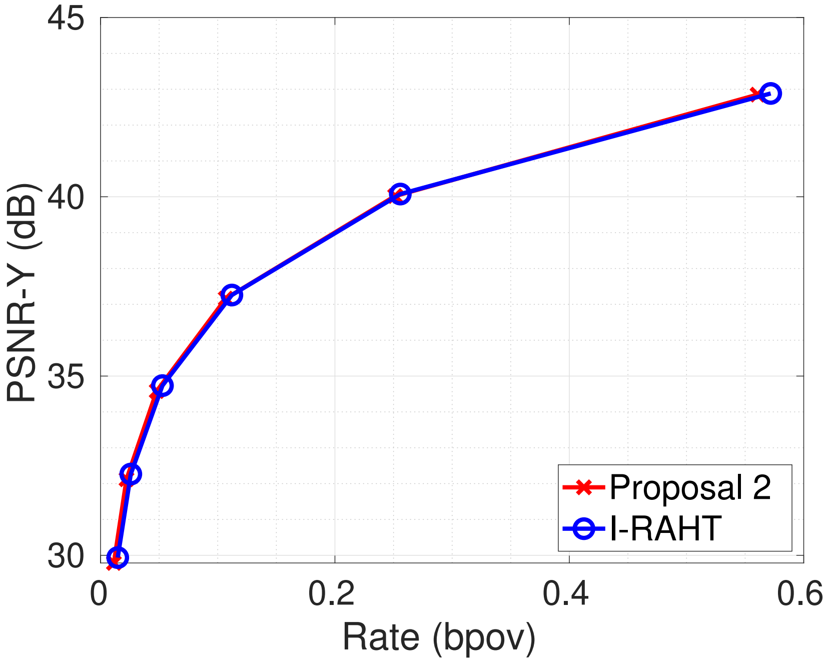


Figure 4.3: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud "David".

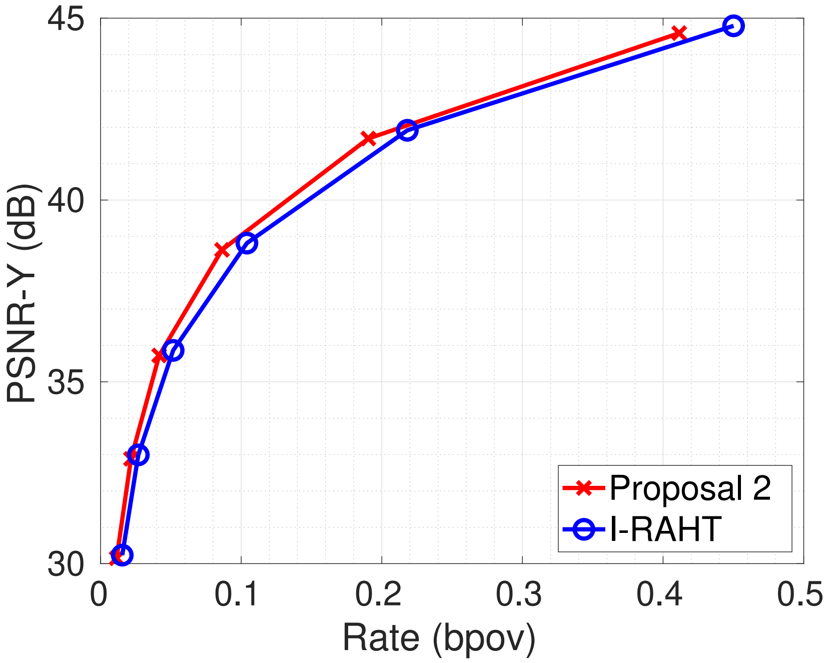


Figure 4.4: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud "Ricardo".

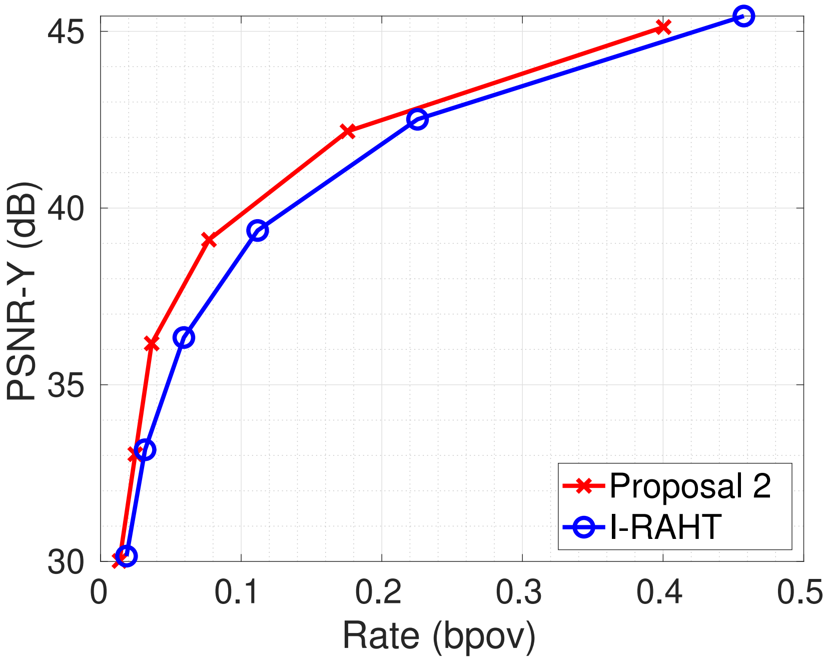


Figure 4.5: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud "Sarah".

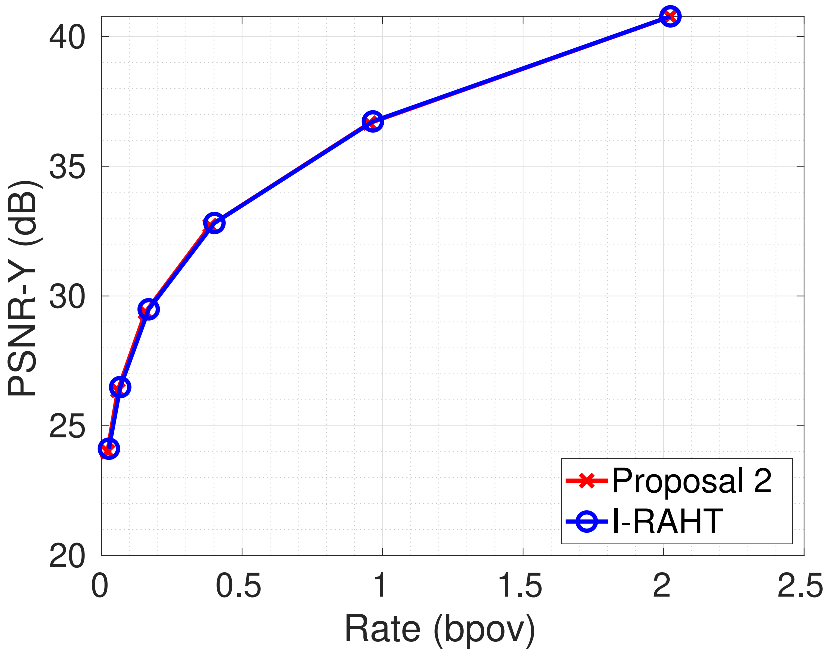


Figure 4.6: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud "Phil".

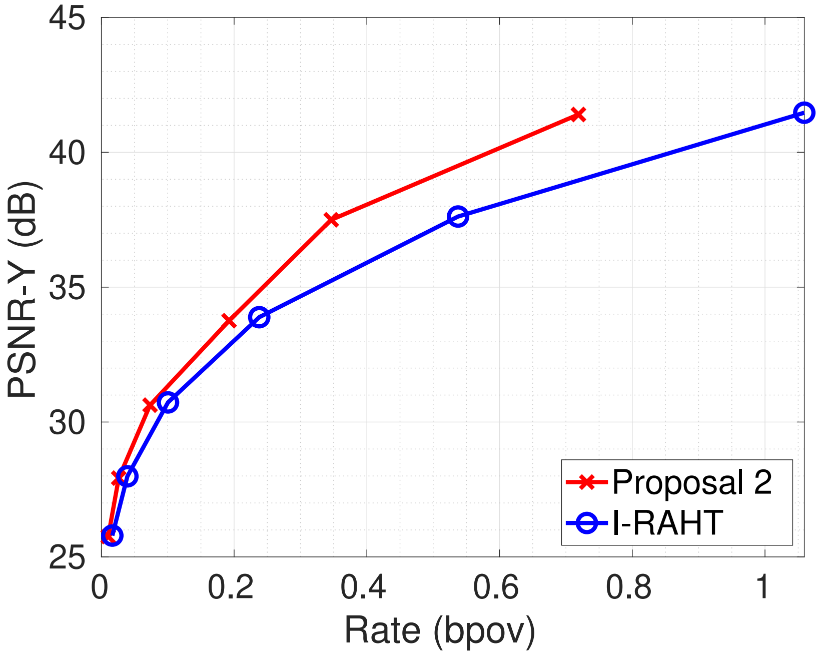


Figure 4.7: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud "Soldier".

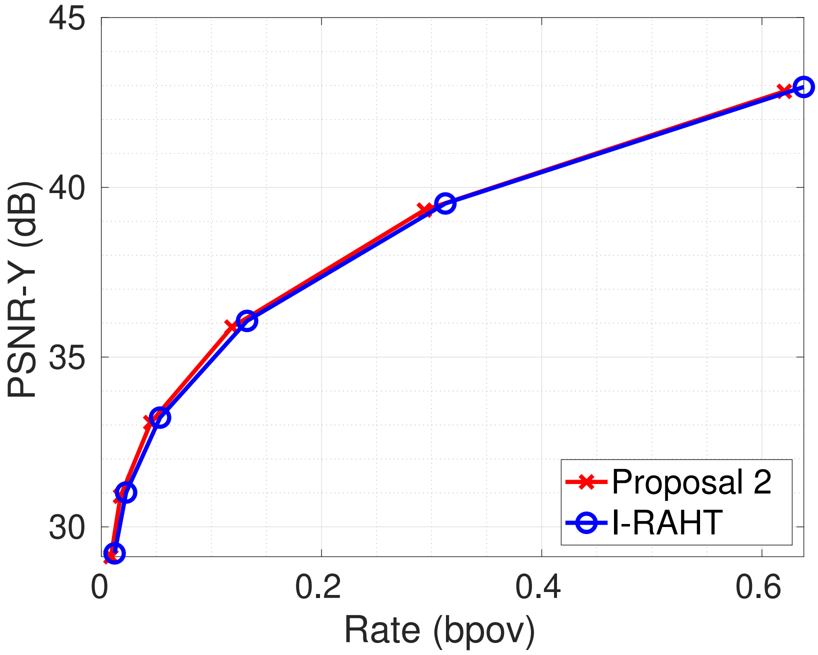


Figure 4.8: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud “Loot”.

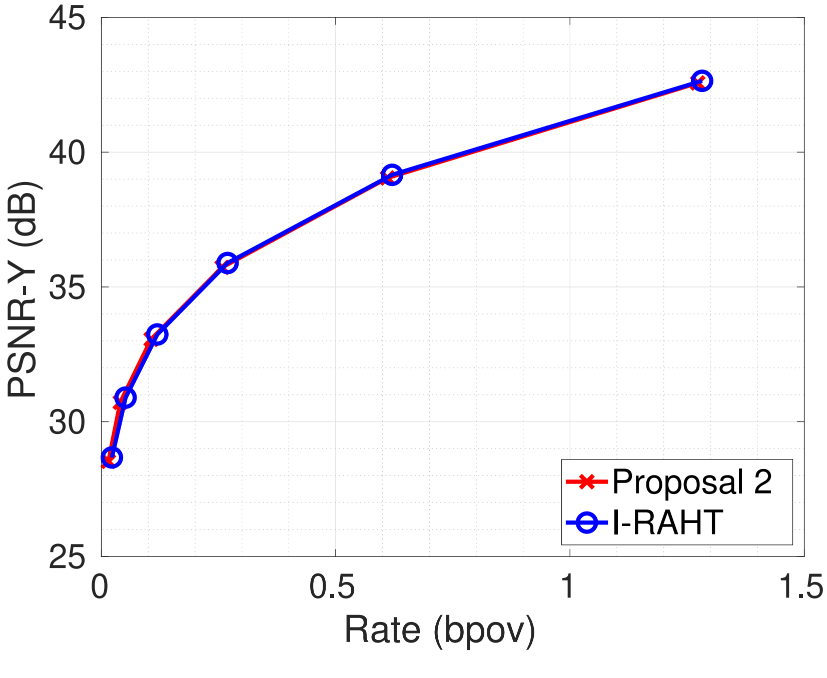


Figure 4.9: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud “Redandblack”.

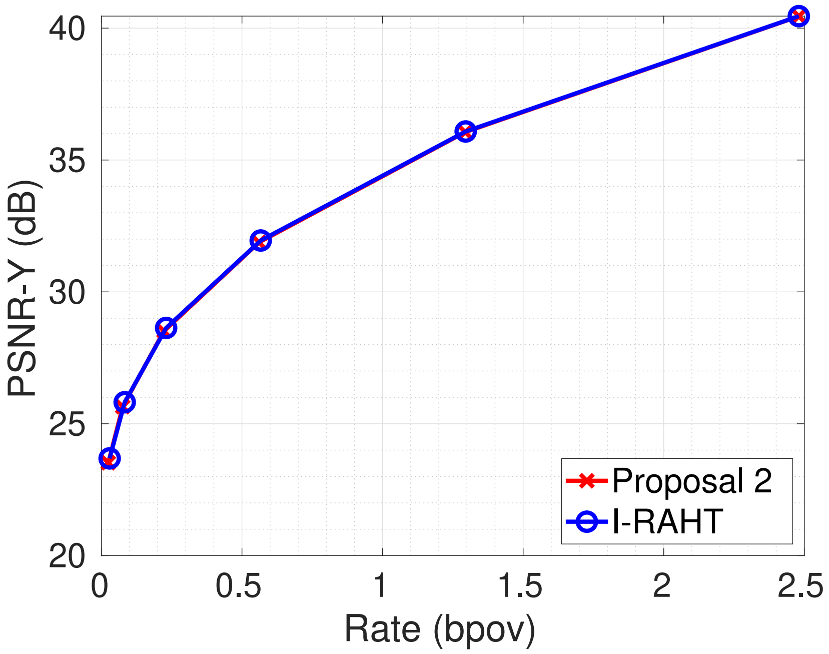


Figure 4.10: Results of the fragment-based multiple decision (Proposal 2) compared to I-RAHT for point cloud "Longdress".

Table 4.2: Average BD PSNR-Y gains and average bitrate savings of the fragment-based multiple decision (Proposal 2) relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Andrew	0.23	-6.74
David	0.16	-5.07
Ricardo	0.61	-13.95
Sarah	1.26	-24.30
Phil	0.12	-3.27
Soldier	1.17	-27.76
Loot	0.27	-8.14
Redandblack	0.16	-5.38
Longdress	0.01	-0.49
Average	0.44	-10.57

4.2 PROPOSAL 4 - LEVEL-BASED MULTIPLE DECISION

In Figs. 4.11 to 4.19, the rate-distortion performances of the level-based multiple decision scheme compared to I-RAHT are depicted. In Table 4.3, average BD PSNR-Y gains and average bitrate savings for each point cloud sequence are presented.

In this case, for the point cloud sequences "Andrew", "Phil", "Redandblack", and "Longdress"

(Figs. 4.11, 4.15, 4.18, and 4.19) small gains over the use of only the intra-frame prediction can be observed. The PSNR gains ranged from 0.14 dB in the case of "Longdress" to 0.43 dB in the case of "Andrew". While the bitrate savings ranged from 4.04% in the case of "Longdress" to 12.96% in the case of "Loot". As mentioned before, the sequences "Redandblack" and "Longdress" are very hard to be predicted due to intense motion in most of their frames and small scale details on the outfits of the depicted people. For this reason, the nearest-neighbor inter-frame prediction performs poorly and our proposed approach does not achieve large gains over the use of only the intra-frame prediction. For the sequences "Andrew" and "Phil", the very small details present on the outfits of the depicted upper bodies represent a harder environment for our inter-frame prediction to perform well. However, even though the gains are still small for the mentioned point cloud sequences, they increase if compared to the achieved results by the fragment-based multiple decision scheme.

For the point cloud sequences "David", "Ricardo", "Sarah", "Soldier", and "Loot" (Figs. 4.12, 4.13, 4.14, 4.16, and 4.17) interesting gains can be observed. The PSNR gains ranged from 0.62 dB in the case of "David" to 2.50 dB in the case of "Soldier". While the bitrate savings ranged from 17.84% in the case of "David" to 49.03% in the case of "Soldier". Again, here the very high gains obtained for "Sarah" in Section 3 were diluted after averaging the results for several frames with different motions, however, the gains remained very interesting. The performance obtained here for "Ricardo", "Sarah", and "Soldier" can be explained by the fact that these sequences are not mostly composed by frames with intense motion, they also contain some frames with low motion and no motion at all. We can also observe that, even for sequences that mostly contain intense motion, such as "David" and "Loot", our devised level-based approach was capable of obtaining interesting gains related to the use of only the intra-frame prediction.

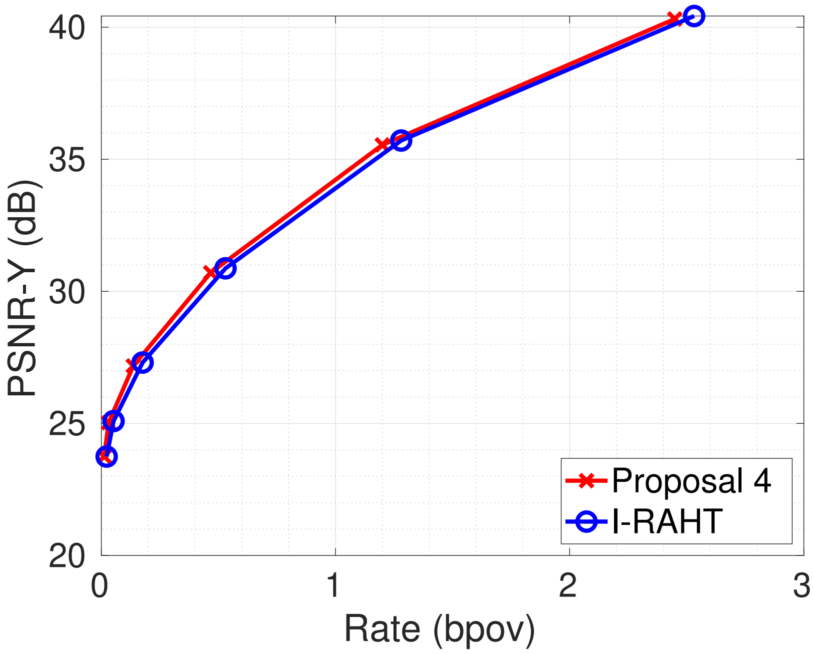


Figure 4.11: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud "Andrew".

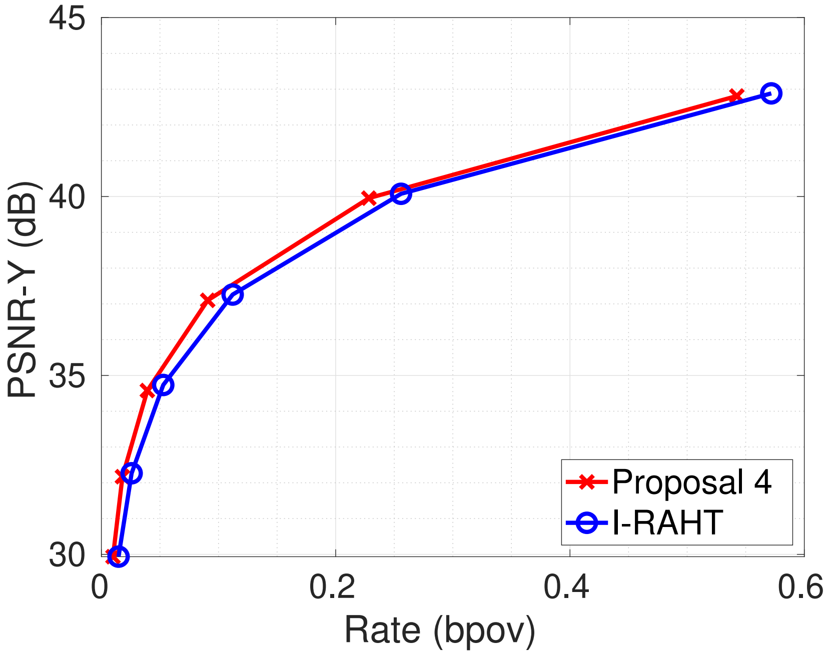


Figure 4.12: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud “David”.

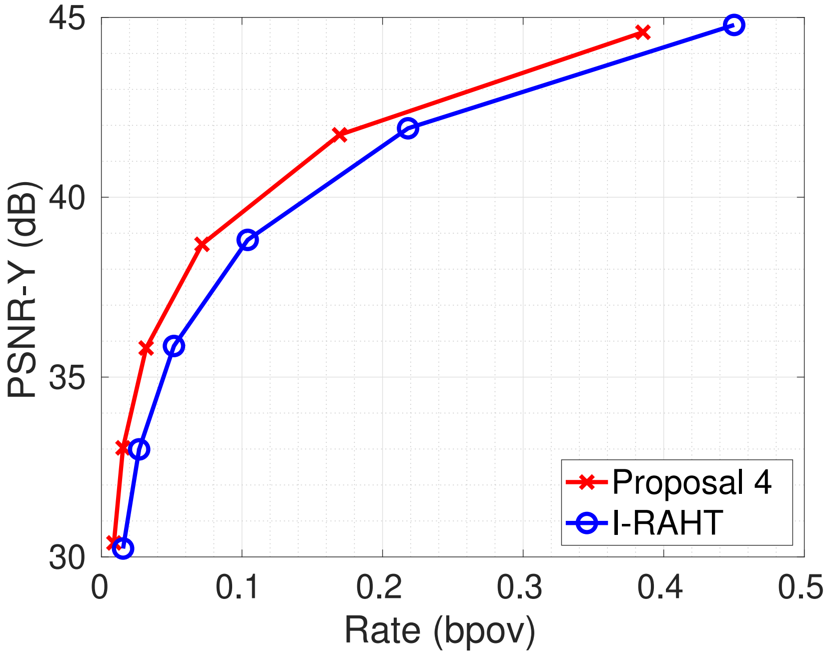


Figure 4.13: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud “Ricardo”.

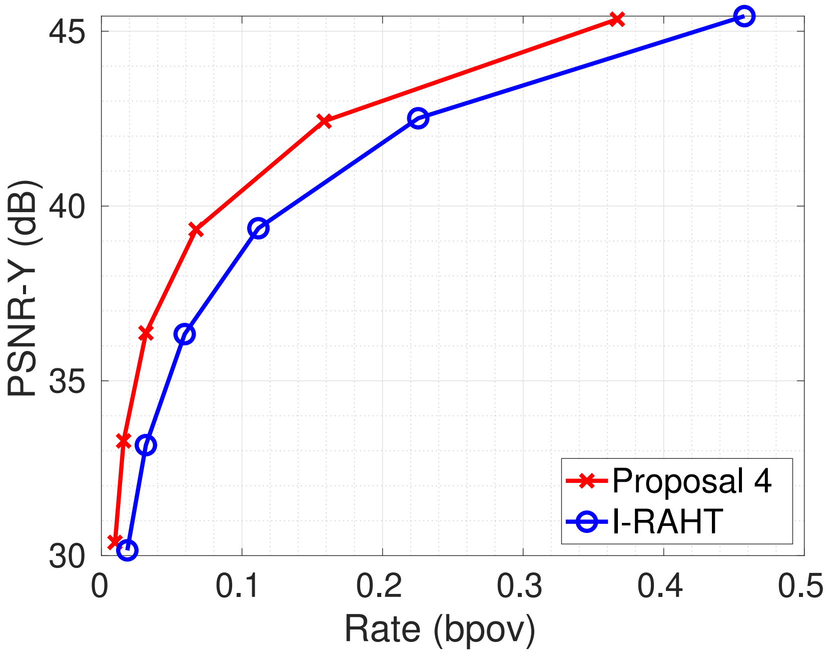


Figure 4.14: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud "Sarah".

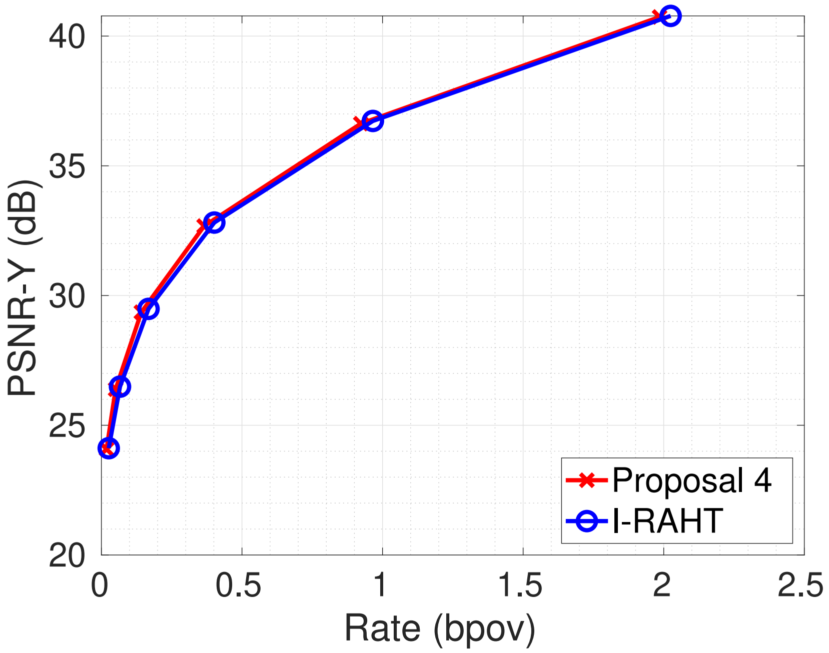


Figure 4.15: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud "Phil".

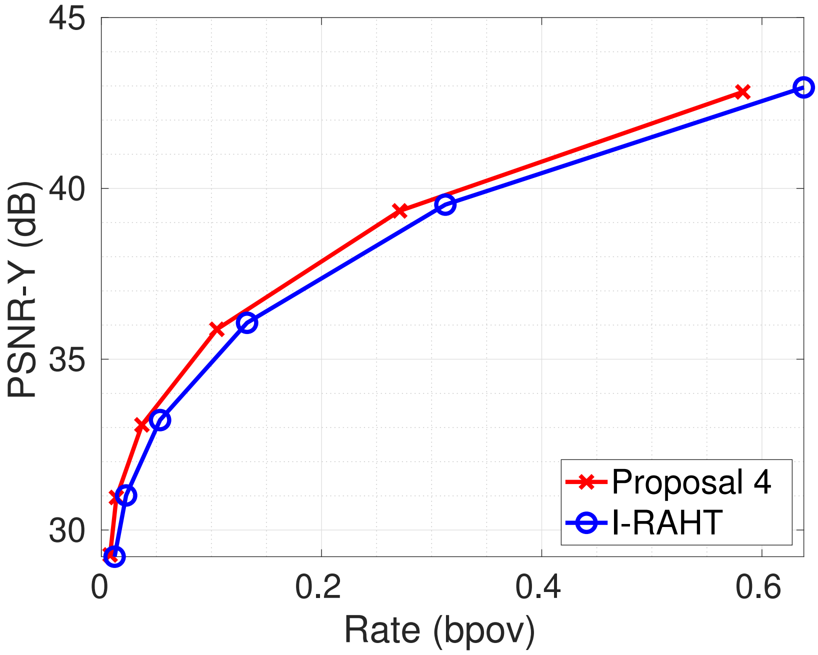


Figure 4.16: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud "Loot".

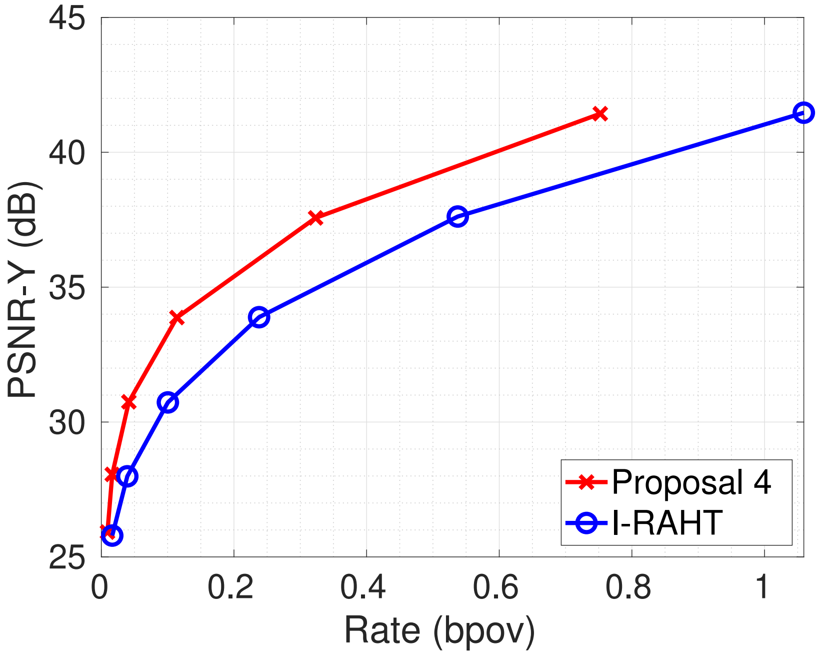


Figure 4.17: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud "Soldier".

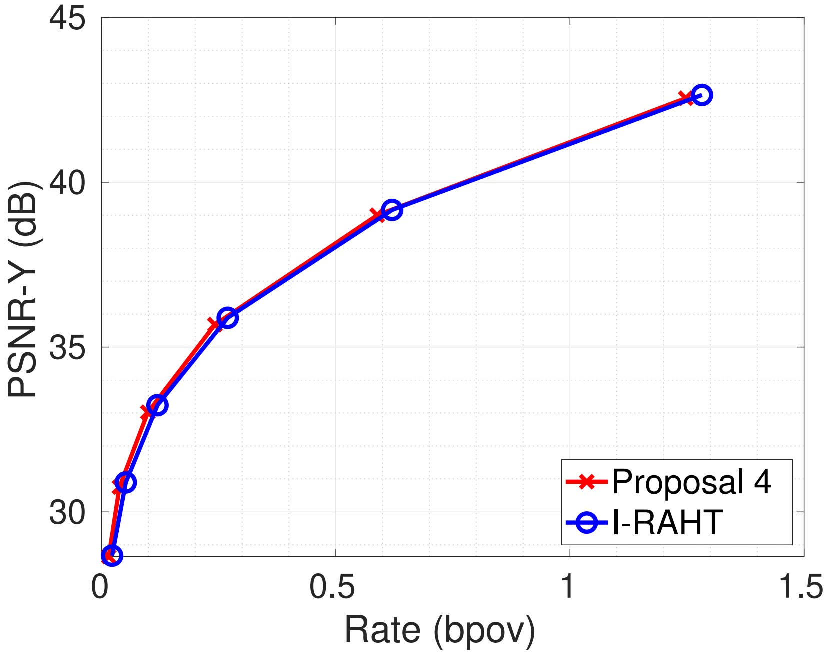


Figure 4.18: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud "Redandblack".

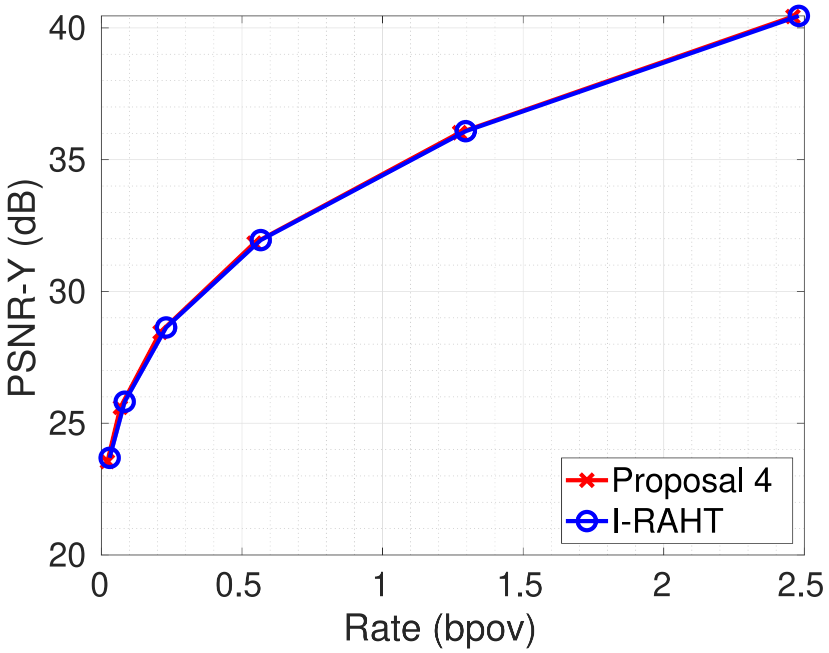


Figure 4.19: Results of the level-based multiple decision (Proposal 4) compared to I-RAHT for point cloud "Longdress".

According to the Table 4.3, the best performance in terms of bitrate and in terms of PSNR are achieved for "Soldier". The worst performance in terms of bitrate and in terms of PSNR are achieved for “Longdress”. The average gains in PSNR and the average bitrate savings for the level-based algorithm relative to I-RAHT are 0.97 dB and 21.73%, respectively. In conclusion, our level-based algorithm can achieve very interesting gains over the use of only the intra-frame prediction for sequences composed of several low motion frames. At the same time, it guarantees a slightly better performance relative to I-RAHT for point cloud sequences consisting mainly or entirely of frames with intense motion.

Table 4.3: Average BD PSNR-Y gains and average bitrate savings of the level-based multiple decision (Proposal 4) relative to I-RAHT.

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Andrew	0.43	-12.96
David	0.62	-17.84
Ricardo	1.45	-31.76
Sarah	2.20	-41.02
Phil	0.37	-9.71
Soldier	2.50	-49.03
Loot	0.71	-19.76
Redandblack	0.31	-9.43
Longdress	0.14	-4.04
Average	0.97	-21.73

4.3 COMPARISON OF PROPOSAL 2 AND PROPOSAL 4

In Figs. 4.20 to 4.28, the rate-distortion performances of the fragment-based multiple decision scheme compared to the level-based multiple decision scheme are shown. In Table 4.4, average BD PSNR-Y gains and average bitrate savings for each point cloud sequence are shown.

From the results obtained, we can observe that for all point cloud sequences, the level-based algorithm performs better relative to the fragment-based algorithm. The PSNR gains ranged from 0.13 dB in the case of “Longdress” to 1.29 dB in the case of “Soldier”. While the bitrate savings ranged from 3.65% in the case of “Longdress” to 29.57% in the case of “Soldier”. For point clouds with small details and consisting mostly (or entirely) of frames with intense motion, such as “Andrew”, “Phil”, “Redandblack”, and “Longdress” (Figs. 4.20, 4.24, 4.27, and 4.28), the advantage of the level-based approach over the fragment-based one was small. In contrast, for the other point cloud sequences, a more interesting advantage of the level-based approach can be seen.

According to the Table 4.4, the best performance of the level-based approach over the fragment-based one in terms of bitrate and in terms of PSNR are achieved for “Soldier”. In contrast, the worst performance in terms of bitrate and in terms of PSNR are achieved for “Longdress”. The average gains in PSNR and the average bitrate savings for the level-based algorithm relative to the fragment-based one are 0.54 dB and 13.36%, respectively. In conclusion, our level-based algorithm performed better for compression of dynamic point clouds for all tested sequences. Gains are somewhat significant for sequences such as "Sarah" and "Soldier" and very small for sequences such as "Longdress" and "Redandblack". Both schemes were shown to be advantageous relative to I-RAHT when compressing dynamic point clouds, but the level-based algorithm was shown here to be the best option in all cases.

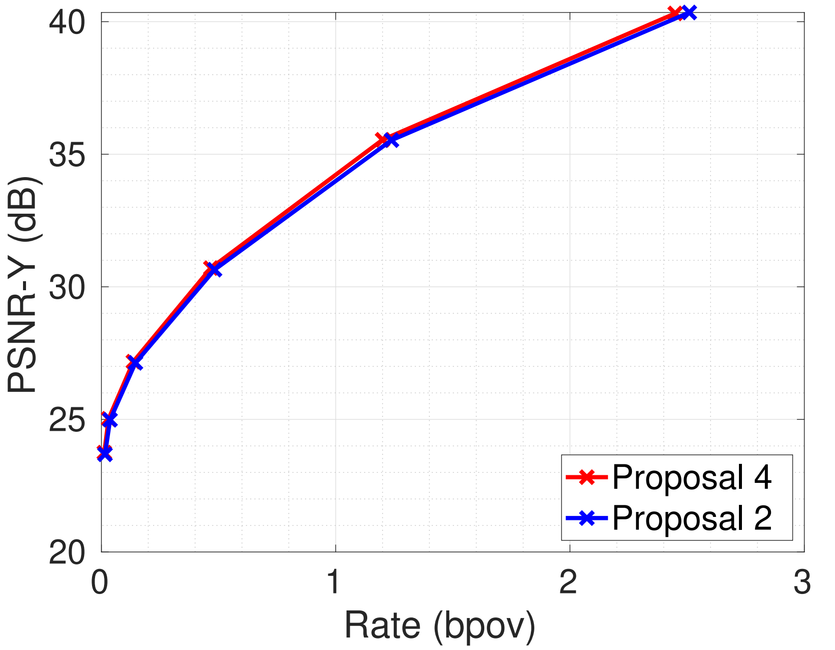


Figure 4.20: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud "Andrew".

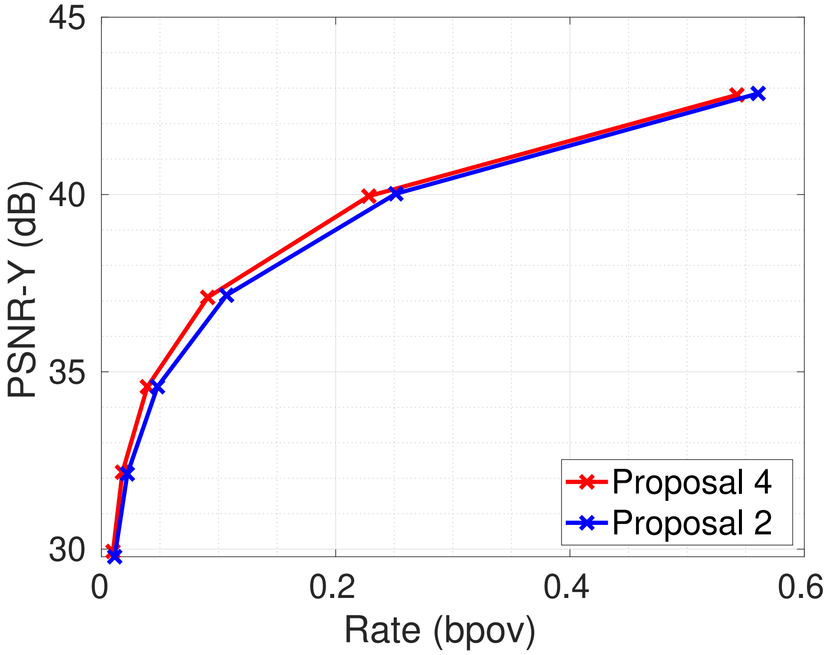


Figure 4.21: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud "David".

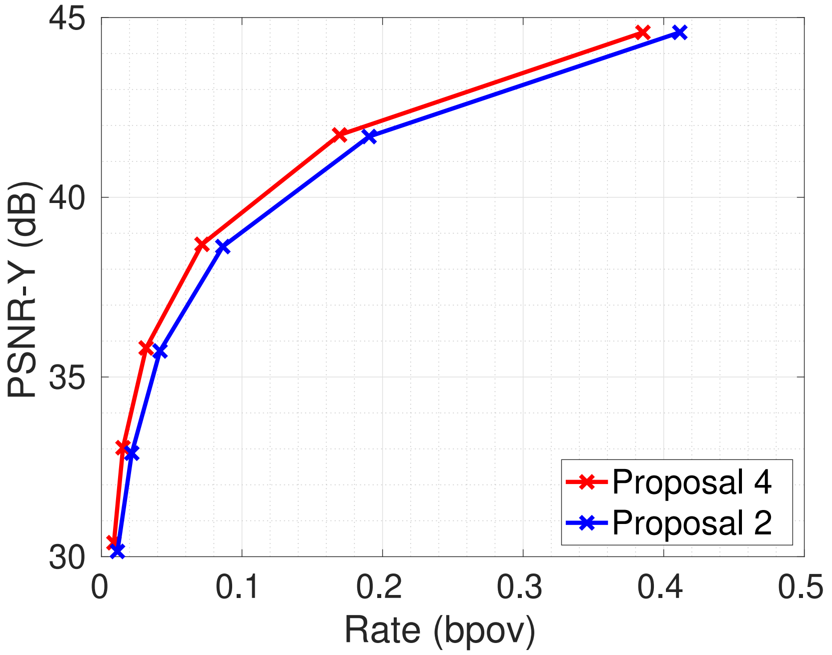


Figure 4.22: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud “Ricardo”.

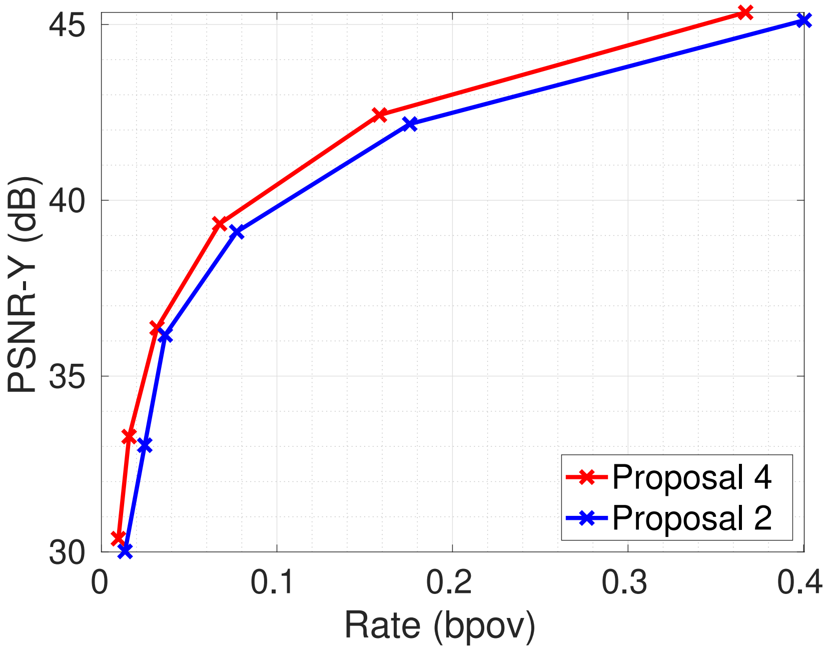


Figure 4.23: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud “Sarah”.

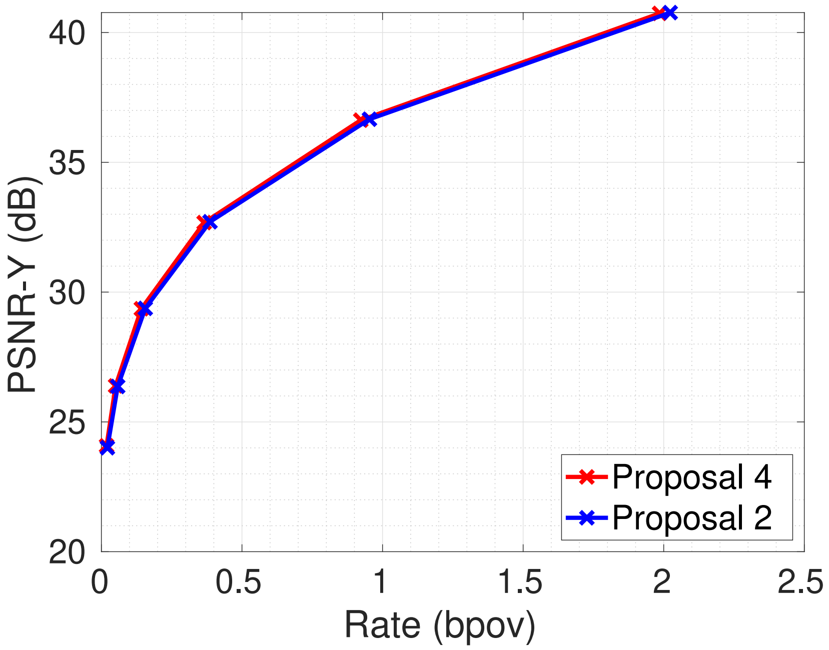


Figure 4.24: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud "Phil".

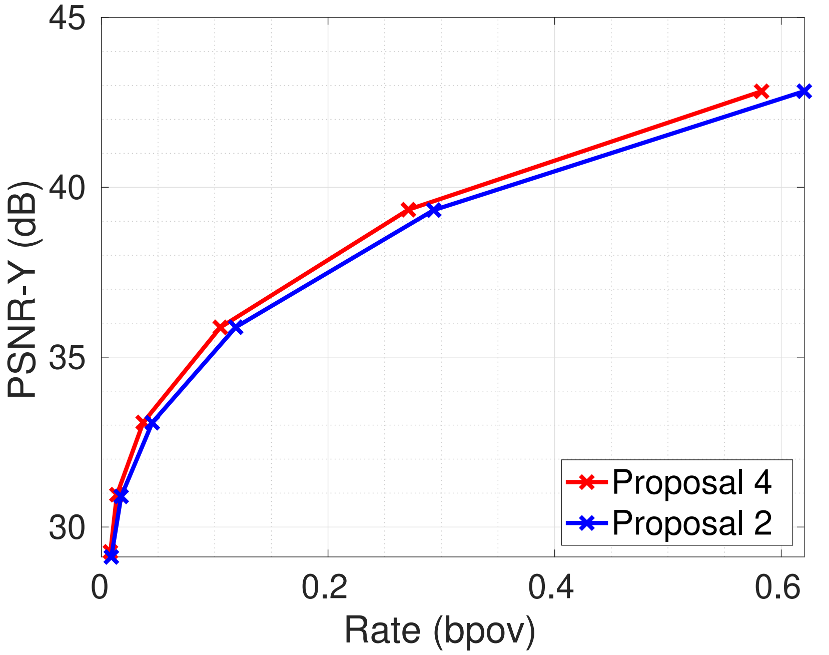


Figure 4.25: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud "Loot".

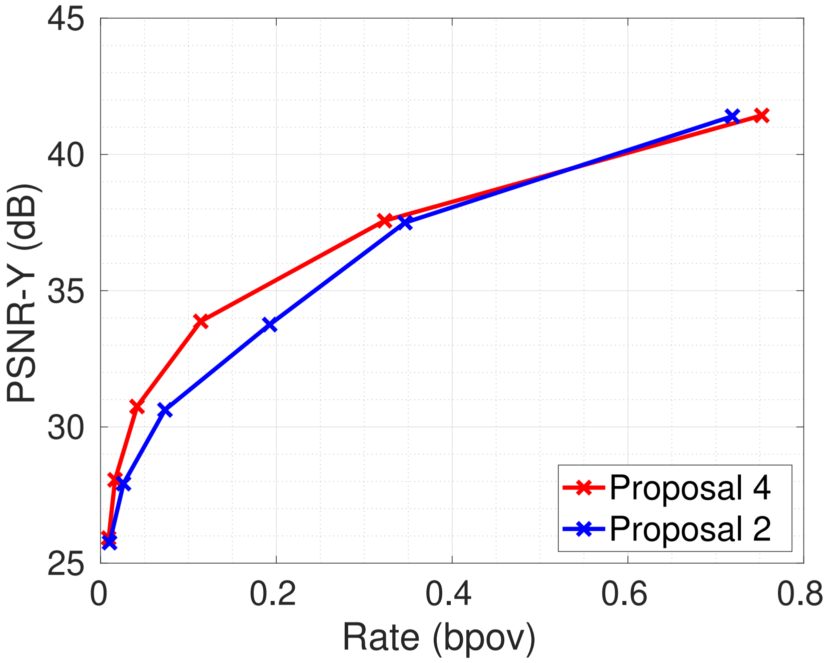


Figure 4.26: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud "Soldier".

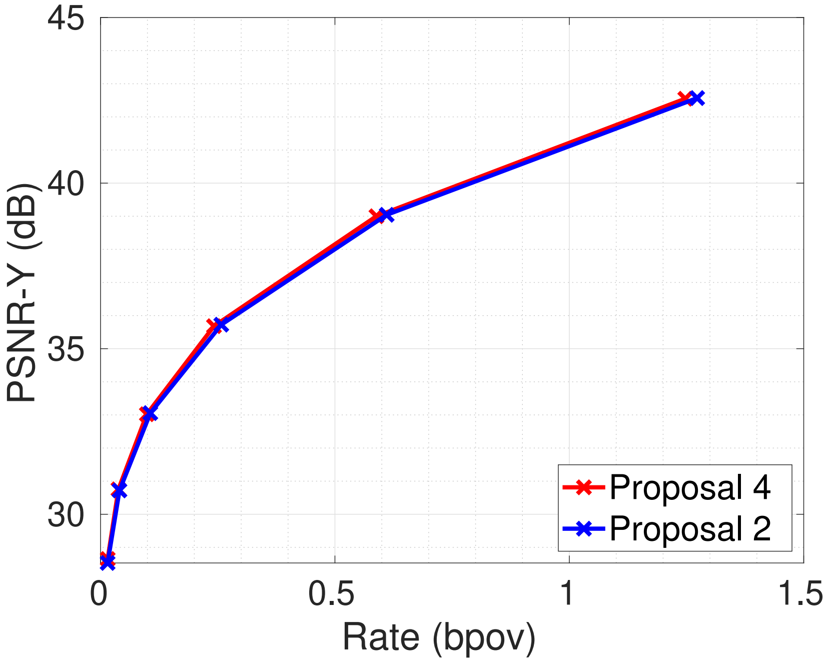


Figure 4.27: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud "Redandblack".

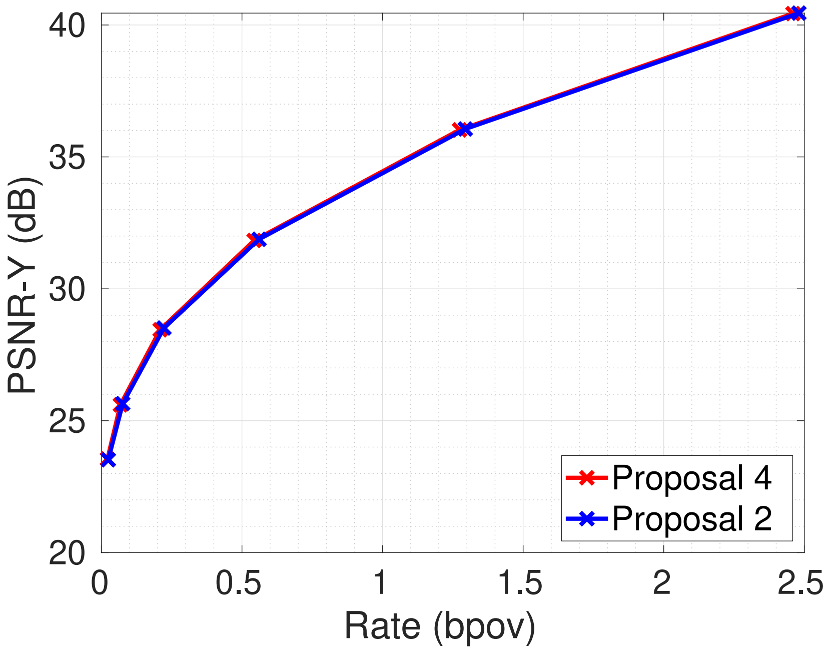


Figure 4.28: Results of level-based multiple decision (Proposal 4) compared to fragment-based multiple decision (Proposal 2) for point cloud "Longdress".

Table 4.4: Average BD PSNR-Y gains and average bitrate savings of the level-based multiple decision (Proposal 4) relative to the fragment-based multiple decision (Proposal 2).

Point Cloud Sequence	BD PSNR-Y [dB]	BD bitrate [%]
Andrew	0.21	-6.65
David	0.47	-13.43
Ricardo	0.89	-20.80
Sarah	1.04	-22.45
Phil	0.25	-6.67
Soldier	1.29	-29.57
Loot	0.44	-12.71
Redandblack	0.14	-4.29
Longdress	0.13	-3.65
Average	0.54	-13.36

5 CONCLUSIONS

The research work presented in this manuscript focused on exploring the use of a simple inter-frame prediction alongside RAHT, in order to improve attribute compression performance of dynamic point clouds. The main idea was to develop a simple scheme combining the latest version of RAHT, with an intra-frame predictive step added, referred to as "I-RAHT" with a low complexity inter-frame prediction.

Since the current motion estimation algorithms for point clouds are highly time-consuming and complex, we opted to use a simple low-complexity inter-frame prediction based on the nearest neighborhood of voxels. This inter-frame prediction was chosen because it can be efficiently implemented for time-constrained applications using k-nearest neighbor search, as in the C++ implementation of the Fast Library for Approximate Nearest Neighbors. Therefore, this work explored adaptive combinations of the intra-frame and the inter-frame predictions to achieve an improvement in compression performance of RAHT for dynamic point clouds. Our devised schemes were compared with I-RAHT because this is currently the best predictive RAHT algorithm in the literature.

We devised two final adaptive schemes: one exploring the encoding of attribute residues referred to as "fragment-based multiple decision" (proposal 2), presented in Section 3.2.2. A second one that explores the encoding of coefficient residues referred to as "level-based multiple decision" (proposal 4), presented in Section 3.3.2. Both of our devised schemes were tested for dynamic point cloud compression against I-RAHT. We use two popular datasets of dynamic point clouds, from 8i Labs and Microsoft Research Labs. The one from 8i Labs consists of four full-body point cloud sequences, and the one from Microsoft Research Labs consists of five upper body point cloud sequences. We tested our devised approaches in 20 point cloud frames from each sequence, always skipping ten frames between two selected frames. Hence, covering a variety of situations and motions. Also, the immediately previous frame was always used to predict the frame to be compressed.

From our rate-distortion curves presented in Sections 3 and 4, it is possible to observe that the fragment-based algorithm is capable of achieving outstanding gains over the use of only the intra-frame prediction for point cloud frames with low or no motion. In the case of point cloud frames with intense motion, our scheme can guarantee a competitive performance against the intra-frame case. Overall, the fragment-based algorithm achieves BD PSNR-Y gains of 0.44 dB and bitrate savings of 10.57%. We can also observe that the level-based algorithm can also achieve outstanding gains over I-RAHT for point cloud frames with low or no motion. For the cases with intense motion, our algorithm can slightly outperform the use of only the intra-frame prediction. Overall, the level-based algorithm achieves BD PSNR-Y gains of 0.97 dB and bitrate savings of 21.73%. Therefore, our devised schemes can be an alternative for the compression of dynamic point clouds, guaranteeing at least a similar performance in the worst-case scenario and achieving significant gains in the best-case scenario.

Finally, comparing the two final devised schemes, it can be observed that the level-based algorithm presents a better compression performance than the fragment-based one, for all tested point cloud sequences and all situations, with low motion or intense motion.

5.1 FUTURE WORK

Future work may focus on the use of better inter-frame prediction algorithms for point clouds that are also time-efficient. An excellent inter-frame predictor will be capable of improving even more, the compression performance of our devised approaches. One alternative is to invest in research to design suitable motion estimation and compensation algorithms for practical applications. Ideas on how to do that efficiently might start by exploring the motion estimation in the 2D since it is already a well-solved problem with plenty of algorithms available. The idea would be to project the 3D point clouds into 2D and perform motion estimation. Then, recover 3D motion vectors from the 2D ones obtained.

Another research line that can be further explored based on this work is to examine the use of inter-frame prediction algorithms alongside RA-GFT. The latter is a recently published transform for attribute compression of point clouds capable of outperforming RAHT. The use of inter-frame prediction algorithms alongside it may allow achieving even better results. Both inter-frame and intra-frame prediction algorithms can be combined with RA-GFT in order to exploit inter-frame and intra-frame redundancies.

REFERENCES

[1] 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.

[2] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai, "An Overview of Ongoing Point Cloud Compression Standardization Activities: Video-based (V-PCC) and Geometry-based (G-PCC)," *APSIPA Transactions on Signal and Information Processing*, vol. 9, 2020.

[3] 3DG, "G-PCC Codec Description v12," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Approved WG 11 document N18891, October 2020.

[4] ——, "V-PCC Codec Description v12," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Approved WG 11 document N18892, October 2020.

[5] 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, August 2016.

[6] S. Lasserre and D. Flynn, "On an Improvement of RAHT to Exploit Attribute Correlation," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Tech. Rep. m47378, March 2019.

[7] G. Bjontegaard, "document VCEG-M33: Calculation of Average PSNR Differences between RD-Curves," in *ITU-T VCEG Meeting*, Austin, Texas, USA, 2001.

[8] C. Tulvan and M. Preda, "Point Cloud Compression for Cultural Objects," ISO/IEC MPEG JTC1/SC29/WG11, Geneva, CH, Document m37240, October 2015.

[9] T. M. Borges, "Fractional Super-Resolution of Voxelized Point Clouds," Master's thesis, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília-DF, Brazil, 2021.

[10] 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.

[11] 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, April 2017.

[12] L. Goode. (2017) Are Holograms the Future of How We Capture Memories. The Verge.

[13] C. Loop, Q. Cai, S. Escolano, and P. Chou, “Microsoft Voxelized Upper Bodies - A Voxelized Point Cloud Dataset,” ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG), input document m38673/M72012, May 2016.

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

[15] E. Pavez, B. Girault, A. Ortega, and P. A. Chou, “Region-Adaptive Graph Fourier Transform for 3D Point Clouds,” pp. 2726–2730, October 2020.

[16] D. Thanou, P. A. Chou, and P. Frossard, “Graph-Based Compression of Dynamic 3D Point Cloud Sequences,” *IEEE Transactions on Image Processing*, vol. 25, no. 4, pp. 1765–1778, 2016.

[17] 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.

[18] 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.

[19] 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, pp. 828–842, April 2017.

[20] C. Dorea and R. L. de Queiroz, “Block-Based Motion Estimation Speedup for Dynamic Voxelized Point Clouds,” in *2018 25th IEEE International Conference on Image Processing (ICIP)*, 2018, pp. 2964–2968.

[21] C. Dorea, E. M. Hung, and R. L. de Queiroz, “Local Texture and Geometry Descriptors for Fast Block-Based Motion Estimation of Dynamic Voxelized Point Clouds,” in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 3721–3725.

[22] A. L. Souto, R. L. de Queiroz, and C. Dorea, “A 3D Motion Vector Database for Dynamic Point Clouds,” *ArXiv e-prints, arXiv: 2008.08438 [eess.IV]*, August 2020.

[23] M. Muja and D. Lowe, “Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration,” in *Proceedings of the 4th International Conference on Computer Vision Theory and Applications*, vol. 1, January 2009, pp. 331–340.

[24] 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.

[25] ——, "Transformada RAHT com Predição Inter-Quadros para Compressão de Nuvem de Pontos," in *Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT)*, November 2020.

[26] A. L. Souto, R. L. de Queiroz, and C. Dorea, "Motion-Compensated Predictive RAHT for Dynamic Point Clouds," *submitted to IEEE Transactions on Image Processing*, 2020.

[27] E. Pavez, A. L. Souto, R. L. de Queiroz, and A. Ortega, "Multi-Resolution Intra-Predictive Coding of 3D Point Clouds Attributes," *submitted to IEEE International Conference on Image Processing (ICIP)*, 2021.

[28] N. Namitha, S. M. Vaitheeswaran, V. Jayasree, and M. Bharat, "Point Cloud Mapping Measurements Using Kinect RGB-D Sensor and Kinect Fusion for Visual Odometry," *Procedia Computer Science*, vol. 89, pp. 209–212, 12 2016.

[29] R. Mekuria, M. Sanna, S. Asioli, E. Izquierdo, D. C. A. Bulterman, and P. Cesar, "A 3D Tele-Immersion System Based on Live Captured Mesh Geometry," in *Proceedings of the 4th ACM Multimedia Systems Conference*. ACM Press, 2013.

[30] 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.

[31] 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, June 2016.

[32] R. Schnabel and R. Klein, "Octree-Based Point-Cloud Compression," in *Proceedings of the 3rd Eurographics / IEEE VGTC Conference on Point-Based Graphics*, 2006, p. 111–121.

[33] Y. Huang, J. Peng, C. . J. Kuo, and M. Gopi, "A Generic Scheme for Progressive Point Cloud Coding," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 2, pp. 440–453, 2008.

[34] C. Loop, C. Zhang, and Z. Zhang, "Real-time High-Resolution Sparse Voxelization with Application to Image-based Modeling," in *Proceedings of the 5th High-Performance Graphics Conference*. ACM Press, 2013.

[35] D. Meagher, "Geometric Modeling using Octree Encoding," *Computer Graphics and Image Processing*, vol. 19, no. 2, pp. 129–147, June 1982.

[36] D. C. Garcia and R. L. de Queiroz, "Context-based Octree Coding for Point-Cloud Video," in *IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 1412–1416.

[37] ——, "Intra-Frame Context-Based Octree Coding for Point-Cloud Geometry," in *IEEE International Conference on Image Processing (ICIP)*, October 2018, pp. 1807–1811.

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

[39] C. Zhang, D. Florêncio, and C. Loop, "Point Cloud Attribute Compression with Graph Transform," in *IEEE International Conference on Image Processing (ICIP)*, October 2014, pp. 2066–2070.

[40] 3DG, "PCC Test Model Category 3 v0," ISO/IEC MPEG JTC1/SC29/WG11, Macau, China, Approved WG 11 document w17249, October 2017.

[41] 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, April 2018.

[42] Shao-Yi Chien, Yu-Wen Huang, Ching-Yeh Chen, H. H. Chen, and Liang-Gee Chen, "Hardware Architecture Design of Video Compression for Multimedia Communication Systems," *IEEE Communications Magazine*, vol. 43, no. 8, pp. 123–131, 2005.

[43] S. Lasserre and D. Flynn, "[PCC] Inference of a Mode using Point Location Direct Coding in TMC3," ISO/IEC MPEG JTC1/SC29/WG11, Gwangju, Korea, Input contribution m42239, January 2018.

[44] G. Sandri, R. L. de Queiroz, and P. A. Chou, "Comments on "Compression of 3D Point Clouds Using a Region-Adaptive Hierarchical Transform"," *ArXiv e-prints, arXiv: 1805.09146v1 [eess.IV].*

[45] H. Malvar, "Adaptive Rrun-Length/Golomb-Rice Encoding of Quantized Generalized Gaussian Sources with Unknown Statistics," in *Data Compression Conference*, April 2006, pp. 23 – 32.

[46] 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, September 2019.