TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

NAHID SHEIKHI POUR

IMPROVEMENTS FOR PROJECTION-BASED POINT CLOUD COMPRESSION

Master of Science Thesis

Examiners: Prof. Moncef Gabbouj
Dr. Sebastian Schwarz
Examiners and topic approved by the
Faculty Council of the Faculty of
Computing and Electrical Engineering
on 30th of November 2017

ABSTRACT

NAHID SHEIKHI POUR: Improvements for Projection-based Point Cloud Compression
Tampere University of Technology
Master of Science Thesis, 63 pages
December 2018
Master's Degree Programme in Information Technology
Major: Signal Processing
Examiners: Prof. Moncef Gabbouj
Dr. Sebastian Schwarz
Keywords: Point Cloud, Volumetric Video Data, Projection, Video Coding, Compression

Point clouds for immersive media technology have received substantial interest in recent years. Such representation of three-dimensional (3D) scenery provides freedom of movement for the viewer. However, transmitting and/or storing such content requires large amount of data and it is not feasible on today's network technology. Thus, there is a necessity for having efficient compression algorithms in order to facilitate proper transmission and storage of such content.

Recently, projection-based methods have been considered for compressing point cloud data. In these methods, the point cloud data are projected onto a 2D image plane in order to utilize the current 2D video coding standards for compressing such content. These coding schemes provide significant improvement over state-of-the-art methods in terms of compression efficiency. However, the projection-based point cloud compression requires special handling of boundaries and sparsity in the 2D projections. This thesis work addresses these issues by proposing two methods which improve the compression performance of both intra-frame and inter-frame coding for 2D video coding of volumetric data and meanwhile reduce the coding artifacts. The conducted experiments illustrated that the bitrate requirements are reduced by around 26% and 29% for geometry and color attributes, respectively compared to the case that the proposed algorithms are not applied. In addition, the proposed techniques showed negligible complexity impact in terms of encoding and decoding runtimes.

PREFACE

The conducted research study in this thesis was done in a collaboration with Nokia Technologies and Laboratory of Signal Processing at Tampere University of Technology (TUT) during the year 2017-2018.
I would like to thank Professor Moncef Gabbouj for highly professional guidance and valuable advice during my studies and working time.
My gratitude and appreciation to my supervisor Dr. Sebastian Schwarz from Nokia Technologies for his strong technical and academic guidance, and insightful comments during the thesis project.
Also I would like to gratefully acknowledge and thank all colleagues in Nokia Technologies, Miska Hannuksela, Jari Hagqvist, Vinod Malamal Vadakital, Alireza Aminlou, Kimmo Roimela, Emre Aksu, Jani Lainema, Deepa Naik, Henri Toukomaa, and Antti Hallapuro for their assistance and providing friendly office environment.
I have had very kind friends whom support me from long distances: Sheida Shahabadi, Simin Shabanpour, and Sara Tabatabaei. Also having chance to find good friends in Tampere: Sounak Bhattacharya, Khazar Khorrami, Zeinab Rezaei, Shuang Luo, Saboktakin Hayati and Aidin Alinezhad.
I would like to express my very special thanks to my dear boyfriend Ramin for his love, patience, encouragement, and support that he has been shown throughout the working and studying process.
And last but not least, I deeply appreciate my family for their love and endless support that I have received during my studies. My mother, Parvaneh Abtin, who always sacrificed her wishes to give me peaceful and happy life. My father, Asadollah, who did his best for family. I wish to thank my brothers, Nima and Hooman for their guidances. And thanks a million to my lovely sister, Anahita who motivates me every day and keeps my smile glowing.

Tampere, November 2018
Nahid Sheikhi pour

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

2D	Two-Dimensional
3D	Three-Dimensional
3D-HEVC	3D High Efficiency Video Coding
6DoF	Six Degrees of Freedom
AI	All Intra
AR	Augmented Reality
BDBR	Bjontegaard Delta Bit Rate
BR	Bit Rate
CfP	Call for Proposals
CU	Coding Unit
DCT	Discrete Cosine Transform
H.265/HEVC	High Efficiency Video Coding
HMD	Head Mounted Display
HVS	Human Visual System
KLT	Karhunen-Loeve Transform
MPEG	Moving Picture Experts Group
MR	Mixed Reality
MSE	Mean Square Error
PCC	Point Cloud Compression
PSNR	Peak Signal-to-Noise Ratio
RA	Random Access
RD	Rate-Distortion
SHVC	Scalable Extension of High Efficiency Video Coding
VR	Virtual Reality

DEFINITIONS

Point Cloud Frame: For a static point cloud it is a representation at a certain time instance and for dynamic, it is defined as capture of point clouds in a period of time.
Geometry: The position of point clouds in 3D space, i.e. in 3D (x,y,z) is coordinates of the points.
Attribute: A feature or set of features, excluding geometry, associated with a point, e.g., (R, G, B) color values, I for reflectance.
Lossy Geometry: The location of decoded compressed point cloud is not entirely similar to the uncompressed one. Even the number of decoded points is not equal to original cloud.
Lossless Geometry: The location of decoded compressed point cloud is identical to the uncompressed one, in another words, the location of coordinates before compression and after compression does not change. Even the number of decoded points are equal to original cloud.
Lossy Attribute: The attribute values of the compressed file after decoding are not necessarily numerically identical to the attribute values of uncompressed.
Lossless Attribute: The attribute values of the compressed file after decoding are numerically identical to the attribute values of uncompressed.

1. INTRODUCTION

In recent years, immersive experience of real world in virtual environment has received remarkable attention. Moreover, the capturing technology of 3D data which enables us to reconstruct highly detailed volumetric data has been developed considerably. Volumetric content represents a three-dimensional (3D) scene or object. Such content is either generated from 3D models, i.e. Computer-generated Imagery (CGI), or can be acquired from real-world scenes using a variety of capturing solutions, e.g. multi-camera setup, advanced 3D laser scanner, depth sensors or other technologies which are able to produce a 3D representation of the surface. Generic representation formats for such volumetric data are triangle meshes, point clouds, or array of voxels [3].
Point cloud is defined as a set of (x, y, z) in 3D coordinates without strict order and local topology. Typically each point in the cloud has the same number of attributes (i.e., color, normal directions, reflectance) to represent a scene or an object [5].
In 360° omnidirectional content capturing, viewport of a scene is recorded regarding one center point in 3D coordinate system. Therefore, there is a limitation for the user to be involved in 3D scene for example to move forward and back of scene or objects since depth information is not recorded. One of the problems of scene model

Figure 1.1 *Tele-immersive experience enabled by point cloud compression.*

is occlusion. Whereas the volumetric content provide six degrees-of-freedom (6DoF) capabilities, in which the viewer can freely move and choose viewpoint in addition to the head movement by using for example a Head Mounted Display (HMD) device. Volumetric video represents 3D data in a way that the observer has freedom to navigate freely in the captured scene. Hence, such data has high importance for virtual reality (VR), augmented reality (AR), or mixed reality (MR) applications, especially for providing 6DOF viewing capabilities [3].

Examples of volumetric data applications are in construction business, agriculture and vegetation management, education, medicine, etc. Figure 1.1 presents an example of tele-immersive experience of point cloud content by HMD.

Apart from the mentioned advantages and applications of such content, the large number of points which are generated to represent the volumetric scenery introduce challenges and limitations to storage and transmission over current network technology. Therefore, efficient compression algorithms are needed to help this technology to expand in all aspects. The aim of the proposed solution is to develop the compression algorithms that cope with the needs of interactive, tele-immersive applications, such as VR, AR or MR with 6DoF capabilities.

Recently, projection-based point cloud compression solution has been introduced. The projection-based point cloud compression significantly improved over the state-of-the-art method [1] in terms of coding efficiency. However, this approach requires special attention on the sharp transitions at the projected object boundaries and sparsity in the 2D projections due to oversampling issues. Consequently, such areas create sub-optimal compression performance using the traditional 2D video compression technology.

The aim of this thesis is to provide solutions for the above-mentioned challenges in order to improve the coding efficiency of the projection-based schemes. In this thesis, two methods have been proposed for this purpose. The first proposal targets improving the coding performance in the generated sharp edges of the projection-based method by applying a smoothing filter in the boundary areas. The Edge Smoothing operation intends to increase the correlation of samples in the boundary areas of the projected content in order to make it suitable for compression operations. Furthermore, Patch Refinement technique is considered for filtering the sparsely projected point cloud data in projection plane for reducing sparsity and improving the compression performance.

1.1 Thesis Outline

The rest of this thesis work is structured as follows:

- **Chapter 2:** Reviews the theoretical background of the study.
- **Chapter 3:** Introduces different projection-based methods and functionality of projection-based point cloud compression.
- **Chapter 4:** Discusses about the challenges of projection-based technique and proposes two methods for improving coding efficiency of the projection-based point cloud compression.
- **Chapter 5:** Analyzes the experimental results and related discussions.
- **Chapter 6:** Provides a conclusion of the proposed methods for improving the projection-based point cloud compression and discusses the potential future works in this domain.

1.2 List of Publications

The conducted research study in this thesis was done in a collaboration with Nokia Technologies and Tampere University of Technology (TUT). This document describes technical details of improvement methods for the Nokia Technologies' submission [3] to category two of the ISO/IEC JTC1/SC29/WG11 Moving Picture Experts Group (MPEG) in response of Call for Proposals (CfP) for 3D point cloud compression [2].
The following publications were the outcome of the work conducted during this thesis:

1. S. Schwarz, M. M. Hannuksela, V. Fakour-Sevom, and **N. Sheikhi-Pour**. "2D video coding of volumetric video data,"In IEEE Picture Coding Symposium, June 2018. San Francisco, California.
2. **N. Sheikhi-Pour**, S. Schwarz, V. Kumar Malamal Vadakital, and M. Gabbouj. "Efficient 2D Video Coding of Volumetric Video Data,"In European Workshop on Visual Information Processing (EUVIP), November 2018. Tampere, Finland.

2. BACKGROUND TO THE STUDY

In the last few years, many research studies have been carried out on point cloud compression and simultaneously quality metrics introduced to measure the distortion in compression process. This chapter provides a brief background study for point cloud compression. Section 2.1 provides a brief overview of early works on point cloud compression. Section 2.2 describes the projection-based compression method for point cloud compression and its challenges. Later, section 2.3 provides quality metrics for point cloud compression.

2.1 Related Work

Several compression solutions for dynamic point cloud data have been proposed in the literature [1,6–8], but many of these solutions suffer from poor spatial and temporal compression performance.

For dynamic 3D data, since both geometry and corresponding attributes may change independently, the motion estimation operation becomes an ill-defined problem [9]. For instance, temporal successive frames do not necessarily have the same number of polygons, points or voxels. Thus, it is difficult to determine spatial and temporal redundancies for intra- and inter-frame compression.

The compression approach which is implemented in [1] do not use spatial prediction in an object. One of the early works on motion estimation coding is introduced in [10]. Utilizing multiview + depth approach for point cloud compression (PCC) is suggested in [11] which shows better results in terms of temporal and spatial compression efficiency. However, the main issue regarding this solution is that it does not cover full scene or object.

In [12], the approach is based on kd-tree structure, where the 3D space is divided recursively, and points in each cell are encoded. For each division, empty cubes are not divided further. A similar approach was implemented in [13] and [14] by applying octrees. By considering the connection between points, these methods achieve better results in terms of compression performance.

In addition to geometry compression, there are many methods which investigate attribute compression. For instance, in [15] an octree structure is implemented and

attributes are treated as signals. In each leaf of the octree, a graph is constructed. The Karhunen-Loeve transform (KLT) is applied to decorrelate color attributes on the graph and compress them.
A novel method for compressing both object and environment data by assisting three-dimensional Discrete Cosine Transform (3D-DCT) is proposed in [16].
The above-mentioned approaches focus on static point cloud compression. Regarding dynamic point cloud compression, in [6] the geometry compression considers the octree structure for each frame of a point cloud sequence. Due to the lack of information regarding point-to-point correspondence in a sequence, motion estimation for compression is challenging. In [1], a hybrid architecture for progressive point cloud compression was proposed, combining an octree-based structure with a common image coding framework. This method was also selected as reference for the ISO/IEC JTC1/SC29/WG11 (MPEG) Call for Proposals (CfP) for point cloud compression technology [2].
In [3, 4], a Projection-based method for efficient compression of point cloud data is proposed, in which the 3D data are projected onto a 2D image plane and coded by making use of the current 2D video coding technology. Spatial and temporal compression efficiency is highly improved over the state-of-the-art [1] for point cloud compression.

2.2 Overview of 2D Video Coding of Point Cloud Data

As was mentioned in the Section 2.1, the projection-based method in [3,4] proposed for efficiently compressing point cloud data. The proposed approach in this work, projects the 3D video data into 2D video and making use of currently available 2D video compression to compress it and then reconstructs the 3D data based on stored metadata. The main advantage of the proposed method is its compatibility with current 2D video coding standards. As 2D video coding standards, such as High Efficiency Video Coding (H.265/HEVC [17]), are already supported by billions of devices and distribution solutions, thus it is possible to integrate this solution in the products and services.
In addition, this method provides remarkable bitrate reduction compared to the reference technology [1], in terms of both objective and subjective quality. Bitrate requirements are reduced by around 75% for geometry and approximately 50% for color attribute over the state-of-the-art compression technology. Out of the mentioned bitrate reductions in [3], almost one third is achieved by the proposed algorithms in this thesis work.
As mentioned before, the proposed method in [3] can take advantage of utilizing video coding standards which are available for 2D video coding purposes, for instance

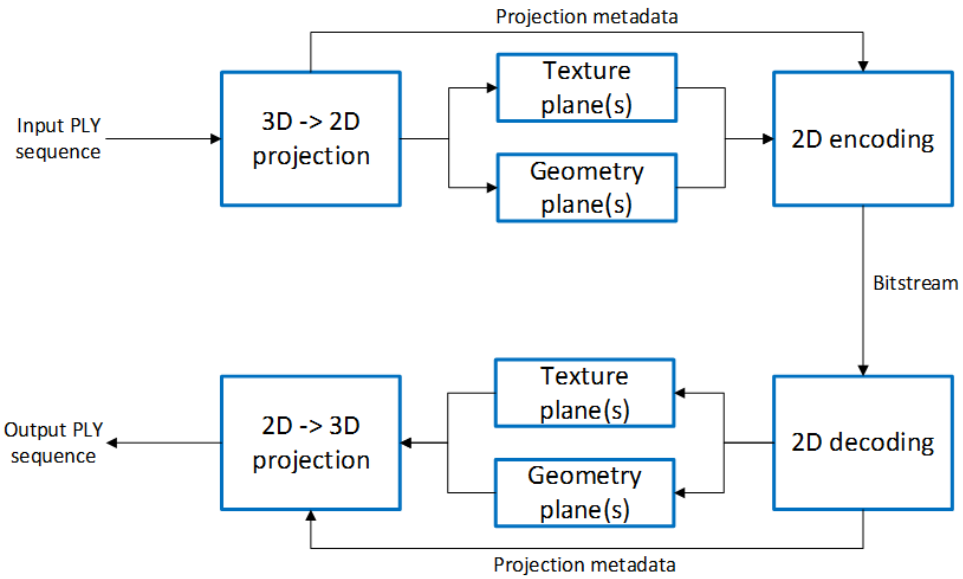


Figure 2.1 *The overall process of projection-based volumetric video coding.*

H.265/HEVC. The 2D video coding which is used for projection-based point cloud compression is scalable extension of HEVC (SHVC) [18] reference software (SHM) version 12.2 [19], in which one texture plane as a base layer and one geometry plane as an enhancement layer is used. It is also feasible to add more enhancement layers or use different video coding standard like 3D High Efficiency Video Coding (3D-HEVC [20]). Since neither of these compression standards are designed for compressing the point cloud data, it is essential to make them compatible for such content. Figure 2.1 illustrates block diagram of overall process of the projection-based point cloud compression. In below, a brief description of the block diagrams that are used in the projection-based technique is provided:

- **3D to 2D projection:** Projecting each individual point cloud of a sequence onto the 2D geometry. One 2D plane is allocated for texture projections and one for geometry.
- **2D encoding:** The 2D planes (geometry and texture) are encoded with the current standard 2D video codecs (e.g HEVC [17])
- **2D decoding:** The encoded 2D planes are decoded with the current standard 2D video codecs.
- **3D to 2D projection:** The PLY can be reconstructed by using decoded planes. In back-projection process, texture plane is used for color intensity value of point and the position of point is determined by corresponding geometry value.

(a) (b) (c)

Figure 2.2 *(a) Original point cloud (b) decoded point cloud at 13 Mbit/s for proposed solution, and (c) reference technology [1]*

Figure 2.2 illustrates subjective performance of the projection-based method compared to the reference technology [1]. As can be seen, this method outperforms the reference technology for compressing the point could data, significantly.

2.3 Objective Evaluation Criteria and Metrics

Different quality metrics have been considered for evaluating the quality of the compressed point cloud data for geometry and color components. For this purpose, two different metrics were calculated for geometry component. The first metric is a point-to-point quality assessment (referred to as D1 metric) which computes the mean square error (MSE) between the reconstructed point and its closest point in the original point cloud. The second metric assesses the point to the plane error (referred to as D2 metric). D2 calculates the MSE between the reconstructed point and the original point cloud surface [21]. The color distortion is computed on a

point-to-point level in YUV domain.

2.3.1 Geometric Distortions

This section describes briefly the objective quality assessment metrics that are used for evaluating the quality of compressed point cloud data.
Lets assume **A** as an original point cloud and **B** as a decoded compressed point cloud. In this case, **A** is the reference for computing the quality of point cloud **B** which is a degraded version of **A** and consist of N points. Where the number of points in original and degraded one are not necessarily equal.
Two-pass computation is performed to evaluate the compression error. Therefore, once compression error which is the distortion of point cloud **B** relative to the reference point cloud **A** is calculated and denoted by $e_{A,B}$. In second pass, the reconstructed point cloud **B** serves as reference and compression error is denoted as $e_{B,A}$. Finally, the worst metric between $e_{A,B}$ and $e_{B,A}$ is selected for final measurement called symmetric compression error [21].
The nearest neighbor (NN) method is used to identify the correspondence between in **B** and **A**. For the sake of computational complexity reduction, KD-tree search is utilized in this process. If points in **B** are denoted by b_j, the corresponding point in **A** is identified by KD-tree to perform nearest neighbor search and denoted by a_j.
In Figure 2.3, the black point is collected from point cloud **B** and the red point is the corresponding point of b_j in the reference point cloud **A**.

Point-to-point Quality Assessment (D1 metric)

After identifying the point of **B** in the uncompressed point cloud **A**, an error vector E(i,j) is calculated by connecting the point in **A** to the corresponding point in **B**. The length of error vector E(i,j) is called point-to-point error and is calculated according to below equation:

$$e_{A,B}^{D1}(i) = ||E(i,j)||_2^2 \tag{2.1}$$

Based on the definition of point-to-point distance for one point in (2.1), the calculation of point-to-point error (D1) for all point (i.e., N_B points) can be done by equation (2.2):

$$e_{A,B}^{D1} = \frac{1}{N_B} \sum_{\forall b_i \in B} e_{A,B}^{D1}(i) \tag{2.2}$$

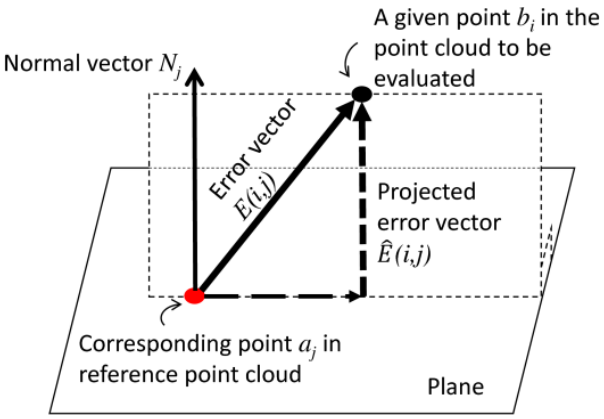


***Figure** 2.3 Illustration of point-to-point distance (D1) and point-to-plane distance (D2) [2]*

Point-to-plane Quality Assessment (D2 metric)

The point-to-plane distances rely on the normal vector for each point. Projection of the error vector E(i,j) along the normal direction N_j result in new error vector $\hat{E}(i,j)$ which is parallel with the normal attributes.
D2 calculates the Mean squared error (MSE) between a reconstructed point and the original point cloud surface [21]. The point-to-plane error is computed as:

$$e^{D2}_{A,B} = ||E(\hat{i}, j)||^2_2 = (E(i,j).N_j)^2 \tag{2.3}$$

Similarly, the point-to-plane error (D2) for all of points is calculated as below equation:

$$e^{D2}_{A,B} = \frac{1}{N_B} \sum_{\forall b_i \in B} e^{D2}_{A,B}(i) \tag{2.4}$$

Geometry PSNR Calculation

The PSNR value for geometry attribute is computed as:

$$PSNR = 10 \log_{10}(\frac{3p^2}{MSE}) \tag{2.5}$$

For each reference point cloud as specified p is defined in Table 5.1 as the peak constant value. And MSE is the mean squared error of one of the errors (point-to-point (D1) or point-to-plane (D2)) which introduced before.

2.3.2 Attribute Distortions

The attribute PSNR is calculated by using following equation:

$$PSNR = 10 \log_{10}(\frac{p^2}{MSE}) \tag{2.6}$$

For color attributes, the color of the original point cloud is compared to the most nearby color in the reconstructed point cloud and calculate PSNR. The MSE for each of the three color components is calculated in YUV domain. Color distortion is performed in YUV color space, since YUV color space is closer to the human visual system (HVS) [2].
A symmetric computation of the distortion is calculated, with the same method which has been done for geometric distortions. The maximum distortion between the two execution is selected as the final distortion. For PSNR calculation of color components, the peak value p in (2.6) is set to 255, hence the bit depth for test data is 8 bits per point.

3. PROJECTION BASED POINT CLOUD COMPRESSION

This chapter studies different 2D projection methods for point cloud compression purposes. The suitable projection is selected based on a certain criteria which are described in section 3.1. The tested projections are described in section 3.2.

3.1 Introduction

There are variety of 3D to 2D projections and it is hardly possible to introduce one projection as the best projection for mapping, since it is highly dependent on the target application.
This work is a response to Call for Proposals (CfP) Category two of the ISO/IEC JTC1/SC29/WG11 (MPEG) for Point Cloud Compression [2], in which the test sets are single 3D model. Thus, the approach and evaluation are provided in this context. Category two consists of scanning real moving people in different time instances. The target projection would cover the exterior part of the model.
The process of Figure 3.1 is used in order to select the best projection format. To that end, the projection format that leads to minimum objective and subjective quality degradation can be selected as the proper projection format for projection-based compression purposes.
In this process, the compression step is removed in order to determine the effect

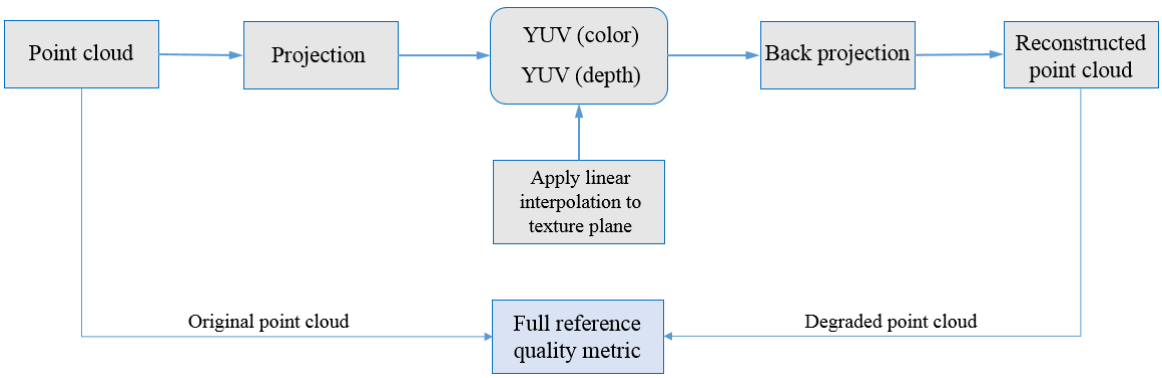
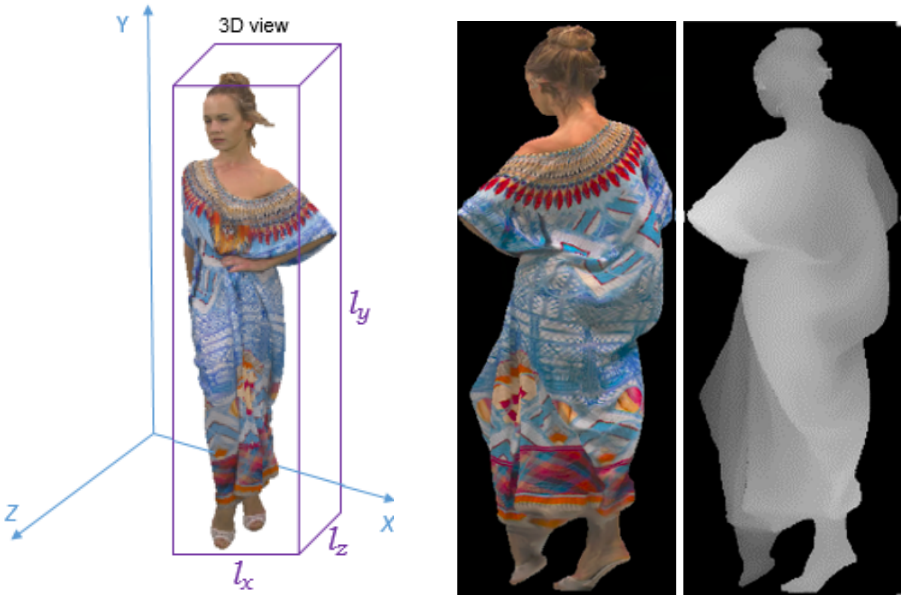


Figure 3.1 *Schematic overview of testing projections*



(a) (b)

Figure 3.2 *First frame of Longdress point cloud data. (a) Bounding box for point cloud, (b) projection of the point cloud on 2D and assigning one image for texture and one for geometry*

of projection itself. For that, the point cloud data is projected onto two 2D image planes, one for texture and one for geometry (or depth). Then, the projected data is back-projected to the point cloud format and compared to the original point cloud by using objective and subjective evaluations.
The desirable projection consists of below features in order to be used for projection-based compression purposes:

- Provides better coverage on the points
- Handles the occlusions in the reconstructed point cloud properly
- Provides high Peak Signal-to-Noise Ratio (PSNR)

3.2 Projection

Here, projection is defined as a transformation which performs translation from 3D Cartesian coordinate to 2D coordinate. Different approaches for projection have been proposed and tested. In the following all the projections which are examined for this study are presented.
Before performing any projection, in order to project samples of point cloud in 2D

space, it is essential to analyze and initialize basic parameters. One of the most important one is bounding box which is based on the attribute (minimum and maximum) of the point cloud. In geometry, the bounding box is the smallest or minimum box which encloses all sets of points in N dimension where in the 3D space, the bounding volume is a cuboid shape. Figure 3.2(a) shows a bounding box for *Longdress* point cloud in which all samples are surrounded by bounding volume where l_x, l_y, l_z are the length of each side of the bounding box. Hence, the edges of th bounding box are derived from the following equations:

$$l_x = max_x - min_x \tag{3.1a}$$

$$l_y = max_y - min_y \tag{3.1b}$$

$$l_z = max_z - min_z \tag{3.1c}$$

Therefore the length of space diagonal is calculated by:

$$d = \sqrt{l_x^2 + l_y^2 + l_z^2} \tag{3.2}$$

Another substantial parameter is primary axis which is defined as longest axis of bounding box. In all testset of this study Y is the primary axis, since between the value ranges of X, Y, and Z, with Y>X>Z order, Y-axis has wider ranges of values, hence it is selected as primary axis and X as an secondary axis.
And the position of 3D points after projection in 2D space is derived from :

$$position_x = \lfloor \frac{x - x_{min}}{x_{max} - x_{min}} \times l_x \rfloor + 1 \tag{3.3a}$$

$$position_y = \lfloor \frac{y - y_{min}}{y_{max} - y_{min}} \times l_y \rfloor + 1 \tag{3.3b}$$

$$position_z = \lfloor \frac{z - z_{min}}{z_{max} - z_{min}} \times l_z \rfloor + 1 \tag{3.3c}$$

In which, length of cube edges is calculated by subtracting the minimum value of coordinate from maximum value and it should be multiple of 8 in order to make it align with the minimum coding unit (CU) size of the HEVC standard [17]. consequently the size of image projection is determined by the 3D bounding box.
A 3D video object, e.g represented as a dynamic sequence of point clouds, is projected onto simple geometries. For each projection process two 2D image planes are allocated, one for color attribute (i.e., the texture image) and one for 3D depth (i.e., the geometry image).
Assuming the 3D point p is located at [x, y, z] and the corresponding RGB attribute [R_p, G_p, B_p] on texture image, the 2D projection of this 3D point on texture image is shows by T and geometry image is shown by D.

$$T(x,y) = \begin{bmatrix} R_p \\ G_p \\ B_p \end{bmatrix} \tag{3.4a}$$

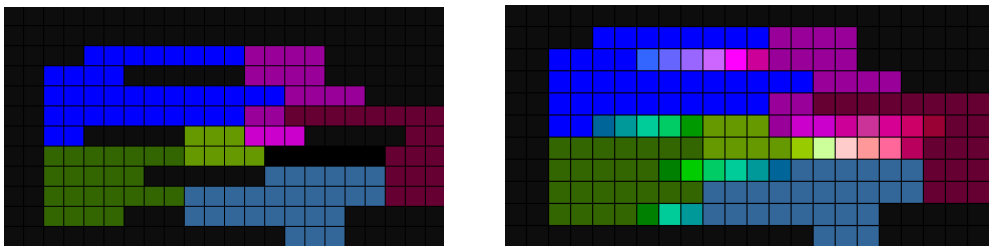
$$D(x,y) = z \tag{3.4b}$$

Where the resolution of D is x×y pixels and it stores the distance of each point from the projection plane. All distance information for each plane is stores in one 2D grid called geometry image. Figure 3.2(b) shows the texture and geometry images for one frame of *Longdress* point cloud projected onto plane projection.
Assuming a cluster of points, the decision for keeping a point is made based on the distance of a point from the plane. Which means the points that are closer to the projection plane are prior to preserve in 2D grid. Therefore, since there is a possibility to map more than one point from 3D space onto the same 2D coordinate, depth buffering is applied to handle this problem. Z-buffer is used to compare the distance to the plane of the occupied coordinate and new point. In other words, the point which has the smaller distance with plane will be stored in depth buffer to make sure that only visible points are mapped. Consequently, its color attribute is stored it in separate 2D grid called texture image. The buffer removes the far point in the case of having same coordinate in 2D grid and replaces it with the closest one. Information about the lost points of a projection is also reported as an evaluation factor to compare the projection methods.

Inpainting

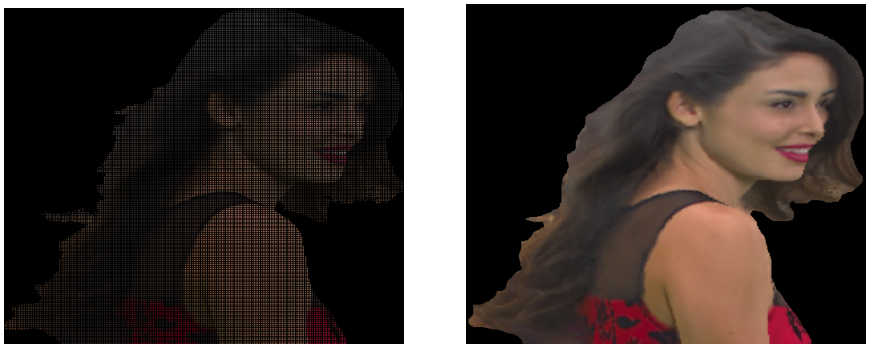
After projecting 3D model to 2D images, due to the possible sparsity of the point cloud data, the 2D projections may contain points without any value assigned. Such points create inefficiencies in the compression process [3]. In order to solve this issue, we use inpainting technique for calculating the missing pixel values between known-value pixels. In the texture plane, these null-value pixels are interpolated in the horizontal direction by using a linear interpolation method. By using such inpainting operation, the missing values will be assigned to a particular color value based on their neighboring color information.
The unknown values can be calculated by the linear interpolation equation in below:

$$f(n+i) = \frac{(D-i) \times f(n) + i \times f(n+D)}{D}, \quad 0 < i < D \tag{3.5}$$



(a) (b)

Figure 3.3 *(a) Sparse projection of the model (b) Inpainted sparse projection by linear interpolation (the colors are exaggerated for illustrative purposes).*



(a) (b)

Figure 3.4 *(a) Sparse projection of model (b) Inpainted by using linear interpolation*

In which, *f(n)* and *f(n+D)* are known values and *D* is the distance between two known values and *i* indicates which unknown pixel is interpolated. By using such inpainting operation, the missing values will be assigned to a particular color value based on their neighboring color information. The higher weights are assigned to closer color intensity value which are unknown.

The inpainting process is only applied to the texture plane and not to the geometry plane, as the geometry plane is essential for the reconstructing the 3D volumetric data. Only pixels with a valid geometry value are reprojected back into 3D space.

The occupancy mask, which is derived from geometry plane, is a binary mask in which occupied pixels are assigned to one, and the empty pixels are set to zero, so only pixels which their corresponding occupancy mask value is equal to one are projected back to the 3D space. If inpainting technique is also applied on the geometry plane, the reconstructed 3D data would include many invalid 3D points, unless generating occupancy mask before interpolating geometry plane.

Figure 3.3(b) depicts the result after applying linear interpolation in horizontal direction and figure 3.4(a) shows the texture plane of *Redandblack* point cloud content. As it can be seen, some null-valued pixels in texture plane are shown as black. The interpolated values will not be reconstructed in the back-projection from 2D to

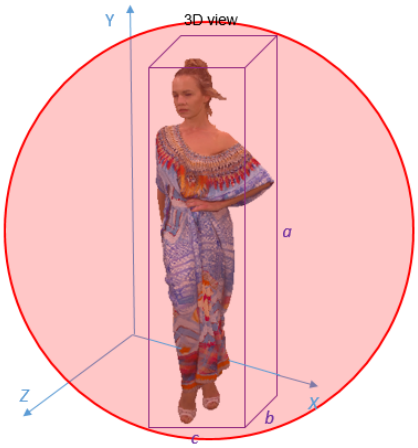


Figure 3.5 *Drawing sphere projection plane around object based on the size of bounding box*

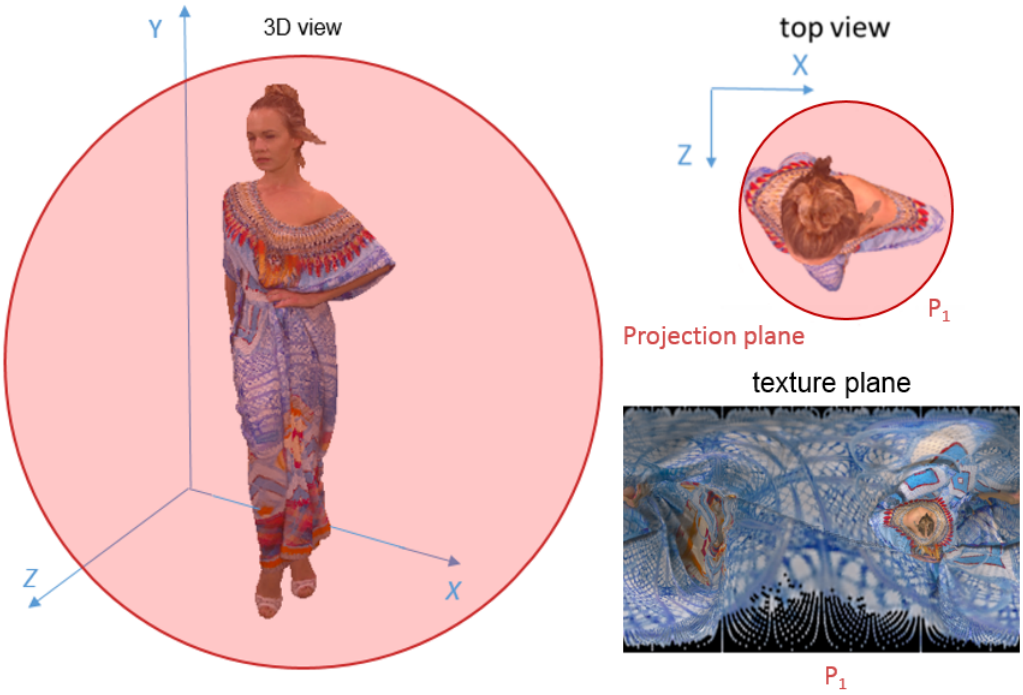
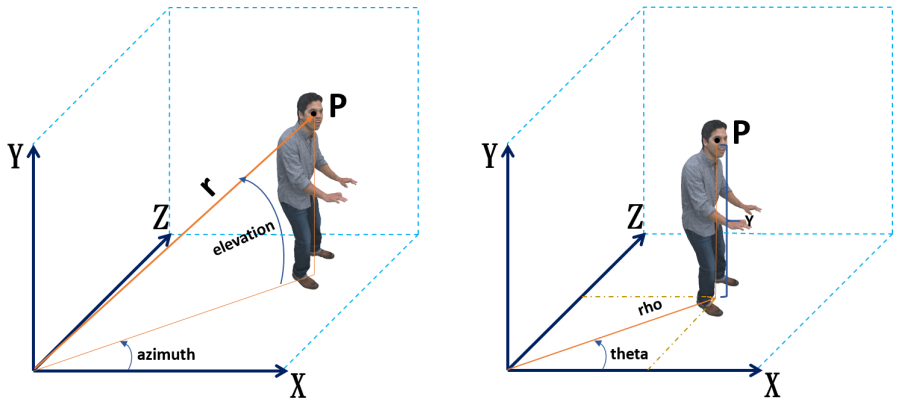


Figure 3.6 *Three-dimensional to two-dimensional projection of Longdress sequence onto sphere*

3D, since the occupancy map for those interpolated values are zero. Figure 3.4(b) illustrates the inpanited version of the content in which the null-valued pixels are interpolated and the projected image contains smoother samples.



(a) (b)

Figure 3.7 *Transforming Cartesian coordinates to (a) spherical (b) cylindrical coordinates*

3.2.1 Planar Projections

Sphere

This section describes the projection to sphere for the projection-based compression scheme. This transformation turns the location of points in Cartesian space to spherical coordinates. In this projection, r_s is maximum radius value between a point and origin. Therefore, for a point which is located in $(a,\ b,\ c)$ and gains the highest distance with the center, r_s is a radius of sphere which passes through $(a,\ b,\ c)$. Hence, in order to project this point to sphere, the radius of projection should be enlarged which is denoted by R_s in 3.6b.
The radius of projection and $(X_s,\ Y_s,\ Z_s)$ coordinates are calculated using equations of (3.6b)-(3.6e).

$$r_s = \sqrt{a^2 + b^2 + c^2} \tag{3.6a}$$

$$R_s = r_s \times 1.2 \tag{3.6b}$$

$$X_s = \arccos(\frac{x}{r_s}) \times R_s \tag{3.6c}$$

$$Y_s = \arccos(\frac{y}{r_s}) \times R_s \tag{3.6d}$$

$$Z_s = \arccos(\frac{z}{r_s}) \times R_s \tag{3.6e}$$

(a) (b) (c) (d)

Figure 3.8 *(a) Original Egyption point cloud (b) reconstructed point cloud in sphere projection (c) Original Longdress point cloud, and (d) reconstructed point cloud in sphere projection*

The conversion from three-dimensional Cartesian coordinates into azimuth and elevation for equirectangular mapping is derived from the following equations:

$$h_s = \sqrt{X_s^2 + Y_s^2} \tag{3.7a}$$

$$azimuth = \text{atan2}(Z_s, X_s) \tag{3.7b}$$

$$elevation = \text{atan2}(Y_s, h_s) \tag{3.7c}$$

Where both *azimuth* and *elevation* are in radian. Azimuth is the counter-clockwise angle in the x-y plane from the positive x-axis as shown in Figure 3.7(a). The elevation is an angle between the radius and x-y plane. In Figure 3.7(a), in order to project an arbitrary point in 3D model into 2D, the pixel coordinates for the equirectangular grid with the size of $W \times H$ is calculated by:

$$x_{pos} = \lfloor \frac{azimuth + \pi}{\frac{2\pi}{W}} \rfloor + 1 \tag{3.8a}$$

$$y_{pos} = \lfloor \frac{elevation + \frac{\pi}{2}}{\frac{\pi}{H}} \rfloor + 1 \tag{3.8b}$$

In back-projection process, pixels with a valid geometry value are reprojected back into 3D space.
The problem with spherical projection is that it is suitable for round 3D models and does not preserve points in other shapes. The 2D projection plane contains only one pixel value per location, hence only points along the normal directions to the 2D plane would be projected onto the planes and other points will be lost during the projection process.
Figures 3.8(b) and 3.8(d) show the reconstructed result of two point clouds with spherical projection. As it can be seen, this projection is highly dependent on the overall shape of the input content and it is not able to preserve all points in the point cloud data in the projection and back-projection processes. In case of *Longdress* point cloud, as can be observed from Figure 3.8(d), the reconstructed point cloud compared to the *Egyption* in Figure 3.8(b), includes higher lost points in the projection and back-projection processes.
Figure 3.17 illustrates the subjective performance of all tested projections in this work according to the proposed evaluation scheme of Figure 3.1. Among that, Figure 3.17(b) presents the result of sphere projection, in which this projection suffers from huge amount of lost points in the reconstructed point cloud compared to other projections.
Table 3.1 provides the objective results for the sphere projection using the process that is described in Figure 3.1. As the table illustrates, the PSNR values of both geometry and color components of the reconstructed point cloud using this projection are significantly lower when compared to other projections in the table. Moreover, this projection results in very high number of lost points in the forward- and back-projection operations. This projection for the tested point cloud was able to preserve only around 24% of the original points and 76% of the points were lost.
Thus, considering both objective and subjective performances, this projection is not suitable for projection-based point cloud compression purposes.

Cylinder

The second type of examined projection is cylinder projection. Similar to sphere projection, it projects points into the corresponding cylindrical coordinates. In this projection, r_c is maximum radius of a circle perpendicular to primary axis (here is y-axis) which passes through the point which has maximum distance to the center. Therefore, similar to the sphere projection, the radius of cylinder is multiplied by a constant to embrace all points (equation 3.9b). Figure 3.7(b) depicts conversion of Cartesian coordinates to cylindrical. The corresponding cylindrical coordinates are

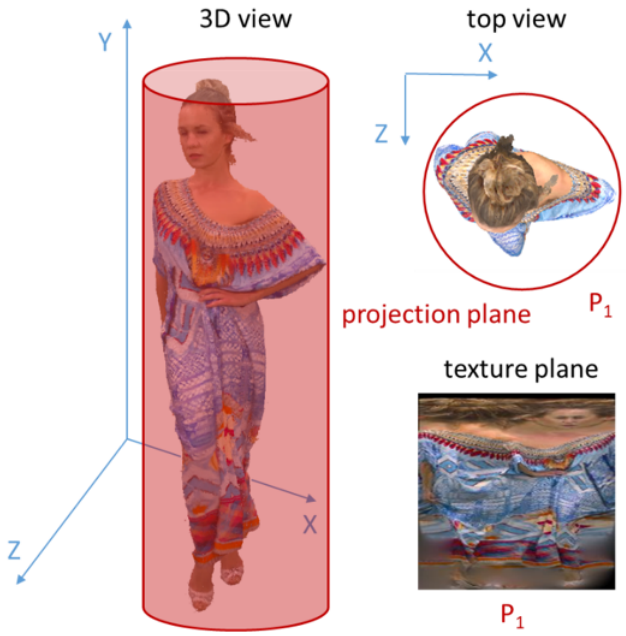


Figure 3.9 *3D to 2D projection of Longdress sequence onto cylinder*

derived from the following equations:

$$r_c = \sqrt{a^2 + c^2} \tag{3.9a}$$

$$R_c = r_c \times 1.2 \tag{3.9b}$$

$$X_c = \arccos(\frac{x}{r_c}) \times R_c \tag{3.9c}$$

$$Y_c = 1 \tag{3.9d}$$

$$Z_c = \arccos(\frac{z}{r_c}) \times R_c \tag{3.9e}$$

The conversion from Cartesian coordinates to cylindrical is done by following equations:

$$h_c = Y_c \tag{3.10a}$$

$$theta = atan2(X_c, Z_c) \tag{3.10b}$$

$$rho = \sqrt{X_c^2 + Z_c^2} \tag{3.10c}$$

In these equations, *theta* is in radian and, similar to *azimuth* in spherical projection, it is the counter-clockwise angle in the x-y plane from positive side of x-axis. The distnace between the projection of point in x-y plane to the origin is denoted by *rho* as depicted in Figure 3.7(b). Therefore, the position of arbitrary point in the 3D

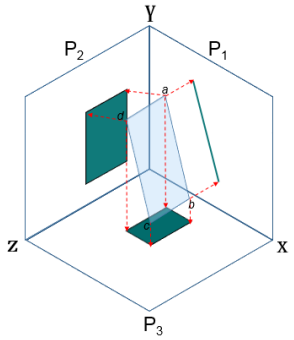


Figure 3.10 *Inclined plane can be seen differently from several angles of the cube*

model in 2D plane is calculated by:

$$x_{pos} = \lfloor \frac{theta + \pi}{2\pi} \times w \rfloor + 1 \tag{3.11a}$$

$$y_{pos} = \lfloor \frac{y - y_{min}}{\frac{y_{max} - y_{min}}{H}} \rfloor + 1 \tag{3.11b}$$

Where y_{min} and y_{max} are the minimum and maximum values of the point cloud, respectively, in y-axis. The projected point cloud data is later unfolded to a 2D image plane. Figure 3.9 depicts the first frame of *Longdress* point cloud surrounded by a cylinder and the unfolded result of the projection in P_1.
Even though table 3.1 indicates cylinder projection gained higher PSNR and lower number of lost point in comparison to sphere projection, however it is not able to preserve around 70% of the original points after reconstruction. This phenomenon is also visible in visual comparison in Figure 3.17(c), in which the reconstructed data contains a lot of holes in the content. Therefore, the cylinder projection is not considered as a suitable projection for point cloud compression.

Cube

This mapping is used for projecting points from 3D space onto 4 sides of the cube. In fact, the goal of this projection is to see the 3D object from 4 different perspectives of the cube. Figure 3.10 shows that a single object can be seen differently from each angle.
As mentioned in section 3.2, the Z-buffering process assists in keeping the closest point to each plane. Therefore, each side of the cube stores the points which their normals are perpendicular to the projection plane. Figure 3.10 shows the process of this projection onto cube faces. As can be seen, projection of the inclined plane onto P_1 is demonstrated as a line that is representative of 'ab' line which is near P_1.

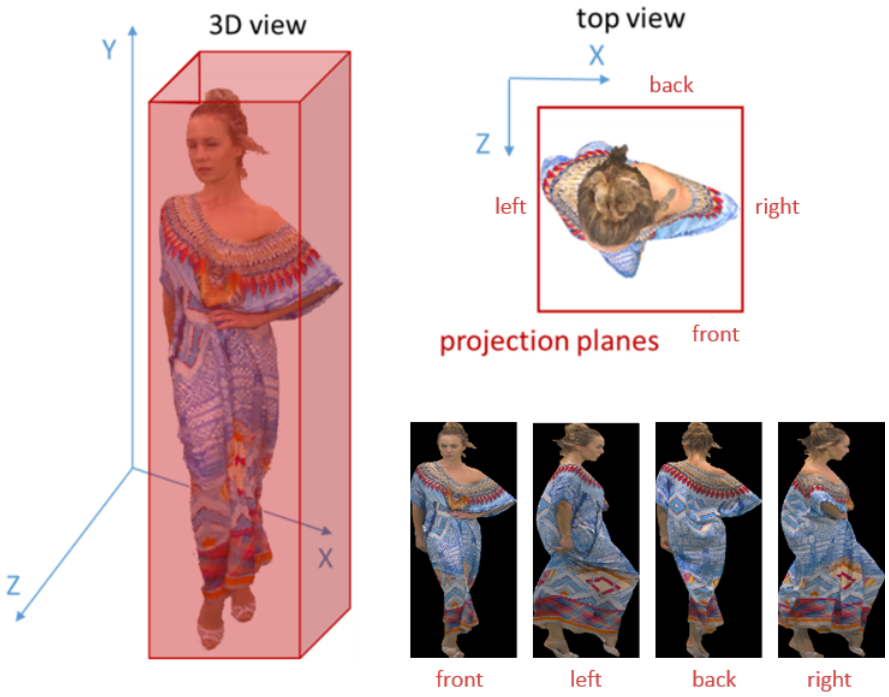


Figure 3.11 *3D to 2D projection of Longdress sequence onto cube*

Projection of two opposite sides of the plane is done simultaneously. Therefore, if a point has a chance to be projected onto one of these two planes, it will be projected to the one that is closer to the point.
According to the results in Table 3.1, the cube projection provides significantly better geometry and color PSNRs compared to sphere and cylinder projections. However, this projection is able to preserve only around 58% of the original points in the reconstructed point cloud and around 42% of the points are lost during the projection and back-projection operations. These lost points result in poor subjective performances after reconstruction. As shown in Figure 3.17(d), this projection provides better visual performance compared to sphere and cylinder. However, due to the occlusion, some concave areas are not covered properly. Hence, in order to cover more points another projection on top of cube projection is studied in the following sections.

3.2.2 Planar Rotation

Simple Planar Rotation

In order to handle occlusion and cover more points, the simple planar rotation method is studied in this section. The functionality of planar rotation is similar to cube projection. This extension adds more projection of surfaces at different rotations to

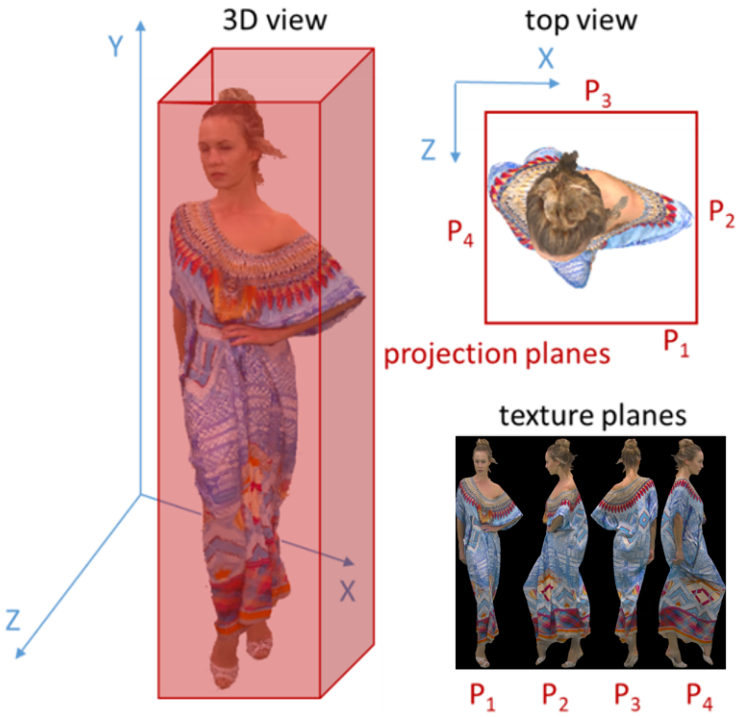


Figure 3.12 *3D to 2D mapping onto four rectangular planes with planar rotation projection*

cover the exterior side of 3D object more coherently. Yet, adding projections does not necessarily cover all the areas. In other words, this projection is a generic form of cube mapping since in this form of projection there is possibility to change the rotation between each plane and also increase the number of rotations in order to cover more points.

Figure 3.12 depicts one example projection with four planes and 90-degree rotation between each plane, where the starting plane is P_1 and it is aligned to X axis.

In Figure 3.13, an offset is applied after first four rotations in order to help better coverage of the 3D object and reducing occlusions. Thus, occlusions can be improved by increasing the number of projection as depicted. But the main objection to this method is the redundant projected points. As can be seen in Figure 3.13, there are significant similarities between the first four rotations and the last four projections which leads to higher bitrate. Therefore, this projection needs some developments in order to save bitrate.

Sequential Decimation

As illustrated in Figure 3.13, there is a considerable similarities between the first four rotations and the last four rotations, thus sequential decimation is studied in

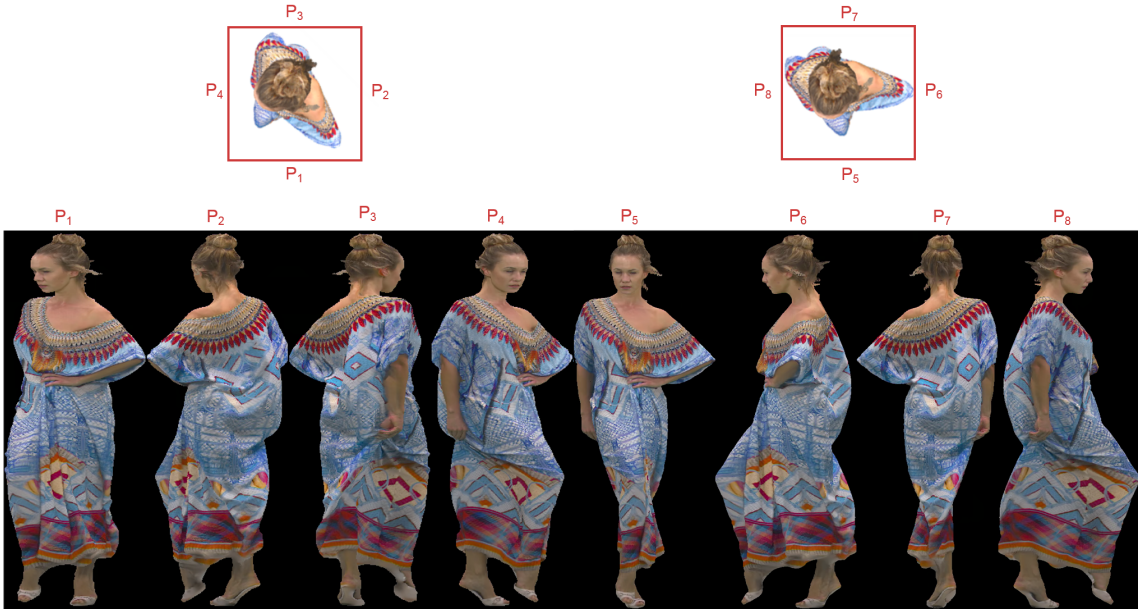
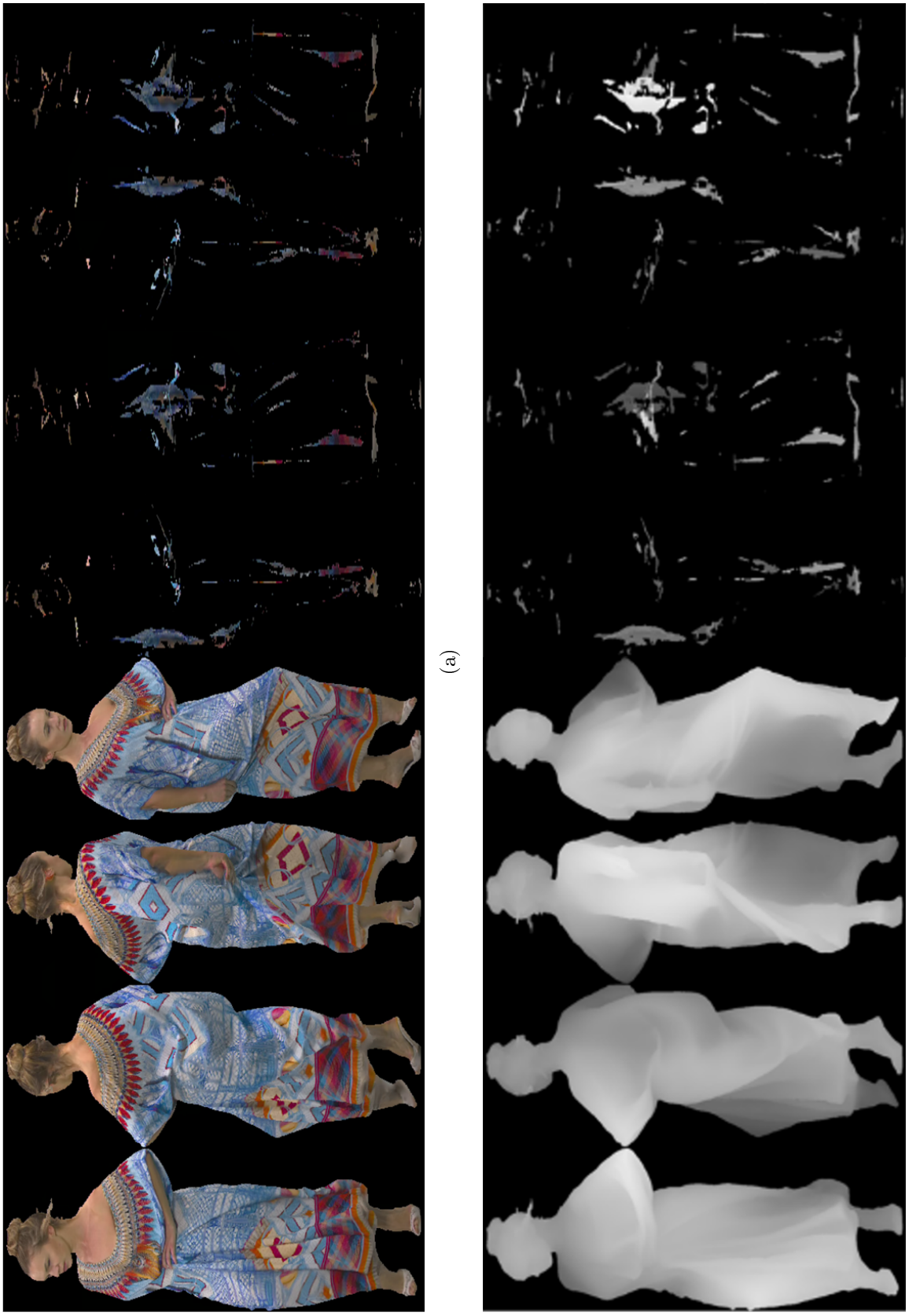


Figure 3.13 *Texture of projection planes for first frame of Longdress sequence, covered by 8 rotations with rotation setp-size 90-degree and 45-degree offset after 4 rotations.*

this section in order to alleviate such issues of planar rotation method.
Figure 3.14 depicts the concept of the sequential decimation method. As shown in the figure, points which are projected successfully removed from point cloud, therefore the last projection is more sparse in comparison to the first one. The number of planes, the angle of first plane to start, number of projection and angle between each rotation can be given in the encoding parameters.
Figure 3.15(a) represents an example of sequential decimation with 4 primary rotations and 4 additional rotations. As it can be observed, those points which are successfully projected in the first 4 rotations are removed, then the rotation plane is shifted by 45 degrees to cover points which do not have chance to project in the earlier projections. Table 3.1 reveals that the sequential decimation method gained remarkable higher PSNR and lower number of lost points in comparison to all examined projections. This method is able to preserve around 84% of the point cloud content in the reconstruction process which is significantly higher compared to other methods that are studied before.
Furthermore, in terms of subjective performance, as shown in Figure 3.17(e), the distortion of reconstructed point cloud is negligible and it is almost similar with the original point cloud.

Figure 3.14 *Effect of sequential decimation after each projection*



(a) (b)

Figure 3.15 *(a) Texture and (b) geometry projection images examples for Longdress sequence.*

Figure 3.16 *Sequential decimation improves the occlusion handling in some concave areas (left)*

Table 3.1 *Objective comparison of the 2D projection methods for point cloud compression*

Projection	Symmetric PSNR					Number of lost points	Number of final points	Original number of points
	D1	D2	Y	U	V			
Sphere	41.08	50.23	22.12	30.34	28.14	585282	180539	765821
Cylinder	56.16	63.90	25.96	35.04	33.23	536253	229568	765821
Cube	64.29	71.62	32.58	42.61	41.28	315868	449953	765821
Sequential decimation	69.43	75.44	31.24	32.94	34.82	116650	649171	765821

(a) (b) (c) (d) (e)

Figure 3.17 *Subjective comparison of studied projection methods for projection-based point cloud compression for (a) Original point cloud, (b) Sphere, (c) Cylinder, (d) Cube, (e) Sequential decimation*

4. PROPOSED METHODS

This chapter includes motivation of the research and implementation details of the proposed methods for improving 2D video coding of volumetric data.
Section 4.1 discusses the challenges ahead for the projection-based methods in intra- and inter-frame coding operations. Section 4.2 includes the proposed solutions for improving the performance of such approach.

4.1 Challenges in 2D Video Coding of Volumetric Data

As described in Section 3.2.2, the projection-based approach provides significant compression performance improvements over the state-of-the-art [1]. However, the projected of 2D data includes sharp boundaries which are not rectangular and therefore it is not aligned with the block partitioning that are used in the 2D video coding process. The sharp edges introduce crucial issues when using conventional block-based DCT video coding schemes [22]. In the object boundaries, some blocks may contain partial data from the projected point cloud content and partially from the image plane background. In de-correlation process of video compression standards (e.g., DCT, DST, etc.), sharp edges in the boundary areas of projected image are not coherent in terms of color intensity, because they contain information from the projected point cloud data and the background of the projection plane. And applying DCT/DST to such non-homogeneous content results in a lot of high-frequency components. In block-based video coding standards, some functionalities (e.g., zig-zag scan of DCT coefficients) are tuned in a way that the high-frequency components are less likely and/or with a smaller values than the low-frequency components, therefore, after de-correlation and quantization processes of blocks in texture plane, these non-homogeneous blocks generate many high-frequency components which leads to visible artifact, in particular ringing effect and requires more bitrate in the compression process [23].
Furthermore, these high-frequency components produce coding artifacts in the decoded point cloud contain flying points in the boundary areas of the point cloud data [3]. In order to alleviate some of the coding artifacts, a depth offset z_0 is defined

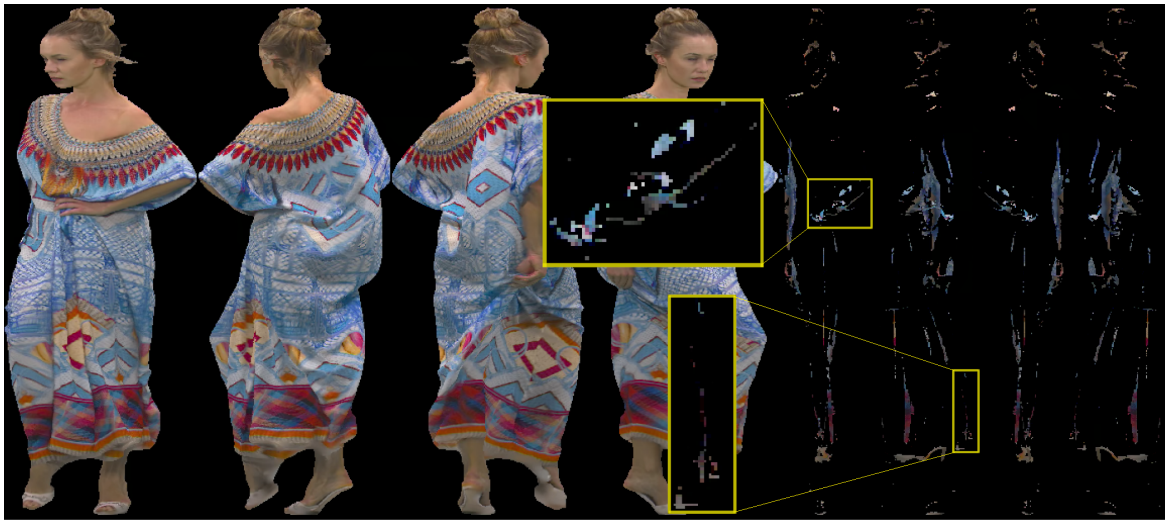


Figure 4.1 *Small patches which are generated in sequential decimation.*

to reduce te effect of coding artifacts. So the 3.4b is changed to:

$$D(x, y) = z + z_0 \tag{4.1}$$

In addition, sparsity of the point cloud data and having small patches in 2D plane leads to inefficiencies in the compression process. As mentioned in Section 3.2 some of values in 2D color plane can be linearly interpolated if the missing points are a few. But as it can be seen in figure 4.1 the sequential decimation produces small patches which leads to the sparse data in the projection plane. And since the mentioned interpolation is applied the inner part of the patches, the interpolation is not sufficient and other solutions must be considered as well. This chapter describes methods in order to resolve the above-mentioned issues of projection-based point cloud compression and improves the coding efficiency.

4.2 Proposed Methods

This section describes the details of proposed algorithms and how the applied methods in [3] helped the coding efficiency and how they alleviate the above-mentioned challenges of projection-based point cloud compression.

4.2.1 Padding

In order to resolve the issues which caused by the sharp edges in the projection boundaries, in this section some padding algorithms are proposed for providing smoother

Figure 4.2 *Applying simple padding to the sub-frame of projected patches edges.*

transition between the projected point cloud data and the background of the image plane. The performances are evaluated based on one frame for each point cloud sequence by using the evaluation method which has been described in Section 2.3. The BD-Rate results are reported in Table 4.1.
For the sake of simplicity, all the tested methods are applied to four rotation technique. The method which results in higher bitrate reduction is selected as the best method.

Simple Padding

With the purpose of achieving smoother boundaries for the projected patches, a simple padding method is studied in this section.
In this method, the border pixel values of the projected point cloud data in each rotation is extended (or padded) in horizontal direction to the entire sub-frame of that rotation. Figure 4.2 depicts the result of applying simple padding to the first frame of *Longdress* sequence. As can be seen, the padded samples improved the correlation of content particularly in the border areas of the projected point cloud. Results in Table 4.1 show that this simple method provides around 7% and 6% BD-Rate improvements in D1 and D2 quality metrics, respectively.

Replication of Patch Edges

This algorithm is follow almost the same principle as simple padding method, but in this experiment there is not any sub-frame for each patch. Therefore, the padding

Figure 4.3 *Applying the replication to the edges of projected patches.*

is applied from one known pixel value to the next known value.
Figure 4.3 shows how this padding affects the background data in the projection plane. The unknown pixel values in left side of the 2D image are replicated from the first pixel value of the edge in vertical axis, but for other values, pixel value of the patch replicated from left to reach the next known value.
In comparison to Figure 4.2, the background is more coherent and the effect of sharp edges in the border of each sub-frame has been disappeared. As can be seen from the results in Table 4.1, this method improved the compression performance. A negligible difference can be observed between this method and the simple padding method in the geometry attributes. However, in terms of color attributes, improvement the color edge extension provides better compression performance.

Interpolation Padding

As discussed in Section 3.2, it is feasible to estimate the color intensity of unknown pixels between two known values by interpolation algorithms. Similar to above-mentioned paddings, the left and right side of the 2D image can be replicated based on the edge pixel value. Other unknown values which are located between two known values can be interpolated based on the equation (3.5).
Figure 4.4 presents the result of applying the Interpolation Padding to the first frame of *Longdress* sequence in which the left and right side of the image is padded similar to aforementioned paddings.
The results in Table 4.1 illustrate that the Interpolation Padding method provides lower compression gain when it is compared to the previous methods for geometry

Figure 4.4 *Applying the Interpolation Padding to the first frame of Longdress sequence.*

attribute. However, this method reduces the bitrate significantly in color attributes.

Edge Smoothing

This section studies a smoothing method for handling the sharp edge issues of the projection-based point cloud compression method. In this method, prior to applying the Edge Smoothing technique, the unoccupied projection planes (i.e., geometry and texture) are initialized with zero values for each pixel. Then, the 3D data is projected onto the corresponding 2D planes. Moreover, the background samples are replaced with gray values. In order to increase the correlation of the background data and the projected point cloud.
Then, Edge Smoothing is applied to the projected point cloud boundaries. This technique includes applying averaging filters in horizontal and vertical directions successively.
For this purpose, the averaging filter of equation (4.2) is used. In this formula, N is the filter kernel size and $X_{i,j}$ is the pixel value at (i,j) location inside the filter kernel. In our experiments, the kernel size of 7 provided the best results.

$$P = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} X_{i,j}}{N^2}, \tag{4.2}$$

Figure 4.6 illustrates an example of applying the proposed Edge Smoothing filter in horizontal direction. In this figure, the blue part indicates a portion of the projected point cloud. The purple, pink and green kernels show the filter kernels that

Figure 4.5 *Applying the replication to the edges of projected patches.*

are applied in horizontal direction. The smoothing process stops at projected data boundaries. Moreover, it does not affect the pixel values inside the projection areas. By employing this algorithm, in both horizontal and vertical directions, the sharp intensity transitions in the boundaries between the point cloud content and background is reduced. And reducing this contrast, results in a coherent and smooth transition between the projected point cloud data and the image plane background. Therefore, the de-correlation process in the compression operation can take place properly and the bitrate would decrease significantly. Figure 4.5 depicts the result image after using Edge Smoothing along horizontal and vertical directions.

It must be noticed that, the proposed padding method is only applied to the texture plane not the geometry plane. The reason is that, if this filtering is applied to the geometry plane, the decoder will not be able to generate the original point cloud data and the decoded point cloud will contain a lot of unwanted data that are used for smoothing the boundaries of the projected data.

Table 4.1 reveals that the Edge Smoothing method shows significant improvement in terms of geometry and color attribute in comparison with other padding methods. therefore Edge Smoothing method is chosen for reducing sharp edges and making homogeneous 2D image and it will apply to sequential decimation which is main projection approach for 2D video coding of point cloud.

The result of applying the proposed Edge Smoothing method is illustrated in Figure 4.7. In this figure, the top figure shows the original projected point cloud data and the bottom one demonstrates the effect of Edge Smoothing operation. As can be observed, the proposed method decreased the contrast of the background data and point cloud data, particularly in the projection boundaries.

As the results in Table 4.1 demonstrate, the Edge Smoothing technique provide

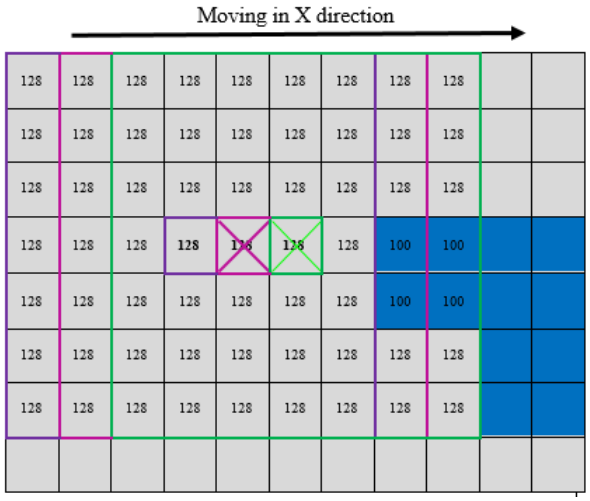


Figure 4.6 *Employing Edge Smoothing in horizontal direction.*

substantial BD-Rate performance in both geometry and color attributes compared to other methods. Hence, this method is selected as the proper one among the studied methods in order to handle the sharp edges of the projection-based point cloud compression method.

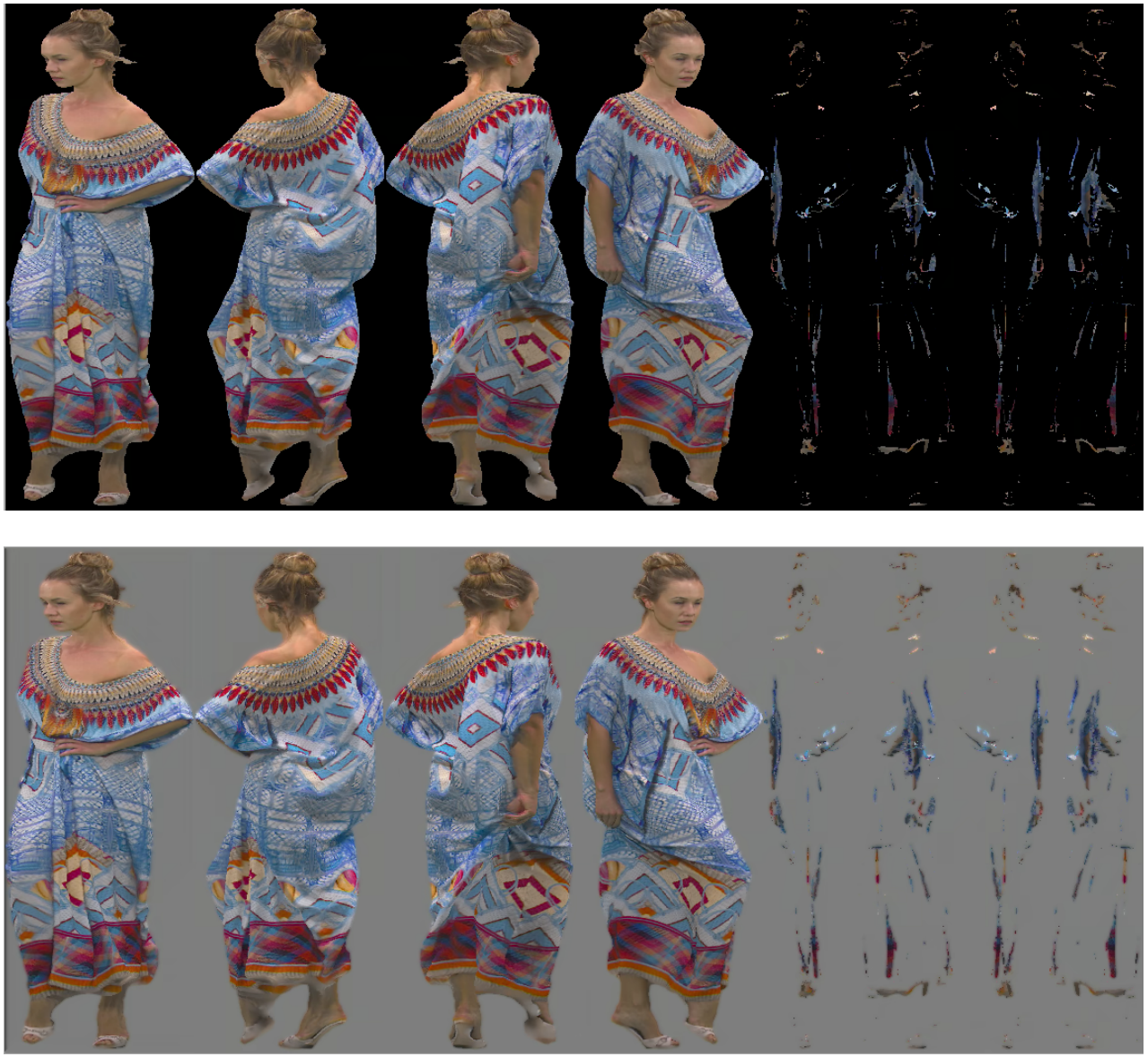


Figure 4.7 *Top) Texture plane for decimation rotation, Bottom) Edge Smoothing applied for projected 3D data*

Table 4.1 *BD-Rate (%) of applying different padding methods to four rotation projection approach for one frame with the same QP.*

Sequence	Simpe Paddign					Replication of Patch Edges					Interpolation Padding					Edge Smoothing				
	D1	D2	Y	U	V	D1	D2	Y	U	V	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-6.5%	-6.3%	-22.4%	-49.0%	-58.7%	-4.7%	-4.8%	-24.5%	-41.4%	-55.7%	-4.6%	-4.6%	-35.6%	-59.1%	-68.7%	-5.6%	-5.5%	-34.8%	-46.0%	-59.7%
Redandblack	-11.5%	-9.6%	-6.4%	-19.5%	-16.1%	-11.5%	-9.5%	-16.0%	-37.6%	-24.6%	-10.4%	-8.3%	-28.1%	-47.2%	-37.1%	-12.8%	-10.4%	-18.1%	-22.1%	-26.4%
Soldier	-9.1%	-7.5%	47.0%	-24.5%	-26.3%	-9.7%	-7.9%	-29.8%	-29.3%	-34.4%	-8.1%	-5.8%	-18.4%	-32.5%	-42.7%	-13.9%	-11.0%	-23.0%	-32.8%	-36.5%
Longdress	-1.3%	0.5%	-12.3%	-100%	-39.0%	-1.6%	-0.4%	-9.3%	-100.0%	-34.8%	-0.4%	1.7%	-10.5%	-98.3%	-45.3%	-6.1%	3.5%	-15.2%	-0.0%	-44.3%
Average	**-7.1%**	**-5.7%**	**1.5%**	**-48.3%**	**35.0%**	**-6.9%**	**-5.5%**	**-5.0%**	**-52.1%**	**-37.4%**	**-5.9%**	**-4.2%**	**23.2%**	**-59.3%**	**-48.4%**	**-9.6%**	**-7.6%**	**-22.8%**	**-25.2%**	**-41.7%**

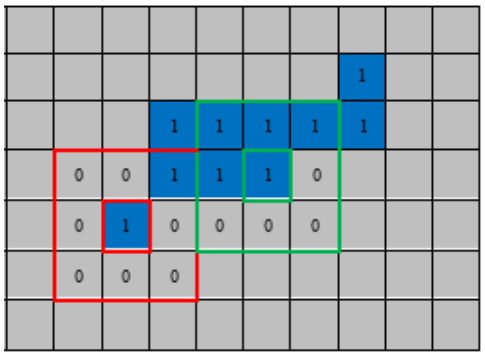


Figure 4.8 *An example of applying Patch Refinement.*

4.2.2 Patch Refinement

As it was mentioned in Section 3.2.2, the sequential decimation method provides additional projections to cover the points which are not initially projected to the projection plane. As a result, their projections have a sparse distribution, as it can be seen in Figure 4.7 (the last 4 projections). Such sparsity is not desirable in image/video coding algorithms and increases the bitrate significantly.

In order to decrease the effect of this sparse content, a method is studied in this section that uses a filtering operation for reducing the number of sparse content in a way that can improve the compression performance and has negligible impact on the quality of the decoded point cloud data.

For this purpose, after the interpolation and Edge Smoothing processes, a binary occupancy mask is extracted from the output texture plane for processing such small patches. The binary occupancy mask is a mask which indicates occupied pixels in texture plane.

The Patch Refinement technique includes a process of removing the samples in a certain area of the sparse data that have less connection to the neighboring samples than a per-defined threshold (i.e., based on number of samples). This process is illustrated in Figure 4.8.

As it can be observed from the figure, a 3×3 filter is selected for this analysis. The filter kernel is applied to the sparse sampling area and detects whether the central point of the kernel has connections to at least three samples inside the kernel. Otherwise, the value of the point will be replaced by the background value in both texture and geometry images. In another words, the more connection a certain point has with its neighborhood samples, the more probable to preserve the point.

In the example of Figure 4.8, the center point of the red kernel is connected to only one point, so this point will be removed. Whereas, for the green kernel the center value is connected to 4 neighbors, so no change will occur.

As a result of the re-filtering operation in Patch Refinement, the discontinuity of

the samples in the sparse area is reduced and consequently the required bitrate will decrease significantly.
Figure 4.9 illustrates the result of applying the Patch Refinement technique to the sparse data in the projection-based method (the last 4 projections). As can be seen from the figure (bottom figure), the Patch Refinement method reduced the sparsity of the last four projections. This effect is highlighted by red circles in the top figure which indicates the samples that are removed in the refinement process in bottom figure.

Figure 4.9 *Top) Texture plane for decimation rotation after applying Edge Smoothing, Bottom) Patch Refinement technique is applied for projected 3D data*

5. EXPERIMENTAL RESULTS

This chapter presents the experimental condition, results and analysis of the implemented methods which are proposed in chapter 4.
Results were provided for dynamic objects testset (category two). Moreover, the experiments were conducted for two coding configurations: 1) All Intra (AI) and, 2) Random Access (RA).

5.1 Point Cloud Data

To evaluate the performance of the proposed algorithms 4 point cloud sequences which are provided by 8i [24] are used for simulations. Table 5.1 shows a list of the 3D point cloud test material dataset and their specifications.

5.2 Implementation and Source Code

The projection-based point cloud compression have been implemented in C++ language on top of scalable extension of the HEVC standard (SHVC), SHM reference software version 12.2 [19], therefore all proposed methods have been implemented in the same layout. It has been compiled and tested on Windows, Linux and MacOS systems. Since the core of projection-based is 2D video coding, it can be integrated on top of any 2D standard video coding.

Table 5.1 *Video sequences of dynamic objects test Category two that are used in this experiment*

Test material dataset filename	Number of frames	Average points per frame	Geometry Precision	Attributes	Peak Value (p)
8i VFB Loot	300	782,000	10 bit	R,G,B	1023
8i VFB RedandBlack	300	700,000	10 bit	R,G,B	1023
8i VFB Soldier	300	1,500,000	10 bit	R,G,B	1023
8i VFB Longdress	300	800,000	10 bit	R,G,B	1023

5.3 Experimental Condition

The evaluation of 2D video coding of volumetric data was performed under MPEG Call for Proposals (CfP) for point cloud compression. For more details, please refer to the CfP document [2] and its corrigenda [25].
The performances were analyzed for five different bitrate targets in the range from 3 to 43 Mbit/s in All Intra and Random Access (RA) configurations.
The performances were analyzed based on the well-known Bjøntegaard Delta Bitrate (BDBR) criterion [26], in which the negative values indicate the bitrate reduction in the same peak signal-to-noise ratio (PSNR) quality. Similarly, the positive values represent how much the bitrate is increased in the same quality level.
In order to compare Point cloud Compression (PCC) solutions, in addition to above-mentioned evaluation criteria the compression rate is also considered. Therefore, BD-rate is is used for assessment.

5.4 Results

This section presents the results of implemented proposed algorithms that were described in Section 4 for 2D compression of point cloud data.
The results of applying Edge Smoothing method of Subsection 4.2.1 is described in subsection 5.4.1. The performance of Patch Refinement method in Section 4.2.2 is demonstrated in Subsection 5.4.2. Furthermore, the combined results of both methods are illustrated in subsection 5.4.3.
Finally, the complexity evaluations of the proposed methods are described in Subsection 5.5.

5.4.1 Applying Edge Smoothing Method

Tables 5.2 and 5.3 show the performances of applying the proposed Edge Smoothing method in the case of All Intra and Random Access configurations, respectively. As can be seen from the tables, the Edge Smoothing method provides significant and consistent bitrate reduction for all the test data regardless of the utilized configuration.
In the AI configuration, the proposed method improves the compression performance on average by around 13% for the geometry component by using D1 and D2 quality metrics. This performance is higher for the color components in which on average around 16%, 52% and 29% bitrate reductions were achieved for Y, U and V components, respectively.

Table 5.2 *BD-Rate (%) and BD-PSNR [dB] performances for Intra-frame and applying Edge Smoothing*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-11.7%	-12.4%	-15.8%	-45.0%	-40.3%	0.48	0.53	0.43	0.71	0.79
Redandblack	-9.8%	-9.3%	-13.2%	-36.7%	-17.1%	0.45	0.49	0.34	0.35	0.21
Soldier	-15.4%	-13.7%	-17.9%	-96.0%	-40.1%	0.68	0.65	0.43	0.28	0.44
Longdress	-16.7%	-14.5%	-17.6%	-31.5%	-19.2%	0.77	0.73	0.35	0.14	0.24
Average	**-13.4%**	**-12.5%**	**-16.1%**	**-52.3%**	**-29.2%**	**0.60**	**0.60**	**0.38**	**0.37**	**0.42**

Table 5.3 *BD-Rate (%) and BD-PSNR [dB] performances for Inter-frame and applying Edge Smoothing*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-18.4%	-18.2%	-18.8%	-35.1%	-39.0%	1.47	1.73	0.85	1.10	1.28
Redandblack	-9.1%	-9.1%	-15.3%	-22.7%	-14.6%	0.64	0.75	0.80	0.56	0.54
Soldier	-24.0%	-23.4%	-26.2%	-29.5%	-32.0%	1.32	1.67	1.00	0.68	0.99
Longdress	-19.3%	-19.2%	-22.3%	-24.8%	-21.8%	1.21	1.41	0.76	0.50	0.62
Average	**-17.7%**	**-17.5%**	**-20.6%**	**-28.0%**	**-26.9%**	**1.16**	**1.39**	**0.85**	**0.71**	**0.86**

For the RA scenario, the Edge Smoothing algorithm provided higher gains compared to AI case. More than 17% bitrate reduction is achieved for geometry component when using D1 and D2 metrics. The bitrate reductions of color components are around 20%, 28% and 27% for the Y, U and V components, respectively.
Another observation from the results is that, the proposed Edge Smoothing method performs better in sequences that have homogeneous texture in the content. For example, in *Longdress* and *Soldier*, the performance is significant for both AI and RA configurations. On the other hand, for the *Redandblack* point cloud, the bitrate reduction is lower when compared to other sequences due to the complex texture in this content.
Figure 5.1 shows the Rate-Distortion (RD) curves of the proposed Edge Smoothing method in the case of AI configuration for *Loot* sequence. As can be observed from the figures, the Edge Smoothing technique provide consistent improvements for quality metrics of D1 (Figure 5.1.a), D2 (Figure 5.1.b) and luma component (Figure 5.1.c).
The behavior is similar in the RA configuration in Figure 5.2. Moreover, in this case, the bitrate reduction is significantly higher in lower bitrates than higher bitrates.

5.4.2 Applying Patch Refinement Technique

The result of applying proposed Patch Refinement method in the case of AI and RA configurations are illustrated in Tables 5.4 and 5.5, respectively.
Similar to the Edge Smoothing results, the Patch Refinement method performs bet-

Table 5.4 *BD-Rate (%) and BD-PSNR [dB] performances for Intra-frame and applying Patch Refinement*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-3.6%	-2.4%	-3.2%	-2.5%	-3.0%	0.15	0.09	0.08	0.03	0.05
Redandblack	-3.3%	-2.2%	-2.4%	-2.1%	-1.9%	0.15	0.11	0.06	0.02	0.02
Soldier	-4.1%	-2.8%	-3.0%	81.0%	-2.6%	0.16	0.13	0.07	0.03	0.03
Longdress	-6.1%	-5.4%	-6.1%	-5.3%	-5.6%	0.27	0.27	0.12	0.02	0.07
Average	**-4.3%**	**-3.2%**	**-3.7%**	**17.8%**	**-3.3%**	**0.18**	**0.15**	**0.08**	**0.02**	**0.04**

Table 5.5 *BD-Rate (%) and BD-PSNR [dB] performances for Inter-frame and applying Patch Refinement*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-12.9%	-12.8%	-12.1%	-13.2%	-13.0%	1.11	1.29	0.58	0.31	0.34
Redandblack	-9.1%	-9.1%	-9.5%	-9.6%	-9.2%	0.63	0.74	0.49	0.21	0.32
Soldier	-10.4%	-10.4%	-10.1%	-9.6%	-9.5%	0.55	0.71	0.37	0.20	0.27
Longdress	-11.4%	-11.3%	-10.8%	-11.1%	-10.2%	0.67	0.77	0.35	0.19	0.27
Average	**-11.0%**	**-10.9%**	**-10.6%**	**-10.9%**	**-10.5%**	**0.74**	**0.88**	**0.45**	**0.23**	**0.30**

ter in the RA configuration than the AI case. In the case of AI configuration, one average of all sequences, around 3% - 4% bitrate reduction is achieved for geometry and color components.

Similar sequence-wise performance variation can be seen from the results. Where, the Patch Refinement method provided higher bitrate reduction in the sequences which uniform texture than the complex ones.

The proposed method reduced the bitrate by more than 10% on average for geometry and color components, when the Random Access configurations is used.

Figures 5.3 and 5.4 illustrate the RD-Curves of the proposed Patch Refinement method for *Loot* point cloud sequence. As can be seen, the proposed method provides better performance compared to the reference. The performance is higher in the case of random access configuration, where the Patch Refinement method reduces the bitrate consistently in all bitrate levels. However, this performance behavior is different in all intra configuration and the proposed method provides better performance in higher bitrates.

5.4.3 Applying Edge Smoothing and Patch Refinement Methods

Tables 5.6 and 5.7 illustrate the performance of applying both Edge Smoothing and Patch Refinement methods when these methods are used in AI and RA configurations, respectively.

Table 5.6 *BD-Rate (%) and BD-PSNR [dB] performances for Intra-frame and applying both Edge Smoothing and Patch Refinement*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-15.0%	-14.5%	-18.6%	-45.8%	-41.9%	0.65	0.64	0.52	0.75	0.85
Redandblack	-13.1%	-11.6%	-15.5%	-37.5%	-18.7%	0.63	0.63	0.40	0.37	0.24
Soldier	-18.4%	-16.0%	-20.0%	-92.9%	-40.7%	0.84	0.76	0.48	0.29	0.44
Longdress	-20.5%	-18.4%	-21.8%	-32.9%	-23.2%	0.99	0.98	0.45	0.16	0.30
Average	**-16.8%**	**-15.1%**	**-19.0%**	**-52.3%**	**-31.1%**	**0.78**	**0.75**	**0.46**	**0.39**	**0.45**

Table 5.7 *BD-Rate (%) and BD-PSNR [dB] performances for Inter-frame and applying both Edge Smoothing and Patch Refinement*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-29.5%	-29.3%	-29.2%	-43.9%	-47.4%	2.36	2.79	1.35	1.39	1.61
Redandblack	-17.5%	-17.5%	-23.3%	-29.8%	-22.5%	1.30	1.53	1.26	0.78	0.87
Soldier	-31.4%	-31.0%	-33.1%	-35.9%	-37.7%	1.79	2.30	1.31	0.86	1.21
Longdress	-27.6%	-27.5%	-29.6%	-32.0%	-28.9%	1.77	2.08	1.03	0.67	0.85
Average	**-26.5%**	**-26.3%**	**-28.8%**	**-35.4%**	**-34.1%**	**1.81**	**2.17**	**1.24**	**0.92**	**1.14**

As can be seen from the results, the combination of both methods provide significant bitrate reduction in both configurations.

In the AI test case, the combined methods reduced the required bitrate by around 16% for geometry component. For the color components, bitrate reductions of 19%, 52% and 31% achieved for Y, U and V components, respectively.

The performance is higher when these methods are used in RA configurations, where around 26% bitrate reduction was achieved for geometry and 29%, 35% and 34% improvements for YUV color components, respectively.

The advantage of proposed methods it that they do not negate the effect of each other, therefore the overall performance will boost by applying both.

As mentioned before, both proposed methods provide higher improvements when they are used in Random Access configuration than All Intra case. This is a welcomed performance behavior since the inter prediction is normally considered for real-world scenarios in point cloud compression.

The RD-Curves of this testing scenario is demonstrated in Figures 5.5 and 5.6 for AI and RA configurations, respectively. As these curves illustrate, the combination of the proposed methods provide significant bitrate reduction in both AI and RA configurations in all bitrate levels. In the case of RA configuration, the bitrate reduction is higher in the lower bitrates compared to higher bitrates.

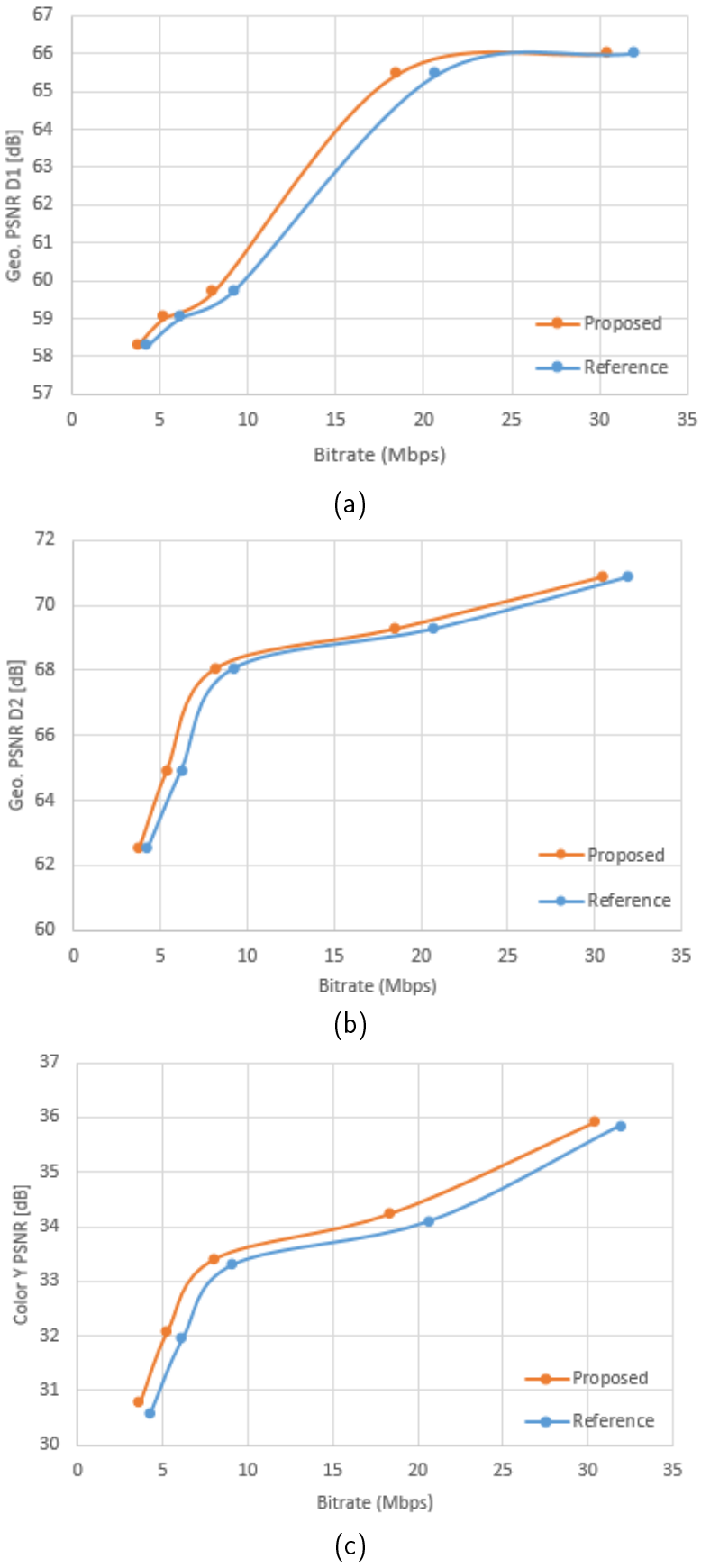


Figure 5.1 *Rate-Distortion curves for the Loot sequence compared to reference for AI configuration and applying Edge Smoothing. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

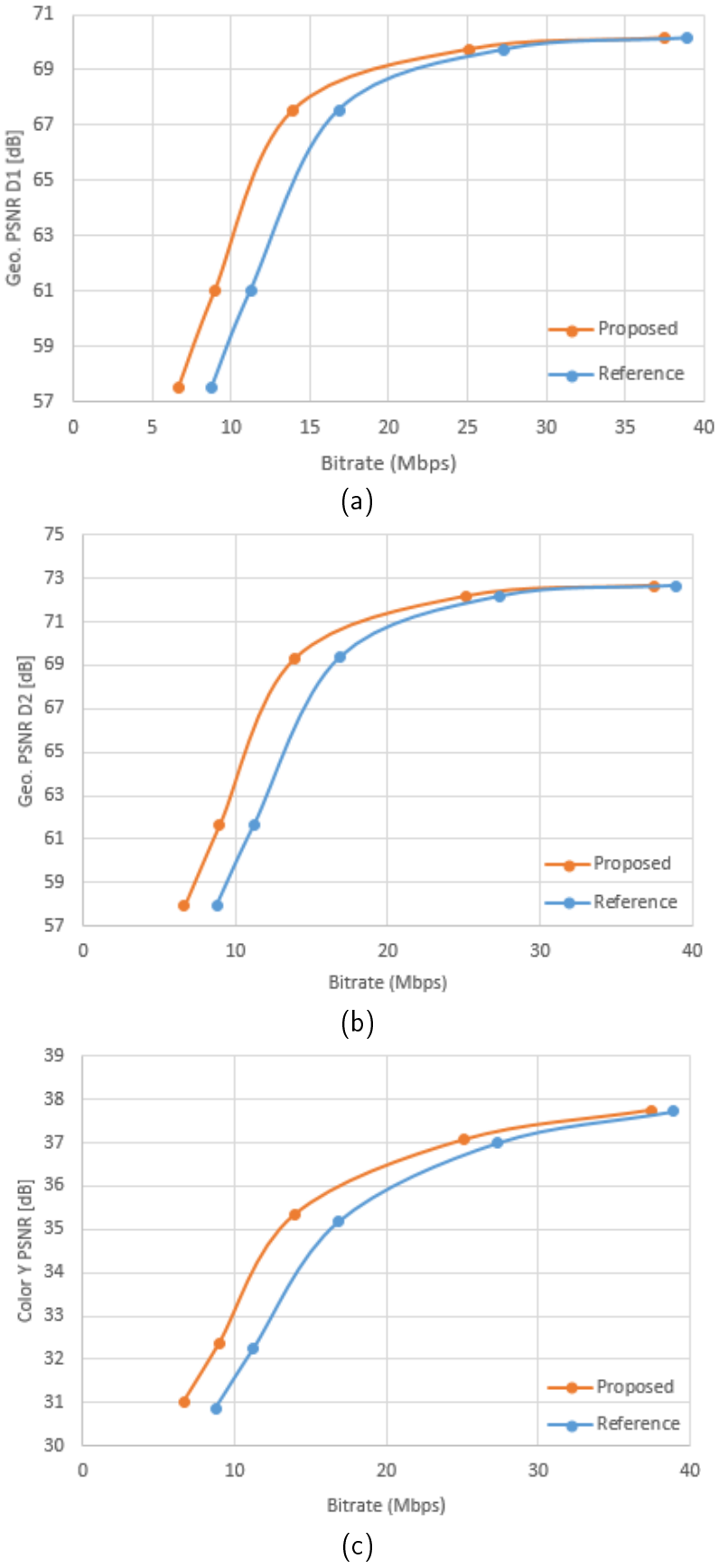


Figure 5.2 *Rate-Distortion curves for the Loot sequence compared to reference for RA configuration and applying Edge Smoothing. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

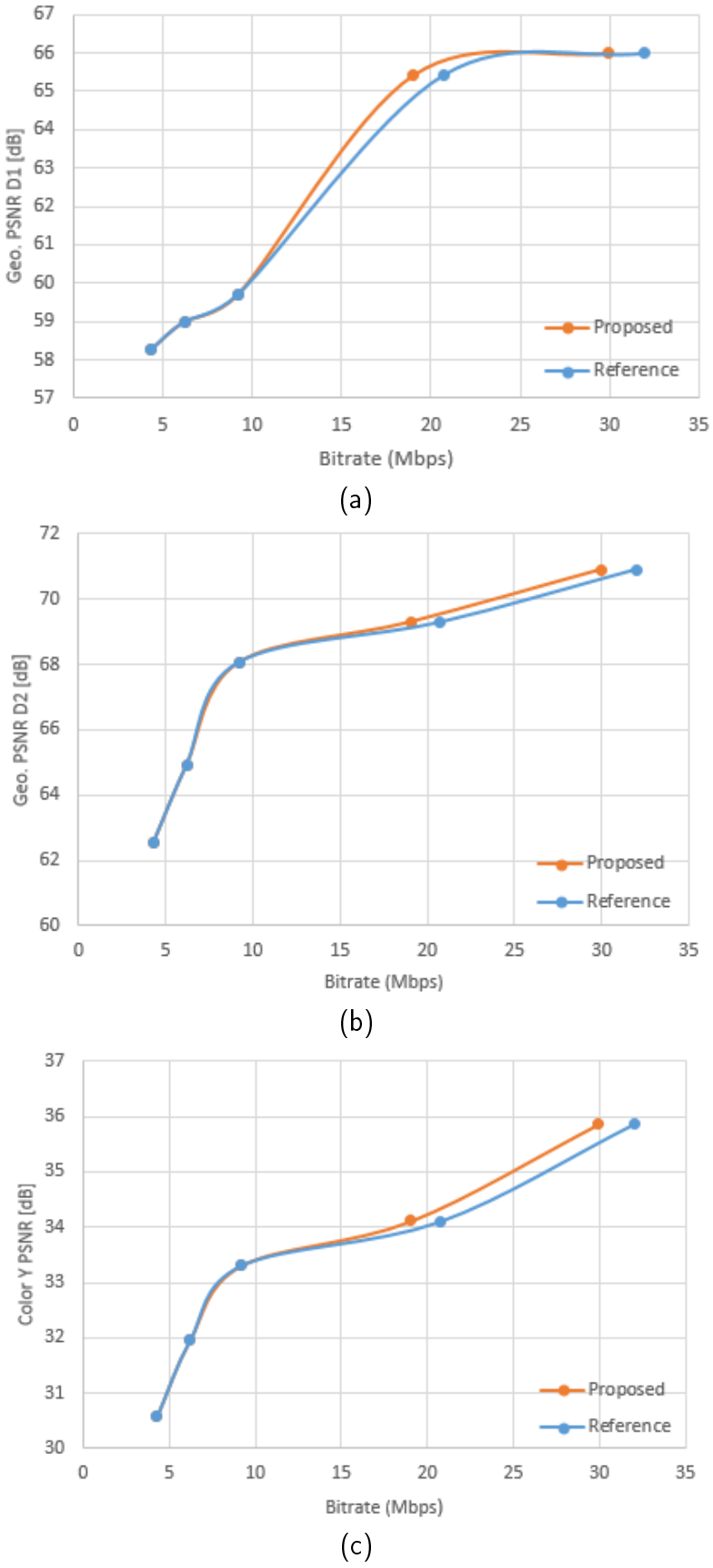
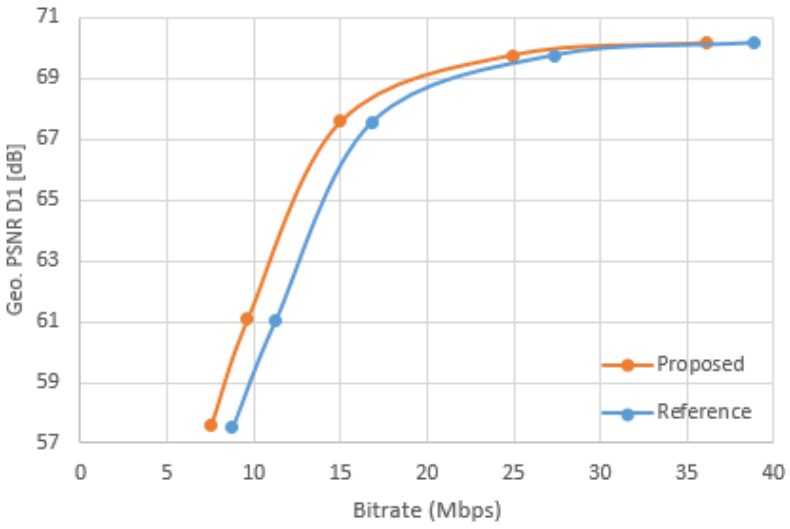
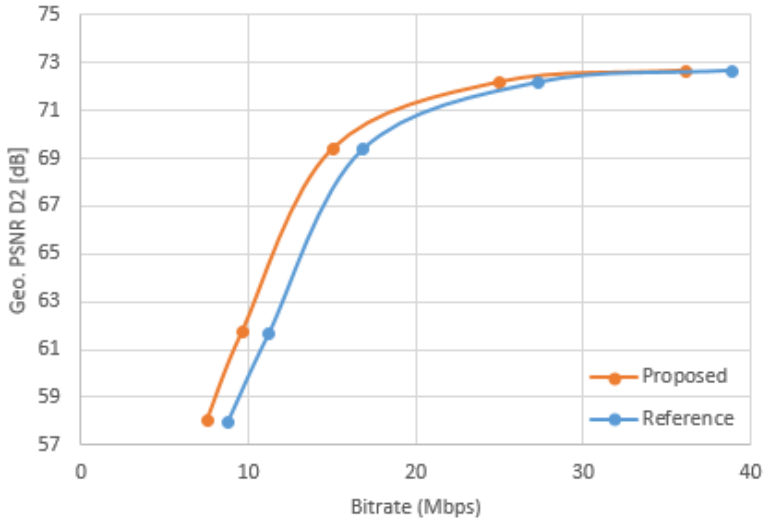


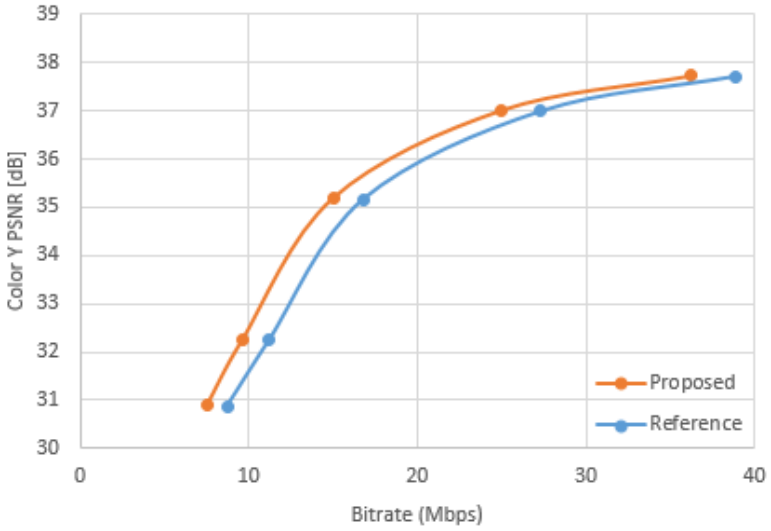
Figure 5.3 *Rate-Distortion curves for the Loot sequence compared to reference for AI configuration and applying Patch Refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*



(a)



(b)



(c)

Figure 5.4 *Rate-Distortion curves for the Loot sequence compared to reference for RA configuration and applying Patch Refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

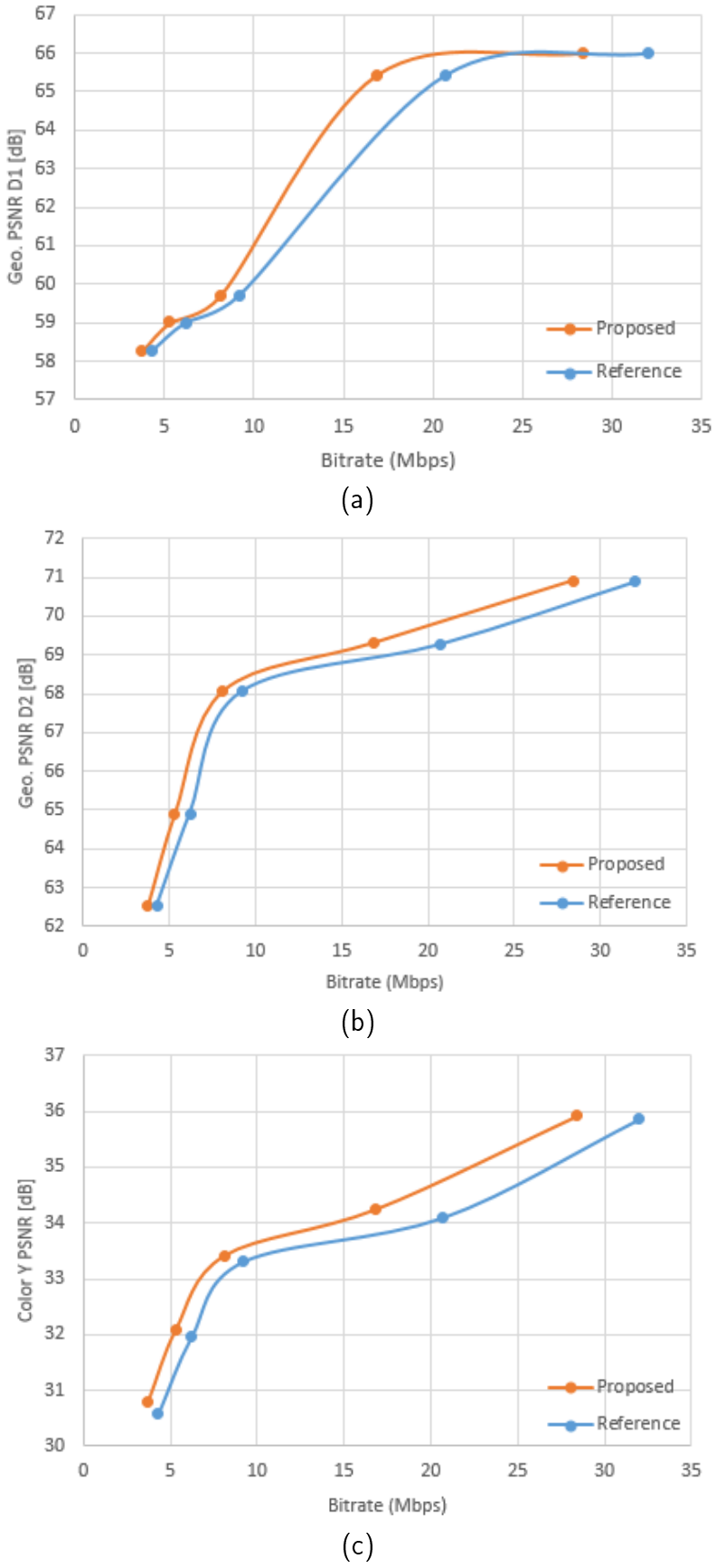


Figure 5.5 *Rate-Distortion curves for the Loot sequence compared to reference for AI configuration and applying Edge Smoothing Patch Refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

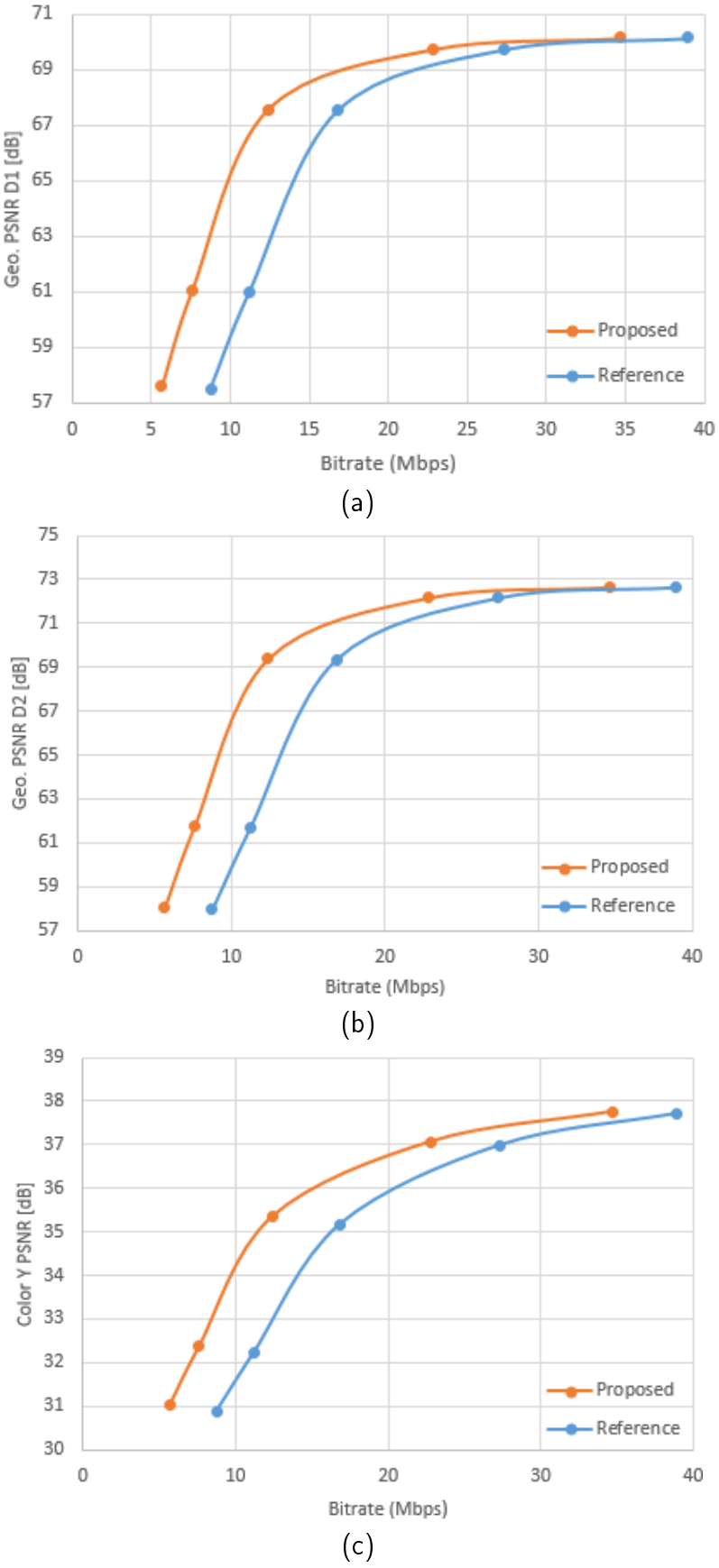


Figure 5.6 *Rate-Distortion curves for the Loot sequence compared to reference for RA configuration and applying edge smoothing Patch Refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

Table 5.8 *BD-Rate (%) and BD-PSNR [dB] performances for Intra-frame comparison of reference technology [1] and improved 2D video coding of volumetric data [3,4]*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-54.9%	-65.9%	-59.6%	-63.3%	-81.0%	4.41	5.06	1.92	1.42	2.83
Redandblack	-39.0%	-48.6%	-26.2%	-75.0%	-100.0%	2.79	3.46	0.69	1.60	3.52
Soldier	-3.0%	-56.8%	1.9%	-6.6%	125.9%	0.36	4.17	-0.03	-0.82	-1.28
Longdress	-68.1%	-53.8%	-56.8%	-88.1%	-100.0%	5.84	4.05	1.62	1.68	3.35
Average	**-41.2%**	**-56.3%**	**-35.2%**	**-58.3%**	**-38.8%**	**3.35**	**4.18**	**1.05**	**0.97**	**2.11**

Table 5.9 *BD-Rate (%) and BD-PSNR [dB] performances for Inter-frame comparison of reference technology [1] and improved 2D video coding of volumetric data [3,4]*

Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-71.2%	-49.2%	-64.8%	-84.9%	-100.0%	8.08	5.51	3.50	3.76	5.22
Redandblack	-63.9%	-35.1%	-17.6%	-66.3%	-79.3%	5.63	3.29	0.56	2.10	4.17
Soldier	-75.7%	-67.2%	-58.9%	-62.0%	-52.3%	6.06	6.78	2.31	1.51	1.48
Longdress	-79.0%	-59.5%	-65.4%	-86.4%	-100.0%	8.26	5.69	2.72	3.18	4.71
Average	**-72.4%**	**-52.8%**	**-51.7%**	**-74.9%**	**-82.9%**	**7.01**	**5.32**	**2.27**	**2.64**	**3.90**

5.4.4 Improved Projection-based Method Versus the State-of-the-Art

Tables 5.8 and 5.9 demonstrate the comparison of improved projection-based and reference technology in point cloud compression [1]. The tables reveal significant bitrate improvement for adding proposed tools to projection-based point cloud compression for all metrics. In the AI scenario, the improved projection-based method reduced the required bitrate by around 40% for geometry component. For the color components, bitrate reductions of 35%, 58% and 38% achieved for Y, U and V components, respectively.

The performances are higher in RA configuration compared to the AI testing case. The required bitrate for D1 metric(point-to-point distortion) is reduced by 72% and for YUV color components improved by 50%, 75% and 83%, respectively.

Figures 5.7 and 5.8 demonstrate the RD-Curves of the projection-based and the reference technology for point cloud compression. As can be observed, the projection-based approach outperforms the reference technology significantly regardless of the coding configuration. However, the performance of the projection-based scheme is higher in the RA configuration than the AI configuration compared to the reference technology.

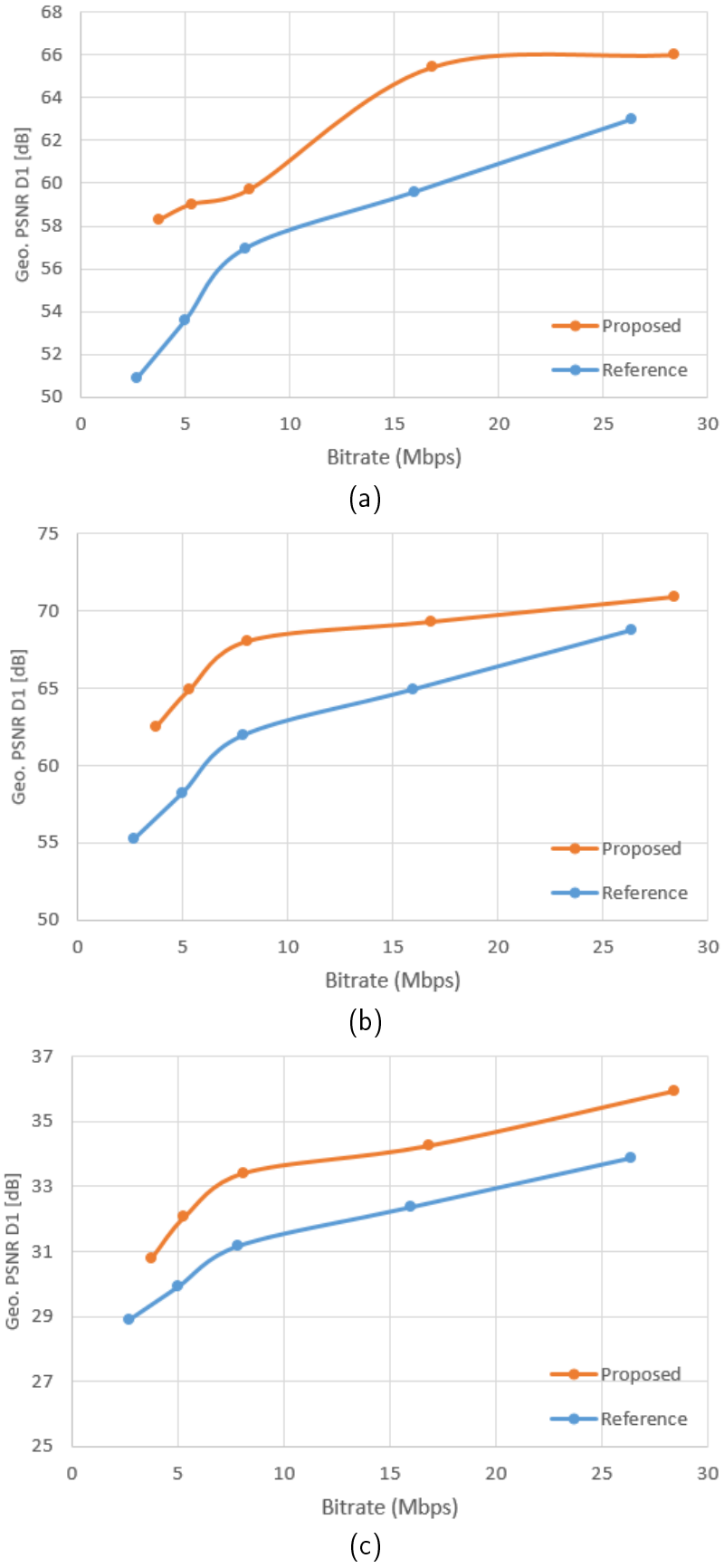
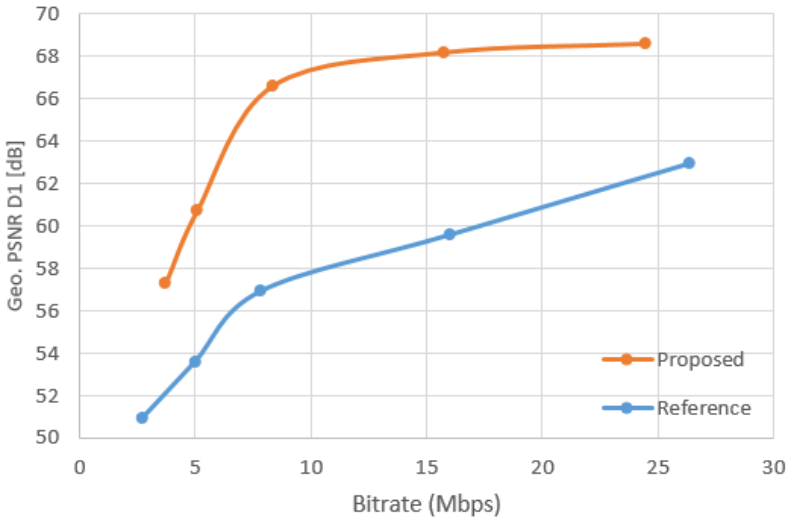
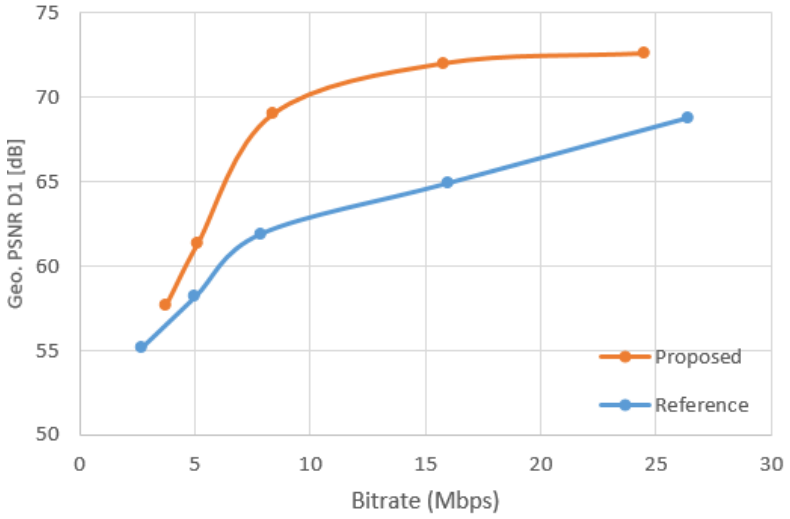


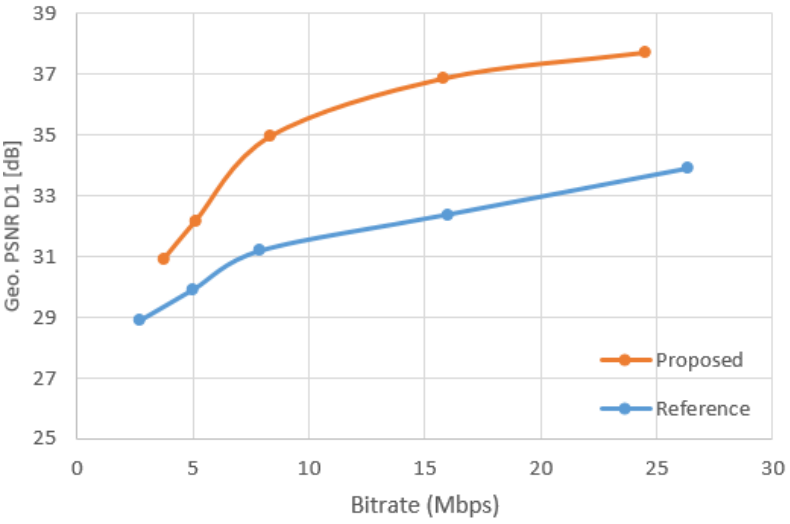
Figure 5.7 *Rate-Distortion curves for the Loot sequence compared to reference for AI configuration and projection-based approach. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*



(a)



(b)



(c)

Figure 5.8 *Rate-Distortion curves for the Loot sequence compared to reference for RA configuration and applying Edge Smoothing Patch Refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

(a) (b) (c) (d) (e)

Figure 5.9 *(a) Original Longdress point cloud. Decoded Longdress point clouds in the 94 Mbit/s (b) without proposed methods, (c) applying Edge Smoothing, (d) applying Patch Refinement, (e) applying both Edge Smoothing and Patch Refinement*

(a) (b) (c) (d) (e)

Figure 5.10 *(a) Original Soldier point cloud. Decoded Soldier point clouds in the 63 Mbit/s (b) without proposed methods, (c) applying Edge Smoothing, (d) applying Patch Refinement, (e) applying both Edge Smoothing and Patch Refinement*

Table 5.10 *Complexity evaluation for all intra config and applying Edge Smoothing and/or Patch Refinement*

Sequence	Edge Smoothing (%)		Patch Refinement (%)		Both (%)	
	Encoder	Decoder	Encoder	Decoder	Encoder	Decoder
Loot	100.7%	100.5	97.7%	100.2	100.4%	101.5%
Redandblack	99.4%	100.5	99.4%	100.4	99.0%	101.1%
Soldier	105.5%	101.7	105.0%	101.6	104.4%	102.2%
Longdress	96.7%	100.5	101.3%	103.6	104.1%	104.3%
Average	**99.3%**	**100.9%**	**100.8%**	**101.4%**	**102.0%**	**102.3%**

Table 5.11 *Complexity evaluation for random access config and applying Edge Smoothing and/or Patch refinement*

Sequence	Edge Smoothing (%)		Patch Refinement (%)		Both (%)	
	Encoder	Decoder	Encoder	Decoder	Encoder	Decoder
Loot	105.7%	101.1	103.5%	100.4	104.3%	101.8%
Redandblack	99.6%	100.9	97.8%	101.4	98.8%	101.4%
Soldier	95.3%	100.1	97.2%	98.5	95.6%	99.7%
Longdress	96.2%	101.0	96.9%	104.7	89.1%	101.9%
Average	**99.1%**	**101.0%**	**98.8%**	**101.2%**	**96.8%**	**101.2%**

Figure 5.9 and 5.10 demonstrate the subjective comparison between the original point cloud, reference (decoded point cloud without using the proposed solutions in this thesis), applied each proposed solutions separately and combined both proposed solutions for *Longdress* and *Soldier* sequences at 94 Mbit/s and 63 Mbit/s, respectively. The decoded point clouds for each sequence are in the same bitrate level in order to observe the quality performance of described methods. It may not be possible to see the differences in the first glance, yet there are some places in which the quality improvement is prominent and visible to the naked eyes. For instance, the coding artifacts in the shoulder of the *Longdress* is removed or the scar in face of *Solider* is almost filled in addition to his sleeves. As it can be seen from the figures, the proposed algorithms have improved the quality of the decoded point clouds significantly compared to the reference method.

5.5 Complexity Analysis

The encoding and decoding complexities of the proposed methods in this work compared to the projection-based scheme are shown in Tables 5.10 and 5.11 for AI and RA configurations, respectively. The evaluations are done based on the encoding and decoding runtimes in the same operating system.

As can be observed from the tables, the proposed methods has negligible impact in the complexities in both encoder and decoder while providing significant bitrate

reductions.
The run time numbers for encoding and decoding is also reported as an comparison factor to examine the complexity of proposed algorithms.
Tables 5.10 and 5.11 represent complexity of applying Edge Smoothing and Patch Refinement methods on top of the projection-based point cloud compression approach.
As shown for applying the Edge Smoothing in both all intra and random access the run time for encoder is decreased by one percent, while in the decoder side the run time is increased by one percent.
The complexity effect of the proposed Edge Smoothing method is on average 1% decrease and 1% increase in encoding and decoding operations, respectively for both AI and RA configurations. The Patch Refinement method increases the encoding and decoding time by around 1% on average for AI configuration. However, in the RA configuration, the decoding time is increased by around 1% while the encoding complexity is reduced by 1%. Such behavior can also be observed when both Edge Smoothing and Patch Refinement tools are enabled. In this case, the encoding and decoding complexities are increased by around 2% on average for AI configurations. In the case of RA configurations, the decoding runtime is increased by around 1% whereas the encoding complexity is decreased by around 3%.

6. CONCLUSION AND FUTURE WORK

This chapter provides a summary of the work which have been done in this thesis, limitations of the studied methods, as well as the future work regarding the research topic of this work.

Many point cloud compression solutions have been suggested in the literature and in the recent study, a projections-based scheme has been proposed to compress point cloud data efficiently. In this approach, the volumetric video data is projected on 2D image planes and 2D video coding standards are utilized in order to compress the projected 2D data.

In this study several projection methods have been examined in order to select the best projection. In this context, a suitable projection was defined as an approach which loses less points, handles the occlusions and gains high PSNR. The conducted studies demonstrated that the sequential decimation method satisfies the above-mentioned criterias for the 2D projection of point cloud content.

In addition, this thesis work highlighted the importance of addressing high-frequency content in projection-based compression of volumetric data using 2D video coding technology. Since most of 2D video coding standards are block-based, the sharp edges of projected patches in 2D plane would generate high-frequency components. Therefore, the sharp edges of projected patches and sparsity in projected 2D images require special attention. Accordingly, two algorithms have been added to the projection-based approach to improve the coding efficiency. First, in order to tackle the high contrast and transition between the background and the projected patches, the Edge Smoothing method is proposed. The proposed method improved the coding efficiency remarkably. By applying this algorithm bitrate is decreased by around 18% and 20% in geometry and color attribute, respectively. Moreover, for the purpose of reducing the sparsity in the 2D plane, Patch Refinement is introduced to preserve points which has most connection with the neighbors. In this method, bitrate is decreased by approximately 18% for geometry and 20% color attribute. Applying the combination of both Edge Smoothing and Patch Refinement methods provided on average bitrate reductions of around 26% for geometry and 29% color attributes.

In overall, the projection-based approach reduced the required bitrate by around

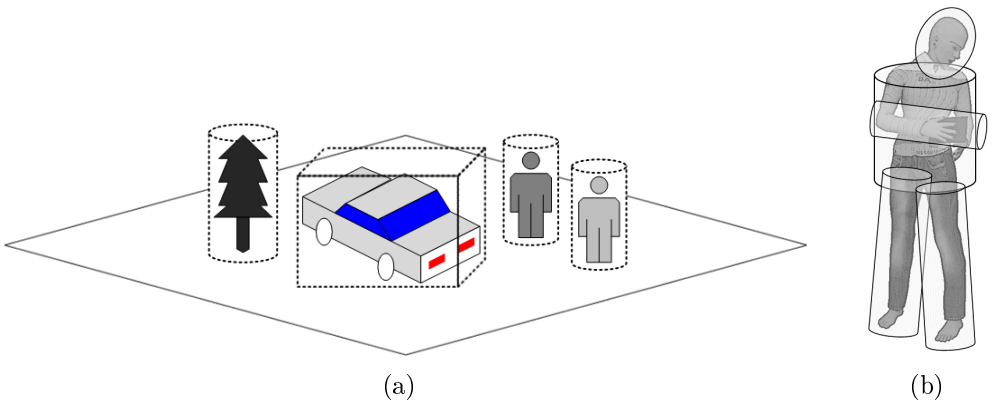


Figure 6.1 *(a)Repartitioning of scene or (b) object into smaller generic projection geometries*

72% for geometry, and 51% color attribute compared to the reference technology in which approximately one third of total bitrates saving is achieved by proposed methods in this thesis. Furthermore, the proposed solutions reduced the coding artifacts in the boundary areas of the projected point cloud data.
The proposed algorithms have been implemented and integrated as part of a contribution to the MPEG CfP on Point Cloud Compression [3,4] and objective evaluation proves the claim of improvement.
Although projection-based method provides significant coding gains compared to the state-of-the-art, we are aware that our research may have two inevitable limitations which originates from 3D to 2D projection. The first one is due to the occlusion in the projections process leads holes in the reconstructed point cloud. The second issue is related to invalid 3D points in the reconstruction point cloud due to the 2D video coding artifacts [3].
The projection-based is a novel and efficient method to compress point cloud data. The principal advantage of this approach is that it uses the standard 2D video coding technology, which is outcome of several decades of experts' work in the field of 2D video coding. Furthermore, current software and hardware are compatible for real-time implementation. Moreover, spatial and temporal compression efficiency is highly improves over the state-of-the-art for point cloud compression [1]. The current form of point cloud compression is suitable for evaluation content for Category 2 of the CfP, i.e. a single, coherent, moving object without any additional scenery or background. Although for future works it is feasible to use current approach for complicated scenes. Figure 6.1 shows repartitioning of the object into smaller projection planes and in larger scene subdivide the scene with different objects and project into smaller projection geometries.

BIBLIOGRAPHY

[1] 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, 2017.

[2] "Call for proposals for point cloud compression V2," *ISO/IECJTC1/SC29/WG11 N16763*, 2017.

[3] S. Schwarz, M. M. Hannuksela, V. Fakour-Sevom, and N. Sheikhi-Pour, "2D video coding of volumetric video data," in *IEEE Picture Coding Symposium*, 2018.

[4] S. Schwarz, M. M. Hannuksela, V. Fakour-Sevom, N. Sheikhi-Pour, V. Malamal-vadakital, and A. Aminlou, "Nokia's response to cfp for point cloud compression (category 2)," *ISO/IECJTC1/SC29/WG11 M41779*, 2017.

[5] "Draft test conditions and complementary test material," *ISO/IEC JTC1/SC29/WG11 N16716*, 2017.

[6] J. Kammerl, N. Blodow, R. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *IEEE ICRA*, 2012.

[7] K. Mamou, N. Stefanoski, H. Kirchhoffer, K. Muller, T. Zaharia, F. Preteux, D. Marpe, and J. Ostermann, "The new MPEG-4/FAMC standard for animated 3D mesh compression," in *3DTV Conference*, 2008.

[8] "Google draco." https://github.com/google/draco, (04.04.2018).

[9] D. Thanou, P. A. Chou, and P. Frossard, "Graph-based motion estimation and compensation for dynamic 3d point cloud compression," in *Image Processing (ICIP), 2015 IEEE International Conference on*, pp. 3235–3239, IEEE, 2015.

[10] 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.

[11] P. Merkle, A. Smolic, K. Muller, and T. Wiegand, "Multi-view video plus depth representation and coding," in *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, vol. 1, pp. I–201, IEEE, 2007.

[12] P. Alliez and M. Desbrun, "Progressive compression for lossless transmission of triangle meshes," in *ACM SIGGRAPH*, 2001.

[13] J. Peng and C. C. J. Kuo, "Progressive geometry encoder using octree-based space partitioning," in *IEEE ICME*, 2004.

[14] Y. Huang, J. Peng, C.-C. J. Kuo, and M. Gopi, "Octree-based progressive geometry coding of point clouds," in *3rd Eurographics / IEEE VGTC Conference on Point-Based Graphics*, 2006.

[15] R. Schnabel and R. Klein, "Octree-based point-cloud compression," in *3rd Eurographics / IEEE VGTC Conference on Point-Based Graphics*, 2006.

[16] L. Wang, L. Wang, Y. Luo, and M. Liu, "Point-cloud compression using data independent method - a 3D discrete cosine transform approach," in *IEEE ICIA*, 2017.

[17] G. J. Sullivan, J.-R. Ohm, W.-J. Han, T. Wiegand, *et al.*, "Overview of the high efficiency video coding(hevc) standard," *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[18] J. M. Boyce, Y. Ye, J. Chen, and A. K. Ramasubramonian, "Overview of SHVC: Scalable extensions of the high efficiency video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 20–34, 2016.

[19] "HEVC scalability extension (SHVC)." https://hevc.hhi.fraunhofer.de/shvc, (04.04.2018).

[20] G. Tech, Y. Chen, K. Müller, J.-R. Ohm, A. Vetro, and Y.-K. Wang, "Overview of the multiview and 3D extensions of High Efficiency Video Coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 35–49, 2016.

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

[22] P. Larbier *et al.*, "Hevc & broadcast content," in *Ateme Whitepaper*, 2013.

[23] R. Ghaznavi-Youvalari, A. Aminlou, M. M. Hannuksela, and M. Gabbouj, "Efficient coding of 360-degree pseudo-cylindrical panoramic video for virtual reality applications," in *Multimedia (ISM), 2016 IEEE International Symposium on*, pp. 525–528, IEEE, 2016.

[24] E. d'Eon, T. M. B. Harrison, and P. A. Chou, "8i voxelized full bodies - a voxelized point cloud dataset," *ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006,Geneva*, 2017.

[25] "Clarifications and corrigenda for cfp point cloud compression," *ISO/IEC JTC1/SC29/WG11 N17091*, 2017.

[26] A. M. Tourapis, D. Singer, Y. Su, and K. Mammou, "BD-Rate/BD-PSNR excel extensions," *ISO/IECJTC1/SC29/WG11 M41482*, 2017.