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 nemidoanm of nemidoanm 2018

ABSTRACT

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

Point clouds for immersive media technology have received substantial interest in the recent years. Such representation of 3D scenery provides freedom of movement for the viewer. But transmitting 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. 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. Such coding schemes provide significant improvement over state-of-the-art methods in terms of coding efficiency. However, the projection-based point cloud compression requires special handling of boundaries and sparsity in the 2D projections. This paper 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. 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.

PREFACE

The research work in this thesis has been carried out from May 2017 - October 2017, at Nokia Technologies in collaboration with Department of Signal Processing, Tampere University of Technology (TUT), Tampere, Finland.
First and foremost, I would like to express my deepest gratitude to my supervisor Prof. Moncef Gabbouj for providing the opportunity for me to conduct my thesis research and his guidance during this project.
My sincere acknowledgment goes to Dr. Miska Hannuksela from Nokia Technologies for his endless support, technical and academic guidance during this work.
I am also grateful to Dr. Alireza Aminlou not only for his excellent co-supervision of this work, but also helping me during the difficulties that I faced throughout this research work.
I would also like to thank my colleagues in Nokia Technologies, Emre Aksu, Jani Lainema, Alireza Zare, Kashyap Kammachi Sreedhar, Antti Hallapuro, Vinod Malamalvadakital, Igor Curcio and Jari Hagqvist for their support and providing friendly office atmosphere.
Special thanks to my dearest friend Saber Kordestanchi for his endless support and friendship during my studies.
I have had very good and supporting friends whom I'd like to thank for all their help: Solmaz Hach, Masoud Malekzadeh, Mohammad Behgam, Saman Bahrampour, Sajjad Nouri, Pouria Hajiani and Sounak Bhattacharya.
And finally I deeply appreciate the support of my parents, my father Ali Ghaznavi and my late mother Effat Heidarzadeh. The people who always supported every decision that I made in my life with their unbelievable kindness and respect.

Tampere, nemidoanm 2018

Nahid Sheikhi pour

I dedicate this thesis to my mother Effat Heidarzadeh. (1956 - 2015)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND SYMBOLS

2D	Two-Dimensional
3D	Three-Dimensional
AMVP	Advanced Motion Vector Prediction
AR	Augmented Reality
BDBR	Bjøntegaard Delta Bit Rate
BR	Bit Rate
CABAC	Context Adaptive Binary Arithmetic Coding
CB	Coding Block
CTB	Coding Tree Block
CTU	Coding Tree Unit
CU	Coding Unit
DCT	Discrete Cosine Transform
H.265/HEVC	High Efficiency Video Coding
H.264/AVC	Advanced Video Coding
HMD	Head Mounted Display
MPEG	Moving Picture Experts Group
MR	Mixed Reality
MSE	Mean Square Error
PSNR	Peak Signal-to-Noise Ratio
QP	Quantization Parameter
RA	Random Access
RD	Rate-Distortion
VR	Virtual Reality

DEFINITIONS

Point Cloud Frame: For a static point cloud is a representation at a certain time instance and for dynamic it a 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, it means 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.
Spatial Random Access: Decoding a pre-defined area from the compressed file.
Progressive/Scalable: Possibility to decode a coarse point cloud in the preliminary step and then refine it with some additional information which are in compressed bit stream.
Random Access:
Intra-only: Independent encoding of each frame from any other frames.
Temporal-random-access:
Low-delay:
Operating Point:
1 Mbit/s: 1,000,000 bits per second.

1. INTRODUCTION

Volumetric video represents a three-dimensional scene or object. The volumetric video is either generated from 3D models, i.e. CGI, or can acquire from real-world scenes using a variety of capture solutions, e.g. multi-camera, laser scan. Generic representation formats for such volumetric data are triangle meshes, point clouds, or array of voxel. Point cloud is defined as a set of (x,y,z) in 3D coordinates without strict order and local topology or meshes. Typically each point in the cloud has the same number attributes (i.e., color, normal directions, reflectance) to represent a scene or object [1]. These points are acquired by advanced 3D laser scanner, multi-camera, depth sensors or other technologies which are able to produce a 3D representation of the surface. The output of 3D scanner can be used directly to create 3D CAD models.
Volumetric video represents 3D data in a way that the observer has freedom to navigate freely in the captured scene. Hence, such data has a high importance for virtual reality (VR), augmented reality (AR), or mixed reality (MR) applications, especially for providing six-degrees-of-freedom (6DOF) viewing capabilities. However, due to the large number of points which are generated to represent the scenery, limitations for storage and transmission over current network technology arise. Therefore, efficient compression algorithms are needed to help this technology to expand in

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

all aspects. And these applications can be use in be in the construction business, agriculture and vegetation management, education and etc. Point clouds mainly use to represent the exterior surface of the object, but it can be utilized in medicine, for example, medical imaging to represent volumetric data.

1.1 Objectives and Scope of the Thesis

The conducted research study in this thesis was done in a collaboration of Nokia Technologies and Tampere University of Technology. This document describe technical details of improvement methods for the Nokia Technology submission [2] to category two of the ISO/IEC JTC1/SC29/WG11 (MPEG) in response of Call for Proposals (CfP) for 3D point cloud compression [1]. The aim of this proposed solution is to develop the needs of interactive, tele-immersive applications, such as VR, AR or MR with Six Degrees of Freedom (6DoF) capabilities. The proposed approach compress 3D video data by using 2D video standards. The main advantage of projection approach is its compatibility with current available 2D video coding standards. In addition, in terms of coding efficiency compare with reference technology it shows remarkable improvement which is proved by objective evaluation. Bitrate requirements is reduced by around 75% for geometry and approximately 50% for color attribute compression over the state-of-the-art compression technology. And from overall improvement almost one third is achieved by proposed algorithms in this thesis.

It is feasible to use video coding standards which are available for 2D video coding for point cloud compression, for instance Advanced Video Coding (H.264/AVC [**?**]) or High Efficiency Video Coding (H.265/HEVC [**?**]). The 2D video coding which is used for projection-based point cloud compression is scalability extension of HEVC (SHVC) reference software (SHM) [3], 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 layer or using different video coding standard like 3D High Efficiency Video Coding (3D-HEVC [**?**]). Since neither of these compression standards are designed for point cloud compression the demand for compression tool which is compatible with such content rise.

The projection-based point cloud compression significantly improved over the state-of-the-art method [4] 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.

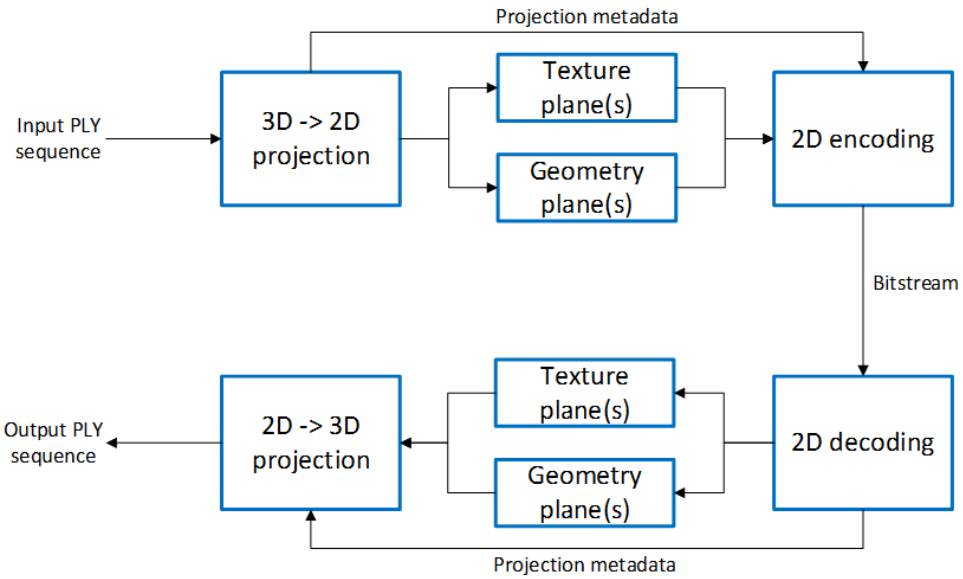


Figure 1.2 *The overall process of projection-based volumetric video coding.*

The aim of this thesis is to provide solutions for the above-mentioned issues in order to improve the coding efficiency of the projection-based schemes. In this study, two processes 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. Furthermore, the patch refinement is considered for filtering the sparsely projected point cloud data in projection plane for reducing sparsity and improving the compression performance. Figure 1.2 illustrates block diagram of overall process of the projection based point cloud compression.

- **3D to 2D projection:** Projecting each individual point cloud of a sequence onto the selected 2D geometry. One 2D plane is allocated for texture projections and one for geometry.
- **2D encoding:** The 2D planes are encoded with the current standard 2D video codecs (e.g SHM)
- **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 (geometry and texture). 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.

1.2 Thesis Outline

The rest of the thesis is organized as follows:

- **Chapter 2:** Introducing different projection methods and functionality of projection based point cloud compression.
- **Chapter 3:** Discussing about the problems and proposing methods to improved coding efficiency of projection based point cloud compression.
- **Chapter 4:** Analyzing the experimental results and related discussions.
- **Chapter 5:** Gives a conclusion and summary of the projection based point cloud and implemented algorithms.

(a) (b) (c)

Figure 1.3 *First frame of Longdress point cloud. Bounding box for point cloud (a), projection of the point cloud on 2D and assigning one image for texture and one for geometry (b)*

2. PROJECTION BASED POINT CLOUD COMPRESSION

2.1 Introduction

There are variety of 3D to 2D projections exists and it is hardly possible to introduce one projection as the best projection for mapping since it is highly depended on the application. Whereas this work is a response to Call for Proposals (CfP) Category two of the ISO/IEC JTC1/SC29/WG11 (MPEG) for Point Cloud Compression, in which a test model are single 3D model. Thus, the approach and evaluation are provided in this context. Category two consist of scaning real moving moving people time instances. The target projection would cover the exterior part of the model.
In order to find the proper projection for the above-mentioned content the compression step which illustrates in Figure 1.2 has been removed to validate performance of the proposed projection method. Figure 2.1 illustrate the general process of the projection test. In fact, in Figure 2.1 the 2D encoder and decoder has been removed to test the functionality of projection. The functionality of projection is evaluated based on subjective and objective evaluation of the reconstructed point cloud. Here a desirable projection defines as a projection which covers more points, handles the occlusion of the reconstructed point cloud and also gain high Peak Signal to Noise Ratio(PSNR).

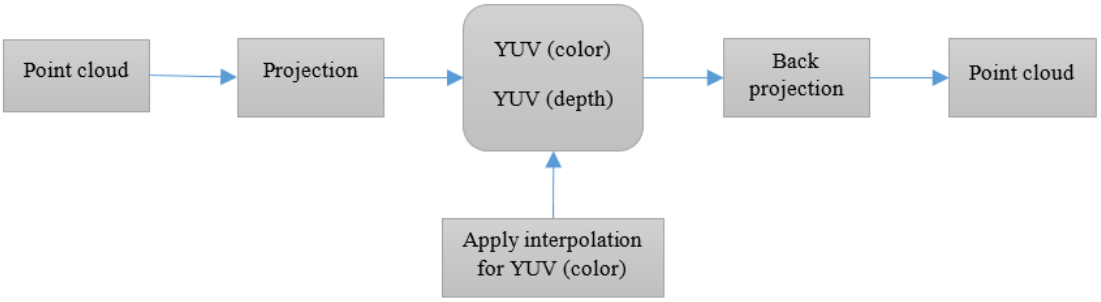
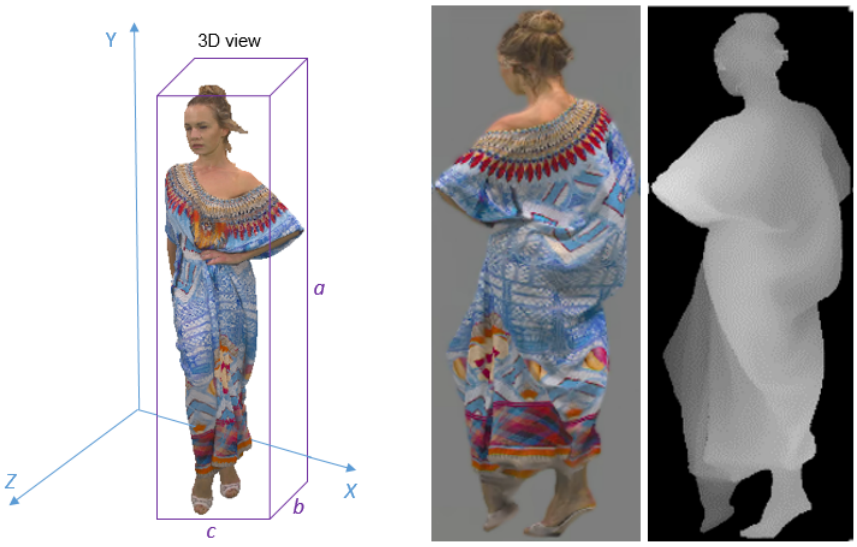


Figure 2.1



(a) (b)

Figure 2.2 *First frame of Longdress point cloud. Bounding box for point cloud (a), projection of the point cloud on 2D and assigning one image for texture and one for geometry (b)*

2.2 Projection

Here projection defines 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 projection which 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 point cloud. In geometry, the bounding box is the smallest or minimum box which encloses all set of points in N dimension where in the 3D space the bounding box is a cuboid shape. Figure 2.2(a) shows a bounding box for *Longdress* point cloud in which all samples are surrounded by bounding box. Another substantial parameter is primary axis which is defined as longest axis of bounding box. The best primary axis for the test data which are used for this experiment is Y-axis.

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 assigned, one for color attribute (texture image) and one for 3D depth (geometry image). Figure 2.2(b) shows the texture and geometry image for one frame of *Longdress* sequence projected onto plane projection.

The length of the space diagonal is calculated by:

$$d = \sqrt{a^2 + b^2 + c^2} \tag{2.1}$$

The position of 3D points after projection in 2D space is derived from :

$$length_x = max_x - min_x \tag{2.2a}$$
$$length_y = max_y - min_y \tag{2.2b}$$
$$length_z = max_z - min_z \tag{2.2c}$$

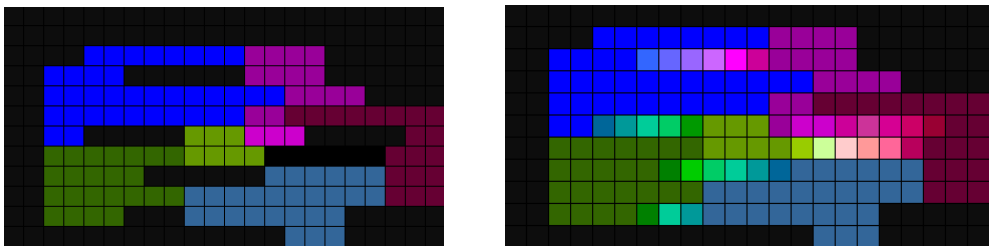
$$position_x = \lfloor \frac{x - x_{min}}{x_{max} - x_{min}} * length_x \rfloor + 1 \tag{2.3a}$$
$$position_y = \lfloor \frac{y - y_{min}}{y_{max} - y_{min}} * length_y \rfloor + 1 \tag{2.3b}$$
$$position_z = \lfloor \frac{z - z_{min}}{z_{max} - z_{min}} * length_z \rfloor + 1 \tag{2.3c}$$

In which length of cube edges is calculated by subtracting minimum value of coordinate from maximum value and it should be multiple of 8 (nemidoanm Ask Ramin).

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 and affect the projection-based scheme sub-optimal. In order to solve this issue, interpolation is applied to calculate the intensity value for missing pixel between known-value pixels. In the texture plane, these null value pixels are interpolated in the horizontal direction by using a linear interpolation method. Linear interpolation:

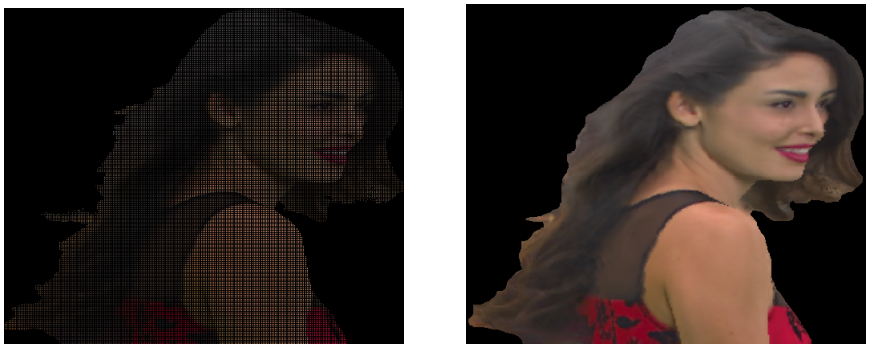
$$f(n+i) = \frac{(d-i) * f(n) + i * f(n+d)}{d}, \quad 0 < i < d \tag{2.4}$$

In which f(n) and f(n+d) are knwon values and d is distance between two known value. 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. Occupancy map which is derived from geometry plane is an image in which occu-



(a) (b)

Figure 2.3 *(a)sparse projection of model (b) inpainted sparse projection by linear interpolation.the colors are exaggerated for illustrative purposes.*



(a) (b)

Figure 2.4 *(a)sparse projection of model (b) small holes are filled by applying interpolation.*

pied pixels are assigned to one and empty ones are zero, so only pixels which their corresponding occupancy map is one are reproject back to 3D space. If inpainting also applied on the geometry plane, the reconstructed 3D data would include many invalid 3D points, unless generating occupancy map before interpolating geometry plane. Figure 2.4(a) shows the texture plane of *Redandblack* sequence as it can be seen some null values in texture plane are shown as black and Figure 2.3(b) depicts the applying linear interpolation in horizontal direction. The interpolated values will not reconstruct in the back projection from 2D to 3D since the occupancy map for those interpolated values are zero.

2.2.1 Sphere

This transformation turns the location of points in Cartesian space to spherical coordinates. Radios (R) of the sphere is calculated by multiplying the arbitrary number larger than one (ex.1.2) with the length of diagonal bounding box.

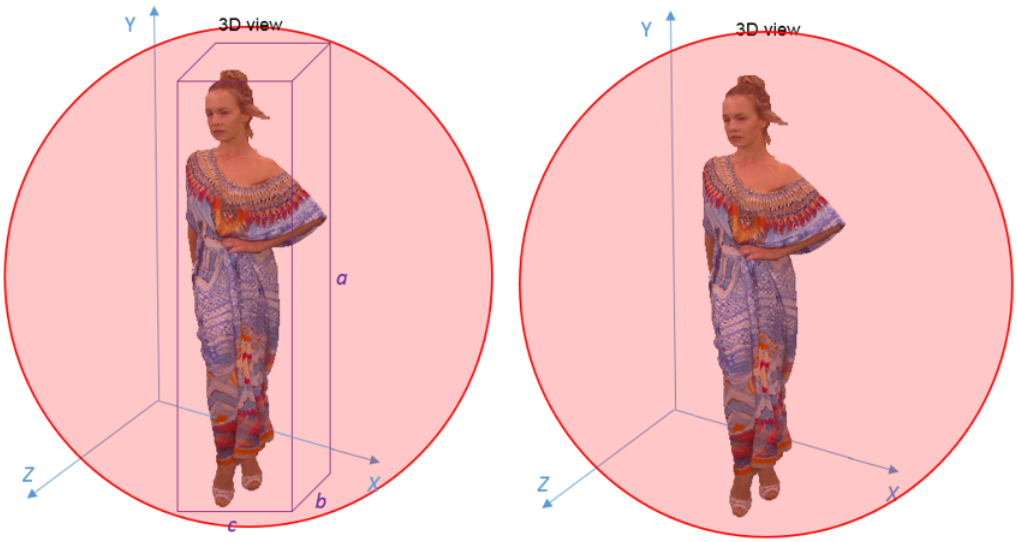


Figure 2.5

$$R_{sp} = d * 1.2 \tag{2.5a}$$

$$X_{sp} = \arccos \frac{X}{R} * d \tag{2.5b}$$

$$Y_{sp} = \arccos \frac{X}{R} * d \tag{2.5c}$$

$$Z_{sp} = \arccos \frac{X}{R} * d \tag{2.5d}$$

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

$$H = \sqrt{X_{sp}^2 + Y_{sp}^2} \tag{2.6a}$$

$$r = \arccos \frac{X}{R} * d \tag{2.6b}$$

$$azimuth = \text{atan2}(Z_{sp}, H) \tag{2.6c}$$

$$elevation = \text{atan2}(Y_{sp}, X_{sp}) \tag{2.6d}$$

Where r is radius and distance from the origin to a point where both azimuth and elevation are in radian. Azimuth is the counterclockwise angle in the x-y plane from the positive x-axis. The elevation is angle in radians from the x-y plane. Pixel

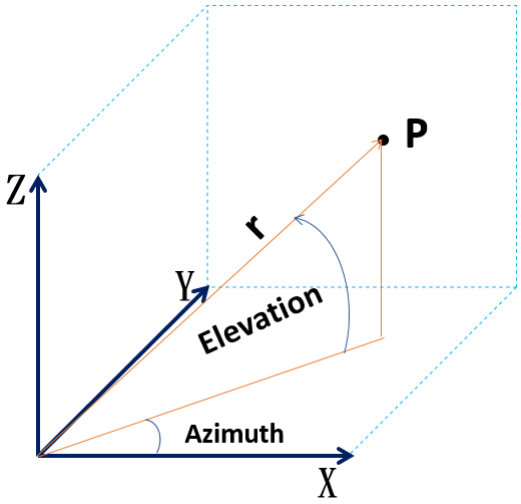


Figure 2.6

coordinates for the equirectangular picture is calculated by:

$$x_{pos} = \lfloor \frac{azimuth + \pi}{\frac{2\pi}{width}} \rfloor + 1 \tag{2.7a}$$

$$y_{pos} = \lfloor \frac{elevation + \frac{\pi}{2}}{\frac{2\pi}{width}} \rfloor + 1 \tag{2.7b}$$

The problem with sphere projection is that it does not preserve all points in 3D models which are not round .(nemidoanm Explain why?) image. Figure 2.7(b) and 2.7(d) shows reconstructed result of two point clouds with same projection, as it can be seen it highly depends on the overall structure of the original content.

2.2.2 Cylinder

The second kind of projection which examined is cylinder projection. Similar to sphere projection it project points into the corresponding coordinates which is follow this relation:

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

Figure 2.7 *(a) Original Egyption point cloud (b) reconstructed point cloud in sphere projection (c) Original Longdress point cloud (d) reconstructed point cloud in sphere projection*

$$r = \sqrt{X^2 + Z^2} \tag{2.8a}$$

$$R_{cy} = r * 1.2 \tag{2.8b}$$

$$X_{SP} = \arccos(\frac{X}{r}) \tag{2.8c}$$

$$Y_{SP} = 1 \tag{2.8d}$$

$$Z_{SP} = \arccos(\frac{Z}{r}) \tag{2.8e}$$

In this projection firstly points project onto cylinder position, then unfold cylinder and use it as 2D image.

2.2.3 Cube

This projection is used for projecting points in 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. For each side, the decision is made based on the distance of the point to plane. 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. Depth

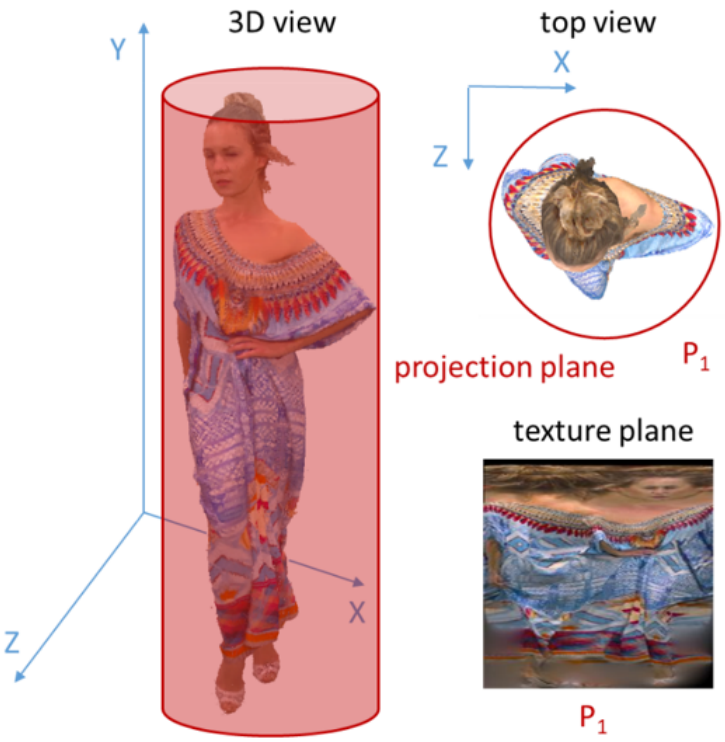


Figure 2.8

buffering stores the distance from the closest point to the 2D plane. Firstly, the projections are done on two opposite sides of the plane. For instance, if there are four planes of projection namely right, front, left and back. Right and left planes which are opposite sides of the cube process (project and back project) simultaneously and same for front and back. Before projecting one point in the texture plane first it should be investigate whether the point which intends to project on texture plain is null or it is occupied in advance.

In the case that more than one 3D point is mapped on the same 2D coordinates Z-buffer is used to compare the distance of the old point and new point to the plane. And the point which has the smaller distance with plane will be store in depth buffer and consequently its color attribute.For the later one since the location of the current point is similar the occupied point comparing the distance is the solution to project the closest points to the 2D plain and store the distance of the corresponding point in the Z-buffer and color channel separately store it in texture plain. Information about the lost points for further processing and comparison can be stored.

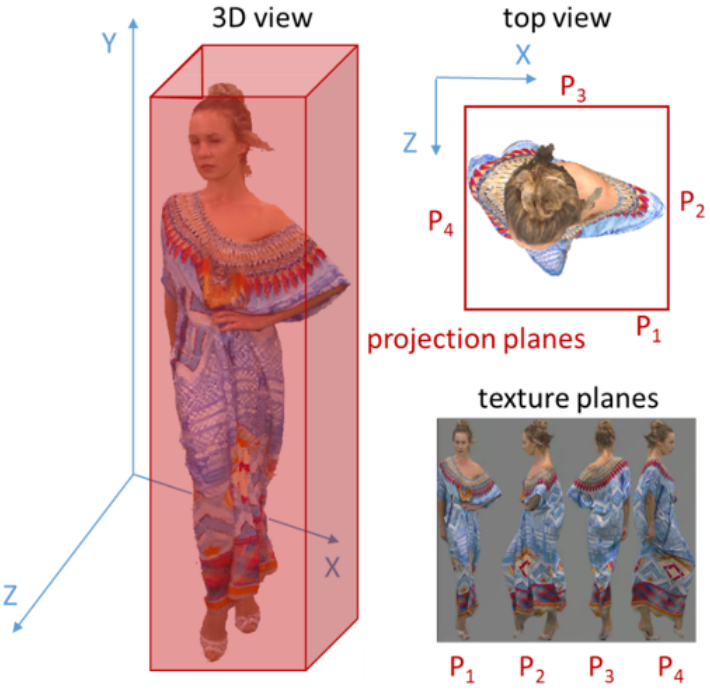


Figure 2.9 *3D to 2D projection onto four rectangular planes and Y is primary axis*

(a) (b)

Figure 2.10 *(a) Texture and (b) geometry projection images examples for Longdress sequence.*

2.2.4 Planer rotation

This projection is similar to projecting on cube ,but all 4 projections (front, left, back and front) are stitched together to make one texture image and one geometry. Figure 2.9 depicts one example projection with four planes and 90-degree rotation between each plane where the starting plane is P_1 and aligned to X axis. In this example Y is primary axis, since between the value ranges of X,Y, and Z, with Y>X>Z, Y axis has wider ranges of values and selected as primary axis and X as an secondary axis. This projection is the base of further implementation for projection-based point cloud compression and for this study, a projection onto series of rectangular planes is

Figure 2.11 *Effect of sequential decimation after each projection*

chosen. Occlusion can improved by increasing the number of projection as depicts in Figure **??**. The offset after the first four rotations is applied to help better coverage of the 3D volume and reducing occlusions.

2.2.5 Sequential Decimation

As illustrated in figure **??** there is a considerable similarities between the first four orations and the last four rotations, thus sequential decimation is introduced. This extension adds more projection of surfaces at different rotations to cover the exterior side of 3D object more coherently. Yet, adding projections does not necessarily cover all the areas. Figure 2.11 depicts the concept of sequential decimation, as shown points which are projected successfully removed from point cloud, therefore the last projection is more spare in comparison the first one.
Figure **??** 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. These additional rotations create sparse data in the projection plane which are not favorable for standard 2D video codecs.

Figure 2.12 *Sequential decimation improves the occlusion handling (left)*

3. PROPOSED METHODS

As described in Section 2.2.5 the projection-based approach provides significant compression performance improvements over the state-of-the-art. However, the projected 2D data has sharp boundaries, which introduce crucial issues when using conventional block-based DCT video coding schemes. In the object boundaries, some blocks may contain partial data from the projected point cloud content and partially from the image plane background. In the encoding process, these are represented as high-frequency DCT components, which result in extra bitrate and decrease reconstruction quality.
Also, 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 and affect the projection-based scheme sub-optimal. In order to solve this issue, we use inpainting 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 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 pixel with a valid geometry value are reprojected back into 3D space. If inpainting would also be applied on the geometry plane, the reconstructed 3D data would include many invalid 3D points.

3.1 Edge Smoothing

First, 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. Prior to the edge smoothing operation, the background samples are replaced with gray values (i.e., equal to 128 in 8 Bit YUV color space). Then, edge smoothing is applied to the projected point cloud boundaries. This technique includes applying averaging filters in horizontal and vertical direction successively. For this purpose, the averaging filter of equation (3.1) 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

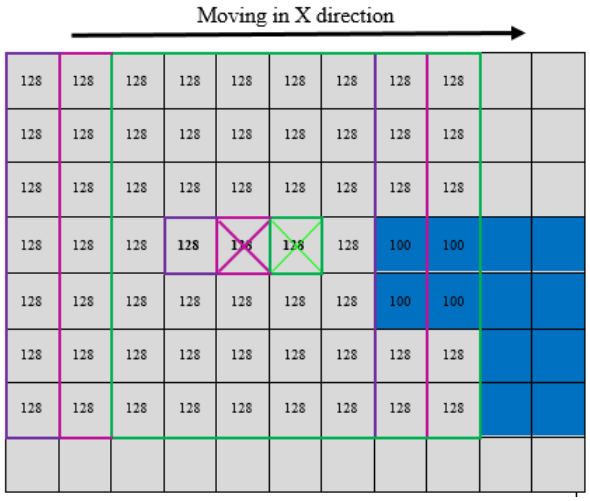


Figure 3.1 *Employing edge smoothing in horizontal direction.*

kernel. In our experiments, the kernel size of 7 provided the best results.

$$P = \frac{\sum_{i=0}^{N-1} \sum_{j=0}^{N-1} X_{i,j}}{N^2}, \tag{3.1}$$

Figure 3.1 illustrates an example of applying the proposed edge smoothing filter in horizontal direction. The blue part indicates a portion of the projected point cloud. The purple, pink and green kernels show the filter kernels that 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 decorrelation process in the compression operation can take place properly and the bitrate would decrease significantly.

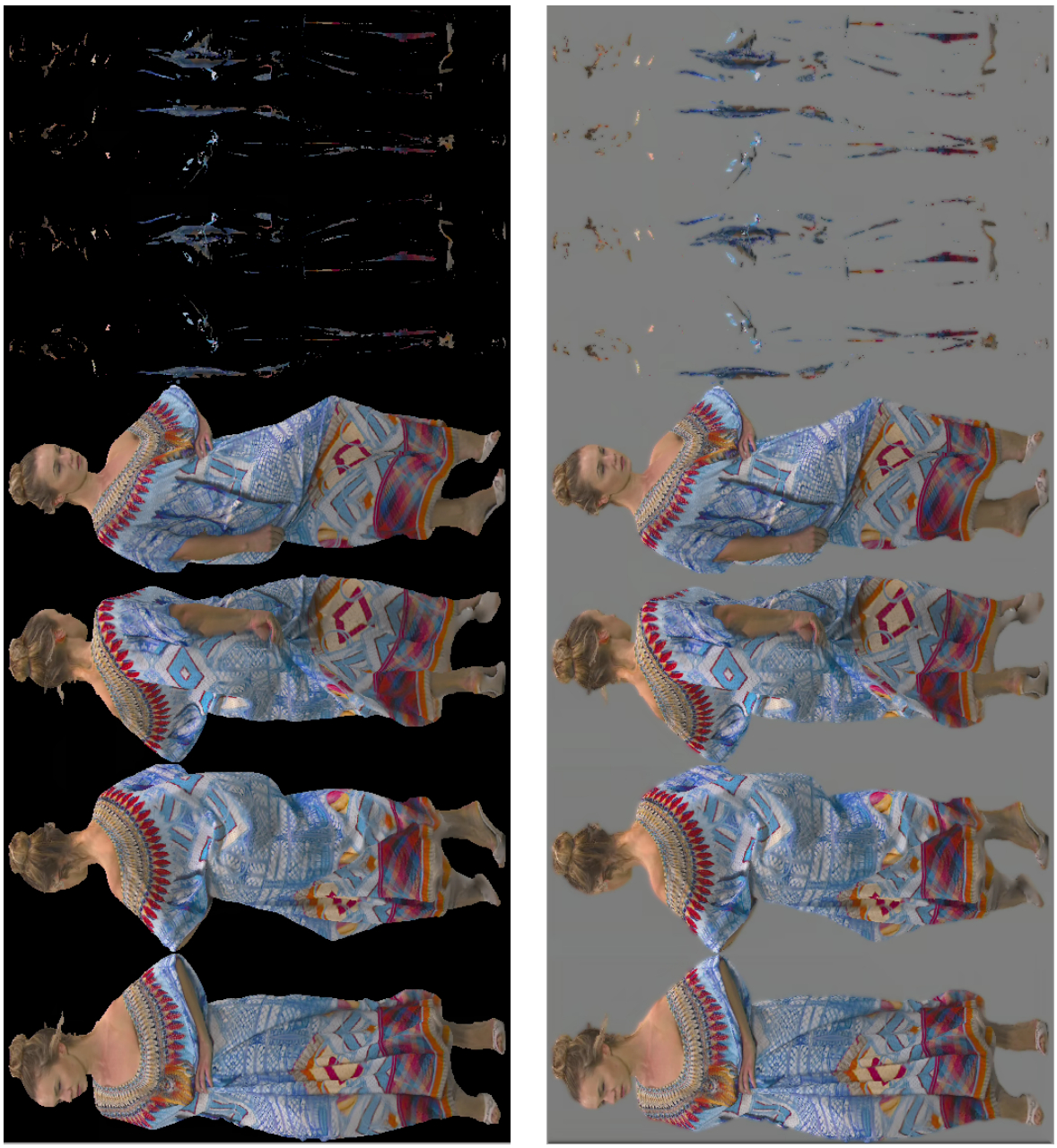


Figure 3.2 Texture plane for decimation rotation (top) edge-Smoothing applied for projected 3D data (bottom)

3.2 Patch Refinement

As it was mentioned in Section 2.2.5, sequential decimation 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 3.2 (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 operation 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 3.3. A it can be seen 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 3.3, 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.

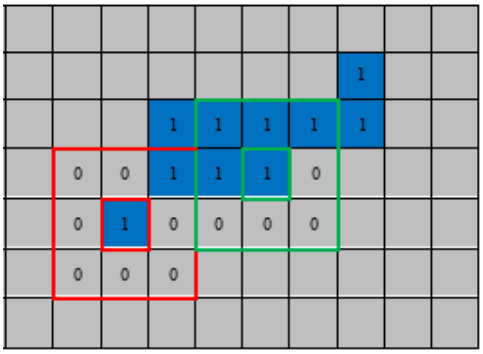


Figure 3.3 *An example of applying patch refinement.*

Figure 3.4 Texture plane for decimation rotation after applying edge-Smoothing(top) patch refinement is applied for projected 3D data (bottom)

4. EXPERIMENTAL RESULTS

This chapter presents the experimental condition, results and analysis of the implemented method which is proposed in chapter 3.
The proposed algorithms are implemented in the HM version 16.6, the HEVC reference software [3] and test condition done under 5 different quantization parameters (QP). Results were provided for dynamic objects test (category two), and testing condition is for all intra and random access. The performance of proposed algorithms was analyzed with the well-known Bjøntegaard delta bitrate (BDBR) metric [16], in which the negative values represent the bitrate reduction in the same peak signal-to-noise ratio (PSNR) and the positive values indicate the bitrate loss for the same PSNR value.

4.1 Point Cloud Data

In the simulations 4 video sequences which are provided by 8i [5] are used to evaluate the performance of the proposed methods. Table 4.1 shows a list of the 3D point cloud test material dataset and their specifications.

4.2 Objective Evaluation Criteria and Metrics

The evaluation of 2D video coding of volumetric data was performed under MPEG CfP for point cloud compression condition. For more details, please refer to the CfP

Table 4.1 *Video Sequences of dynamic objects test Category two are used in this experiment*

Test material dataset filename	Number of frames	Pts	Geometry Precision	Attributes
8i VFB Loot	300	782,000	10 bit	R,G,B
8i VFB RedandBlack	300	700,000	10 bit	R,G,B
8i VFB Soldier	300	1,500,000	10 bit	R,G,B
8i VFB Longdress	300	800,000	10 bit	R,G,B

document [1] and its corrigenda [6]. The proposed methods have been implemented within the scalable extension of the HEVC standard (SHVC) SHM reference software version 12.2 [3]. The performance was analyzed for five different bitrate targets in the range from 3 to 43 Mbit/s in random access (RA) configuration. Four point cloud sequences were evaluated, each consisting of 300 frames.

Two different metrics were calculated to assess the geometry distortion. 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 [7]. The color distortion is computed on a point-to-point level in YUV domain. The performances were analyzed based on the well-known Bjøntegaard Delta Bitrate (BDBR) criterion [8], 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.

4.2.1 Geometric Distortions

The point cloud is define as a set of points without specific order in 3D space and all points have the same number of attributes. The most crucial attributes are geometry location (x,y,z) and color components (r,g,b) or (y,u,v). Let assume **A** as an original point cloud and **B** as a decoded compressed point cloud. **A** is reference for computing the quality of a **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. Compression error which is point cloud **B** relative to the reference point cloud **A** is denoted by $e_{A,B}$. Nearest neighbor is used to identify the correspondence between in **B** and **A**. n this case in order to reduce the computational complexity KD-tree search is utilized. If points in **B** is 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 the Figure 4.1 the black point is collected from point cloud **B** and the red point is the corresponding point of b_j in point cloud **A**.

Computing D1

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

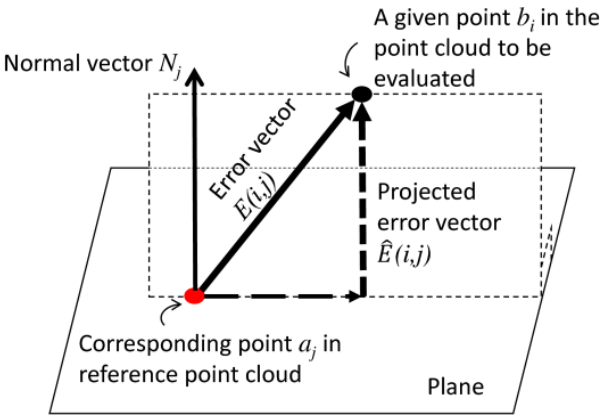


Figure 4.1 *Illustration of point-to-point distance (D1) and point-to-plane distance (D2) [1]*

$$e^{D1}_{A,B}(i) = ||E(i,j)||^2_2 \tag{4.1}$$

Based on the definition of point-to-point distance for one point in 4.1, for calculating point to point error (D1) for all the points in **B**, with N_B number of points, is defined as:

$$e^{D1}_{A,B} = \frac{1}{N_B} \sum_{\forall b_i \in B} e^{D1}_{A,B}(i) \tag{4.2}$$

Computing D2

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 normal. D2 calculates the Mean squared error (MSE) between a reconstructed point and the original point cloud surface [7]. The point-to-plane error is computed as:

$$e^{D2}_{A,B} = ||E(\hat{i},j)||^2_2 = (E(i,j).N_j)^2 \tag{4.3}$$

Similarly the point-to-plane error (D2) for all of points is calculated by:

$$e^{D2}_{A,B} = \frac{1}{N_B} \sum_{\forall b_i \in B} e^{D2}_{A,B}(i) \tag{4.4}$$

Geometry PSNR Calculation

The PSNR value for geometry attribute is computed as:

$$PSNR = 10 \log_{10}(\frac{3p^2}{MSE}) \tag{4.5}$$

For each reference point cloud as specified in Table [nemidoanm] p is defined as the

Table 4.2 *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**

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.

Color PSNR Calculation

The PSNR value for color attribute is computed as:

$$PSNR = 10\log_{10}(\frac{p^2}{MSE}) \tag{4.6}$$

For color attributes, the MSE for each of the three color components is calculated. Color distortion is performed in YUV color space, since YUV color space is closer to human perception. Therefore, if the color space of the point cloud is other than YUV color conversion is performed. A symmetric computation of the distortion is calculated, with the same method which done for geometric distortions. The maximum distortion between the two execution is selected as the final distortion. For PSNR calculation of color the peak vale p is 255, hence the bit depth for test data is 8 bits per point.

4.3 Results for applying Edge-smoothing and patch refinement

This section presents the results of implemented proposed algorithms for 2D compression of point cloud data.

4.4 Applying Edge-smoothing

All Intra Coding Results

Random Access Coding Results

Table 4.3 *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**

Table 4.4 *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**

4.5 Applying Patch refinement

All Intra Coding Results

Random Access Coding Results

4.6 Applying Edge-smoothing and patch refinement

All Intra Coding Results

Random Access Coding Results

Table 4.5 *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**

Table 4.6 *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 4.7 *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	-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**

4.7 Improved Projection-based Versus the Stat-of-the-art

All Intra Coding Results

Random Access Coding Results

Figure 4.10 demonstrates the subjective comparison between the reference and proposed methods for the decoded *Loot* sequence in the same bitrate level. As it can be seen from the figure, the proposed solution has improved the quality of the decoded point cloud significantly compared to the reference method.

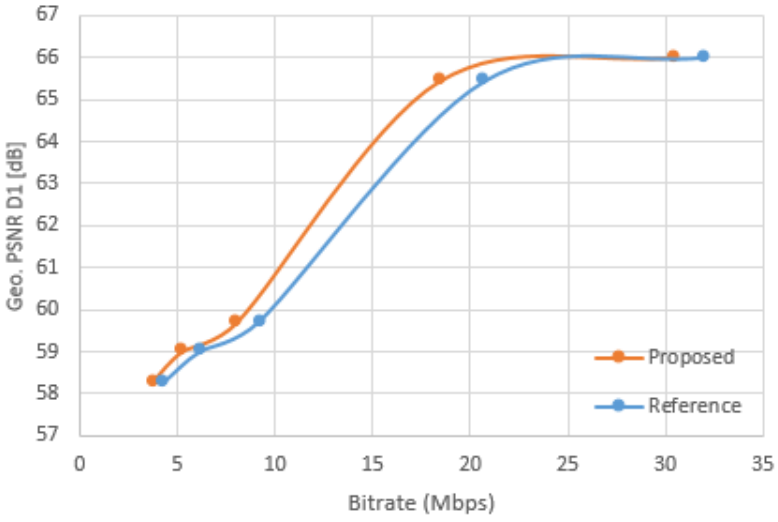
4.8 complexity

Table 4.8 *BD-Rate (%) and BD-PSNR [dB] performances for Intra-frame comparison Ruf and Seb*

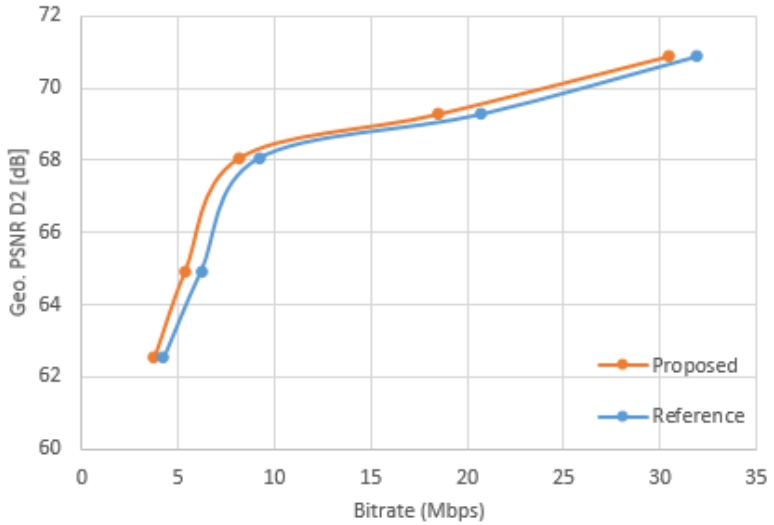
Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-54.9%	-65.9%	-59.6%	-63.3%	-81.0%	4.41	5.06	1.92	1.42	2.83
Redandblack	-39.0%	-48.6%	-26.2%	-75.0%	-100.0%	2.79	3.46	0.69	1.60	3.52
Soldier	-3.0%	-56.8%	1.9%	-6.6%	125.9%	0.36	4.17	-0.03	-0.82	-1.28
Longdress	-68.1%	-53.8%	-56.8%	-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 4.9 *BD-Rate (%) and BD-PSNR [dB] performances for Intra-frame comparison Ruf and Seb*

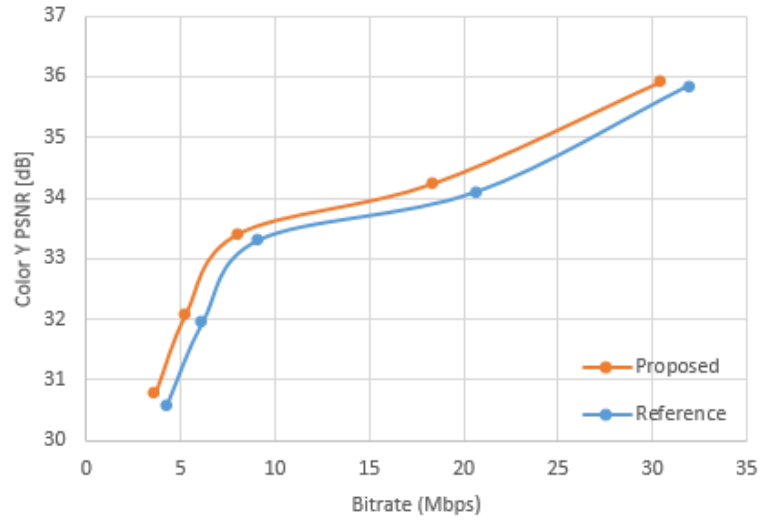
Sequence	BD-Rate (%)					BD-PSNR [dB]				
	D1	D2	Y	U	V	D1	D2	Y	U	V
Loot	-71.2%	-49.2%	-64.8%	-84.9%	-100.0%	8.08	5.51	3.50	3.76	5.22
Redandblack	-63.9%	-35.1%	-17.6%	-66.3%	-79.3%	5.63	3.29	0.56	2.10	4.17
Soldier	-75.7%	-67.2%	-58.9%	-62.0%	-52.3%	6.06	6.78	2.31	1.51	1.48
Longdress	-79.0%	-59.5%	-65.4%	-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**



(a)



(b)



(c)

Figure 4.2 *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*

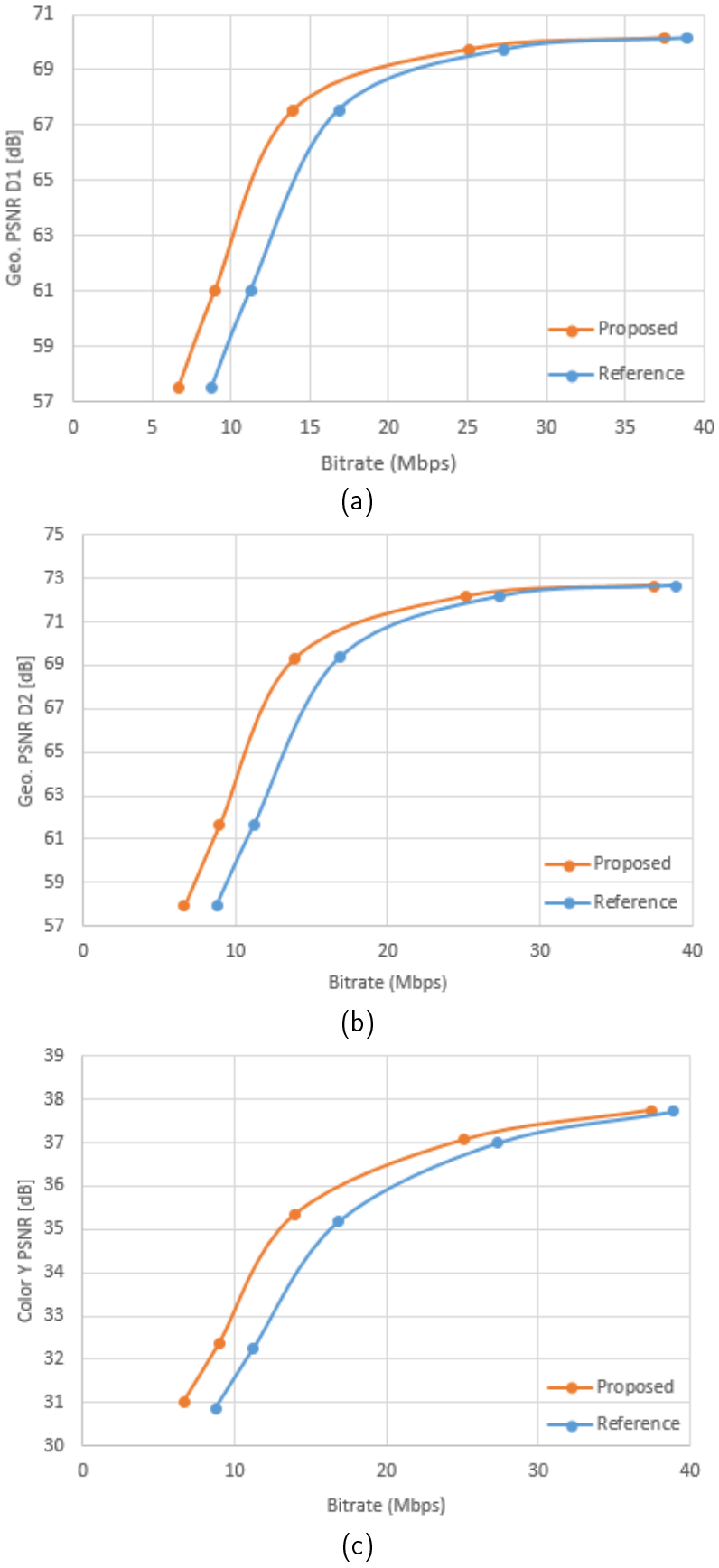


Figure 4.3 *Rate-Distortion curves for the Loot sequence compared to reference for RA configuration and applying edge smoothing. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

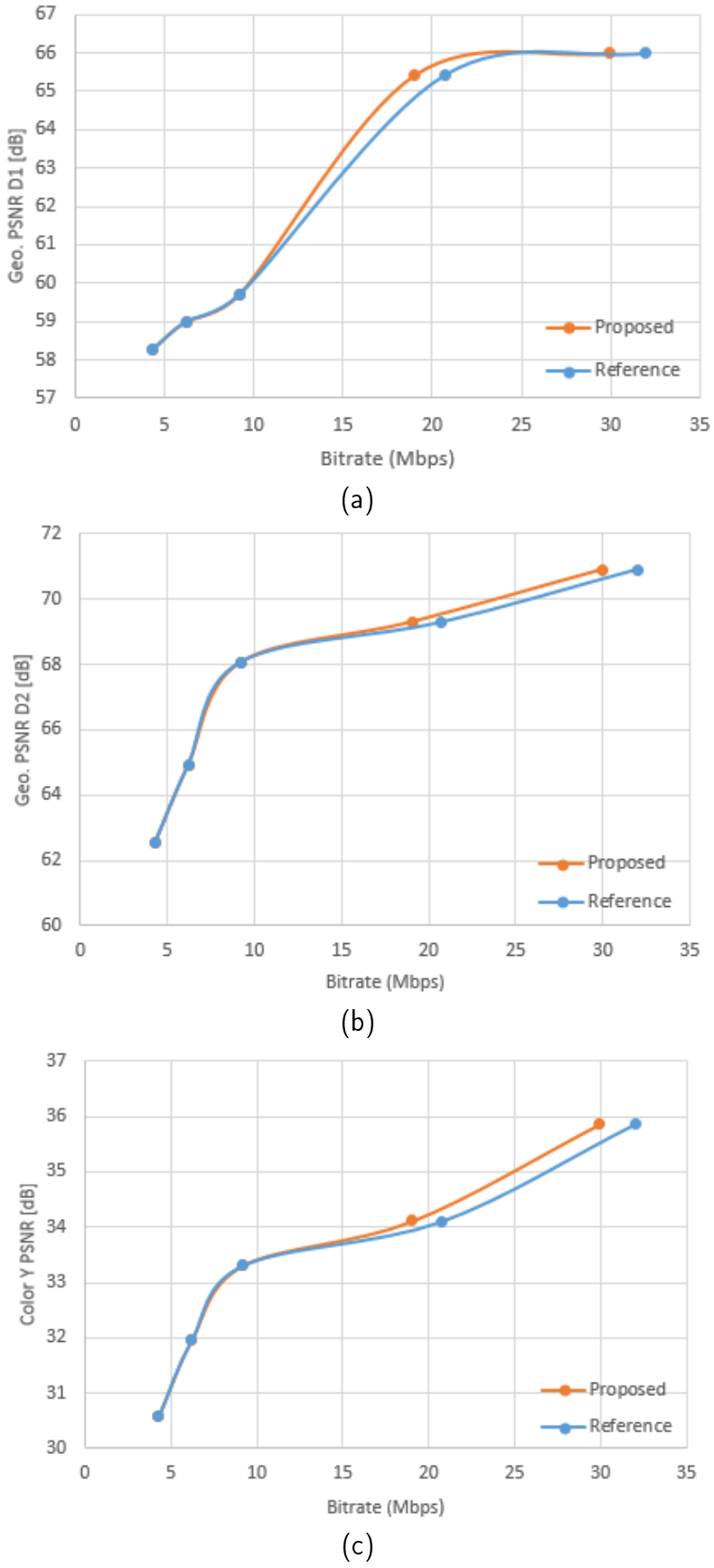


Figure 4.4 *Rate-Distortion curves for the Loot sequence compared to reference for AI configuration and applying patch refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

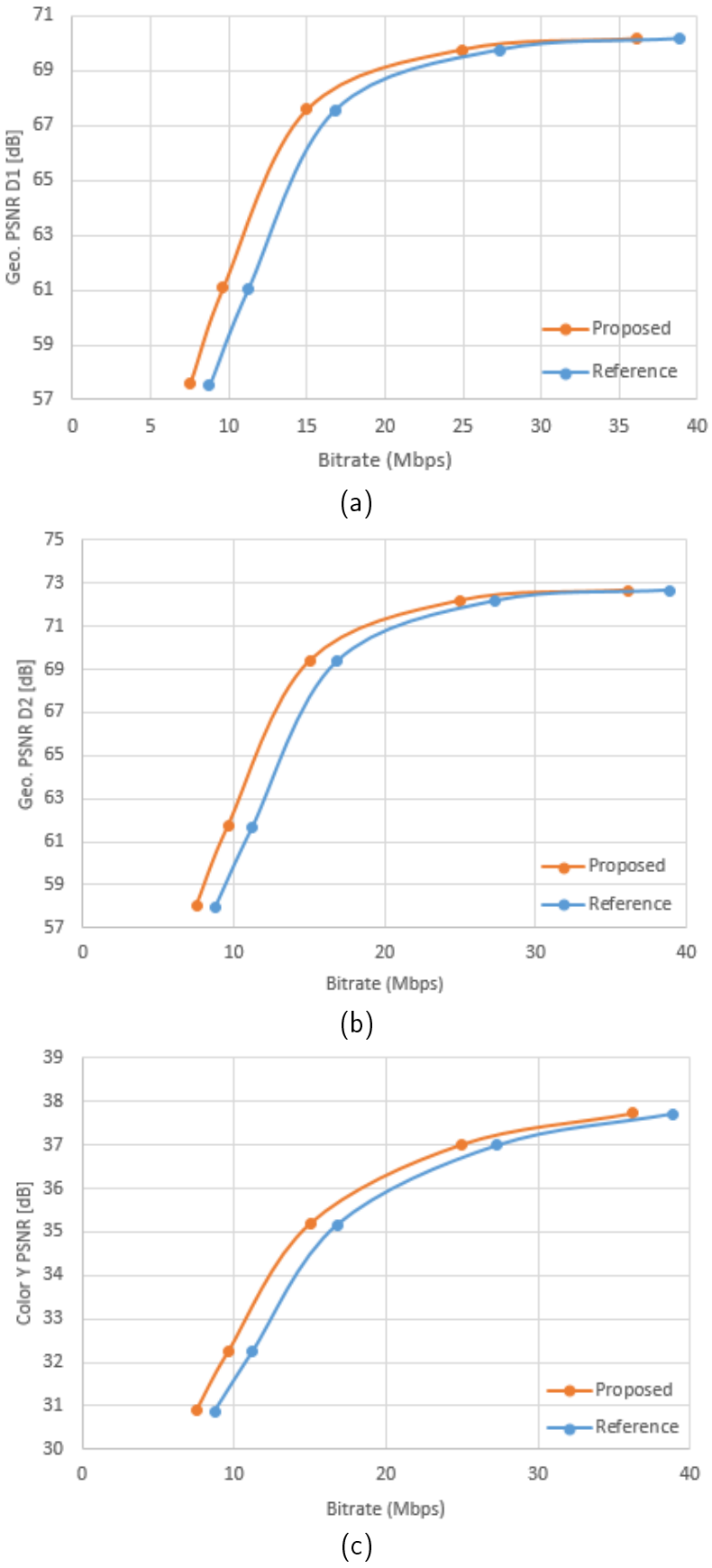


Figure 4.5 *Rate-Distortion curves for the Loot sequence compared to reference for RA configuration and applying patch refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

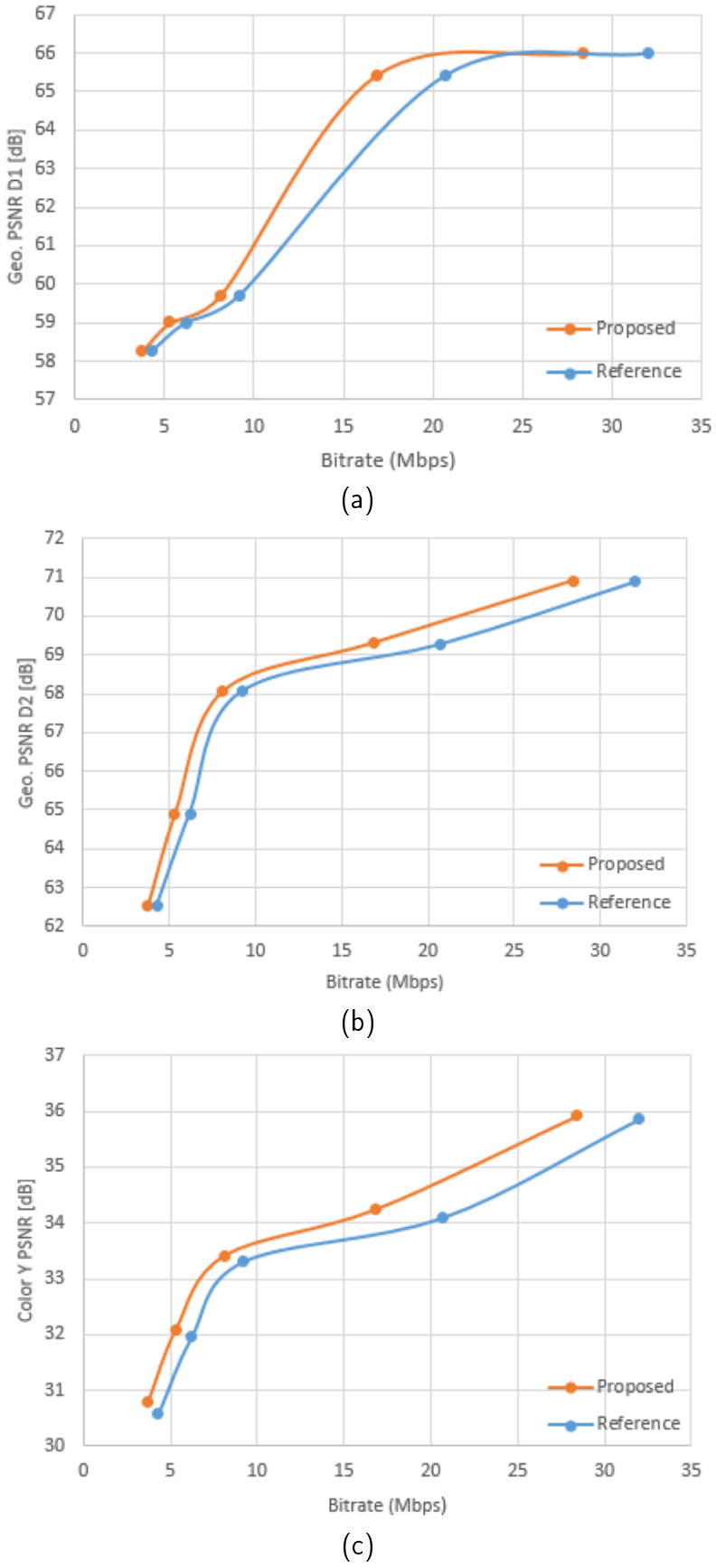


Figure 4.6 *Rate-Distortion curves for the Loot sequence compared to reference for AI configuration and applying edge smoothing patch refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*

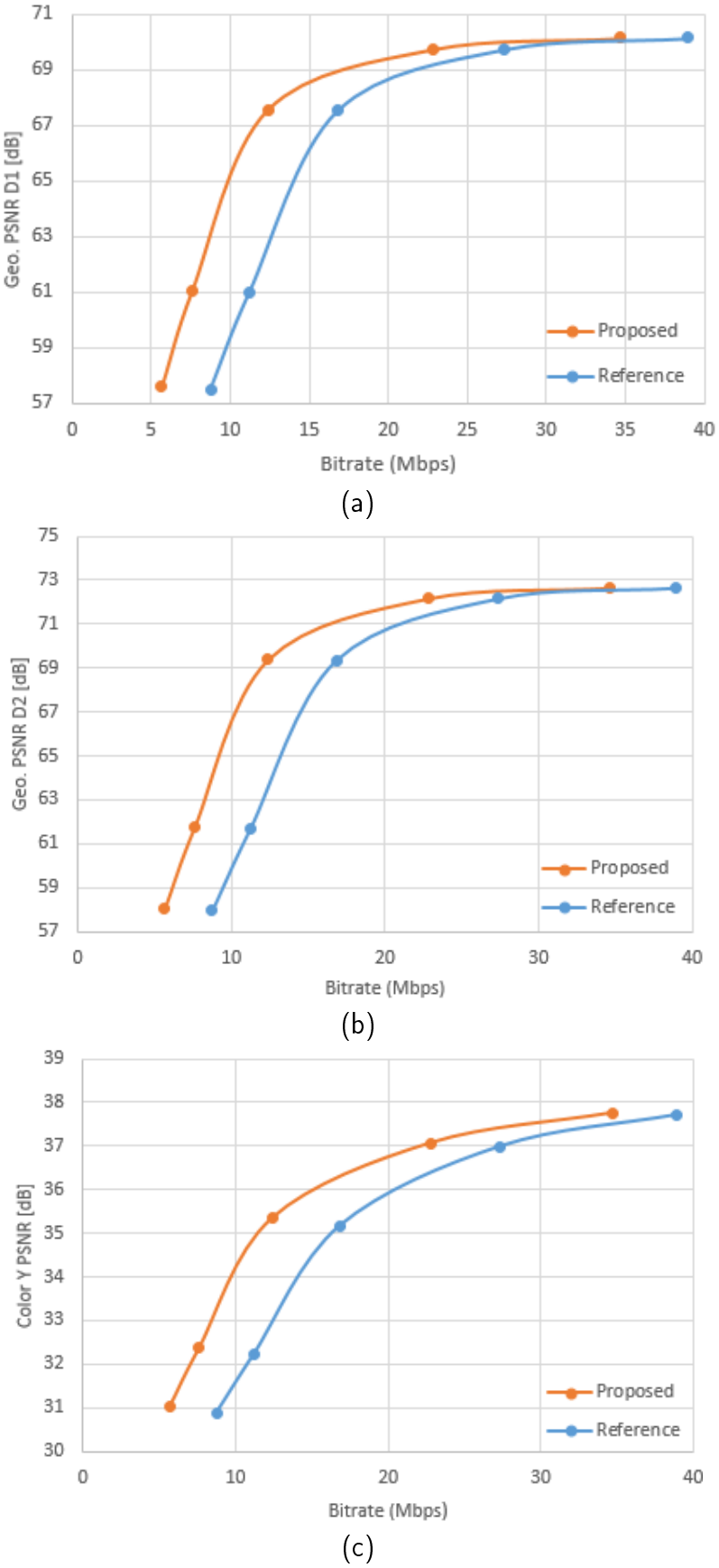
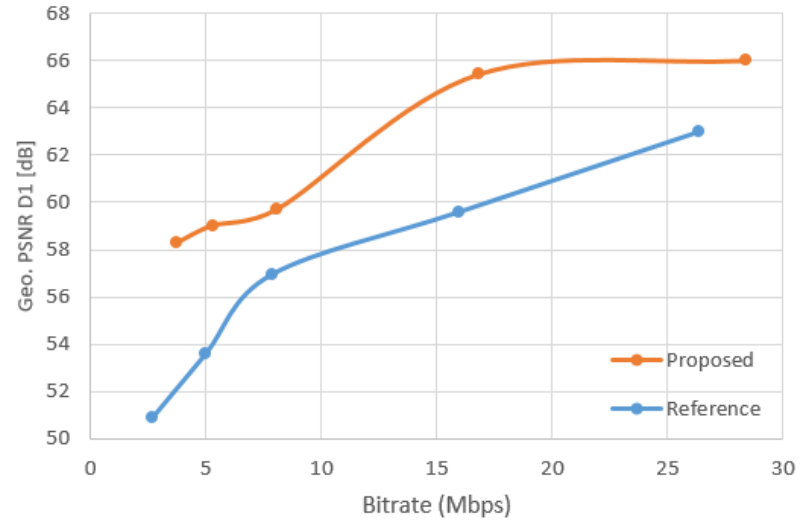
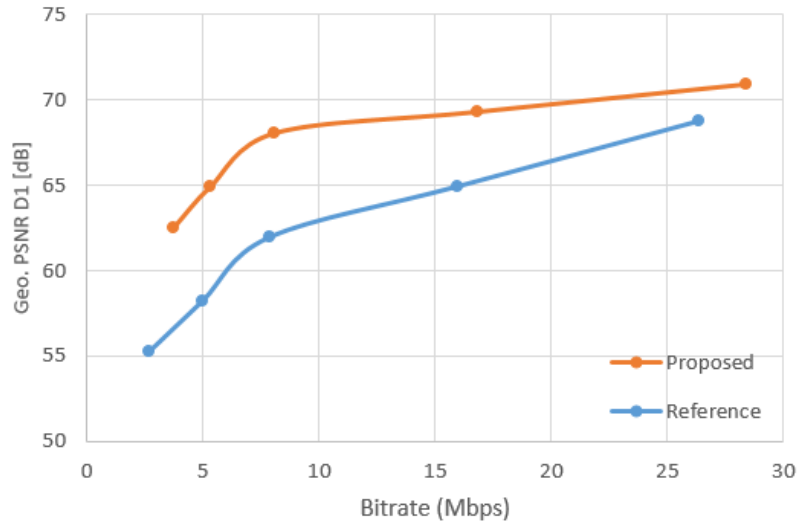


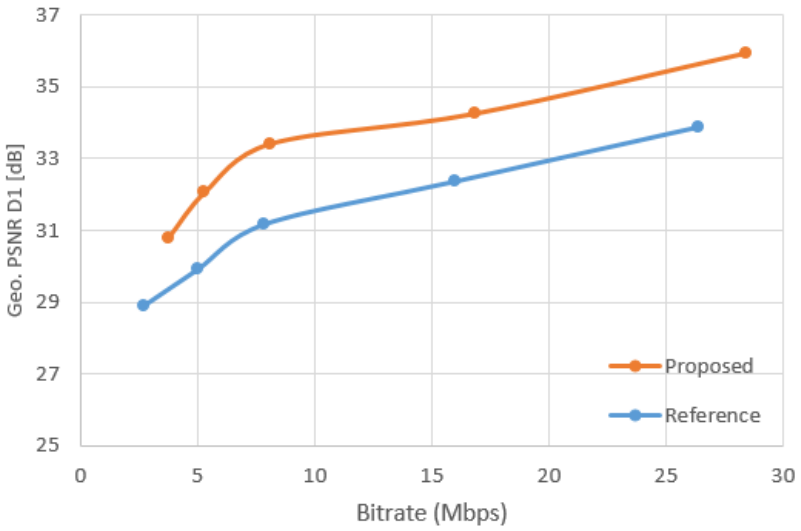
Figure 4.7 *Rate-Distortion curves for the Loot sequence compared to reference for RA configuration and applying edge smoothing patch refinement. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*



(a)

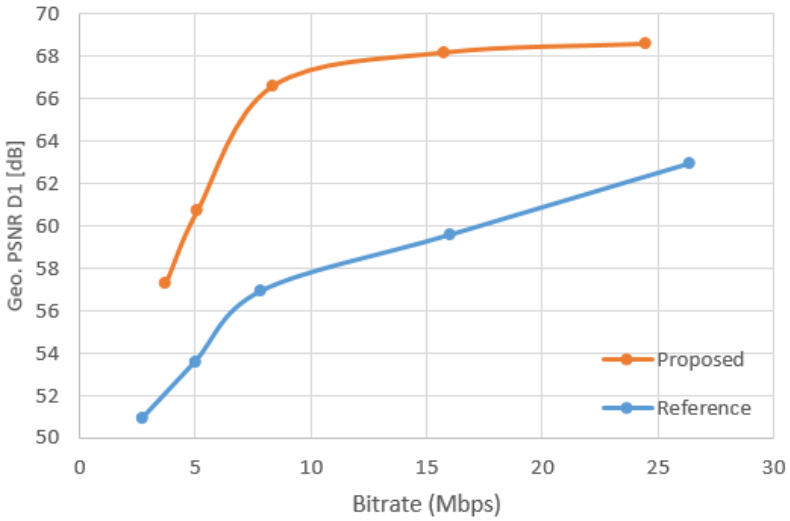


(b)

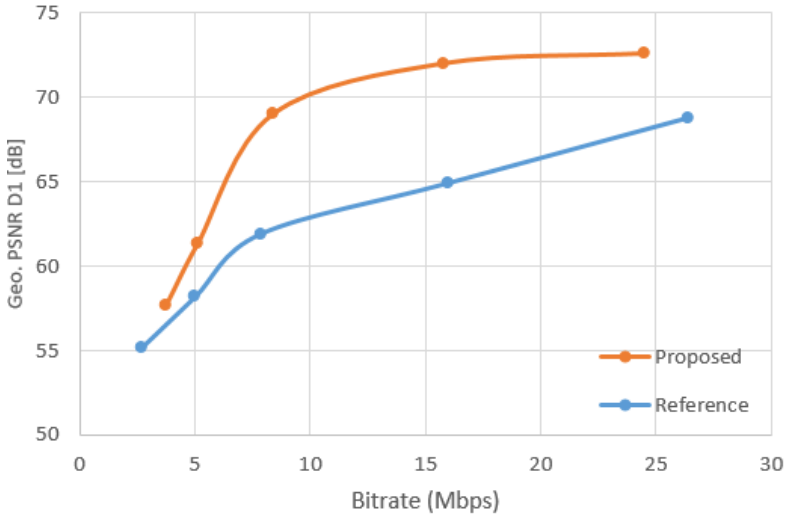


(c)

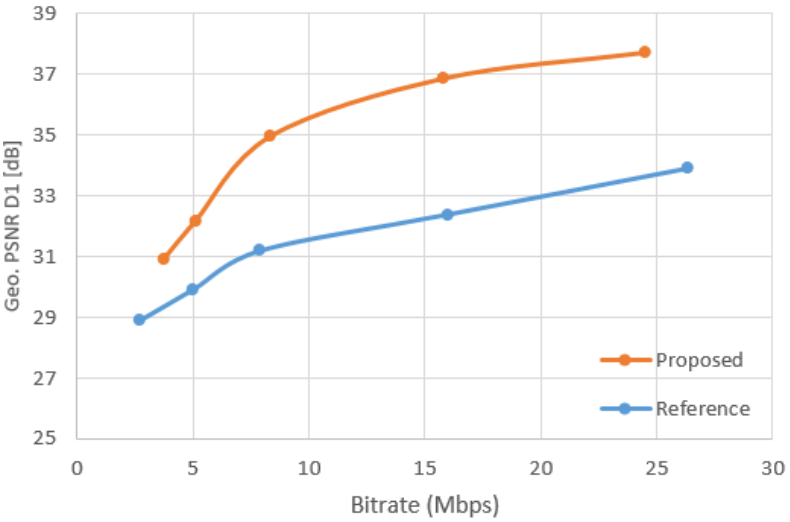
Figure 4.8 *Rate-Distortion curves for the Loot sequence compared to reference for AI configuration and projection-based approach. (a) D1 point-to-point , (b) D2 point-to-plane, and (c) luma(Y) PSNRs*



(a)



(b)



(c)

Figure 4.9 *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)

Figure 4.10 *Decoded point cloud in the 18 Mbit/s for (a) reference, (b) proposed solution*

5. CONCLUSION AND FUTURE WORK

what is point cloud? Due to the large number of points which generated to represent the object or scene the limitation for storage and transmission with the current network technology would arise, therefore the necessity of having efficient compression would help this technology to expand in all aspects. The compression can be lossless to lossy. Projection-based is a novel and efficient method to compress point cloud data. This thesis work highlighted the importance of addressing high frequency content in projection-based compression of volumetric data using 2D video coding technology. Two proposed methods improves the coding efficiency significantly. The proposed algorithms have been implemented and integrated as part of a contribution to the MPEG CfP on Point Cloud Compression [2] and objective evaluation proves the claim of improvement.

BIBLIOGRAPHY

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

[2] 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.

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

[4] 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.

[5] 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.

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

[7] D. Tian, H. Ochimizu, C. Feng, R. Cohen, and A. Vetro, "Geometric distortion metrics for point cloud compression," in *IEEE ICIP*, 2017.

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