

Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Thesis presented for Master's in ICT for Internet and Multimedia

No-reference point cloud quality assessment based on subjective and objective scores

Student:

Ahmed Omar Younus Shahhat

Supervisor:

Prof. Federica Battisti

Matricola:

2008561

Presented 27/02/2023

Contents

Contents	2
List of Figures	4
1 Introduction	9
1.1 Motivation	9
1.2 Challenges	9
2 Point Clouds	11
2.1 General View	11
2.2 Point Clouds Acquisition	12
2.3 Compression	13
2.3.1 Geometry-based point cloud coding	14
2.3.2 Video-based point cloud coding	16
2.4 Point Clouds Rendering	17
2.5 Point Clouds Use Cases	18
3 Quality	19
3.1 Quality of Experience	19
3.2 Quality of Experience for Point Clouds	20
3.3 Objective Metrics	21
3.3.1 Full-Reference Point cloud metrics	21
3.3.2 Geometric Full-Reference Point cloud metrics	21
3.3.3 Colour Full-Reference Point cloud metrics	24
3.3.4 Point Clouds objective quality metrics – Joint Geometry-Color	25
3.4 Point Clouds subjective quality	27
3.4.1 Data Analysis of the Subjective tests	29
4 Objective Metrics Evaluation	31
4.1 Method	32
4.2 Data Preparation	33
4.2.1 voxel Down-sampling	33
4.3 D1-MSE and D1-CMSE	34
5 Subjective Tests	38
5.1 Data Preparation	39
5.1.1 Meshing	39
5.1.2 Ball pivoting	40
5.2 Unity	42
5.3 The Tests	45
5.3.1 Tests Results	48
6 PointNet	52

6.1	PointNet Quality Classification	53
6.2	Results of The First training	55
6.3	Modified PointNet Loss Function	57
6.4	Model Carbon Impact	60
7	Conclusion	61
	References	63

List of Figures

1	Example of point clouds acquisition by LiDAR	12
2	Example of point clouds acquisition by photogrammetry	12
3	Comparing Photogrammetry with LiDAR for the same scene . . .	13
4	Example of the Octree representation	14
5	Example of the Mesh structure	14
6	Example of the Voxelization the process	15
7	Arco Valentino PC: left) original PC; and right) MPEG G-PCC decoded PC. Original texture used for recoloring.	16
8	Examples of different rendering results with different noises . . .	18
9	Example of a point cloud use case in the Automatic navigation systems	18
10	Illustration of the point-to-surface correspondence computation .	27
11	Example of a person doing a subjective test using Oculus 2 . . .	29
12	Examples of point clouds dataset	31
13	Example of a code to down sample point clouds	33
14	Examples of point clouds dataset	35
15	Results of the Objective Metrics Evaluation	36
16	Results of the Objective Metrics Evaluation	36
17	Results of the Objective Metrics Evaluation	37
18	Results of the Objective Metrics Evaluation	37
19	Results of the Objective Metrics Evaluation	37
20	Meta oculus 2	38
21	Before and after meshing for point cloud	40
22	MashLab Software	41
23	Blender Software	42
24	Unity Software	43
25	Unity Software	43
26	built-in support for Android development in Unity Software . .	44
27	SideQuest	44
28	Meta environment	45
29	Experiment start	45
30	Training Sample	46
31	Example of the experiment Samples	47
32	Evaluation screen	47
33	Results from the subjective tests	48
34	Results from the subjective tests	49
35	Results from the subjective tests	49
36	Results from the subjective tests	50
37	Results from the subjective tests	50
38	Results from the subjective tests	51
39	Results from the subjective tests	51
40	Results from the subjective tests	51
41	PointNet Model	53

42	PointNet Accuracy	55
43	PointNet Loss Function Graphs	56
44	PointNet Modified Loss Function	58
45	Modified PointNet Loss Function Graphs	58
46	Modified PointNet Accuracy Graphs	59
47	Absolute Quality Difference	60

List of Equations

1	Distance Calculation Point-to-Point Metric	22
2	D1-PSNR calculation	22
3	Error Calculation Point-to-Plan Metric	22
4	D2-PSNR calculation	23
5	PSNR for each colours components	24
6	PSNR for total colours components	24
7	P-PSNR for Joint Geometry-Color metric	25
8	MSE for Joint Geometry-Color metric	25
9	curvatures calculation for PCQM metric	26
10	minimization of the quadratic surface Q for PCQM	26
11	calculation of the quadratic surface Q	26
12	calculation of PCQM	26
1	calculation of MOS	30
2	standard deviation of MOS calculation	30
3	standard error of MOS calculation	30
4	ci calculation	30

Summary

Recently, thanks to the increasing capability of 3D acquisition devices, point cloud has emerged as the most popular format for immersive media. In practice, a variety of distortions could be involved and affect human perception. Developing point cloud quality assessment can help to understand the distortions and carry out the quality optimization for distorted point clouds. Therefore in this work we are investigating the quality assessment of point clouds and what are the main factors effecting the quality of the point clouds. And presented as well the existing subjective and objective methods for evaluation of the quality of point clouds and applying these methods to evaluate the point clouds quality.

After that, we made different experiments to apply the subjective and objective methods to estimate the quality of different data sets we have, And as well using the results of previous experiments made for farther investigations .

However, applying these methods for estimating the quality is quite challenging approach as we will see later in this study. Therefore, we created various solutions to overcome the challenges that we faced during this research .

At the end, we are investigating the using of the Neural Networks in the quality assessment of point clouds and how this can simplify the measurement of point clouds quality.

1 Introduction

1.1 Motivation

Point clouds are widely used in various fields such as computer graphics, robotics, and virtual reality. As a result, it is crucial to have effective methods to assess the quality of point clouds. Traditional point cloud quality assessment methods rely on reference models, which can be time-consuming and unrealistic in many real-world scenarios. Additionally, these methods are limited in their ability to accurately reflect the subjective perception of human viewers.

With the recent advancements in deep learning, there is an opportunity to improve the accuracy and efficiency of point cloud quality assessment. The combination of subjective and objective scores, along with deep learning, offers a more comprehensive and accurate evaluation of point cloud quality.

In summary, the motivation for this research lies in the need for a more comprehensive and accurate approach to point cloud quality assessment that considers both objective metrics and subjective perception. The integration of deep learning into this approach offers a valuable tool for evaluating the quality of point clouds in various applications, and provides a new level of accuracy and efficiency.

1.2 Challenges

The integration of subjective and objective scores and deep learning in point cloud quality assessment presents several challenges that must be overcome in order to achieve accurate and efficient results. Some of the challenges include:

Challenges with subjective tests:

Data collection: Collecting subjective scores for a large number of point clouds can be a time-consuming and resource-intensive process.

Data variability: Point clouds can vary widely in terms of their quality and subjective perception, which can make it difficult to obtain consistent subjective scores.

Model evaluation: Evaluating the accuracy of the subjective scores is a critical challenge, as the results will depend on the quality and quantity of the subjective data.

Challenges with objective metrics:

Data collection: Collecting objective metrics for a large number of point clouds can be a time-consuming and resource-intensive process

Data variability: Point clouds can vary widely in terms of their characteristics and objective metrics, which can make it difficult to obtain consistent objective scores.

Deep learning model development:

Integration of subjective and objective scores: Developing a deep learning model that effectively integrates both subjective and objective scores is a challenging task that requires careful consideration of the relationships between the two types of scores.

Model evaluation: Evaluating the accuracy of the deep learning model is a critical challenge, as the results will depend on the quality and quantity of the training data as well as the effectiveness of the deep learning model itself.

Generalization: The deep learning model must be capable of generalizing to new point clouds that have not been seen during training, in order to be useful in real-world scenarios.

In conclusion, these challenges must be addressed in order to achieve accurate and efficient results with the proposed approach to point cloud quality assessment based on subjective and objective scores and deep learning. However, with careful consideration and rigorous experimentation, these challenges can be overcome, and the proposed approach has the potential to provide a valuable tool for evaluating the quality of point clouds in various applications.

2 Point Clouds

2.1 General View

A point cloud is a set of data points in space. The points may represent a 3D shape or object. Each point position has its set of Cartesian coordinates (X, Y, Z) . The point cloud data set is composed of multiple points in a coordinate system by floating point values. 3D coordinates can be also converted into integer representation based a required spatial precision. While point clouds can be directly rendered and inspected, point clouds are often converted to polygon mesh or triangle mesh models, NURBS surface models, or CAD models through a process commonly referred to as surface reconstruction.

There are many techniques for converting a point cloud to a 3D surface. Some approaches, like Delaunay triangulation, alpha shapes, and ball pivoting, build a network of triangles over the existing vertices of the point cloud, while other approaches convert the point cloud into a volumetric distance field and reconstruct the implicit surface so defined through a marching cubes algorithm.

In geographic information systems, point clouds are one of the sources used to make digital elevation model of the terrain. They are also used to generate 3D models of urban environments. Drones are often used to collect a series of RGB images which can be later processed on a computer vision algorithm platform such as on AgiSoft Photoscan, Pix4D or DroneDeploy to create RGB point clouds from where distances and volumetric estimations can be made.

Point clouds can also be used to represent volumetric data, as is sometimes done in medical imaging. Using point clouds, multi-sampling and data compression can be achieved.[1]

However, point cloud data is the term used to refer to the data points collected for a given geographical area, terrain, building or space. A LiDAR point cloud dataset is created when an area is scanned using light detection and ranging.

2.2 Point Clouds Acquisition

PCs can be classified according to acquisition methods such direct and indirect. Direct acquisition they are defined as specially designed imaging systems to collect 3D information, either as a series of scattered points or a cloud of dense points where, in each pixel of the imaging sensor, a depth value is associated which can then be converted into a dense point cloud. This definition includes systems such as LiDARs(Light Detection and Ranging) and time-of-flight cameras.

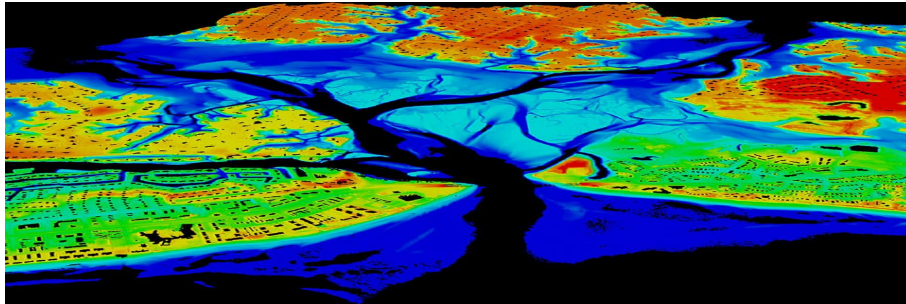


Figure 1: Example of point clouds acquisition by LiDAR

As for the indirect acquisition, methodology involves algorithms that do not directly measure 3D information. This can be exemplified by algorithms that generate PC information from a set of 2D images which match points according to the images of an object, taken at different angles (e.g. photogrammetry).[2]



Figure 2: Example of point clouds acquisition by photogrammetry



Figure 3: Comparing Photogrammetry with LiDAR for the same scene

2.3 Compression

Point clouds are 3D scenes and objects with given geometry characteristics (e.g., shape, size, position) and additional attributes (e.g., colour, reflectance and even temporal changes), commonly represented by volumetric visual data . These data can be either generated from computer-based 3D models, or captured from real-world settings using multiple cameras or especial ones, such as LIDARs. It should be noted that volumetric data are mostly represented in the form of polygon meshes or point clouds.[3]

MPEG started its point cloud compression (PCC) standardization with a Call for Proposal (CfP) in 2017. Three categories of point clouds were identified: category 1 for static point clouds, category 2 for dynamic point clouds, and category 3 for LiDAR sequences (dynamically acquired point clouds). Two technologies were finally defined: G-PCC (Geometry-based PCC, ISO/IEC 23090 part 9) for category 1 and category 3; and V-PCC (Video-based PCC, ISO/IEC 23090 part 5) for category 2. The first test models were developed in October 2017, one for G-PCC (TMC13) and another one for V-PCC (TMC2).[1] However, now we can divide the point clouds compression to tow main types, Geometry-based point cloud coding and Video-based point cloud coding .

2.3.1 Geometry-based point cloud coding

Before defining the G-PCC coding we defined some of its tools:

Octree : The octree representation partitions the three-dimensional spatial region of the point cloud dividing it recursively into octanes, according to a tree structure in which each node intern has eight sectors. It introduces compression if the points of an octane are represented by a smaller set of original dots and this is referred to as octree pruning.

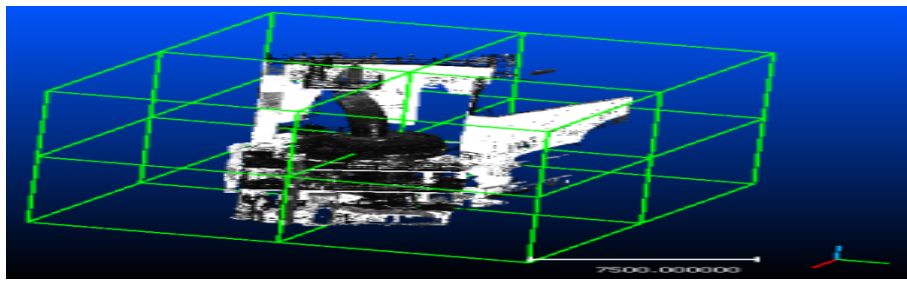


Figure 4: Example of the Octree representation

Mesh : A 3D mesh is the structural build of a 3D model consisting of polygons. 3D meshes use reference points in X, Y and Z axes to define shapes with height, width and depth. While it can take large numbers of polygons to make a 3D mesh approach photorealism, these relatively simple shapes allow for faster processing than other techniques, like NURBS, that produce smooth curves. The polygons used are typically quadrangles or triangles; these geometric shapes can be further broken down into vertices in X, Y, Z coordinates and lines.

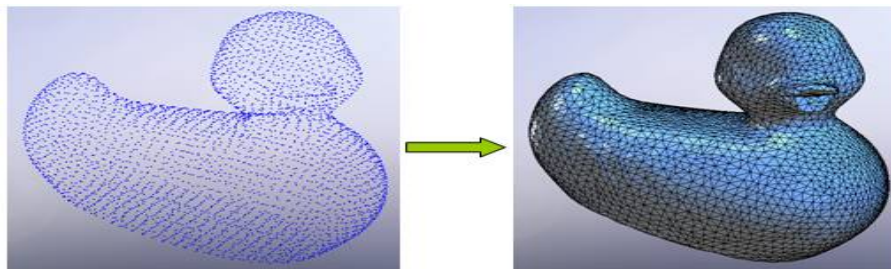


Figure 5: Example of the Mesh structure

Voxelization: Voxelization is the process of converting a data structures that store geometric information in a continuous domain (such as a 3D triangular mesh) into a rasterized image.

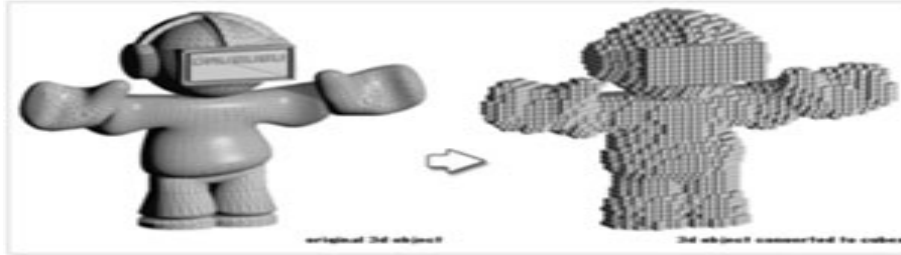


Figure 6: Example of the Voxelization the process

Surface approximation via trisoup: The geometry may be represented by a pruned octree, constructed from the root to an arbitrary level where the leaves represent occupied sub-blocks that are larger than a voxel. The object surface is approximated by a series of triangles, and since there is no connectivity information that relates the multiple triangles, the technique is called “triangle soup” (or trisoup). It is an optional coding tool that improves the subjective quality in lower bitrate as the quantization gives the rough rate adaptation. If trisoup is enabled, the geometry bitstream becomes a combination of octree, segment indicator, and vertex position information.[6]

G-PCC: In the G-PCC geometry coder, points are first transformed and voxelized into an axis-aligned bounding box before geometry analysis using trisoup or octree scheme. In the trisoup coder, geometry can be represented by a pruned octree plus a surface model. This model approximates the surface in each leaf of the pruned octree using 1 to 10 triangles. In contrast, the octree coder partitions voxelized blocks until sub-cubes of dimension one are reached. First, the coordinates of isolated points are independently encoded to avoid “polluting” the octree coding (Direct Coding Mode - DCM). To encode the occupancy pattern of each octree node, G-PCC introduces many methods to exploit local geometry information and obtain an accurate context for arithmetic coding, such as Neighbour-Dependent Entropy Context, intra prediction, planar/angular coding mode, etc. The lossless geometry coding mode of G-PCC is based on octree coding only.[3]



Figure 7: Arco Valentino PC: left) original PC; and right) MPEG G-PCC decoded PC. Original texture used for recoloring.

2.3.2 Video-based point cloud coding

It is based on the generation of 3D surface segments by dividing the point cloud into a number of connected regions, which we call 3D patches. Each 3D patch is independently projected in a 2D patch. Orthographic projections are used to avoid re-sampling problems and allow lossless compression. It is equivalent to having several acquisitions via camera e they are combined as in a mosaic. This is then done through the methods compression for 2D images and videos. Several associated projections were evaluated with several compressions. In panoramic images mapped from the point cloud are used and then JPEG, JPEG-2000 and PNG are used. In a hybrid approach is used: the map of depth is compressed with a lossless compression method, that of the attribute of color through JPEG.[4]

V-PCC is suitable for point clouds with an approximately uniform distribution of points and is considered to be the most efficient compared to other existing alternatives. If the clouds are distributed in a more sparse way, a geometry-based class, G-PCC, is commonly employed.[6]

This approach is derived from the combination of LIDAR point cloud compression (L-PCC) for dynamically acquired data and Surface point cloud compression (S-PCC) for static point cloud data, due to their similarities.

G-PCC approach includes decomposition of the 3D space into a cubical hierarchical structure and assigning every point to a corresponding index of the cube. While V-PCC coding method focused on 3D to 2D projections. G-PCC, in contrast, compresses the content of 3D space in a direct way. For this purpose G-PCC employs the octree pruning method which has been described previously.

It should be noted that there is no initial assumption on the input point cloud coordinates, but there is internal integer-based value, converted from a floating point value.[5]

Another important feature of G-PCC is characterized by tiles and slices, composed of a set of points (geometry and attributes) that enable a parallel coding. Similar to V-PCC, G-PCC method is limited with lack of temporal prediction tool.

2.4 Point Clouds Rendering

PC rendering is the process of producing a visual representation that can be consumed by users using an available display, e.g. conventional 2D, stereo, auto-stereoscopic, headmounted displays, etc. Since it effectively selects the information to be seen, the rendering process has a significant impact on the quality perceived by the user.

There are two main approaches; the first, directly uses the PC data (point-based) while the second converts the PC data into another representation format, very commonly a surface, e.g. a polygonal mesh. The decision on the rendering approach to adopt mostly depends on the application requirements which may be very different.

The PC conversion to another representation format more rendering friendly may bring some information loss and, in some cases, it may not even be possible due to the complexity of the visual scene in terms of geometry or the low PC density. By directly rendering PCs, massive amounts of points can be visualized. Rather often, these PCs do not fit into the available memory and require special algorithms to stream, process and render only a small subset of the entire PC data. This is easier to perform with a point-based model due to the lower complexity associated to the rendering process in comparison to a polygonal mesh representation where surface reconstruction and interpolation are usually needed.[7]

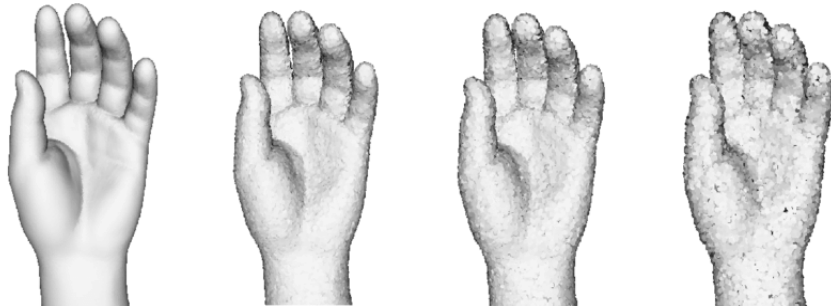


Figure 8: Examples of different rendering results with different noises

2.5 Point Clouds Use Cases

Point Clouds are very versatile and have many uses as the following examples [8] :

- Virtual, augmented and mixed reality.
- 3D content creation.
- Medical Applications.
- Construction and manufacturing.
- Consumer and retail.
- Cultural heritage preservation.
- Remote sensing and geographical information systems.
- Automatic navigation systems.
- Surveillance.

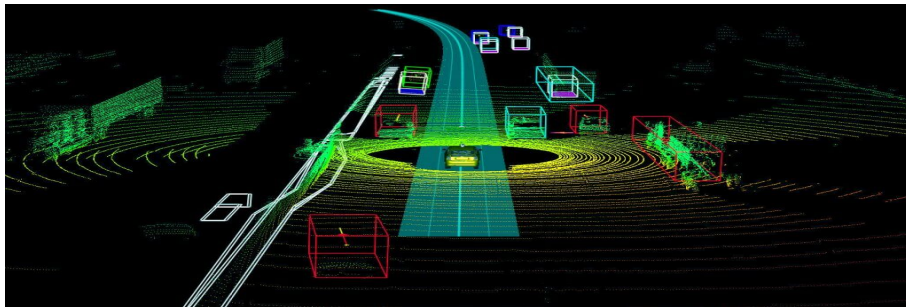


Figure 9: Example of a point cloud use case in the Automatic navigation systems

3 Quality

The availability of 3D range scanners and RGB-D cameras is pushing the spreading of point cloud-based applications. One of the main issues of this technology, in applications where the end user is a human observer, is the presentation of the data. Three-dimensional visual information represented as point clouds can be displayed in several ways, e.g. as sets of points with varying point size or as a surface rendered using one of several available methods.

Therefore it is very important to understand the user preference of visualization of point clouds in terms of different rendering devices and methods and the quality of experience for the user.

3.1 Quality of Experience

Before talking about the quality of experience lets see some useful definitions:

Event: An observable occurrence. An event is determined in space (i.e. where it occurs), time (i.e. when it occurs), and character (i.e. what can be observed).

Perception: on the other hand, is a process which involves the recognition and interpretation of stimuli which register our senses.

Experience: An experience is an individual's stream of perception and interpretation of one or multiple events

Quality: Is the outcome of an individual's comparison and judgment process. It includes perception, reflection about the perception, and the description of the outcome. In contrast to definitions which see quality as "qualities", i.e. a set of inherent characteristics, we consider quality in terms of the evaluated excellence or goodness, of the degree of need fulfillment, and in terms of a "quality event"

Influence Factor: Any characteristic of a user, system, service, application, or context whose actual state or setting may have influence on the Quality of Experience for the user.

Fundamental for these definitions is the understanding of both terms quality and experience from an individual's point of view. Thus, in contrast to performance they cannot be solely described by only physical properties or the achievement of a certain objective goal (e.g. intelligibility).

Application: A software and/or hardware that enables usage and interaction by a user for a given purpose. Such purpose may include entertainment or information retrieval, or other.

Service: An episode in which an entity takes the responsibility that something desirable happens on the behalf of another entity.

Quality of Experience (QoE): is the degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations with respect to the utility and / or enjoyment of the application or service in the light of the user's personality and current state.[9]

3.2 Quality of Experience for Point Clouds

As we previously saw that the point clouds are used in applications where the end user is a human observer. Therefore, it is very important to understand the user preference of visualization of point clouds. And for this the subjective tests are the most effective way to understand the quality of the point clouds.

In the other hand the subjective test is very expensive in terms of time and for this it is very common to use the objective test as well for measuring the point clouds quality.

However, the quality of the point clouds is very effected by compression and rendering and for this reason most of the studies of the point clouds quality are focusing on the effects of the compression and rendering on the point clouds quality.

In this work we will investigate these quality metrics rather than the compression and rendering effects on point clouds quality and what are the challenges on applying these methods for the determination of the quality .

Previous investigations showed that the most important criteria in the PCs selection include the following i) the PC density, ii) the semantic content (e.g.PCs from Buildings and People), iii) the PC geometry attributes (e.g.PCs with holes), and iv) the colour characteristics. Apart from these characteristics, during acquisition step, some original point clouds can contain significant amount of noise. Thus, it was concluded that noise and density are essential factors involved in the subjective variables given by users.[10]

3.3 Objective Metrics

Objective quality metrics refer to algorithms that computationally predict the visual quality of distorted stimuli. They are easily operated, yet, their performance depends on their ability to assess distortions in perceptual terms. The prediction accuracy of objective quality metrics is validated through benchmarking and, specifically, after comparison with subjective ground truth.[11]

Objective metrics have shown relevant results in the performance assessment of compression techniques for volumetric data, although these metrics cannot explain how the user visually perceives point cloud objects from a given perspective.

Objective quality metrics can be classified based on their requirement for the original content (i.e., reference) at execution time as full-reference, reduced-reference, and no-reference. Point cloud metrics can additionally be categorized as projection-based and point-based.[11]

3.3.1 Full-Reference Point cloud metrics

In these metrics the presence of the reference content is necessary, due to the fact that the distorted content should be compared with the original to calculate the degradation. In the state of the art, objective metrics can be divided into three categories based on the type of degradation they are able to capture: geometric degradations, textures or both.

The full-reference point cloud metrics are the most used type of metrics for the objective tests for point clouds quality tests, and we can divide this type of metrics as following :

3.3.2 Geometric Full-Reference Point cloud metrics

These are the metrics that use only geometry to evaluate the point cloud quality and they can be divided in four different types as following :

• **Point-to-point (D1)**: the geometric distance of the associated points between the reference content and the content under evaluation is evaluated. We can define that for any point b_k of the content under evaluation B there is a point a_k of the reference content A through the nearest neighbor algorithm. The distance error is then evaluated Euclidean through the following equation :

$$E(a_i, b_k) = \|v_{a_i}^{b_k}\|^2 \quad (1)$$

where $v_{a_i}^{b_k}$ represents the vector starting from point a_i and ending in b_k .

In practice it is popular to use Mean Square Error (MSE) or peak signal-to-noise ratio ($PSNR$) to measure the point to point objective metric. However, for Point to point (D1) : geometric distance between point cloud A and point cloud B $D1 - MSE$

• For every point in A , find closest point in B and compute $d_j = distance(A_j, B_{nearst}(A_j))$.

• Compute average of squared d_j i.e. $MSE(A, B)$.

• Switch A and B and repeat top get $MSE(B, A)$.

• $D1 - MSE$ is maximum($MSE(A, B), MSE(B, A)$).

And for $D1 - PSNR$ we have :

• Compute $MSE(A, B)$ and $MSE(B, A)$ as above.

• Compute $D1 - PSNR$ as following :

$$D1 - PSNR = 10 \log \left(\frac{3 \cdot peak^2}{\max(MSE(A, B), MSE(B, A))} \right) \quad (2)$$

where $peak = 2^b - 1$ and b is the voxelization precision in bits. [12]

• **Point-to-plane (D2)** : the error is projected along the normal of a reference point. Given b_k I can identify a_i using the nearest neighbor algorithm. Then calculate the error along the normal n_{a_i} as following:

$$\dot{E}(a_i, b_k) = v_{a_i}^{b_k} \cdot n_{a_i} \quad (3)$$

In practice we have that for the calculation of Point to plane (D2) geometric distance between point cloud A and point cloud B we can use one of the following ways :

D2 – MSE :

- For every point in A, find closest point in B.
- Fit plane tangent to B surface at point $B_{nearst(A_j)}$
- Find normal to plane and project vector $A_j \rightarrow B_{nearst(A_j)}$ onto normal.
- Compute projection length p_j .
- Compute average of squared p_j i.e. $PMSE(A, B)$.
- Switch A and B and repeat top get $PMSE(B, A)$.
- $D2MSE$ is maximum($PMSE(A, B), PMSE(B, A)$).

D2 – PSNR:

- Compute $PMSE(A, B)$ and $PMSE(B, A)$ as above.
- Compute $D2 – PSNR$ as following :

$$D2 – PSNR = 10 \log \left(\frac{3 \cdot peak^2}{\max(MSE(A, B), MSE(B, A))} \right) \quad (4)$$

where $peak = 2^b - 1$ and b is the voxelization precision in bits. [12]

- **Plane-to-plane**: based on the angular similarity of the tangent planes that match at the points associated between the content under evaluation and the reference content. Given b_k and a_i , solved by the nearest neighbor algorithm, we obtain the normals n_{a_i} e n_{b_k} . We defined the angular similarity of the tangent plane as $1 - \frac{2\sigma}{\pi}$ where σ is the angle smaller of the two angles formed by the intersection of the tangent planes.

- **Point-to-mesh**: based on the projected distances from the reference point cloud to the content to evaluate, defined on mesh.

In the last tow objective quality metric that has been mentioned so far, a single error is associated to each point that belongs to the content under consideration. In order to obtain a value of total degradation, the Root Mean Squared Error ($RMSE$), the Mean Squared Error (MSE), the Hausdor distance, or even a simple average of the individual values, even in weighted form.

Referencing the distorted point cloud instead the original, different pairs of associated points are obtained and, therefore, different objective scores. Therefore, it is very common to use both contents as a reference, calculate both the errors, selecting the maximum value.

3.3.3 Colour Full-Reference Point cloud metrics

For each point b_k of the content under evaluation B , a point a_k is identified which belongs to it to the reference point cloud A through the nearest neighbor algorithm.

After formed each pair of associated points, the color attributes are used to calculate a total color degradation value via standard formulas such as $PSNR$.

In practice we can adapt $D1$ to compute a color-only point cloud quality measure.

Assuming there are three color components, e.g. Y, Cb, Cr , an average per component MSE ($CMSE$) and $PSNR$ are computed as follows:

- For every point in A , find spatially closest point in B and compute the color component distance

$$cd_j = distance(ColorComp(A_j), ColorComp(B_{nearest(A_j)})).$$

- Compute average of squared cd_j i.e. $CMSE(A, B)$.
- Switch A and B and repeat top get $CMSE(B, A)$.
- $CMSE$ is maximum($MSE(A, B), MSE(B, A)$).

afterwards a logarithmic measure, $PSNR$, is computed for each color component as :

$$PSNR = 10 \log \left(\frac{p^2}{CMSE} \right) \quad (5)$$

where p depends on the component bit depth $b, p = 2^b - 1$.

Finally, the per component $PSNRs$ are weighted as in the following:

$$PSNR_{colour} = 10 \log \left(\frac{6PSNR_y + PSNR_{Cb} + PSNR_{Cr}}{8} \right) \quad (6)$$

3.3.4 Point Clouds objective quality metrics – Joint Geometry-Color

An approach to calculate the MSE and $PSNR$ for the point clouds objective quality metrics-Joint Geometry-Color is based on voxelized, projected point clouds orthographically on 2D planes. Notably, the point cloud is initially voxelized at a specified voxel grid depth. The resulting voxelized versions are then projected orthographically onto 2D image planes. Whereas 3D interactive content can be viewed from an infinite number of points of view, it is possible to get an infinite number of projected planes. As this can be considered both impractical unwieldy, a subset of these viewpoints are naturally employed.

After getting the projections, typical image metrics are used 2D. We define P_A and P_B as the projected images of two point clouds A and B from the same direction d . Considering the 8-bit color attributes, the projected $PSNR$ value, called $P-PSNR$ is calculated as following:

$$P-PSNR = 10 \log \left(\frac{255^2}{MSE(P_A; P_B)} \right) \quad (7)$$

Where $MSE(P_A; P_B)$ is calculated as :

$$MSE(P_A; P_B) = \frac{1}{N \cdot M} \sum_{n=1}^N \sum_{m=1}^M [P_A(n, m) - P_B(n, m)]^2 \quad (8)$$

with N and M the dimensions of the projections.

After calculating $P-PSNR_d$ for each selected viewpoint d , you can use a average or a weighted average to provide a single objective score as a prediction subjective visual quality.

Some measures have been introduced recently to measure the quality of the Geometry and Color of point clouds like PCQM [13] and Point SSIM.[14]

PCQM:

Local features curvatures ρ_p and $\dot{\rho}_p$ are computed according to :

$$\rho = \frac{(1 + d^2)a + (1 + e^2)b + 4abc}{(1 + e^2 + d^2)^{\frac{3}{2}}} \quad (9)$$

Where a, b, c, d and e can be obtained from the minimization of the quadratic surface Q respect to the z coordinate that is calculated as following :

$$\sum_i ||z_i - Q(x_i, y_i)||^2 \quad (10)$$

Where $Q(x_i, y_i)$ is obtained from the following equation:

$$Q = ax^2 + by^2 + cxy + dx + ey + f \quad (11)$$

Then set of local features are computed relating point p and point \dot{p} properties.

For the first three features based on curvature values, the values μ_p^ρ and $\mu_{\dot{p}}^\rho$ represent the gaussian-weighted averages of the curvature on the set of neighbours $N(p, h)$ and $N(\dot{p}, h/2)$, respectively.

While σ_p^ρ , $\sigma_{\dot{p}}^\rho$ and $\sigma_{p\dot{p}}^\rho$ are the standard deviations and covariance over these neighborhoods.

Using these parameters and the statistical variables for lightness and chroma we can calculate the variables from f_1 to f_7 while f_8 uses the gaussian-weighted average of this variable at the neighbourhood $N(p, h)$.

f_4 to f_6 use these same statistical variables for lightness and chroma, and f_3 uses the curvature structure.[13]

The features are normalized and the quality is computed as

$$PCQM = 0.18f_3 + 0.44f_4 + 0.38f_6 \quad (12)$$

However, for point SSIM has some stability problems with sparse point clouds, namely lack of monotonicity with compression rates and inconsistent agreement with subjective scores.

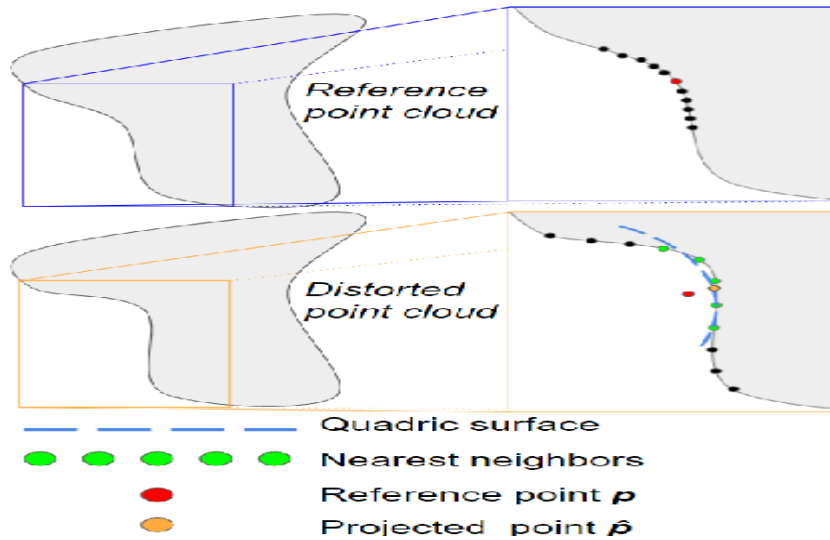


Figure 10: Illustration of the point-to-surface correspondence computation

3.4 Point Clouds subjective quality

There are common methodologies employed in the subjective assessment of the quality of images. These image assessment methods are using measurements derived from the direct reactions of users who view the images under the test. Objective metrics might not always completely describe performance of the tested systems; subsequently, it is necessary to conduct both objective tests together with subjective tests.

In a typical subjective quality assessment experiment, subjects look at a series of original and degraded stimuli and rate their quality on a numerical scale. The subjective quality is often expressed as a Mean Opinion Score (MOS) which represents the degree of quality attributed from a standard average observer to a given stimulus.

MOS degrees are collected following well-defined methods aiming to ensure identical experimental settings and conditions, facilitating the reproducibility of an experiment.

In Recommendation ITU-R BT.500-13 [15], some of the most popular and widely adopted methods for subjective quality assessment in 2D imaging are described, while for stereoscopic 3DTV systems the ITU-R Recommendation BT.2021-1 [16].

A common single stimulus subjective assessment method is Absolute Category Rating (ACR). In this technique each point cloud is assessed individually based on the ACR scale.

The labels assigned on each scale are as follows: "bad", "poor", "fair", "good", and "excellent". For calculation of MOS values, these labels are converted into the values 1, 2, 3, 4 and 5.

ACR method may also include a hidden reference, which is original unimpaired sequence to be presented along with the impaired sequences, without awareness of the subjects of its presence. Results in this case are based on ratings derived from differential scores between the reference and the impaired types [15].

Another class of the commonly used methodologies in subjective tests refer to DSIS (Double Stimulus Impairment Scale) or, in other words, DCR (Degradation Category Rating). This technique is characterized by observers who first see an unimpaired reference video, then the same video, but in impaired version, and subsequently, they vote on the second video according to the 'impairment scale' (i.e., from "impairments are imperceptible" to "impairments are very annoying").

The double-stimulus method is considered to be cyclic as the viewer is first presented with an unimpaired reference, then with the same image impaired [15].

In [18] the results obtained using two different methodologies were compared subjective testing (ACR and DSIS) and several objective metrics, with two different biases (Gaussian noise and octree-pruning).

The ACR and DSIS methodologies resulted statistically equivalent. Comparison of DSIS and ACR methodology shows that in the first case the subjects tend to classify content better in the presence of Gaussian noise.

The methodology DSIS is more consistent in terms of identifying the level of deterioration.

However, subjects tend to judge based on relative differences (i.e., geometric distances or number of points) and not explicitly based on visual quality; however, the latter is obtained through the ACR methodology.



Figure 11: Example of a person doing a subjective test using Oculus 2

3.4.1 Data Analysis of the Subjective tests

In a subjective experiment, large quantities of data. This data must be condensed by statistical techniques to produce results in graphical and numerical form that summarize the performance of the systems under test.

The most important statistic measure is the Mean Opinion Score MOS which, as previously mentioned, represents the degree of quality attributed by an average standard observer to a point clouds stimulus.

$$MOS = \frac{1}{N} \sum_{i=1}^N score_{i,j,k,r} \quad \forall j, k, r \quad (1)$$

Where N are the participants, j are the stimuli, k are the various conditions and r are repetitions. The mean scores are accompanied by a confidence interval that comes from the standard deviation and size of each sample. The standard deviation is defined as follows:

$$\sigma = \sqrt{\frac{1}{M} \sum_{i=1}^M (score - m)^2} \quad (2)$$

with M = number of samples, m = average (MOS); while the standard error is :

$$SE = \frac{\sigma}{\sqrt{M}} \quad (3)$$

A 95% confidence interval is proposed as a standard in the literature is calculated through the ci value derived from SE :

$$ci = 1.96 \cdot SE \quad (4)$$

which translates for the MOS scores:

$$[MOS - ci, MOS + ci]$$

4 Objective Metrics Evaluation

Objective metrics are measurements or evaluations that are based on objective criteria, rather than subjective opinions or judgments. They are often used in scientific research and engineering to evaluate the performance of different algorithms or techniques, as well as to compare the results of different studies.

In the context of point clouds, objective metrics might be used to evaluate the accuracy or precision of different techniques for aligning, registering, or segmenting point clouds, or to compare the quality of different point cloud datasets. The specific metrics used will depend on the specific goals and objectives of the research or application, and may include measures such as mean squared error, mutual information, or other statistical or functional measures.

Objective metrics evaluation of point clouds can be an important tool for determining the quality and suitability of a point cloud dataset for a given application

The dataset used for this evaluation is from the paper [19] where they are raw data in *ply* format which is the Stanford format, and they are different point clouds shapes for example:

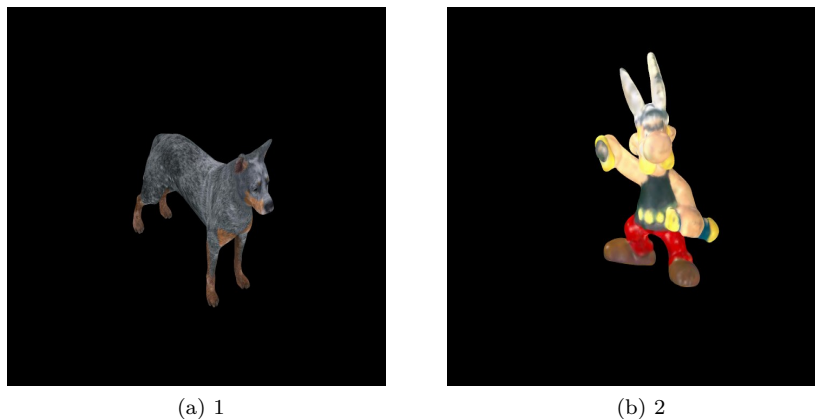


Figure 12: Examples of point clouds dataset

4.1 Method

The methods we used for the objective metrics evaluation are the Full-Reference Point cloud metrics.

Full-reference point cloud metrics are evaluation methods that compare a target point cloud dataset to a reference point cloud dataset in order to assess the quality or accuracy of the target dataset. These metrics are often used when a ground truth or reference dataset is available, and are designed to provide an objective and unbiased measure of the similarity between the two datasets.

Full-reference point cloud metrics can be an important tool for evaluating the quality and accuracy of a point cloud dataset, particularly when a reference dataset is available for comparison.

There are a few reasons why full-reference point cloud objective metrics may be preferred over non full-reference metrics:

Unbiased evaluation: Full-reference point cloud metrics compare a target dataset to a reference dataset, which can provide an objective and unbiased measure of the quality or accuracy of the target dataset. Non full-reference metrics, on the other hand, may be subject to bias or subjectivity, as they do not have a reference dataset against which to compare the target dataset.

Ground truth comparison: Full-reference point cloud metrics allow for comparison to a known ground truth or reference dataset, which can be especially useful in applications where high levels of accuracy are required. Non full-reference metrics do not have this capability, as they rely on other criteria to assess the quality of the target dataset.

Comparison to real-world data: Full-reference point cloud metrics can be used to compare a target dataset to real-world data, which can be especially useful in fields such as surveying or construction where the accuracy of the data is critical. Non full-reference metrics may not have this capability, as they rely on other criteria to assess the quality of the target dataset.

Overall, full-reference point cloud metrics can provide a more objective and unbiased evaluation of a point cloud dataset, particularly when a reference dataset is available for comparison.

4.2 Data Preparation

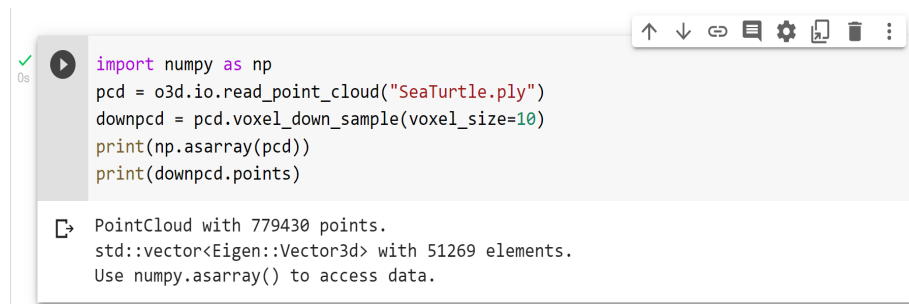
For full-reference point cloud metrics we need target point cloud dataset to compare it to the reference point cloud dataset that we have from [19], And to do that we proceed with the creation of the target dataset by voxel down-sampling the reference dataset that we have.

4.2.1 voxel Down-sampling

Voxel down-sampling is a method for reducing the number of points in a point cloud dataset by representing the points as small cubic units, or voxels. This can be useful for reducing the computational cost of processing large point cloud datasets, as well as for simplifying the overall structure of the point cloud.

In the Python library Open3D, voxel down-sampling can be performed using the voxel down sample() function. This function takes as input a point cloud object and a voxel size, and returns a new point cloud object with the points represented as voxels.

For example, the following code demonstrates how to use the voxel down sample() function to down-sample a point cloud with a voxel size of 10:



```
import numpy as np
pcd = o3d.io.read_point_cloud("SeaTurtle.ply")
downpcd = pcd.voxel_down_sample(voxel_size=10)
print(np.asarray(pcd))
print(downpcd.points)

PointCloud with 779430 points.
std::vector<Eigen::Vector3d> with 51269 elements.
Use numpy.asarray() to access data.
```

Figure 13: Example of a code to down sample point clouds

It's important to note that voxel down-sampling can introduce some loss of information, as it involves aggregating points within each voxel and discarding points that do not fit within a voxel. However, this trade-off can be acceptable in many cases where the overall structure of the point cloud is more important than the individual points.

4.3 D1-MSE and D1-CMSE

MSE (Mean Squared Error) is a measure of error that can be used to evaluate the performance of algorithms for point cloud processing, particularly in the context of geometry and color. Point clouds are collections of points in 3D space, and are often used to represent 3D objects or scenes in fields such as computer vision, robotics, and geometry processing.

There are several reasons why MSE is a useful measure of error for evaluating the geometry and color of point clouds:

It provides a simple, intuitive way to quantify the difference between two point clouds. By calculating the distance between each point in one point cloud and its closest point in the other point cloud, and then averaging these distances, MSE can give a sense of how well the two point clouds match up in terms of their geometry and color.

It is easy to compute and interpret. MSE is calculated using a straightforward formula, and the results can be easily understood in terms of the distance between points.

It is scale-invariant, meaning that it does not depend on the size or orientation of the point clouds. This makes it useful for comparing point clouds that may have different scales or orientations.

It is robust to noise and outliers. Because it is based on the average distance between points, D1-MSE is less sensitive to individual points that may be outliers or corrupted by noise.[20]

However, calculating MSE for all of the points in a large point cloud can be computationally expensive, especially if the point clouds are large or the algorithms being evaluated are computationally intensive. By using a random subset of points, it is possible to reduce the computational cost without sacrificing too much accuracy as we saw from the results obtained.

Using random points for MSE in point cloud geometry and color comparison can produce results that are similar to using all points in the cloud. This is particularly true if the target point cloud is obtained through the application of a uniform filter to the reference point cloud.

In MSE, the distance between each point in the target cloud and its nearest neighbor in the reference cloud is calculated, and the mean of these distances is taken as the overall measure of error between the two clouds. By using random points, rather than all points, to calculate this error measure, some level of noise is introduced into the calculation. However, if the target point cloud has been obtained through the application of a uniform filter to the reference point cloud, this noise is likely to be minimized, as the filtered cloud will retain much of the same overall structure as the reference cloud.

Overall, the use of random points in MSE for point cloud comparison may not produce results that are significantly different from using all points, especially in cases where the target cloud has been obtained through the application of a uniform filter to the reference cloud.

For example for these two point clouds :

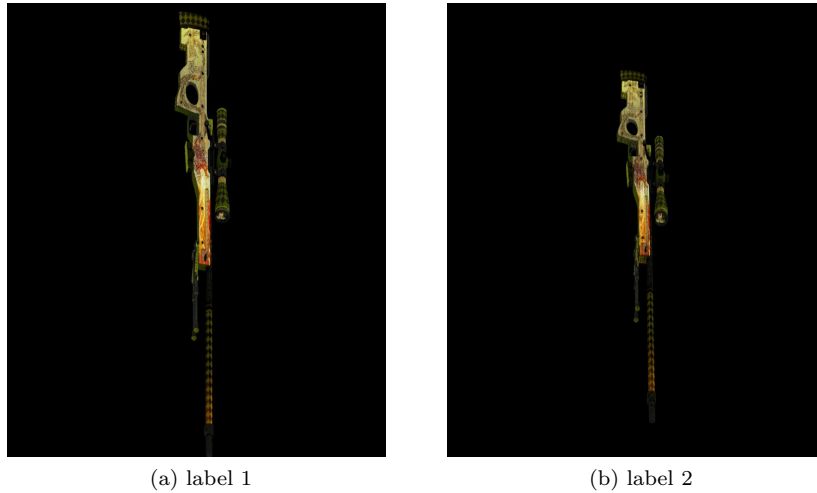


Figure 14: Examples of point clouds dataset

The D1-MSE for 1000 random points is 12.42 while for the whole data is 12.12.

Here we can see some result of applying D1-MSE and D1-CMSE on some different point clouds with different voxel sizes for the down-sampling.

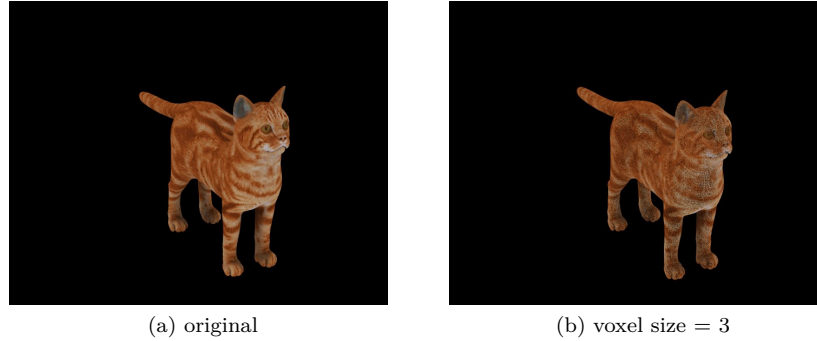


Figure 15: Results of the Objective Metrics Evaluation

D1-MES = 5.01 R-MSE = 37.6 G-MSE = 47.58 B-MSE = 52.41

We can see higher the voxel size (less points) we get higher results for D1-MSE and D1-CMSE as we expect.

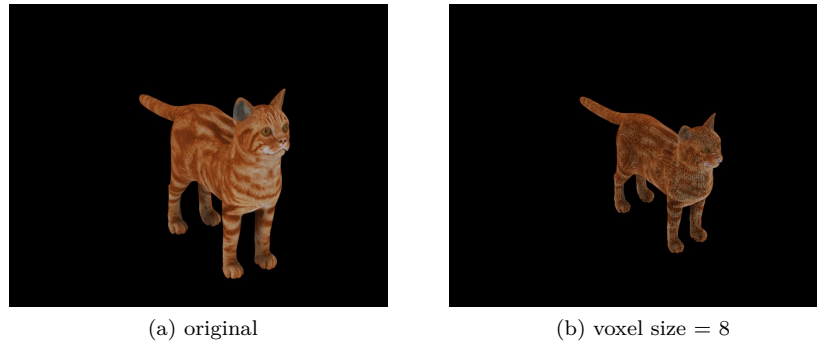


Figure 16: Results of the Objective Metrics Evaluation

D1-MES = 10.87 R-MSE = 55 G-MSE = 68 B-MSE = 85

However, it is important to note that the value of D1-MSE and D1-CMSE are not absolute measures, but rather are dependent on the specific characteristics and shape of the point clouds being compared. To obtain an absolute measure about the quality we should use PSNR that could be obtained from the MSE values. in the last case we would need to have the bit precision of the point clouds.

Here we can see different point clouds for which we applied the D1-MSE and D1-CMSE:



Figure 17: Results of the Objective Metrics Evaluation
 D1-MES = 0.73 R-MSE = 192 G-MSE = 184 B-MSE = 176



Figure 18: Results of the Objective Metrics Evaluation
 D1-MES = 5.68 R-MSE = 291 G-MSE = 268 B-MSE = 253



Figure 19: Results of the Objective Metrics Evaluation
 D1-MES = 5.99 R-MSE = 450 G-MSE = 408 B-MSE = 350

5 Subjective Tests

Subjective tests for point clouds are a type of evaluation method that aims to assess the quality and perceived realism of point clouds, which are 3D datasets that represent the surface of an object or environment. These tests rely on the subjective judgment of human testers, who are asked to evaluate the point clouds based on various criteria such as visual quality, spatial awareness, and overall immersion.

To conduct a subjective test for point clouds, a variety of methods can be used, such as asking testers to complete a series of tasks or assessments while interacting with the point cloud, or simply asking them to provide their impressions and opinions about the point cloud. These methods can be applied in a variety of contexts, including in virtual reality (VR) environments, on traditional computer displays, or even on paper printouts.

One tool that is commonly used for subjective tests of point clouds is the Meta Oculus 2, a VR headset that is capable of rendering point clouds in a virtual environment. By using the Meta Oculus 2, testers can interact with the point cloud in a more immersive and intuitive way, allowing them to better evaluate the quality and realism of the point cloud.



Figure 20: Meta oculus 2

Overall, subjective tests for point clouds are an important tool for evaluating the effectiveness and realism of these types of visualizations, and the use of the Meta Oculus 2 can help to provide a more immersive and interactive testing experience.

5.1 Data Preparation

In order to perform a subjective test on point clouds using the Unity platform, it is necessary to convert the raw point cloud data into a mesh. This is because Unity does not have native support for raw point cloud data and can only work with 3D meshes. Converting the point cloud data into a mesh involves creating a surface representation of the points, typically by using a technique such as mesh reconstruction. Once the point cloud data has been meshed, it can then be imported into Unity and used for the subjective test.

5.1.1 Meshing

Point clouds are sets of points in three-dimensional space that can be created using 3D scanning technologies such as laser scanners or structured light scanners. These point clouds can be used to represent the geometry of a 3D object or environment. Meshing is the process of taking a point cloud and generating a mesh, or a collection of interconnected triangles, from it. This mesh can then be used for a variety of purposes, such as 3D printing, visualization, or as input for finite element analysis.

There are several methods that can be used to generate a mesh from a point cloud, including ball pivoting meshing, volumetric meshing, and parametric meshing. Each method has its own set of advantages and disadvantages, and the choice of which method to use depends on the specific requirements of the application.

Ball pivoting meshing is a popular method for generating meshes from point clouds because it is able to produce high-quality meshes with a relatively low number of triangles. It works by starting from a seed point in the point cloud and pivoting a sphere around it until it touches a set of neighboring points. The points that the sphere touches are then connected by triangles, and the process is repeated for each of these points until the entire point cloud has been meshed.

One of the advantages of ball pivoting meshing is that it is able to handle complex geometry and topology well. It is also relatively fast and easy to implement, making it a practical choice for many applications.[21]

5.1.2 Ball pivoting

The ball pivoting algorithm is an iterative process that generates a mesh from a point cloud. The basic steps of the algorithm are as follows:

Select a seed point from the point cloud and place a sphere of a given radius around it. Find all of the points in the point cloud that are within the sphere. Connect the seed point to these points with triangles. Set each of these points as the new seed points and repeat the process until all points in the point cloud have been meshed. The radius of the sphere is an important parameter in the ball pivoting algorithm, as it determines the density of the resulting mesh. If the radius is too small, the resulting mesh will have a high triangle count and may be too fine to be useful. If the radius is too large, the resulting mesh will have a low triangle count and may not capture the detailed geometry of the point cloud.

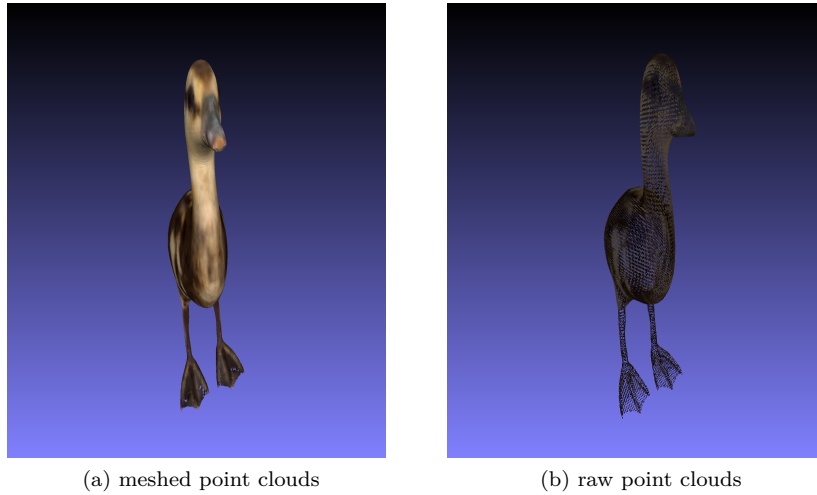


Figure 21: Before and after meshing for point cloud

However, to do mesh the point clouds we used MeshLab software. MeshLab is a free and open-source 3D mesh processing software that can be used to generate meshes from point clouds. One of the advantages of using MeshLab for meshing is that it offers a range of meshing algorithms, including the ball pivoting algorithm, which is known for producing high-quality meshes with a relatively low triangle count.

In addition to its meshing capabilities, MeshLab is also a good choice because it is free and open-source. This makes it an attractive option for those who are working with limited budgets or who prefer to use open-source software.

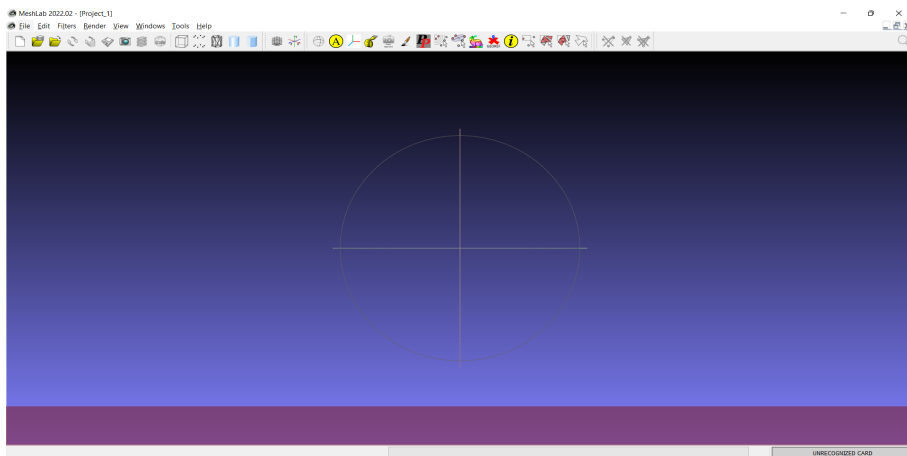


Figure 22: MashLab Software

However, using point clouds in the .fbx format can be beneficial when working with Unity. This is because the .fbx format is a widely-supported 3D file format that is natively supported by Unity. This means that point clouds in this format can be easily imported and used in Unity projects without the need for additional conversion or processing steps.

On the other hand, point clouds in the .ply format may not be natively supported by Unity, or may require additional processing steps in order to be used. As a result, it may be more efficient to use point clouds in the .fbx format when working with Unity.

To convert point clouds from the .ply format to the .fbx format, a 3D modeling software such as Blender can be used. Blender is a free and open-source 3D modeling software that has a range of features and tools for importing, exporting, and converting 3D models and point clouds. It is a good choice for converting point clouds to the .fbx format because it is free, open-source, and has a range of features that make it easy to manipulate and convert 3D models.

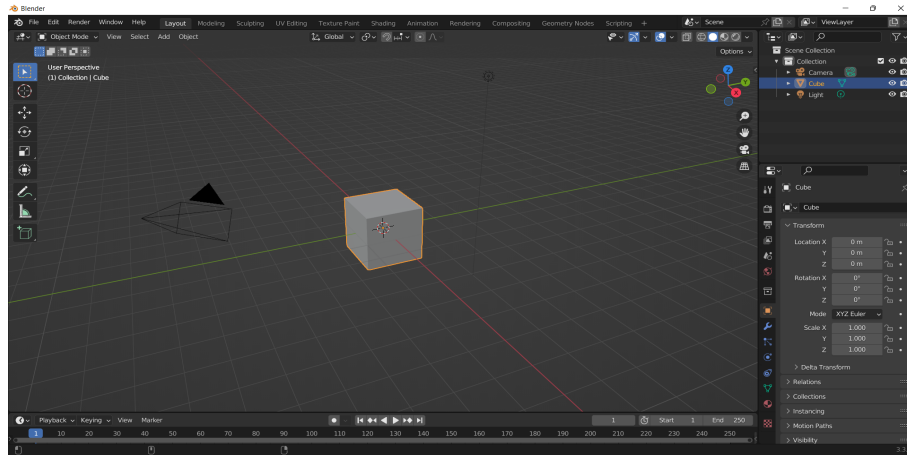


Figure 23: Blender Software

5.2 Unity

Unity is a powerful game engine and development platform that allows developers to create interactive 3D content for a variety of platforms, including virtual reality (VR) devices like the Oculus meta quest 2. One popular use of Unity in VR is creating virtual environments, such as rooms, for use in subjective testing.

Using Unity to create a virtual room for use in subjective testing has several advantages. First, Unity provides a wide range of tools and assets that can be used to create highly detailed and realistic virtual environments. This can help to create a more immersive and engaging experience for users, which is important for getting accurate and useful feedback

Second, Unity also provides a wide range of scripting and programming tools that can be used to create interactive elements in the virtual environment. This can help to add more realism and complexity to the environment, and can also help to make the environment more engaging and interactive for users. Here i am using 2021.3.5f1 version for this study.

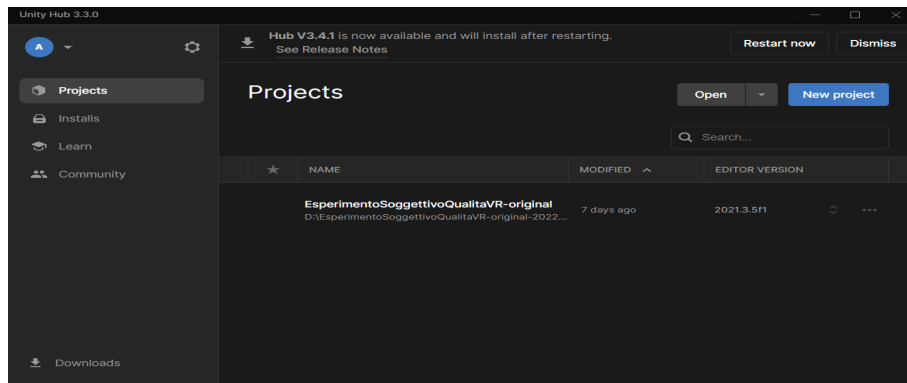


Figure 24: Unity Software

A virtual reality (VR) room created using Unity can provide an immersive and interactive experience for users. The room can be highly detailed and realistic, with accurate lighting, textures, and physics as we can see from the Figure 25:

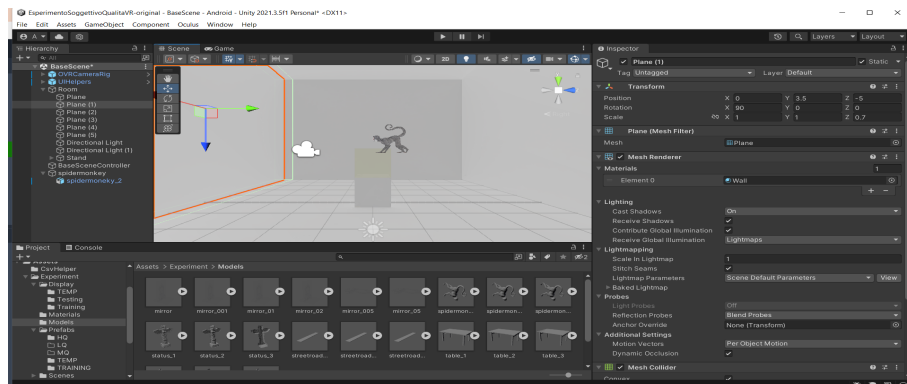


Figure 25: Unity Software

After preparing the experiment in the virtual reality (VR) room built using Unity, the next step is to create an Android application for the experiment to upload it on the Oculus Quest 2. This can be done by using Unity's built-in support for Android development, which allows developers to export their VR projects as Android applications.

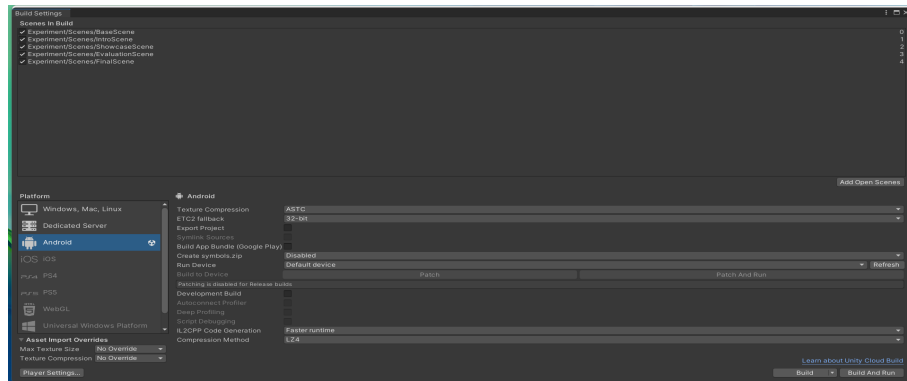


Figure 26: built-in support for Android development in Unity Software

After building an Android application of the experiment using Unity, it can be uploaded to the Oculus Quest 2 via the SideQuest application. SideQuest is a third-party application that allows developers to easily sideload VR content onto the Oculus Quest 2, including games and apps that are not available on the official Oculus store.

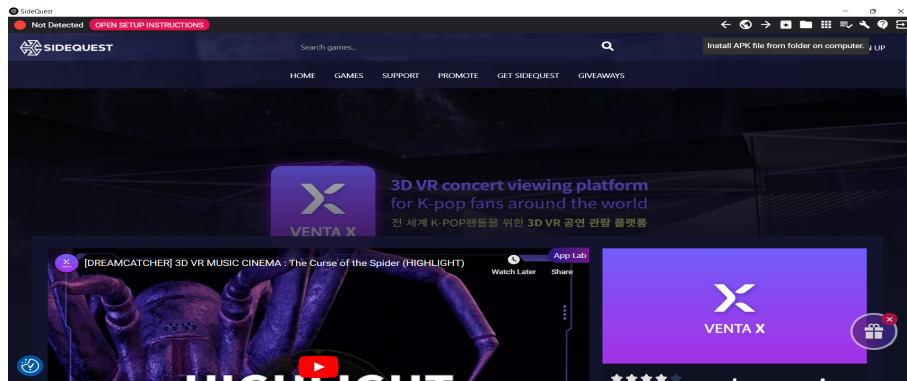


Figure 27: SideQuest

5.3 The Tests

First thing of the test is to prepare the borders for the Oculus 2 so the participant can move freely in this space and be warned in case they crossed them. After that i found it quite helpful to give the participant the ability to take a look about the meta environment first and then search about the experiment application by them self. As this helping them to get less stressed and more used to the VR environment before starting the experiment.



Figure 28: Meta environment

After finding the application the starting of the experiment looks as in the Figure 29 :



Figure 29: Experiment start

The protocol that was implemented for the experiment is the Absolute Category Rating with a Hidden reference (ACR-H), which aligns with the guidelines outlined in ITU-R BT.500-13. This protocol is widely used as a benchmark in the point cloud evaluation, it is a reliable and robust method for evaluating the quality of point clouds.

One of the key features of this protocol is that the participants were not provided with any information regarding which point cloud was the original, or the level of degradation associated with the models. This was done to ensure that the evaluation was impartial and unbiased.

Additionally, the stimuli were presented in a random order during each session, to reduce any potential bias that may have been introduced due to the ordered presentation of the models or previous knowledge of the different levels of degradation. This method of randomization ensures that the results are representative of the true quality of the point clouds, rather than being influenced by external factors.

The initial point clouds that participants are presented with are training samples that are not included in the final results, but serve to familiarize participants with the experiment.



Figure 30: Training Sample

Upon completion of the training sample, the evaluation process will commence using actual point cloud samples to determine the results. These samples will undergo a thorough evaluation, aimed at producing accurate and conclusive results.

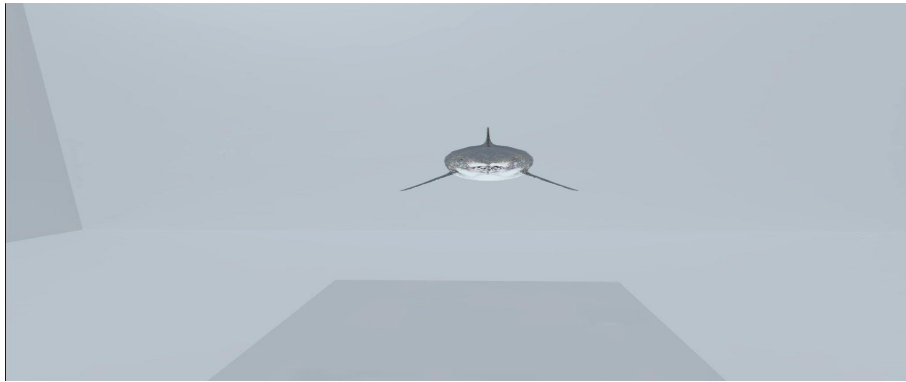


Figure 31: Example of the experiment Samples

Following the presentation of each sample, the participant will have a 16-second interval to inspect and examine the sample. Subsequently, the evaluation screen will be presented, enabling the participant to rate the quality of the point cloud using a scale ranging from 1 to 5. The number 5 indicates the highest level of quality, providing the participant with the opportunity to accurately reflect their assessment of the point cloud's quality.

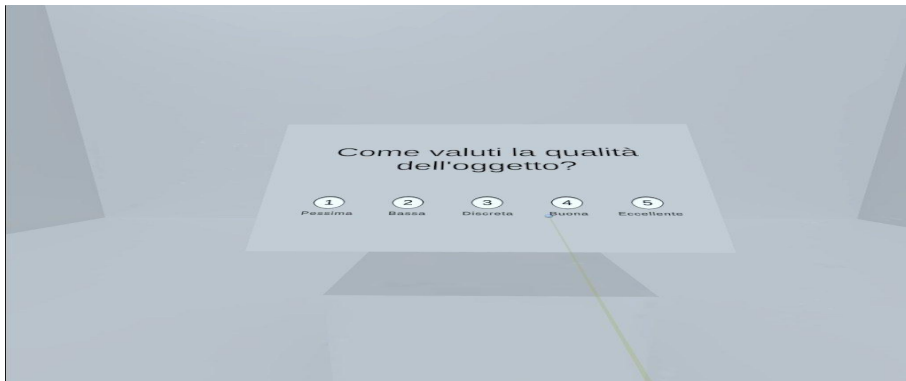


Figure 32: Evaluation screen

5.3.1 Tests Results

We have conducted 22 extensive subjective evaluations, in which a diverse range of point clouds were shown to the participant. The overall number of point clouds that underwent evaluation was 347, derived from 93 unique point cloud models. Through the application of the down sampling procedure previously mentioned in the objective metrics evaluation, these 93 models were transformed into 347 point cloud objects, making them suitable for comprehensive testing and analysis.

The following examples showcase a selection of the quality values obtained from subjective evaluations, highlighting the relationship between the results of these assessments and the corresponding values determined.:

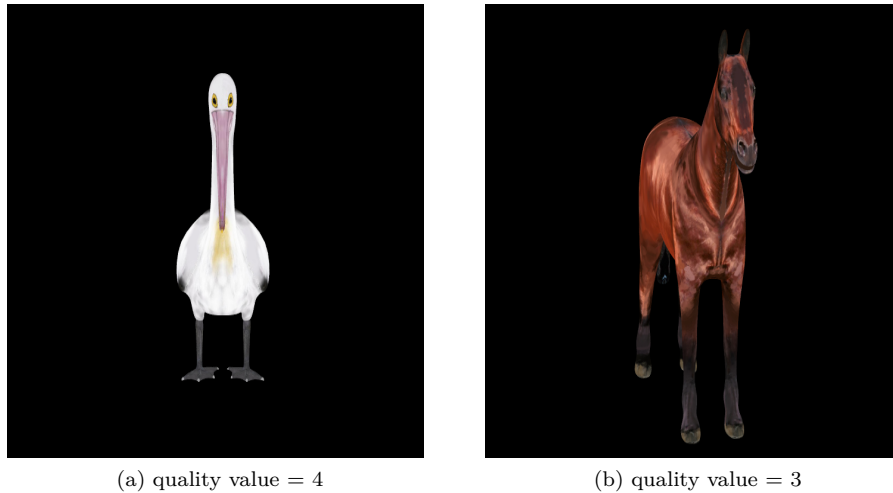


Figure 33: Results from the subjective tests

It is worth mentioning that the evaluation of quality in a three-dimensional environment, such as that of the Oculus Quest 2, can be vastly different from the data presented in the form of figures. This is due to the subjective nature of perceiving quality in a virtual environment.

Here, we are presented with a collection of examples showcasing the same model but with varying degrees of downsampling. These examples have been evaluated subjectively to determine their quality and the scores have been recorded alongside each instance for reference:

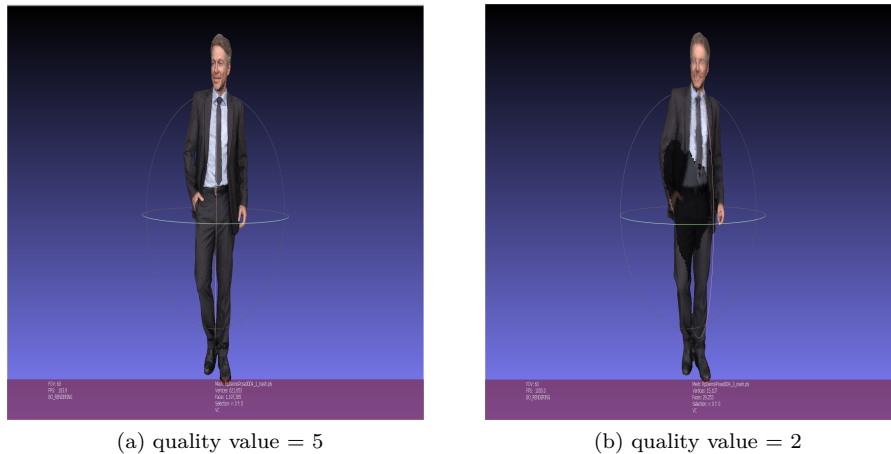


Figure 34: Results from the subjective tests

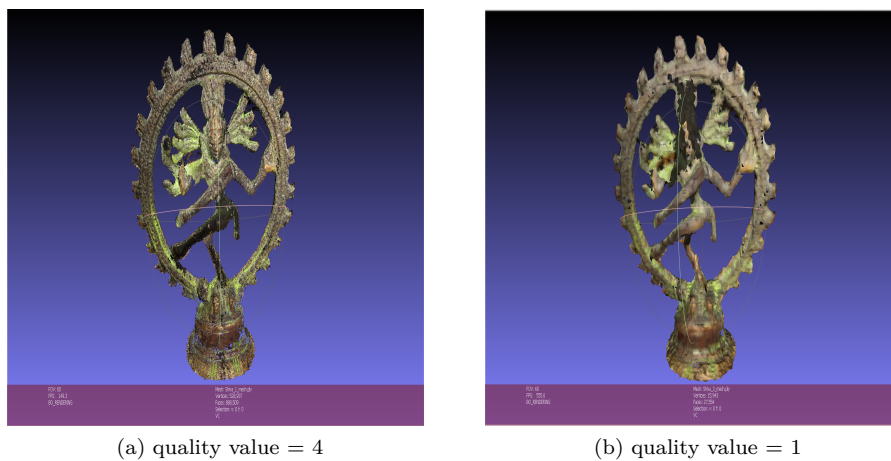
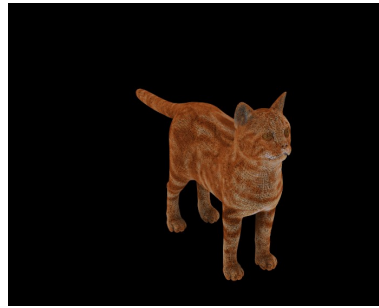


Figure 35: Results from the subjective tests

It has been noted that the higher the D1-MSE or CMSE values, the lower the subjective evaluation scores tend to be, as evidenced by the figures and results presented below. This correlation between objective evaluation metrics (D1-MSE or CMSE) and subjective evaluations is observed in the data as we can see from following figures and results:



(a) original, Subjective quality value = 5



(b) voxel size = 3, Subjective quality value = 4

Figure 36: Results from the subjective tests

D1-MES = 5.01 R-MSE = 37.6 G-MSE = 47.58 B-MSE = 52.41



(a) original, Subjective quality value = 5



(b) voxel size = 8, Subjective quality value = 1

Figure 37: Results from the subjective tests

D1-MES = 10.87 R-MSE = 55 G-MSE = 68 B-MSE = 85



(a) original, Subjective quality value = 5



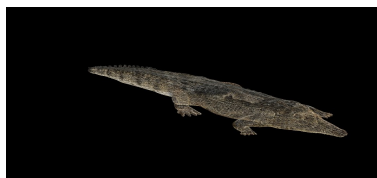
(b) voxel size = 4, Subjective quality value = 3

Figure 38: Results from the subjective tests

D1-MES = 0.73 R-MSE = 192 G-MSE = 184 B-MSE = 176



(a) voxel size = 4, Subjective quality value = 3



(b) voxel size = 16, Subjective quality value = 2

Figure 39: Results from the subjective tests

D1-MES = 5.68 R-MSE = 291 G-MSE = 268 B-MSE = 253



(a) original, Subjective quality value = 5



(b) voxel size = 16, Subjective quality value = 2

Figure 40: Results from the subjective tests

D1-MES = 5.99 R-MSE = 450 G-MSE = 408 B-MSE = 350

6 PointNet

PointNet is a deep learning model for processing point clouds, which are collections of 3D coordinates representing objects in the real world. It was introduced in 2016 by Qi et al. in their paper "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation"[22]

PointNet consists of a simple architecture with a symmetrical encoder-decoder structure. The encoder network maps the input point cloud to a low-dimensional feature representation, while the decoder network maps the features back to the output space. The architecture allows PointNet to directly process unordered point clouds without any manual feature engineering or preprocessing steps.

The encoder of PointNet is implemented as a multi-layer perceptron (MLP) network, which takes in a set of N points in 3D space, represented as a $N \times 3$ matrix. The MLP consists of multiple fully connected layers, followed by a max pooling operation, which is used to reduce the dimensionality of the features and capture the most important information from the point cloud. The max pooling operation is implemented as a learnable function that takes the maximum value of each feature vector across all N points in the cloud. This operation is repeated several times to produce a compact and hierarchical representation of the point cloud.

The decoder of PointNet is also implemented as an MLP network, which takes in the compact feature representation produced by the encoder and produces a reconstructed point cloud representation. The decoder consists of multiple fully connected layers, followed by a MLP-based symmetric function that maps the compact feature representation back to a $N \times 3$ matrix.

PointNet achieved state-of-the-art results on several benchmark datasets for 3D object classification and segmentation tasks. In the ModelNet40 dataset for 3D object classification, PointNet achieved an accuracy of 89.2%. In the ShapeNet dataset for semantic segmentation, PointNet achieved an average intersection over union (IoU) of 0.76 .

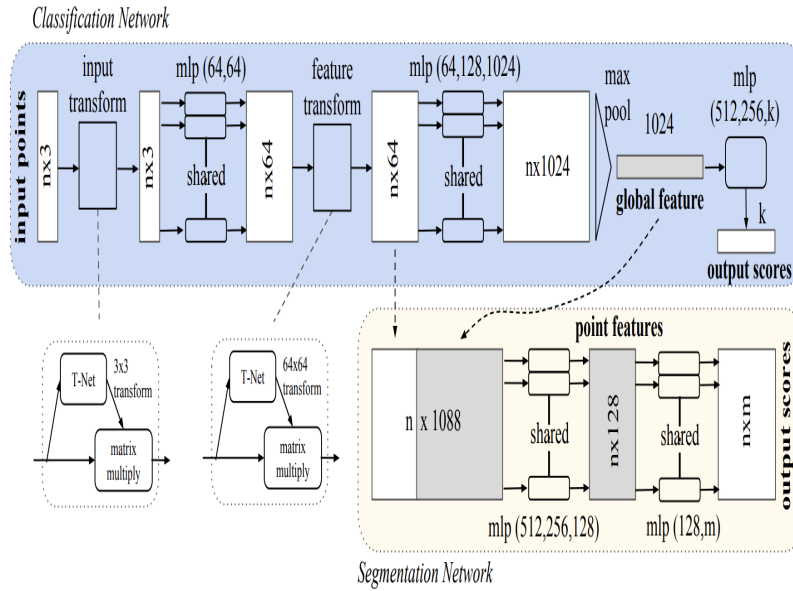


Figure 41: PointNet Model

6.1 PointNet Quality Classification

PointNet was selected for this task because it has several advantages over other models. First, PointNet can directly process unstructured 3D point cloud data, which makes it well-suited for handling noisy and unordered point cloud data. Second, the model's symmetric encoder-decoder architecture enables it to capture the most important features from the point cloud and produce a compact and hierarchical representation of the data. This makes PointNet highly suitable for handling complex and large-scale point cloud data.

Additionally, PointNet has demonstrated high accuracy in various point cloud classification tasks, making it a suitable choice for point cloud quality classification. The model's ability to learn from large datasets of point clouds, combined with its high accuracy in classification tasks, makes PointNet a compelling option for automating the assessment of point cloud quality.

In summary, PointNet was thought to be a suitable model for point cloud quality classification due to its ability to directly process unstructured 3D point cloud data, its symmetric encoder-decoder architecture, and its demonstrated high accuracy in point cloud classification tasks.

We utilized the results and point clouds obtained during the subjective test to train our model. The PointNet model was modified to have an input with six dimensions, including the color values, so that the model can learn the quality scale from both the geometric and color information.

The output of the model is five classes, which correspond to the five quality scale. The process of training the model can be represented as follows:

Let R be the results obtained from the subjective test, where R is a set of data points that represent the subjective ratings of the quality of the point clouds. Let P be the point clouds, where P is a set of 3-dimensional points that represent the spatial information of the point clouds. Let C be the color values associated with each point in the point clouds.

To train our model, we modified the PointNet model, to have an input with six dimensions. The input with six dimensions includes both the spatial information of the point clouds (P) and the color values (C) associated with each point.

The additional information from the color values was included so that the model can learn the quality scale from both the geometric and color information.

The output of the model is five classes, represented as $C1$, $C2$, $C3$, $C4$, and $C5$, which correspond to the five quality scale and represent the predicted quality of the point clouds. The modified PointNet model can be represented as $M(X)$, where M is the modified PointNet model, and X is the input data with six dimensions (P and C).

The training process can be represented as: $M = \text{train}(R, P, C)$ where train is a function that trains the model on the input data (R , P , and C) and returns the trained model M .

During the training process, the model is optimized to minimize the difference between its predicted classes and the actual subjective ratings (R) obtained during the subjective test.

The final trained model M can then be used to predict the quality of new point clouds, taking into account both the geometric and color information.

6.2 Results of The First training

The PointNet model was used to directly classify the point clouds into five classes based on their subjective test results. The point clouds were fed into the network as input and the network was trained to predict the quality class of the point clouds. Despite the potential of PointNet, the results obtained were not as expected and are presented in the Figure 42.

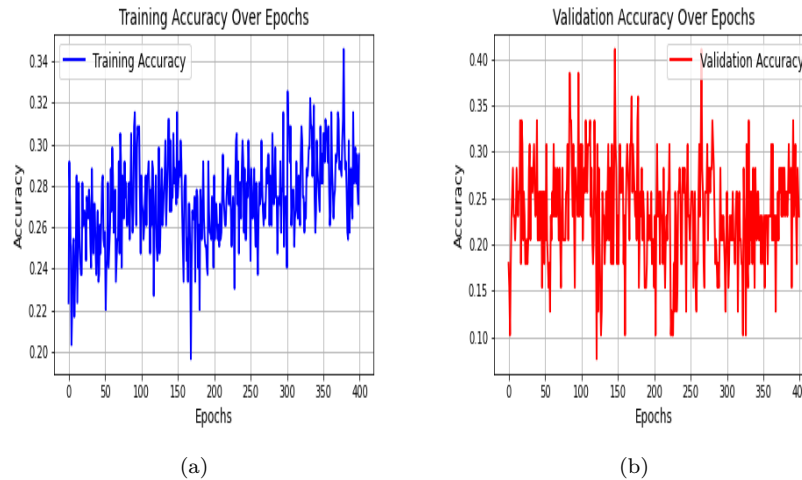


Figure 42: PointNet Accuracy

The results of our model training indicate that the training accuracy is improving at a slow rate, but the validation accuracy is not showing any significant improvement. This suggests that the model is having difficulties learning the underlying pattern in the data.

However, while the results of our model training show a lack of improvement in the validation accuracy, the loss function graph for the training data suggests a decreasing trend. This implies that the model is making progress in minimizing the loss on the training data, but the improvement is not translating to the validation accuracy as we can see from Figure 43.

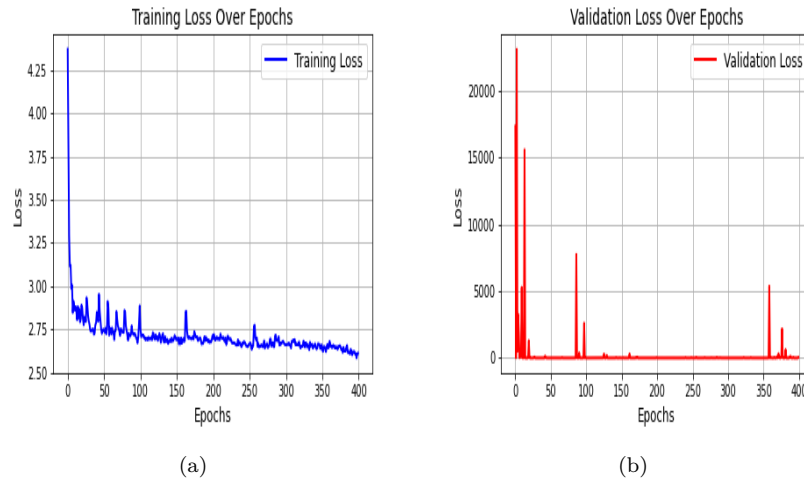


Figure 43: PointNet Loss Function Graphs

The disparity between the decreasing trend in the loss function for the training data and the lack of improvement in the validation accuracy could be due to the subjective nature of the point cloud quality classification. The subjective test results used to evaluate the quality of the point clouds can be confusing and unreliable, as different people may assign different values to the same point cloud. This lack of consistency in the subjective test results makes it challenging for the model to understand and accurately classify the point cloud quality.

The subjective nature of the point cloud quality evaluation can lead to a lack of uniformity in the training data, which makes it difficult for the model to learn the underlying patterns in the data.

To address the challenges posed by the subjective nature of the point cloud quality evaluation, we introduced a modified version of the PointNet model with a modified loss function. By incorporating this new loss function into the model, we aimed to improve its ability to learn the underlying patterns in the data and achieve better results in the point cloud quality classification task.

6.3 Modified PointNet Loss Function

The application of PointNet in the point cloud quality classification task is faced with a challenge due to the subjective nature of the evaluation process. The results obtained from the subjective tests used to evaluate the quality of the point clouds are prone to variation, as different people may assign different values to the same point cloud.

To tackle this challenge, we have proposed a modified version of the loss function used by PointNet. The sparse categorical crossentropy loss function, which is commonly used in classification tasks, is not suitable for the point cloud quality classification task due to its limitation in handling the subjective nature of the evaluation.

The modified loss function takes into account the subjective nature of the evaluation by incorporating the insight that while different people may assign different values to the same point cloud, it is highly probable that these values will not be far apart. For example, if one person assigns a value of 5 to a point cloud in a subjective test, it is more likely that other people will assign values of 5, 4, or 3 to the same point cloud, rather than values of 2 or 1.

By incorporating this insight into the loss function, the performance of the PointNet model on the point cloud quality classification task is expected to improve. The modified loss function, shown in Figure 44, is designed to account for the subjective nature of the evaluation and helps the model learn the underlying patterns in the data. This, in turn, leads to a better performance of the PointNet model in classifying the quality of point clouds.

```
[ ] def custom_loss(y_true, y_pred):
    # Calculate the sparse categorical crossentropy loss
    sparse_loss = tf.keras.losses.sparse_categorical_crossentropy(y_true, y_pred)

    # Convert the ground truth labels to float data type
    y_true = tf.cast(y_true, tf.float32)

    # Calculate the mean of the absolute difference between the ground truth and predicted values raised to the power of 3
    loss = tf.reduce_mean((tf.abs(y_true - y_pred))**3)

    # Return a weighted combination of the sparse loss and the new loss, with the new loss given a weight of 0.2
    return .2*loss+sparse_loss
```

Figure 44: PointNet Modified Loss Function

By utilizing this custom loss function, two additional hyperparameters will be introduced. These hyperparameters represent the power of the absolute difference between the predicted quality class and the target quality class. Through various trial and error iterations, we found that the values shown in Figure 44 work well for our specific task and provide desirable results.

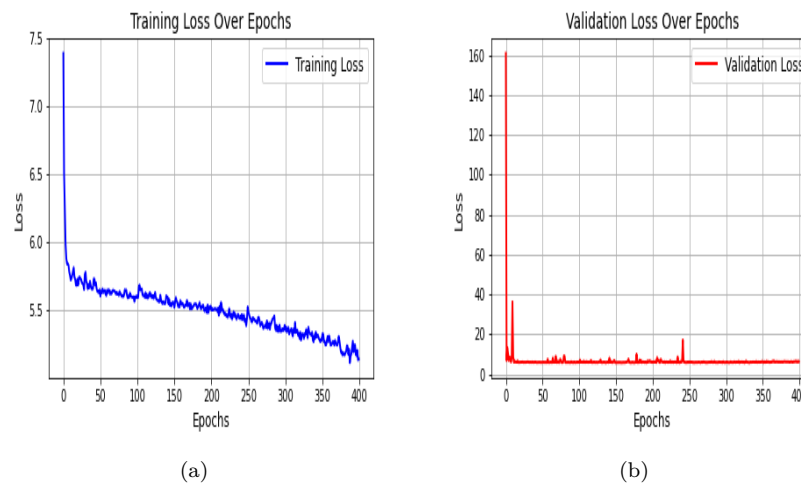


Figure 45: Modified PointNet Loss Function Graphs

By analyzing the results of the custom loss function, we can observe that the model is making progress in learning the underlying patterns in the data. This is evident from the decreasing trend in the graphs of the loss function over time. Additionally, the accuracy of the model can be seen to increase, further confirming that the model is learning and improving with each iteration as in the Figure 46.

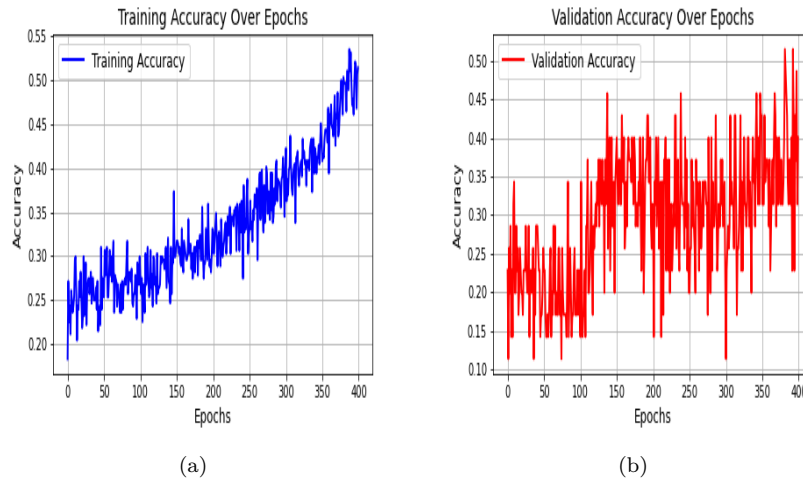


Figure 46: Modified PointNet Accuracy Graphs

The implementation of the custom loss function has resulted in a noticeable improvement in the accuracy of the model when compared to the previous loss function. In order to preserve the generality of the model and prevent it from overfitting, we decided to halt the training process after 400 epochs. Although the model achieved an accuracy of approximately 52 percent on the training data, it is crucial to consider that the actual accuracy of the model can be more accurately assessed through the results shown in Figure 47. It is important to keep in mind that accuracy metrics can be influenced by various factors, and that the results presented in the figure provide a more comprehensive evaluation of the model's performance.

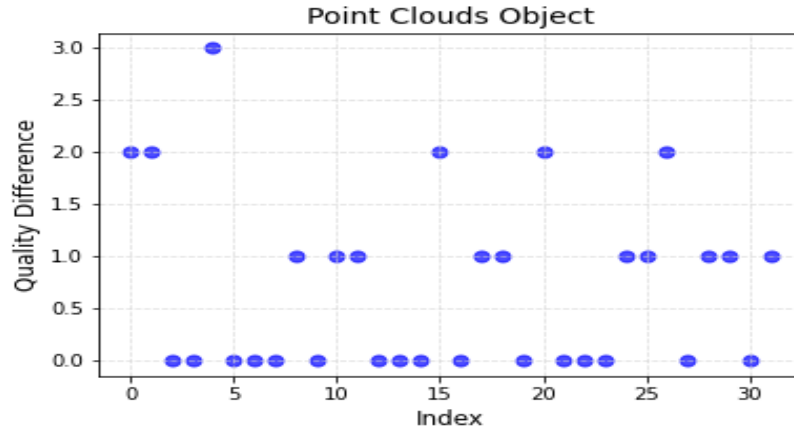


Figure 47: Absolute Quality Difference

Figure 47 displays the absolute difference values between the predicted quality scores of point clouds and the actual quality scores obtained from subjective tests for the test dataset, which consists of 32 point clouds. The plot reveals that most of the absolute difference values are either 0 or 1. This implies that if we consider a tolerance of 1 class difference between the predicted and actual quality scores, then the accuracy of our model is around 82%.

Furthermore, the plot indicates that there are very few 2-class differences and almost no 3 and 4-class differences between the predicted and actual quality scores. This suggests that the model has a good level of accuracy, as the majority of the predicted scores are very close to the actual scores.

The results from Figure 47 demonstrate the effectiveness of the proposed approach, which includes the use of a modified loss function.

6.4 Model Carbon Impact

The research community is promoting the notion of disclosing the carbon footprint of training deep learning models, due to the significant carbon impact they have. Our training produced an estimated carbon consumption of 3.72 KG CO₂, equivalent to driving 15 Km in a conventional internal combustion engine car[23].

7 Conclusion

The study of point cloud quality is still an active area of research and development. It is an essential component of many applications such as 3D modeling and autonomous driving. In order to evaluate the quality of point clouds, researchers have been exploring various techniques such as objective metrics, subjective testing, and deep learning models.

Objective metrics provide a reliable way to estimate the quality of point clouds. These metrics measure the geometric and topological properties of point clouds and can provide an objective measure of quality. However, this technique requires significant computational resources, and the results may not always be reliable in real-world scenarios. In some cases, these metrics are based solely on mathematical calculations and do not take into account human perception.

On the other hand, subjective testing provides a direct opinion of end-users. This approach involves conducting user studies where participants are asked to evaluate the quality of point clouds. While this technique offers a realistic evaluation of point cloud quality, it is time-consuming and can be expensive. Additionally, subjective testing may not be reliable for applications that require a continuous evaluation of quality.

Deep learning models show promise for the future evaluation of point cloud quality. These models use neural networks to learn the features of point clouds and can provide accurate predictions of quality. However, the accuracy of these models is still under development, and there is a need for further research to achieve high accuracy that can be reliable and replace subjective and objective methods.

In conclusion, the quality evaluation of point clouds is an essential component of many applications, and continued research is necessary to improve the accuracy and efficiency of evaluation techniques. Researchers need to balance the advantages and disadvantages of different methods to choose the most appropriate technique for their applications. The development of reliable and efficient evaluation methods will lead to better applications in fields such as 3D modeling, autonomous driving, and beyond.

References

- [1] https://en.wikipedia.org/wiki/Point_cloud
- [2] LiDAR point clouds correction acquired from a moving car based on CAN-bus data Pierre Merriaux¹, Yohan Dupuis¹, Rémi Bouteau¹, Pascal Vasseur and Xavier Savatier.
- [3] Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip A Chou, Robert A Cohen, Maja Krivokuca, Sébastien Lasserre, Zhu Li, et al. Emerging mpeg standards for point cloud compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(1):133–148, 2018.
- [4] A Comprehensive Study and Comparison of Core Technologies for MPEG 3D Point Cloud Compression Hao Liu, Hui Yuan, Senior Member, IEEE, Qi Liu, Junhui Hou, Member, IEEE, Ju Liu, Senior Member, IEEE.
- [5] D Graziosi, O Nakagami, S Kuma, A Zaghetto, T Suzuki, and A Tabatabai. An overview of ongoing point cloud compression standardization activities: Video based (v-pcc) and geometry-based (g-pcc). *APSIPA Transactions on Signal and Information Processing*, 9, 2020
- [6] Learned Point Cloud Geometry Compression Jianqiang Wang, Hao Zhu, Zhan Ma, Tong Chen, Haojie Liu, and Qiu Shen Nanjing University.
- [7] Point Cloud Rendering after Coding: Impacts on Subjective and Objective Quality A. Javaheri, C. Brites, Member, IEEE, F. Pereira, Fellow, IEEE, J. Ascenso, Senior Member, IEEE.
- [8] ISO/IEC JTC1/SC29/WG1 N86012, “JPEG Pleno Point Cloud – Use Cases and Requirements v1.3”, Sydney, Australia, January 2020.
- [9] Qualinet White Paper on Definitions of Quality of Experience Output from the fifth Qualinet meeting, Novi Sad, March 12, 2013.
- [10] Alireza Javaheri, Catarina Brites, Fernando Pereira, and Joao Ascenso. Point cloud rendering after coding: Impacts on sub-

- jective and objective quality. *IEEE Transactions on Multimedia*, 23:4049–4064, 2020.
- [11] PointPCA: Point Cloud Objective Quality Assessment Using PCA-Based Descriptors Evangelos Alexiou, Irene Viola, Member, IEEE and Pablo Cesar, Senior Member, IEEE.
- [12] GEOMETRIC DISTORTION METRICS FOR POINT CLOUD COMPRESSION Dong Tian, Hideaki Ochimizu, Chen Feng, Robert Cohen, Anthony Vetro.
- [13] G. Meynet, Y. Nehmé, J. Digne, and G. Lavoué, “PCQM: a full-reference quality metric for colored 3d point clouds,” in 2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX), 2020.
- [14] E.Alexiou and T.Ebrahimi, “Towards a Point Cloud Structural Similarity Metric,” 2020 IEEE International Conference on Multimedia Expo Workshops (ICMEW), London, United Kingdom, 2020.
- [15] International Telecommunication Union/ITU Radiocommunication Sector, January 2. ITU-R BT.500-13, “Methodology for the subjective assessment of the quality of television pictures.
- [16] International Telecommunication Union/ITU Radiocommunication Sector, February 2. ITU-R BT.2021, “Subjective methods for the assessment of stereoscopic 3DTV systems.
- [17] E. Dunic, F. Battisti, M. Carli and L. A. da Silva Cruz, “Point Cloud Visualization Methods: a Study on Subjective Preferences,” 2020 28th European Signal Processing Conference (EUSIPCO), 2021, pp. 595-599, doi: 10.23919/Eusipco47968.2020.9287504.
- [18] Alexiou, E. ; Ebrahimi, T.: On the performance of metrics to predict quality in point cloud representations, 2017, S. 53
- [19] author = Liu, Yipeng and Yang, Qi and Xu, Yiling and Yang, Le, title = Point Cloud Quality Assessment: Dataset Construction and Learning-Based No-Reference Metric, year = 2022

-
- [20] Rusu, R. B., Blodow, N., Beetz, M. (2009). Fast point feature histograms (FPFH) for 3D registration. In Robotics and Automation (ICRA), 2009 IEEE International Conference on (pp. 3212-3217). IEEE.
- [21] Ball Pivoting Algorithm for Surface Reconstruction, F. Bernardini, G. Patané, and D. Panozzo, ACM Transactions on Graphics (TOG), Vol. 32, No. 4, July 2013.
- [22] Qi, Charles R., Hao Su, Kaichun Mo, and Leonidas J. Guibas. "PointNet: Deep learning on point sets for 3D classification and segmentation." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.
- [23] Quantifying the Carbon Emissions of Machine Learning, Lacoste, Alexandre and Luccioni, Alexandra and Schmidt, Victor and Dandres, Thomas, arXiv preprint arXiv:1910.09700