Quality and User Experience (2023) 8:4
https://doi.org/10.1007/s41233-023-00057-4

RESEARCH ARTICLE

Crafting the MPEG metrics for objective and perceptual quality assessment of volumetric videos

Jean-Eudes Marvie[1] · Yana Nehmé[2] · Danillo Graziosi[3] · Guillaume Lavoué[4]

Received: 5 December 2022

Abstract
Efficient objective and perceptual metrics are valuable tools to evaluate the visual impact of compression artifacts on the visual quality of volumetric videos (VVs). In this paper, we present some of the MPEG group efforts to create, benchmark and calibrate objective quality assessment metrics for volumetric videos represented as textured meshes. We created a challenging dataset of 176 volumetric videos impaired with various distortions and conducted a subjective experiment to gather human opinions (more than 5896 subjective scores were collected). We adapted two state-of-the-art model-based metrics for point cloud evaluation to our context of textured mesh evaluation by selecting efficient sampling methods. We also present a new image-based metric for the evaluation of such VVs whose purpose is to reduce the cumbersome computation times inherent to the point-based metrics due to their use of multiple kd-tree searches. Each metric presented above is calibrated (i.e., selection of best values for parameters such as the number of views or grid sampling density) and evaluated on our new ground-truth subjective dataset. For each metric, the optimal selection and combination of features is determined by logistic regression through cross-validation. This performance analysis, combined with MPEG experts' requirements, lead to the validation of two selected metrics and recommendations on the features of most importance through learned feature weights.

Introduction

Technological advances in the acquisition, rendering, and visualization devices (e.g., VR/AR Head-Mounted Displays) have led to the emergence of a new type of multimedia: volumetric videos. A volumetric video generally consists of a temporal sequence of 3D meshes or 3D point clouds (Fig. 1 illustrates the concepts of point clouds and meshes), resulting from a multi-view capture process and a reconstruction process. This new type of data induces a whole range of new scientific challenges from capture to rendering [2–4]. A crucial issue is the efficient compression of those assets. Indeed, a few seconds of raw volumetric video can represent gigabytes of data. Important efforts are therefore undertaken by the scientific community and the MPEG and JPEG consortiums on this topic. Since compression processes may obviously alter the appearance of 3D content, efficient perceptual metrics are thus needed to evaluate the visual impact of compression artifacts on the visual quality of the volumetric videos.

The problem of objective quality assessment of volumetric videos has only been considered by a few authors [5, 6]. Proposing a robust quality/fidelity metric for volumetric videos is actually very challenging, particularly because volumetric videos may have different intrinsic representations (meshes or point clouds) and also different representations for colors (texture maps or vertex/point colors). Moreover, a volumetric video is usually composed of *incoherent* frames, i.e., without correspondence between points along the frames.

✉ Jean-Eudes Marvie
jean-eudes.marvie@interdigital.com

Yana Nehmé
yana.nehme@insa-lyon.fr

Danillo Graziosi
danillo.graziosi@sony.com

Guillaume Lavoué
guillaume.lavoue@enise.fr

[1] InterDigital INC, Rennes, France

[2] INSA Lyon, Univ Lyon, CNRS, UCBL, LIRIS, UMR5205, Lyon, France

[3] Sony Corporation of America, San Jose, USA

[4] Centrale Lyon, Univ Lyon, CNRS, INSA Lyon, UCBL, LIRIS, UMR5205, ENISE, Lyon, France

Published online: 06 June 2023

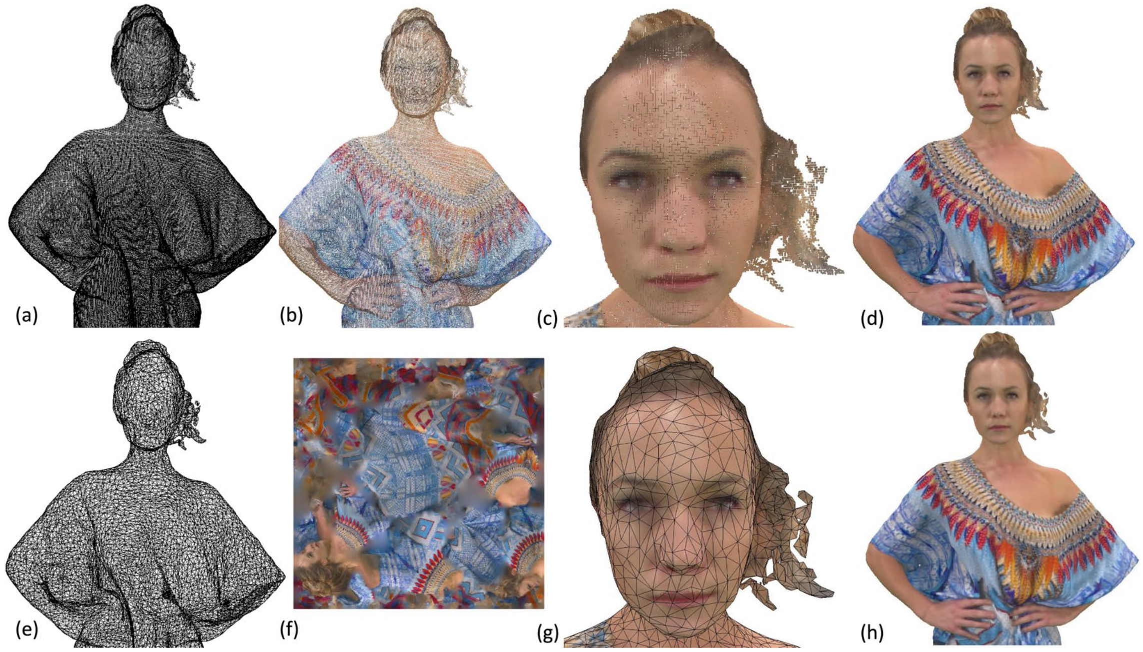


Fig. 1 Illustration of point cloud (top row) and mesh (bottom row) representations. A 3D point cloud is a discrete set of points in the 3D space (**a**). Each point can be assigned a color (**b**). Basic rendering of a point cloud leads to non continuous surfaces (**b**, **c**). Hence, point clouds often require a huge amount of points for the rendering (e.g. millions). Some splatting methods allow to fill these inter-point spaces at the rendering stage (**d**). A 3D mesh is a collection of vertices (3D points), edges and faces that defines a polyhedral surface (**e**). Faces usually consist of triangles. Some color images, named texture maps (**f**), can be mapped onto these triangles to colorize the surface (**g**). The final rendering of a textured mesh produces a continuous surface (**h**). Longdress model from MPEG dataset, courtesy 8i [1]

This paper presents an MPEG effort to create, benchmark and calibrate objective quality assessment metrics for volumetric videos. This effort is composed of three steps:

1. **Creation of the ground-truth dataset**. We created a challenging dataset of 176 volumetric videos (represented by textured meshes) impaired with various distortions and conducted a subjective experiment to gather human opinions (more than 5896 subjective scores were collected);
2. **Selection and creation of metrics and features**. Two *model-based* state-of-the art metrics were selected [7, 8]. These metrics operate on the 3D data directly and involve the computation of several geometry and color features. An *image-based* (i.e., operating on rendered snapshots) metric was also implemented; it computes standard image-based features (e.g., mean square errors (MSE) on YUV components) but also two novel features: one involving the depth buffer and a mask buffer that specifically detects holes in the data, and the other aiming at detecting temporal distortions.
3. **Calibration, learning and evaluation**. Each metric presented above is calibrated (i.e., selection of best values for parameters such as the number of views or grid sampling density) and evaluated on the ground-truth dataset presented above. More specifically, for each metric, the optimal selection and combination of features is determined by logistic regression through cross-validation. This performance analysis, combined with MPEG experts' requirements, led to the adoption of two metrics, one model-based and one image-based that will be released publicly in open source.

The main contributions of our work are as follows:

- The benchmarking of several state-of-the-art metrics and features, originally designed for static 3D models, for the specific case of volumetric videos (i.e., temporal sequences of 3D models) using our new subject-rated dataset.
- The proposition of new image-based features involving the depth buffer, a specifically designed mask buffer and the modeling of temporal distortions. These new features greatly improve the performance.
- The release in open-source of ready-to-use metrics for volumetric videos represented as sequences of textured meshes.

The rest of the paper is organized as follows. Section 2 discusses the related work about quality assessment of volumetric videos. Section 3 presents the subjective experiment we conducted. In Sect. 4 and 5 we respectively detail the adaptation of some point-based metrics for texture mesh assessment and the newly introduced image-based metric. We finally provide an evaluation of the different metrics using a learning-based approach to select best features in Sect. 6 before concluding in Sect. 7.

Related work

This overview of the state of the art describes describes recent advances on quality assessment of volumetric videos. We refer the reader to [9] for a very recent and exhaustive survey on this topic.

Volumetric videos usually refer to temporal sequences of 3D meshes, or 3D point clouds, which are often accompanied by color information, either represented in the form of texture maps or as color values associated with points/vertices. These sequences of meshes/point clouds usually have no temporal coherence, meaning that from one frame to another there are no correspondences between points/vertices/triangles and even not the same number of points/vertices/triangles. This lack of coherence, combined with the heterogeneous nature of the data (mesh vs point clouds, texture maps vs colors) makes their quality assessment particularly challenging.

The topics of subjective and objective quality assessment of 3D meshes and point clouds (with color information) have been very active for the last five years. As detailed in [9], many subjective experiments have been conducted leading to the public release of a number of subject-rated datasets [10–16]. Each of these datasets contains a number of stimuli rated by observers using different subjective protocols. These stimuli are created by applying distortions on pristine source models. For instance, Guo et al. [10] propose a dataset of 136 textured meshes obtained by applying distortions on both geometry and texture of 6 source meshes. Nehmé et al. [12] propose a dataset of 480 distorted colored meshes obtained by applying geometry quantization, color quantization and simplification on a set of 5 source meshes. For point clouds, Su et al. [15] propose a dataset of 740 stimuli obtained by applying Gaussian noise and down sampling to 20 source point clouds. The largest available dataset is for textured meshes [11] and contains 3000 stimuli from 55 sources. The datasets mentioned above are precious assets for evaluation, creation and calibration of objective metrics; however, they all concern *static* 3D content while we are interested in *temporal sequences* of meshes with time-varying texture maps.

Many objective quality metrics have also been proposed [7, 17–24]. They usually rely on a number of features, related to color and geometry, which are then combined using hand-crafted weights or regression algorithms. Most of these metrics are model-based, i.e., their features are computed directly on meshes/points clouds. Still, several image-based metrics [25–27] have also been considered for assessing the quality of 3D data, they consist in applying well-known image metrics (e.g., SSIM [28]) to projected views (rendered snapshots) of 3D meshes/point clouds.

All the work presented above concern *static* 3D models, i.e., not specifically volumetric videos which are *temporal sequences* of 3D models. Only a few authors specifically tackled the problem of volumetric video quality assessment. Zerman et al. [29] and Cao et al. [30] conducted subjective quality assessment experiments with volumetric videos, they both compared mesh and point cloud representations concerning their rate-distortion tradeoff. Viola et al. [6] compared how the level of interactions (e.g., 3DoFs vs 6DoFs) influence the perceived quality of volumetric videos. Finally, Ak et al. [5] compared several point-based and image-based metrics for the objective quality evaluation of volumetric videos from the V-SENSE subject-rated dataset [29] and explored how reducing the temporal sampling affects the results.

As can be seen above, the field of objective quality assessment of volumetric videos remains mostly unexplored. Ak et al. [5] found, in their comparison, that the best results were obtained by the image-based SSIM and the model-based RMS metrics; however, they did not test recent state-of-the-art metrics such as PCQM. Moreover, the subjective data they consider [29] remains relatively easy to predict since a simple PSNR results in more than 0.8 Pearson correlation with subjective data.

To further explore this topic, we created a new subjectively-rated challenging dataset of volumetric videos, represented as textured meshes. We then used it to calibrate, optimize and compare several state-of-the-art metrics. We also propose an image-based metric integrating two new features: the first is based on the depth buffer and a mask buffer that specifically detects holes, while the second is designed to detect temporal artifacts.

Subjective experiment

We conducted a subjective experiment to evaluate the visual impact of different kinds of compression-related distortions on the visual appearance of volumetric videos. As recommended by Nehmé et al. [31] we selected a double stimulus impairment scale (DSIS) protocol. This section provides details on the subjective study.

Stimuli

Source models and distortions

Our dataset consists of 11 source volumetric videos (VVs), subject to 4 types of distortions, each associated with 4 levels, producing a total of 176 stimuli. The source VVs were selected from the MPEG repository updated for the V-Mesh call for participation (CfP) [1]. They are illustrated in Fig. 2 and Table 1 summarizes their characteristics.

These source models have been corrupted by four types of distortions, each applied with four different strengths:

- *Position quantization* - We consider uniform geometric quantization. The geometric quantization levels are driven by Q_p, the precision in number of bits per coordinate. $Q_p \in \{8, 9, 10, 11\}$.
- *Texture coordinate quantization* - We consider uniform quantization, like for geometry. The texture coordinate quantization levels are driven by Q_t, the precision in number of bits per coordinate. $Q_t \in \{8, 9, 10, 11\}$.
- *HEVC compression*: We consider the libBPG implementation (https://bellard.org/bpg/). Compression levels are driven by Q_{map}. $Q_{map} \in \{30, 40, 45, 50\}$.
- *Triangle holes*: To simulate potential cracks due to encoding, we remove arbitrary triangles from each frame. The distortion levels are driven by the hole density D_h; for instance $D_h = 1000$ means that 1 triangle every 1000 triangles is removed. $D_h \in \{150, 300, 500, 1000\}$.

The strengths of these distortions were selected in order to span the whole range of visual quality from imperceptible levels to high levels of impairment.

Stimuli generation

As raised in many studies about quality assessment of 3D content, while it is important for the observer to have access to different viewpoints of the 3D object, the problem of allowing free interaction is the cognitive overload which may alter the results. In the specific case of volumetric videos most of studies consider *non-interactive* protocols, some of them generate animations by considering predefined camera paths (e.g., [29]), while others consider fixed viewpoints (e.g., [32]). Since our dataset is composed of *characters* and in order to simplify the cognitive task, we determined for each VV the most representative viewpoint (i.e., corresponding to the front view of each character) and produced an animation by simply displaying the mesh sequence at 30 frames per second. Note that we considered an orthogonal front face projection to prevent any distortion due to perspective projection. For the rendering, we considered a top right directional lighting and a Lambertian reflectance model. Figure 3 illustrates several reference and distorted stimuli.

Table 1 Details on our volumetric videos: vertex number (per-frame), face number (per-frame), sizes of texture images and number of frames

Source	#Vertices	#Faces	Texture size	#Frames
Basket	20k	40k	2048×2048	300
Dancer	20k	40k	2048×2048	300
Exercise	20k	40k	2048×2048	300
Football	27k	40k	2048×2048	300
Longdress	22k	40k	2048×2048	300
Loot	22k	40k	2048×2048	300
Mitch	16k	30k	4096×4096	300
Model	20k	40k	2048×2048	300
Redandblack	22k	40k	2048×2048	300
Soldier	22k	40k	2048×2048	300
Thomas	16k	30k	4096×4096	300

Experimental procedure and apparatus

A recent study compared three subjective protocols for the quality assessment of 3D graphics [31]. This study pointed out the fact that, contrarily to images and videos, the simultaneous presence of the pristine reference is of great importance for the human subjects to correctly assess the visual quality of impaired 3D models. Following the recommendation from [31], we adopted the Double Stimulus Impairment Scale (DSIS) method: observers see the source VV and the distorted VV side by side and rate the impairment of the second stimulus in relation to the source model using a five-level impairment scale, displayed after the presentation of each pair of stimuli. The scale is as follows: 5-Imperceptible, 4-Perceptible, but not annoying, 3-Slightly annoying, 2-Annoying, 1-very annoying.

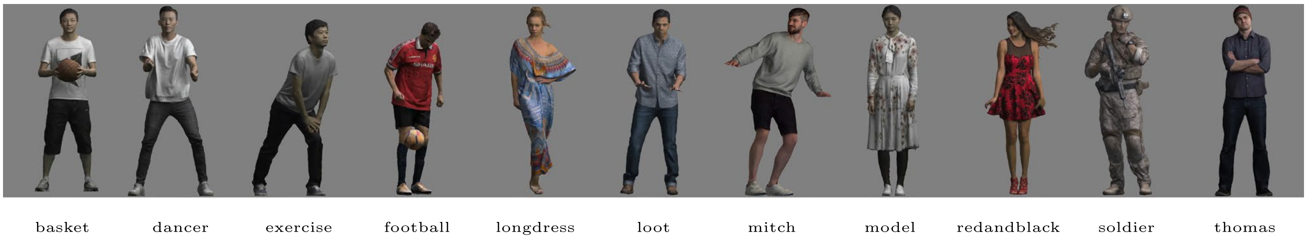
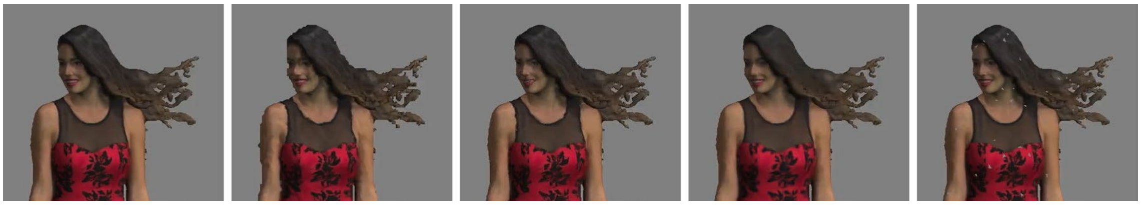


Fig. 2 The 11 source models used to create our dataset. They are sequences of textured meshes composed of 300 frames each



Reference

Geometry quant. $Q_p = 8$

Text. coord. quant. $Q_t = 8$

HEVC compression $Q_{map} = 50$

Triangle holes $D_h = 150$

Fig. 3 Illustrations of the visual effects of the introduced distortions, for the *redandblack* volumetric video

For each stimulus, we thus generate a video of the pair { pristine source, distorted stimulus} side by side. The video resolution is 1920 x 1080 and the duration is 10 s. Those videos were then compressed with FFMPEG H265 at the highest quality.

Due to COVID-19 restrictions, participants performed the experiment online. To design and implement it, we used the PsychoPy framework [33], combined with Pavlovia platform (www.pavlovia.org) for hosting. This way, only a web browser was required to run the experiment. As mentioned above, we implemented the Double Stimulus Impairment Scale (DSIS) protocol: each participant is presented with videos illustrating pairs of stimuli (distorted + pristine reference) during 10 s, and then rates the impairment of the distorted stimulus in relation to the reference using a five-level impairment scale [34]. Participants cannot provide their score until the videos have been played completely. There is no time limit for voting and videos of the stimuli are not shown during that time (and cannot be replayed).

At the beginning of the session, all videos are first downloaded with a progress bar showing the status of the loading process to ensure there is no latency during the test. Then instructions are displayed to the user. A training set of three videos (using the volumetric video *Ballerina*, an asset not used in the test session) is then displayed to familiarize the user with the rating task; for the fist two, example scores assigned to the presented distortions are illustrated, and for the last the participant rates the impairment himself. Of course those training ratings are not considered in the results. After this training, the real test session starts.

Note that to ensure consistent viewing conditions, participants were given requirements before running the experiment (full HD monitor requirement, zoom level set to 100%) and the web platform maintained full screen mode throughout the experiment.

Participants

35 participants (31 males and 4 females) from different international universities and companies did the subjective test. They were recruited by emails and did the subjective test in two sessions (88 stimuli each). For each session, each participant received a link by email to launch the experiment. The link for the second session was sent after completion of the first one. For each participant, two unique random playlists of stimuli were generated offline for the two sessions. 3 out of 35 participants could not complete the second session due to technical problems. Overall, each of the 176 stimuli is associated with between 32 and 35 quality scores, for a total of 5896 collected scores.

Analysis of subjective data

We computed the mean opinion score (MOS) and corresponding confidence interval for each of the 176 stimuli, according to the ITU recommendations [35]. Figure 4 illustrates the results for each distortion type and source model. We can observe the strong annoying effect of the triangle holes on the perceived quality. On the contrary, the smooth artifacts due to compression are judged as much less visible. In Sect. 6, those subjective data will be used to optimize and evaluate several objective metrics (presented in Sects. 4 and 5)

Point-based objective quality metrics

With the conclusion of two international standards for point cloud compression (the Video-based Point Cloud Compression, V-PCC, and the Geometry-based Point Cloud Compression, G-PCC [36]), many metrics to measure point cloud distortion have also been proposed [7, 8]. In order to leverage the available point cloud distortion tools, dynamic textured meshes can be converted into colored point clouds and then the distortion between two point clouds can be measured with the available tools. The advantage of such solution is that dynamic textured meshes with different connectivity varying in time can be analyzed with the proposed scheme. However, the fidelity of the metric heavily depends on the mesh surface sampling strategy. Mesh sampling is devised in next section. We then present the two point-based metrics we evaluated; the MPEG PCC [8] and the PCQM [7].

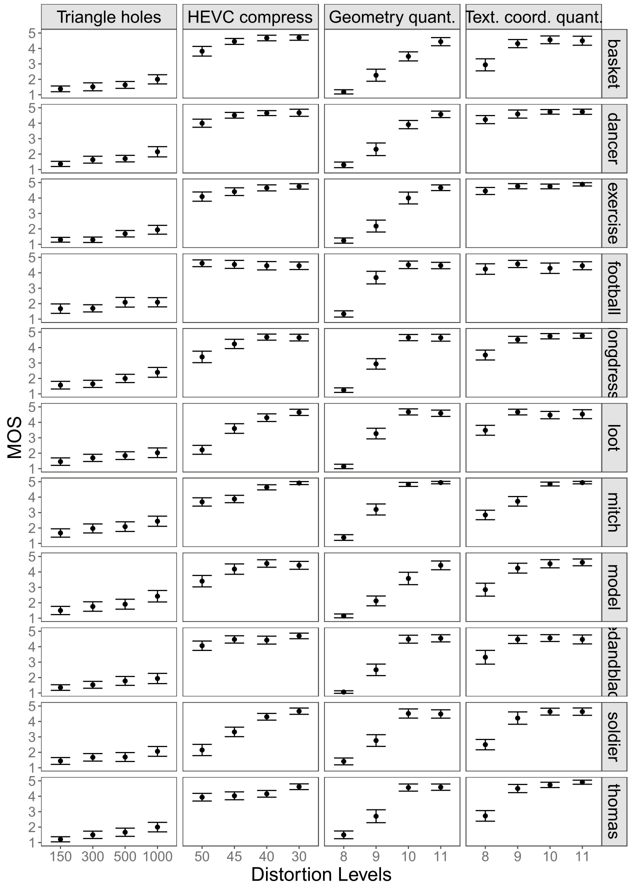


Fig. 4 Mean Opinion Scores (MOS) and 95% confidence intervals for all stimuli of our dataset

Mesh surface sampling

Mesh surface sampling is used in mesh evaluation tools like metro [37], which implements surface sampling strategies such as Monte Carlo sampling, face subdivision or similar triangle sampling. In our experiments, we sample both reference meshes and distorted ones with a grid sampling approach. We did experiments with several types of sampling and the grid one, presented hereafter, provided the most homogeneous results. Our method divides the sides of the sequence bounding box into a grid, and samples the surface of each mesh of the sequence by performing ray-casting in the axis direction (x,y,z) from each point of the grid. The sequence bounding box is the union of all bounding boxes from each frame. For each triangle, the sampling direction selected (x, y or z) depends on the normal of the triangle, whereby the direction closer to the triangle's normal is used. A hit test determines if the casted ray touches the triangle, then the color is obtained by barycentric interpolation (to determine the UV coordinate of the point), and then bilinear interpolation (to get the RGB value from texture map) (Fig. 5).

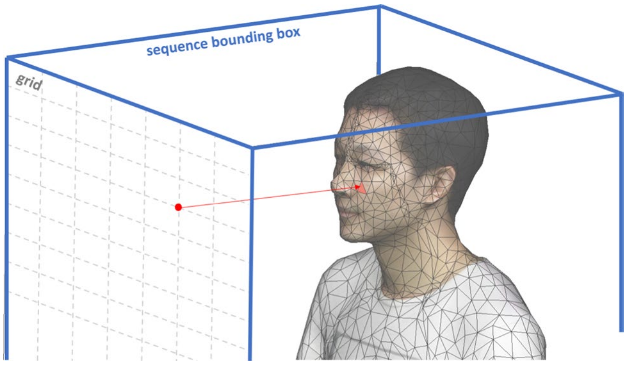


Fig. 5 Mesh grid sampling. The mesh surface points sampled by ray-casting from the grid at the sequence bounding box are added to the final point cloud

The MPEG PCC metric [8]

Geometry-based features

For geometry distortion, the MPEG PCC tool includes the point-to-point f_{D1} and point-to-plane f_{D2} metric features from [8]. Considering **A** and **B** as the original and distorted point clouds, respectively, f_{D1} is calculated in the following way:

$$f_{D1} = \max\Big(\frac{1}{N_A}\sum_{\forall a_j \in \mathbf{A}} \|E(i,j)\|_2^2, \quad \frac{1}{N_B}\sum_{\forall b_j \in \mathbf{B}} \|E(i,j)\|_2^2\Big) \tag{1}$$

where the variables N_A and N_B represent the total number of points in the sampled meshes **A** and **B**, and the error distance between two points is given by $E(i, j)$, which indicates the 3D distance between a point i and its nearest neighbor j, obtained using a kd-tree search.

For f_{D2}, the metric is obtained using the following equation:

$$f_{D2} = \max\Big(\frac{1}{N_A}\sum_{\forall a_j \in \mathbf{A}} (E(i,j) \cdot N_j)^2, \quad \frac{1}{N_B}\sum_{\forall b_j \in \mathbf{B}} (E(i,j) \cdot N_j)^2\Big) \tag{2}$$

where N_j is the normal of the point j, directly provided by the sampling algorithm in our use case.

Color-based features

In the case of texture, a similar metric as used for f_{D1} is applied to the point cloud colors. The only difference is that the range of colors is limited by the maximum color value, typically 255 for an 8-bit colored point cloud. Each color channel can be calculated separately, and usually a conversion from RGB to YUV is conducted using ITU-R BT.709 [38] before any measure is taken, since YUV space correlates better with human perception. A symmetric computation of the distortion is utilized, in the same way as is done for geometric distortions. The maximum distortion between the two passes is selected as the final distortion. The color distortion is similar to the f_{D1} metric, whereby the formula for one of the color channels is given as:

$$f_c = \max(\frac{1}{N_A} \sum_{\forall a_j \in \mathbf{A}} E_C(i,j)^2, \\ \frac{1}{N_B} \sum_{\forall b_j \in \mathbf{B}} E_C(i,j)^2 \tag{3}$$

where $E_C(i,j)$ indicates the difference between color values at point i and at its nearest neighbor j, obtained using a kd-tree search and $c \in [y,u,v]$.

The PCQM metric [7]

PCQM is based on the following steps: first a correspondence is established between the point clouds being compared; for each point p from the reference pristine point cloud R its corresponding point $\hat{p}$ on the distorted point cloud D is computed (using a local least-squares fitting of a quadric surface). Then, a set of geometric and color features is computed locally, i.e. over local neighborhoods around each point p of R and their corresponding points $\hat{p}$ on D.

Geometry-based features

The geometry-based features of PCQM are based on the *mean curvature* information ρ and were inspired by the SSIM image metric from Wang et al. [39], who consider that the human visual system is highly adapted for extracting structural information. Hence the geometry-based features aim to capture differences of surface structure (captured via curvature statistics).

$$\text{Curvature comparison} \quad f_1^p = \frac{\left\| \mu_p^\rho - \mu_{\hat{p}}^\rho \right\|}{max(\mu_p^\rho, \mu_{\hat{p}}^\rho) + k_1} \tag{4}$$

$$\text{Curvature contrast} \quad f_2^p = \frac{\left\| \sigma_p^\rho - \sigma_{\hat{p}}^\rho \right\|}{max(\sigma_p^\rho, \sigma_{\hat{p}}^\rho) + k_2} \tag{5}$$

$$\text{Curvature structure} \quad f_3^p = \frac{\left\| \sigma_p^\rho \sigma_{\hat{p}}^\rho - \sigma_{p\hat{p}}^\rho \right\|}{\sigma_p^\rho \sigma_{\hat{p}}^\rho + k_3} \tag{6}$$

where k_i are constants to avoid instability when denominators are close to zero. μ_p^ρ, $\mu_{\hat{p}}^\rho$ are Gaussian-weighted averages of curvature over the 3D points belonging to local neighborhoods. Similarly σ_p^ρ, $\sigma_{\hat{p}}^\rho$ and $\sigma_{p\hat{p}}^\rho$ are standard deviations and covariance of curvature over these neighborhoods.

Color-based features

The color-based features of PCQM are extensions for 3D points clouds of the work from Lissner et al. [40], who proposed several features for the quality evaluation of 2D images. Their features are inspired by SSIM but also integrate chromatic components and demonstrated excellent results for image quality assessment. As in [40], the color-based features of PCQM are computed in the perceptually uniform color space LAB200HL [41], in which each vertex p has of a lightness and two chromatic values (L_p, a_p, b_p). The chroma of p is defined as: $Ch_p = \sqrt{a_p^2 + b_p^2}$.

$$\text{Lightness comparison} \quad f_4^p = \frac{1}{k_4.(\mu_p^L - \mu_{\hat{p}}^L)^2 + 1} \tag{7}$$

$$\text{Lightness contrast} \quad f_5^p = \frac{2\sigma_p^L \sigma_{\hat{p}}^L + k_5}{\sigma_p^{L^2} + \sigma_{\hat{p}}^{L^2} + k_5} \tag{8}$$

$$\text{Lightness structure} \quad f_6^p = \frac{\sigma_{p\hat{p}}^L + k_6}{\sigma_p^L \sigma_{\hat{p}}^L + k_6} \tag{9}$$

$$\text{Chroma comparison} \quad f_7^p = \frac{1}{k_7.(\mu_p^c - \mu_{\hat{p}}^c)^2 + 1} \tag{10}$$

$$\text{Hue comparison} \quad f_8^p = \frac{1}{k_8.\overline{\Delta H_{p\hat{p}}}^2 + 1} \tag{11}$$

with

$$\Delta H_{p\hat{p}} = \sqrt{(a_p - a_{\hat{p}})^2 + (b_p - b_{\hat{p}})^2 - (c_p - c_{\hat{p}})^2}$$

and $\overline{\Delta H_{p\hat{p}}}$ is the Gaussian-weighed average over neighborhoods. These color-based features are inverted ($f_i^p = 1 - f_i^p$) so that a value of 0 indicates that there is no local geometric and color distortion around p. All features $\in [0, 1]$.

The geometric and color features detailed above are computed for each point of R. In order to obtain global features f_j, local values of each feature are then averaged:

$$f_i = \frac{1}{|R|} \sum_{p \in R} f_i^p \tag{12}$$

The global distortion score is then defined as a linear combination of f_j, optimized through logistic regression.

$$PCQM = \sum_{i \in S} w_i f_i \tag{13}$$

S is the set of feature indices of the linear model. w_j weights the contribution of each feature to the overall distortion score. The optimal subset of features and their corresponding weights w_j are usually determined through an optimization computed through cross-validation.

Image-based objective quality metric

The point-based metrics presented in the previous section involves multiple nearest points search, based on KD-tree, leading to very long execution times (see Sect. 6.1), especially since accurate sampling generates many points and our test models present long sequences of 300 frames. Inspired by Lindstrom's work [25] and more recent works such as [26, 27, 42] we implemented an image-based metric applied to the output of the rendering of the volumetric videos. This approach is suitable for our kind of contents that are localized, generally watertight and presents few occlusions. It is also possible to implement very efficient parallel implementation of such metric by leveraging modern graphics hardware. We present in the next section the definition of our Image Based Sampling Metric (IBSM), which makes use of rendered images but also of rendered masks and accompanying depth buffers generated during the rendering.

The image-based sampling metric (IBSM)

The computation of the IBSM features, is based on image and depth buffer MSE processing. An overview of the approach is given in Fig. 6. For each frame, the reference and the distorted models are rendered for several view directions vd_i, using an orthographic projection (see Fig. 7). The images obtained from the rendering of reference and distorted models are then compared using some adapted image MSE metrics. The results are averaged over a set of view directions for the frame and over the frames of the sequence.

Rendering of one view

The rendering of one view is illustrated in Fig. 7. The pixels of the image could be obtained by ray tracing or rasterization of the mesh. We used rasterization for performance reasons. The bounding sphere is obtained by summing the axis aligned bounding box of the distorted and reference objects and taking the diagonal and center of the resulting bounding box. The view directions vd_i always points toward the center of the bounding sphere in 3D space. The mesh is rendered using an orthogonal projection. The projection plane for the direction vd_i is the plane tangent to the bounding sphere and perpendicular to the view direction vector. The mesh is rendered using clockwise (CW) back-face culling which suits all the models of our dataset in terms of visual rendering.

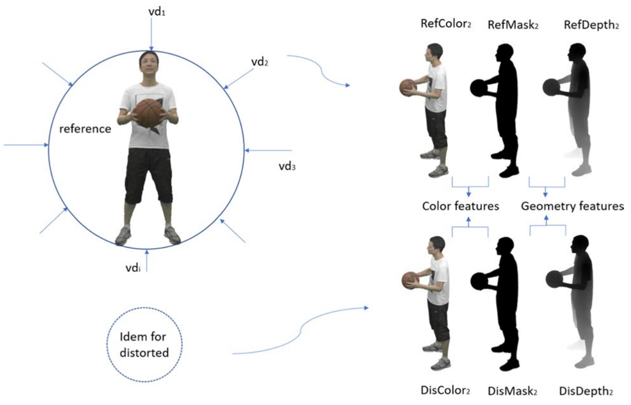


Fig. 6 Overview of the image-based metric. Ref/DisColor are the color images/buffers. Ref/DisMask are binary images where pixel[i,j]=1 if a projection exists in associated color buffer, and 0 otherwise. Ref/DisDepth are the depth buffers. All the buffers have same square dimensions of w pixels

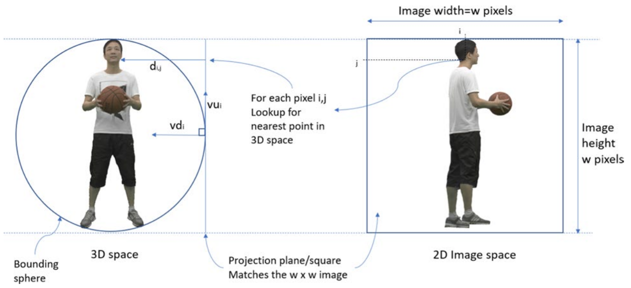


Fig. 7 Rendering of image buffers for a given view direction vd_i

The rendering step generates, for each view, a *color*, a *mask*, and a *depth* buffer. The *color buffer* contains for each pixel i,j the RGB value, of the nearest projected triangle. In case of textured meshes, the RGB color is obtained by bilinear interpolation of the texture map using triangle UV coordinates. In case of color per vertex meshes (no texture map), the vertex colors are blended using barycentric coordinates. Once RGB values are generated, additional YUV values are computed per pixel by using ITU-R BT.709 [38], since YUV space is known to better correlate with human perception.

The *mask buffer* contains for each pixel i,j a binary value set to 1 if a projection for this pixel exists and 0 otherwise. Finally, the *depth buffer* contains for each pixel i,j the distance $d_{i,j}$ from the projection plane to the 3D surface in 3D space.

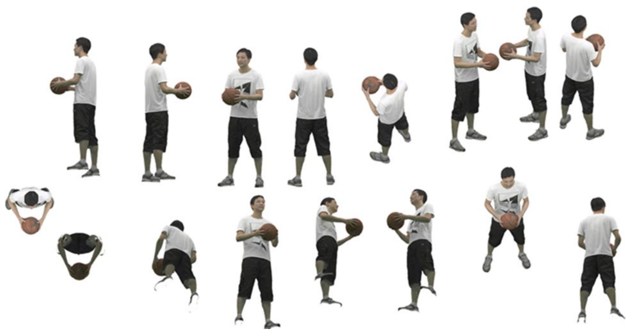


Fig. 8 Example of color images generated for the basketball player using $nv = 16$ views

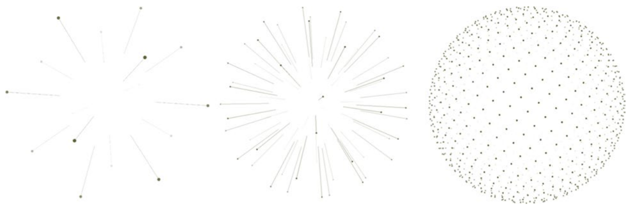


Fig. 9 Fibonacci sphere sampling. Left: $nv = 16$ samples with view directions. Center: $nv = 128$ samples with view directions. Right: $nv = 1024$ samples to better visualize the sampling distribution

Selection of views

The positioning of the views, that is to say the set of orientations, is obtained by using a Fibonacci sphere lattice. This distribution aims at generating points over a sphere in an evenly spaced manner (see Fig. 9). Once one has the points P_i, the directions vd_i are the vectors passing through these points and pointing toward the center of the sphere. Figure 8 gives an example of generated images for 16 views using this method.

The Fibonacci sphere samples (i.e. the view directions) are computed as follows. Let nv be the target number of views (i.e. of camera directions) and $i \in [1, nv]$ be the index of the view for which we need to compute the camera direction vd_i. Let $y = 1 - 2\frac{i-1}{nv-1}$ vary from 1 to -1. Let $r = \sqrt{1 - y^2}$ be the radius at y. Let ϕ be the golden angle, $\phi = \pi(3 - \sqrt{5})$. The view direction vd_i is defined as follows:

$$vd_i = \begin{bmatrix} r \cdot \cos(\phi \cdot i) \\ y \\ r \cdot \sin(\phi \cdot i) \end{bmatrix} \tag{14}$$

A special attention is also given to generate the up vectors vu_i (see Fig. 7). The up vector vu_i determines, for a given direction vd_i, the rotation of the camera on the direction axis, and by side effect the final orientation of the model in the image. At the end, we obtain the camera matrix using the well-known LookAt function from OpenGL. For a given view direction vd_i, the up vector (vu_i) is defined by Eq. 15.

$$vu = \begin{cases} (0,0,1), \text{ if } (vd_x, |vd_y|, vd_z) = (0,1,0) \\ (0,1,0), \text{ otherwise} \end{cases} \tag{15}$$

In other terms, whenever view direction vu_i is not a north or south pole, we use y axis vector as the up vector vu_i, otherwise we use the z axis vector as the up vector vd_i. In practice we use a threshold of angle between vd and (0,1,0) due to limited floating point precision. Those values were selected so most of our models visually renders head at top of image when rendered by side or front views. The special case of poles is the simplest positive default value for vu_i.

Distortion features computation

The calculus of the distortion is based on the general mean squared error formula. Let $Y_{i,j}$ be a sample of an original image, $\hat{Y}_{i,j}$ a sample of a distorted image and n the width of both images in pixels. The MSE for the two images is calculated as follows:

$$MSE = \frac{1}{n^2} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} (Y_{i,j} - \hat{Y}_{i,j}) \tag{16}$$

In the rest of this document, we adapt this formula to restrict the processing to parts where the mask generated from the reference and distorted models is equal to one. Let m be a matrix, of same size as references mask RefMask noted mr and distorted mask DisMask noted md, which contains for each pixel i, j a binary value set to 1 if a projection for this pixel exists and 0 otherwise. Let m_v, mr_v and md_v be the masks for the given view v.

$$m_{vij} = \begin{cases} 1, \text{ if } mr_{vij} + md_{vij} = 2 \\ 0, \text{ otherwise} \end{cases} \tag{17}$$

Let w be the width of the buffers in pixels. The combined number of projected pixels ns of all the view directions nv for one frame is:

$$ns = \sum_{v=1}^{nv} \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} m_{vij} \tag{18}$$

Color-based features

Let c_{vij} be a sample i, j of the color image/buffer of the reference model for view direction v, $\hat{c}_{vij}$ a sample i, j of the color image/buffer of the distorted model for view direction v. The color MSE feature for the YUV images, noted f_c, is calculated as follows on each color component c in $[y, u, v]$:

$$f_c = \frac{1}{ns} \sum_{v=1}^{nv} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} m_{vij} \cdot (c_{vij} - \hat{c}_{vij})^2 \tag{19}$$

A summary color feature f_{yuv} can also be computed, as commonly used in MPEG standards, as follows:

$$f_{yuv} = \frac{6.f_y + f_u + f_v}{8} \tag{20}$$

Geometry-based features

Let d_{vij} be a sample of the depth image/buffer of the reference model for view direction v, $\hat{d}_{vij}$ a sample of the depth image/buffer of the distorted model for view direction v. Let *sigDynamic* be the dynamic of the depth signal initialized with the diagonal of the bounding box of both models. The depth MSE feature for the depth images/buffers, noted f_d, is calculated as follows:

$$f_d = \frac{1}{ns} \sum_{v=1}^{nv} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} m_{vij} \cdot \frac{((d_{vij} - \hat{d}_{vij}) \cdot 255)^2}{sigDynamic} \tag{21}$$

The depth re-normalization to 255 is used to get geometric MSEs comparable to the color ones.

The metric also reports the ratio of unmatched samples feature f_{ru}, which corresponds to holes and unmatched silhouettes, defined as follows:

$$f_{ru} = \frac{nus}{ns} \tag{22}$$

Where the number of unmatched samples *nus* value is given by:

$$nus = \sum_{v=1}^{nv} \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} \begin{cases} 1, \text{ if } mr_{vij} + md_{vij} = 1 \\ 0, \text{ otherwise} \end{cases} \tag{23}$$

An additional feature f_{rh} provides the distinction between holes and silhouette with a minimum cost of additional processing and few false positives. More precisely, it provides the percentage of pixels occupied by holes *rh* not taking into account silhouette differences. The silhouette unmatched ratio f_{rs} is computed by subtracting the percentage of holes from the percentage of unmatched pixels: $f_{rs} = f_{ru} - f_{rh}$. Indeed, the unmatched pixels correspond to the pixels where there is a projection for one model but not for the other. It thus corresponds to the silhouette distortion plus the eventual holes on the surface of the distorted model. To compute f_{rh} we render an additional pass for each view v of the *distorted* model with back-face culling disabled where internal side of the model faces can thus be seen through potential holes and obtain an $md'_v ij$ mask. Let mh_{vij} be the mask detecting holes defined as follows:

$$mh_{vij} = \begin{cases} 1, \text{ if } mr_{vij} + md'_{vij} = 2 \text{ and } md_{vij} = 0 \\ 0, \text{ otherwise} \end{cases} \tag{24}$$

We then compute f_{rh} the ratio of pixels corresponding to holes as:

$$f_{rh} = \frac{1}{ns} \sum_{v=1}^{nv} \sum_{i=0}^{w-1} \sum_{j=0}^{w-1} mh_{wij} \tag{25}$$

Temporal distortion features

We introduce temporal features as an extension of the features described above. The idea is to compare if the distorted signal evolves in the same manner as the reference signal over time. We evaluate the evolution of the signal between frame n and $n - 1$, by computing color and depth difference between those frames. We then compute the distances between reference and distorted difference buffers by computing an MSE for each component (depth, y,u and v). The obtained MSEs are the temporal features. Details are provided below.

The first frame has no temporal features. Then starting from the second frame we first compute, for each pixel of color and depth buffers of all views, the *differences* between frame n and frame $n - 1$. We store those results into intermediate buffers.

Reference color difference *crd*, with $cr(n)$ the reference color of frame n:

$$crd_{vij}(n) = cr_{vij}(n) - cr_{vij}(n-1) \tag{26}$$

Distorted color difference *cdd*, with $cd(n)$ the distorted color of frame n:

$$cdd_{vij}(n) = cd_{vij}(n) - cd_{vij}(n-1) \tag{27}$$

Reference depth difference *drd*, with $dr(n)$ the reference depth of frame n:

$$drd_{vij}(n) = dr_{vij}(n) - dr_{vij}(n-1) \tag{28}$$

Distorted depth difference *ddd*, with $dd(n)$ the distorted depth of frame n:

$$ddd_{vij}(n) = dd_{vij}(n) - dd_{vij}(n-1) \tag{29}$$

These formulas are used to compare if the distorted signal evolves in a similar manner as the reference signal over time. We compute the MSE between reference and distorted difference buffers to obtain our two temporal features, respectively related to color and depth:

$$f_{tc} = \frac{1}{ns} \sum_{v=1}^{nv} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} tm_{vij} \cdot (crd_{vij} - cdd_{vij})^2 \quad (30)$$

$$f_{td} = \frac{1}{ns} \sum_{v=1}^{nv} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} tm_{vij} \cdot \frac{((drd_{vij} - ddd_{vij}) \cdot 255)^2}{sigDynamic} \quad (31)$$

With tm the temporal mask with tm_{vij} set to one if m_{vij} of $distorted(n)$, $distorted(n-1)$, $reference(n)$ and $reference(n-1)$ are all non null. Otherwise $tm_{vij} = 0$.

Calculus of PSNRs

Having the different MSE features previously presented it is also possible to compute PSNR values since those can be more convenient for human interpretation. The $PSNR_k$ with $k \in [d, y, u, v]$ is defined as follows:

$$PSNR_k = 10 \log_{10} \left(\frac{255^2}{f_k} \right) \quad (32)$$

Summary

To summarize, the *IBSM* metric features are the following, their values are obtained by computing their mean for the entire sequence (Table 2).

Metric calibration

We present in this section the calibration process aiming at selecting the most suitable parameters of IBSM for our context of volumetric videos representing character performances.

Sensitivity to size and number of views

Our objective is to study the behavior of the IBSM metric when varying the generated image resolution w and the number of views nv. For this test we experimented with the simple IBSM version, and ran the metric on the first frame of all our models with varying $Qp \in \{8, 10, 12, 14, 16, 18, 20, 24, 28, 30\}$, metric parameters $w \in \{512, 1024, 2048, 4092\}$ and $nv \in \{16, 18, 24, 32, 64\}$. Figure 10 shows the $PSNR_d$ obtained for the Mitch model by varying these parameters. These results are representative of the metric sensitivity for all the other sequences, all the other parameter variations and all the other features of IBSM. In this example we can see that resolutions of 512 and 1024 are usually unstable unless we use many views to compensate (see 1024 with 64 and 32 views). We can also see that 64 views with 4K res is very stable and shall be considered the reference to compare with. By analyzing the results for all models, we extracted some parameters described in Table 3. In the context of the MPEG V-Mesh activity we proposed to use 16 views of 2K resolution as a good compromise between results stability and execution performance.

Table 2 The different IBSM metrics experimented. *IBSM* is the set of features used for MPEG V-Mesh standard. $IBSM_h$ includes additional holes features and $IBSM_{ht}$ adds temporal features on top of $IBSM_h$

Name	f_d	f_y	f_u	f_v	f_{rus}	f_{rh}	f_{td}	f_{ty}	f_{tu}	f_{tv}
IBSM	✓	✓	✓	✓	✓					
$IBSM_h$	✓	✓	✓	✓	✓	✓				
$IBSM_{ht}$	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

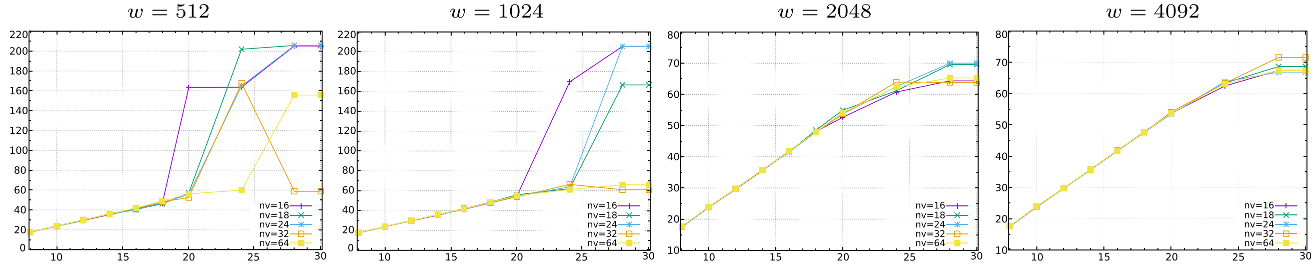


Fig. 10 For each chart, the horizontal axis is Q_p (the positions quantization bits) and the vertical axis is $PSNR_d(Q_p, w, nv, Q_t = 30)$ for the first frame of Mitch model using software rasterization

Table 3 Expert classification of *nv* and *w* parameters according to experimentation results. See Sect. 6.3 for the specifications of the hardware used for the tests

nv	*w*	Quality/stability	Seconds per frame
64	4K	Reference, “perfect”	up to 50
32	2K	Very stable	6 to 10
18	2K	Stable	4 to 6
16	2K	Stable	3.5 to 5
32	1K	Acceptable	3 to 4

Sensitivity to rig rotation

The purpose of this experimentation is to evaluate the impact of the orientation of the camera rig (set of view directions) on the results of $IBSM(nv = 16, w = 2K)$. We selected three orientations of the rig (defined by the polar angle and the azimuth angle of the rotation axis and the rotation magnitude around the center of the sequence bounding sphere) in [(0, 0, 0), (0, 0, 90), (45, 45, 45)].

In a first test we did run the metric over the available sequences of the MPEG V-Mesh Anchor which are based on this paper sequences including several distortions such as decimation, geometry quantization, texture resizing and texture compression. In our observations we noticed that the rate point that shows higher discrepancies was R5 (highest rate), but we found that the largest difference between the different rig orientations is only 0.4 dB in a single frame, and in average is usually around 0.04 dB for geometry $PSNR_d$.

We also tested the effect on our models including only texture compression based on the MPEG HM encoder. We observed that the averaged PSNR is very close for each rotation. As an example, for basketball player at texture quantization $Q_m = 42$, the maximum $PSNR_y$ difference is about 0.08 dB. The reason is that the geometry plays an important role in the discrepancy due to silhouettes penalty, in this experiment the geometry is lossless, so the PSNR difference from different camera angles is smaller than for previous experiments.

To conclude, experimentation with the rig rotation did not emphasise strong discrepancies when using different orientations. We thus recommend using the default parameters of the metric (no rotation) for the MPEG V-Mesh CfP and subsequent uses.

Results and evaluation

This section presents a comprehensive evaluation and comparison of the metrics presented above. For each of them, we adopt the scheme from PCQM: each metric is defined as a linear combination of its features; the optimal subset of features and their corresponding weights are obtained through an optimization computed by logistic regression. This protocol and the results are described below.

Performance evaluation measures

In order to evaluate the performance of objective metrics presented above, we compare the predicted quality scores given by these metrics to the ground truth subjective data from our subjective experiment. The standard performance evaluation measure consists in computing the Pearson Linear Correlation Coefficient (PLCC) and the Spearman Rank Order Correlation Coefficient (SROCC) between the metric predictions and subjective scores (MOS). These indices measure, respectively, the accuracy and the monotonicity of the predictions. Note that the Pearson correlation (PLCC) is computed after a logistic regression which provides a non-linear mapping between the objective and subjective scores. This allows the evaluation to take into account the saturation effects associated with human senses.

Metric optimization and evaluation protocol

Each metric involves a number of different features. All these features are not necessarily significant and some may be redundant. Moreover, integrating too many features may lead to over-fitting. Therefore, to select an optimal subset of features for each metric and fairly evaluate them, we adopt the cross-validation and feature selection protocol from [7, 12], as follows: we split the dataset into a *training set* that serves to optimize feature weights using logistic regression and a *test set* used for testing the obtained metrics. The splitting is done according to the source models, ensuring that the test set does not contain any of the models used for training, regardless of the distortion. Given that there are 11 source volumetric videos (VVs) in our dataset, we consider 8 sources and their distortions out for training (i.e., a total of $8 \times 16 = 128$ models), and the remaining 48 VVs (from 3 sources) for testing. We repeat this operation using the $\binom{11}{8} = 165$ possible splittings, and report the average performance.

For each metric, considering that we have n features, there are 2^{n-1} possible combinations of features. We exhaustively search through all possible combinations of features, and select the feature subset that generates the best average performance over all the test sets (165 folds) in terms of the mean of PLCC and SROCC.

Table 4 Learning and testing *including* all distortions from dataset

PCQM	config	Pearson	Pear. Std-Dev	Spearman	Spear. Std-Dev	sec. per seq.	f1	f2	f3	f4	f5	f6	f7	f8		
PCQM	Grid 512	0.459	0.055	0.498	0.064	1287.930	−132.558	118.385	11802.780	−121.877	8.420	−1.773	0.000	38.244		
PCQM	Grid 1024	0.582	0.057	0.612	0.061	5302.230	−19.004	0.000	73.743	−224.137	0.000	6.038	0.000	−14.821		
PCQM	Grid 2048	0.545	0.051	0.565	0.058	35683.730	80.949	−85.972	6.947	−268.751	0.000	5.723	0.000	−14.602		
PCC	config	Pearson	Pear. Std-Dev	Spearman	Spear. Std-Dev	sec. per seq.	D1	D2	y	u	v					
PCC	Grid 512	0.394	0.038	0.523	0.069	808.700	0.000	1.638	0.000	0.000	0.000					
PCC	*Grid 1024*	0.394	0.038	0.523	0.069	2781.347	0.000	1.643	0.000	0.000	0.000					
PCC	Grid 2048	0.394	0.038	0.524	0.068	10942.530	0.000	1.645	0.000	0.000	0.000					
PCC	Grid 4092	0.369	0.061	0.525	0.069	42504.540	1.114	0.000	0.000	0.000	0.000					
IBSM	config	Pearson	Pear. Std-Dev	Spearman	Spear. Std-Dev	sec. per seq.	d	y	u	v	ru	rh	td	ty	tu	tv
IBSM	16 × 512	0.604	0.038	0.638	0.051	240.032	1.400	0.000	0.000	0.000	0.000	na	na	na	na	na
IBSM	*16 × 2048*	0.604	0.038	0.638	0.051	876.380	1.399	0.000	0.000	0.000	0.000	na	na	na	na	na
IBSM$_h$	16 × 512	0.874	0.030	0.850	0.047	312.667	0.814	0.010	0.000	0.000	0.000	10.453	na	na	na	na
IBSM$_h$	16 × 1024	0.874	0.030	0.850	0.047	635.585	0.813	0.010	0.000	0.000	0.000	10.454	na	na	na	na
IBSM$_h$	16 × 2048	0.874	0.030	0.850	0.047	1572.485	0.814	0.010	0.000	0.000	0.000	10.452	na	na	na	na
IBSM$_h$	16 × 4092	0.874	0.030	0.850	0.047	5740.434	0.814	0.010	0.000	0.000	0.000	10.452	na	na	na	na
IBSM$_{ht}$	16 × 512	**0.875**	0.032	**0.865**	0.049	**363.647**	0.000	0.011	0.000	0.000	0.000	10.481	0.428	0.000	0.000	0.000
IBSM$_{ht}$	16 × 1024	**0.875**	0.031	**0.865**	0.049	**632.015**	0.000	0.011	0.000	0.000	0.000	10.481	0.428	0.000	0.000	0.000
IBSM$_{ht}$	16 × 2048	**0.876**	0.031	**0.865**	0.049	***2626.47***	0.000	0.011	0.000	0.000	0.000	10.479	0.428	0.000	0.000	0.000

Italic rows represents metric version/configuration used for the MPEG V-Mesh CfP. Bold represents best Pearson/Spearman/execution time results. Bolditalic highlights high execution times among those best metrics

Table 5 Learning and testing *excluding* triangle holes distortions (surface cracks) from dataset

PCQM	config	Pearson	Pear. StdDev	Spearman	Spear. StdDev	sec. per seq.	f1	f2	f3	f4	f5	f6	f7	f8		
PCQM	Grid 512	0.83	0.09	0.72	0.10	1288.75	−194.91	59.03	0.00	−191.22	−2.81	1.02	−168.79	55.72		
PCQM	Grid 1024	**0.90**	0.06	**0.79**	0.06	***5310.74***	103.43	−115.07	53.38	−205.47	0.00	−2.05	−121.90	24.92		
PCQM	Grid 2048	**0.93**	0.05	**0.80**	0.07	***35836.91***	31.58	−36.14	2.57	−256.80	−0.60	−2.03	0.00	8.97		
PCC	config	Pearson	Pear. StdDev	Spearman	Spear. StdDev	sec. per seq.	D1	D2	y	u	v					
PCC	Grid 512	0.877	0.058	0.760	0.098	816.570	0.000	2.550	1180.977	0.000	0.000					
PCC	*Grid* 1024	0.872	0.057	0.760	0.095	2785.068	0.000	2.631	1187.430	0.000	0.000					
PCC	Grid 2048	0.872	0.056	0.757	0.097	10948.650	0.000	2.666	1194.089	0.000	0.000					
PCC	Grid 4092	0.872	0.055	0.759	0.096	42538.380	0.000	2.675	1195.708	0.000	0.000					
IBSM	config	Pearson	Pear. StdDev	Spearman	Spear. StdDev	sec. per seq.	d	y	u	v	ru	rh	td	ty	tu	tv
IBSM	16 × 512	0.877	0.049	0.711	0.097	241.445	0.967	0.015	0.000	0.000	0.000	na	na	na	na	na
IBSM	*16 × 2048*	0.878	0.049	0.711	0.097	878.424	0.967	0.015	0.000	0.000	0.000	na	na	na	na	na
IBSM_h	16 × 512	0.877	0.048	0.713	0.096	314.173	0.928	0.016	0.000	0.000	0.000	8.720	na	na	na	na
IBSM_h	16 × 1024	0.877	0.048	0.712	0.096	637.893	0.927	0.016	0.000	0.000	0.000	8.738	na	na	na	na
IBSM_h	16 × 2048	0.877	0.048	0.712	0.096	1572.307	0.927	0.016	0.000	0.000	0.000	8.758	na	na	na	na
IBSM_h	16 × 4092	0.877	0.048	0.712	0.096	5811.036	0.927	0.016	0.000	0.000	0.000	8.760	na	na	na	na
IBSM_{ht}	16 × 512	**0.872**	0.061	**0.736**	0.102	**365.055**	0.000	0.016	0.000	0.000	0.000	0.000	0.520	0.000	0.000	0.000
IBSM_{ht}	16 × 1024	0.872	0.060	0.736	0.102	634.020	0.000	0.016	0.000	0.000	0.000	0.000	0.519	0.000	0.000	0.000
IBSM_{ht}	16 × 2048	0.872	0.060	0.736	0.102	2632.7	0.000	0.016	0.000	0.000	0.000	0.000	0.520	0.000	0.000	0.000

Italic rows represents metric version/configuration used for the MPEG V-Mesh CfP. Bold represents best Pearson/Spearman/execution time results. Bolditalic highlights high execution times among those best metrics

Performance of the metrics and comparisons

Table 4 summarizes the Pearson and Spearman correlations obtained for the PCQM, PCC and IBSM metrics, with varying parameters, after the learning approach described above. The standard deviations of the Pearson and Spearman correlations are also reported. The lower the standard deviation, the more confidence we can have in the learning results. It means that each splitting leads to similar results. The execution times for a sequence of 300 frames are reported in seconds. All the experiments were run on an Intel(R) Xeon(R) CPU E5-2687W v3 with 20 cores cadenced at 3.10 GHz and 32 GB of RAM. PCC and PCQM benefit from a parallel implementation leveraging this multi-core architecture, whereas IBSM uses a single-threaded implementation. This timing information is important in the MPEG context since enhancing a codec prototype requires validation with many executions of the metric over all the test contents, which can quickly become a limitation if execution time is too important. Finally, the weights of the different features obtained by the learning (averaged over all splittings) are also reported.

From the results of Table 4 one can see that the $IBSM_{ht}$ outperforms all the other metrics and parameter configurations, in terms of Pearson/Spearman correlation with reasonable execution times. One can also see that with the range of distortions used in the dataset, using the $IBSM_{ht}(nv = 16, w = 512)$ configuration provide similar results as more costly configurations.

Our dataset is particularly challenging especially because of the triangle hole distortions. If we remove the *triangle holes* distortion from the dataset and re-run the learning/testing protocol we obtain the results presented in Table 5. In this case the PCQM is providing best results in terms of correlation, but at the cost of impracticable execution times. We can also conclude that PCQM and PCC are very weak at handling hole distortions. Finally, we see that the PCC and IBSM metrics produce quite similar correlation results in the case of no holes in the dataset. Still, IBSM produces good correlated results with the best execution time.

Conclusion

In this work, we designed and produced a dataset of 176 subjectively-rated volumetric videos (VVs), represented as sequences of textured meshes. This dataset was created by introducing distortions on 11 pristine source VVs. Subjective scores were obtained through a subjective study based on the DSIS protocol and gathering 5896 subjective scores. This dataset allowed us to benchmark and calibrate several objective quality metrics: two model-based metrics (PCC and PCQM) and one image base approach (IBSM) for which we introduced two new features that specifically detect holes and temporal defects. Tested metrics are defined as linear combinations of several features. To optimize and fairly compare them, we trained them and selected the optimal subset of features using cross-validation on our dataset.

Extensive evaluations show that the image-based metric IBSM offers the best results, especially thanks to our newly introduced features for hole detection and temporal effect modeling. Moreover, it is the fastest to compute in less than 370 s per VV of 300 frames. Much further research remains to be done on the quality assessment of volumetric videos. In the near future, we plan to further explore temporal pooling and improve the modeling of temporal distortions. For this task, we should take inspiration from the recent 2D video quality metrics like VMAF to extend the IBSM metric. Integrating an attention model, e.g., for increasing the perceptual importance of salient regions like faces and hands, is also a perspective of great interest. Finally, proposing a large dataset with mixed distortions (like did Nehmé et al. [11] for static 3D meshes) would be very beneficial for the volumetric video quality assessment community.

Acknowledgements The authors would like to thank their colleagues at InterDigital and in the MPEG-3DGC group, for valuable discussions that led to the formulation of this work and for cross-validation of results. We also thank all the contributors (individuals, universities and companies) to the perceptual evaluation tests for spending time in running the experimentation from their premises and providing feedback on the remote experimental setup.

Declarations

conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. MPEG CfP for Dynamic Mesh Coding. [Online]. Available: https://www.mpeg.org/wp-content/uploads/mpeg_meetings/136_OnLine/w20972.zip
2. Valenzise G, Zerman E, Ozcinar C (eds) (2022) Immersive Video Technologies. Elsevier
3. Jvd Hooft, Vega MT, Wauters T, Timmerer C, Begen AC, Turck FD, Schatz R (2020) From capturing to rendering: Volumetric media delivery with six degrees of freedom. IEEE Commun Mag 58(10):49–55
4. Zell E, Castan F, Gasparini S, Hilsmann A, Kazhdan M, Tagliasacchi A, Zarpalas D, Zioulis N (2021) Volumetric video - acquisition, compression, interaction and perception. Eurographics 2021 - Tutorials, p 5
5. Ak A, Zerman E, Ling S, Callet PL, Smolic A (2021) The effect of temporal sub-sampling on the accuracy of volumetric video quality assessment. In: Picture Coding Symposium (PCS). IEEE
6. Viola I, Subramanyam S, Li J, Cesar P (2022) On the impact of VR assessment on the quality of experience of highly realistic digital humans. Quality and User Experience, vol 7
7. Meynet G, Nehmé Y, Digne J, Lavoué G (2020) Pcqm: A full-reference quality metric for colored 3D point clouds. In: 2020

Twelfth international conference on quality of multimedia experience (QoMEX). IEEE, pp 1–6

8. Tian D, Ochimizu H, Feng C, Cohen R, Vetro A (2017) Geometric distortion metrics for point cloud compression. In: 2017 IEEE international conference on image processing (ICIP). IEEE, pp 3460–3464
9. Alexiou E, Nehmé Y, Zerman E, Viola I, Lavoué G, Ak A, Smolic A, Callet PL, Cesar P (2022) Subjective and objective quality assessment for volumetric video. In: Valenzise G, Alain M, Zerman E, Ozcinar C (eds). Immersive video technologies, Elsevier
10. Guo J, Vidal V, Cheng I, Basu A, Baskurt A, Lavoué G (2016) Subjective and objective visual quality assessment of textured 3D meshes. ACM Trans Appl Percept 14:1–20
11. Nehmé Y, Dupont F, Farrugia J-P, Le Callet P, Lavoué G (2022) Textured mesh quality assessment: Large-scale dataset and deep learning-based quality metric. arXiv preprint arXiv:2202.02397
12. Nehmé Y, Dupont F, Farrugia J-P, Le Callet P, Lavoué G (2021) Visual quality of 3D meshes with diffuse colors in virtual reality: Subjective and objective evaluation. IEEE Trans Visual Comput Graphics 27(3):2202–2219
13. Alexiou E, Viola I, Borges TM, Fonseca TA, de Queiroz RL, Ebrahimi T (2019) A comprehensive study of the rate-distortion performance in MPEG point cloud compression. APSIPA Trans Signal Inf Process 8:e27
14. Perry S, Cong HP, da Silva Cruz LA, Prazeres J, Pereira M, Pinheiro A, Dumic E, Alexiou E, Ebrahimi T (2020) Quality evaluation of static point clouds encoded using MPEG codecs. IN: 2020 IEEE International Conference on Image Processing (ICIP), pp 3428–3432
15. Su H, Duanmu Z, Liu W, Liu Q, Wang Z (2019) Perceptual quality assessment of 3D point clouds. In: IEEE international conference on image processing (ICIP), pp 3182–3186
16. Yang Q, Chen H, Ma Z, Xu Y, Tang R, Sun J (2020) Predicting the perceptual quality of point cloud: A 3D-to-2D projection-based exploration. In: IEEE Transactions on Multimedia 1
17. Viola I, Subramanyam S, Cesar P (2020) A color-based objective quality metric for point cloud contents. In: Twelfth international conference on quality of multimedia experience (QoMEX), pp 1–6
18. Alexiou E, Ebrahimi T (2020) Towards a point cloud structural similarity metric. In: IEEE international conference on multimedia expo workshops (ICMEW), pp 1–6
19. Yang Q, Ma Z, Xu Y, Li Z, Sun J (2020) Inferring point cloud quality via graph similarity. In: IEEE transactions on pattern analysis and machine intelligence 1
20. Liu Q, Yuan H, Su H, Liu H, Wang Y, Yang H, Hou J (2021) PQA-Net: Deep no reference point cloud quality assessment via multi-view projection. In: IEEE transactions on circuits and systems for video technology 1
21. Alexiou E, Viola I, Cesar P (2021) PointPCA: Point cloud objective quality assessment using PCA-based descriptors. Available: https://arxiv.org/abs/2111.12663
22. Zhang Y, Yang Q, Xu Y (2021) MS-GraphSIM: Inferring Point Cloud Quality via Multiscale Graph Similarity. Association for Computing Machinery, New York, pp 1230–1238
23. Diniz R, Freitas PG, Farias MCQ (2021) Color and geometry texture descriptors for point-cloud quality assessment. IEEE Signal Process Lett 28:1150–1154
24. Quach M, Chetouani A, Valenzise G, Dufaux F (2021) A deep perceptual metric for 3D point clouds. Electron Imag 2021(9):257
25. Lindstrom P, Turk G (2000) Image-driven simplification. ACM Trans Graph 19(3):204–241. https://doi.org/10.1145/353981.353995
26. Larkin M, O'Sullivan C (2011) Perception of simplification artifacts for animated characters. In: Proceedings of the ACM SIGGRAPH symposium on applied perception in graphics and visualization, ser. APGV '11. New York, NY, USA: Association for Computing Machinery, p. 93-100. Available: https://doi.org/10.1145/2077451.2077469
27. Lavoué G, Larabi C, Mohamed Váša L (2016) On the efficiency of image metrics for evaluating the visual quality of 3D models. IEEE Trans Vis Comput Graph 22(8):1987–1999
28. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612
29. Zerman E, Ozcinar C, Gao P, Smolic A (2020) Textured mesh vs coloured point cloud: A subjective study for volumetric video compression. In: 12th international conference on quality of multimedia experience (QoMEX)
30. Cao K, Xu Y, Cosman P (2020) Visual quality of compressed mesh and point cloud sequences. IEEE Access 8:171–203
31. Nehmé Y, Farrugia J-P, Dupont F, Callet PL, Lavoué G (2020) Comparison of subjective methods for quality assessment of 3D graphics in virtual reality. ACM Trans Appl Percept 18(1):1–23
32. Torkhani F, Wang K, Chassery JM (2015) Perceptual quality assessment of 3D dynamic meshes: subjective and objective studies. Signal Process Image Commun 31:185–204
33. Peirce JW (2007) Psychopy-psychophysics software in python. J Neurosci Methods 162(1):8–13
34. ITU (1999) Subjective video quality assessment methods for multimedia applications, recommendation ITU-T p.910. Int Telecommun Union
35. ITU (2012) Methodology for the subjective assessment of the quality of television pictures bt series broadcasting service. Int Telecommun Union. 13:1–48
36. Graziosi D, Nakagami O, Kuma S, Zaghetto A, Suzuki T, Tabatabai A (2020) An overview of ongoing point cloud compression standardization activities: Video-based (v-pcc) and geometry-based (g-pcc). APSIPA Trans Signal Inf Process 9:e13
37. Cignoni P, Rocchini C, Scopigno R (1998) Metro: measuring error on simplified surfaces. In: Computer graphics forum. vol 17(2). Wiley Online Library, pp 167–174
38. ITU (2015) Recommendation ITU-R BT.709-6. parameter values for the HDTV standards for production and international programme exchange," International Telecommunication Union. Available: https://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.709-6-201506-I!!PDF-E.pdf
39. Wang Z, Bovik A, Sheikh H, Simoncelli E (2004) Image quality assessment: From error visibility to structural similarity. In: IEEE transactions on image processing, vol. 13, no. 4, pp. 600–612. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1284395
40. Lissner I, Preiss J, Urban P, Lichtenauer MS, Zolliker P (2013) Image-difference prediction: from grayscale to color. IEEE Trans Image Process 22(2):435–446
41. Lissner I, Urban P (2012) Toward a unified color space for perception-based image processing. IEEE Trans Image Process 21(3):1153–1168
42. Krivokuća M, Wünsche BC, Abdulla W (2012) A new error metric for geometric shape distortion using depth values from orthographic projections. In: Proceedings of the 27th conference on image and vision computing New Zealand, ser. IVCNZ '12. New York, NY, USA: Association for Computing Machinery, p. 388-393. Available: https://doi.org/10.1145/2425836.2425911

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.