INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC 1/SC 29/WG 7
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC 1/SC 29/WG 7 **m56119**
Online – January 2021

Title: [PCC-MESH] Metric for mesh compression evaluation

Author: Danillo B Graziosi, Alexandre Zaghetto, Ali Tabatabai

Abstract

For the evaluation of mesh compression algorithms, a suitable objective metric is desired. Here we present a proposal for a metric based on the well-known D1/D2 metric used in point cloud compression. We extend the pcc_error software to accept mesh inputs, and texture from either png files or bgr/yuv videos and perform a surface sampling to generate a point cloud representation of the input mesh. With the sampled surface point cloud, we calculate D1/D2 and Y-PNSR metrics between two meshes. Notice that the metric can also be used between a mesh and a point cloud. We provide in the software implementation three alternative for mesh surface sampling (orthogonal projection, texel back-projection and regular surface sampling) and show objective and subjective results to validate the proposed metric.

Introduction

As presented in [1], the proposed evaluation metric shall be able to evaluate two meshes with different topology and vertex density. Moreover, the metric shall the able to cope with topology varying per frame. In addition to the evaluation of geometry distortion, the metric shall also provide distortion metrics to photometry characteristics of the mesh, such as color per vertex or colors from texture map.

In order to use the distortion metric software for point clouds, the pcc_error [2], a possible solution would be to sample the mesh's surface to obtain a point cloud, and then use pcc_error to evaluate the distortion of the sampled surface. Even though there are some known issues with this approach (for instance, holes and temporal inconsistencies are not well captured), the metric was thoroughly used during the point cloud standardization and could be a good starting point for a metric suitable for mesh compression.

Here we show results of the implementation of a surface sampling routine inside pcc_error, as shown in Figure 1. The implementation is available at the MPEG git repository, `6-mesh-support`.The software now accepts OBJ as inputs for the mesh content, and either a png file, or a video sequence in YUV420p or BGR444p formats. Internally the software samples the surface of the mesh according to three possible options: orthogonal projection, texel back-projection, or regular surface sampling, which will be explained in further detail ahead. Limitations and possible improvements of the implemented methods will also be discussed.

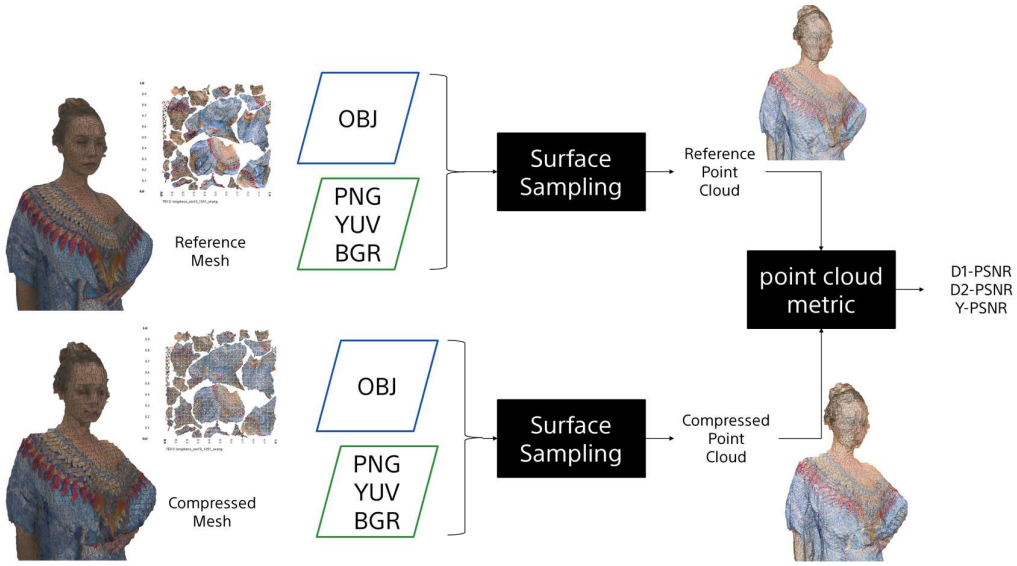

Figure 1: pcc_error modification for mesh evaluation

Mesh surface sampling

Orthogonal projection

The point cloud is created by performing ray-casting in the axis direction (x,y,z), depending on the normal of the triangle. A hit test determines if the casted ray hits 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). The coordinate of the point in the normal direction is rounded, while the other directions are the same, since they are obtained from in integer positions of the casted ray.

Since the meshes are voxelized, all the vertices are in an integer position of the 3D grid. If an orthogonal projection of each triangle is used, the exact position of the vertices will be part of the sampled surface. The color of those points is obtained through bilinear interpolation, if UV mapping is used, otherwise the vertex color should be used (however, in the current implementation vertex color was not tested). Figure 2 shows the surface sampling for the uncompressed longdress mesh, frame 1051.

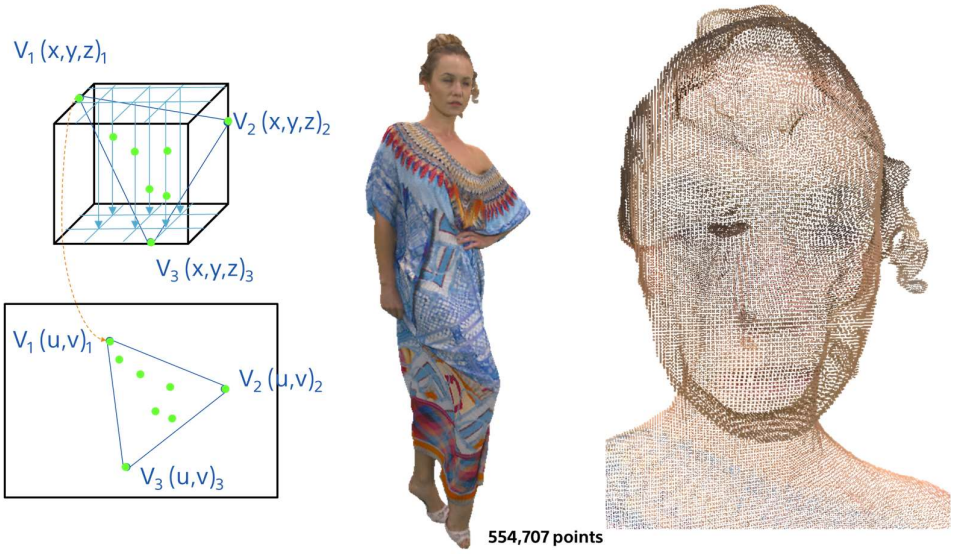

Figure 2: Orthogonal Mesh Surface Sampling

The sampling process is performed individually in each triangle, so it creates a lot of duplicate points. But the pcc_metric software has the option to either remove the duplicates by keeping the first point or averaging all the duplicates. In our experiments, duplicate points were averaged. The sampling grid used can also be an input parameter, but currently we do not change the resolution of the input mesh and perform sampling in a grid with similar size. For instance, for the 8i sequences, the grid size is 1024x1024x1024.

Texel back-projection

The point cloud is created by performing back projection of the texels, that is, the integer positions in the UV maps. The color of each point is directly obtained from the texture map, but the position of the point in 3D space is obtained by performing barycentric interpolation of the vertices of the triangle. Notice that in this case, the vertices of the mesh may not be present in the point cloud, since the UV coordinates of the vertices are usually not an integer position (and do not represent a texel). Figure 3 shows the surface sampling for the uncompressed longdress mesh, frame 1051.

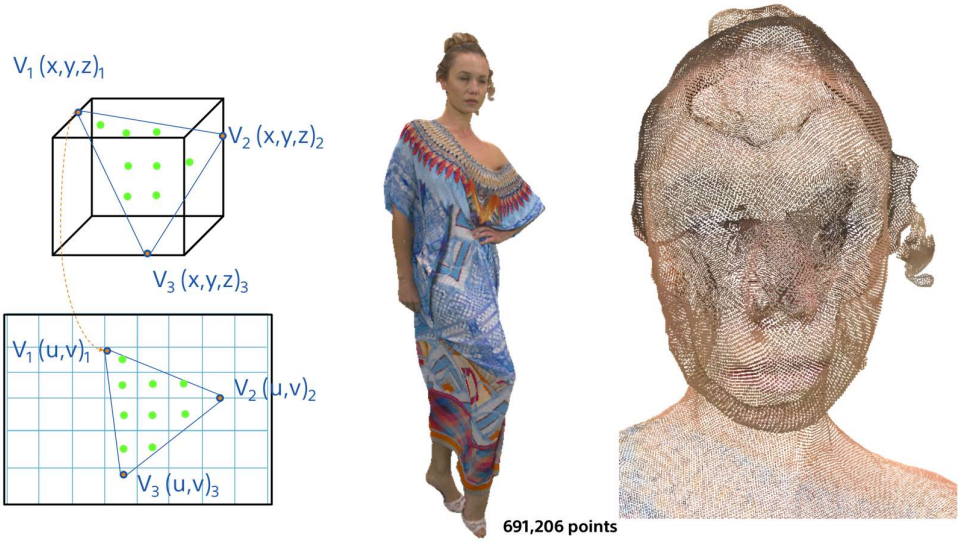

Figure 3: Texel back-projection Mesh Surface Sampling

Since the position of the points is obtained by combining the position of the vertices, the result is a floating-point value, which avoid duplicates in the resampled point cloud. The density of the point cloud will depend on the size of the texture map.

Regular surface sampling (Owlii method)

In [3], point cloud content was provided by Owlii from sampling a mesh surface. In their contribution, they also provided a software that performs the conversion. In summary, each triangle is sampled in a grid following the direction of the sides of the triangle. The sampled position is rounded to an integer value, and the color value is obtained by first estimating the UV coordinates of the sampled position using barycentric interpolation, and then doing bilinear interpolation in the UV map domain. In the provided software, the size of the grid at the sized of the triangles was given as an input parameter, and multiple layers could also be created along the normal direction.

In the proposed modified pcc_error software, sampling following Owlii's approach is also possible. However, the positions of the sampled surface are not rounded and are still maintained in floating-point resolution. Because with this approach the edge and the vertices of the triangles are part of the sampling set, duplicate points occur in the final point cloud. Similar to the orthogonal projection approach, we also allow the software to remove duplicate points, either by choosing the attribute of the first one or just averaging all their attributes. Figure 4 shows the surface sampling for the uncompressed longdress mesh, frame 1051.

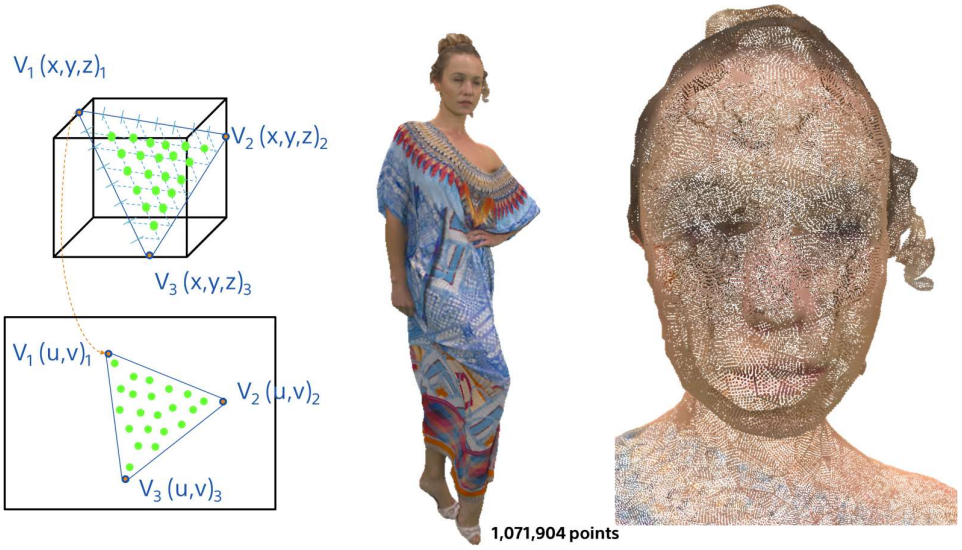

Figure 4: Regular surface sampling Mesh Surface Sampling

Experimental Results

We present in Figure 5 results for one frame of the longdress sequence, coded with TFAN using the SC3DMC software [4]. The quantization parameters varied from 5 to 16 for the vertices position and for the texture coordinates.

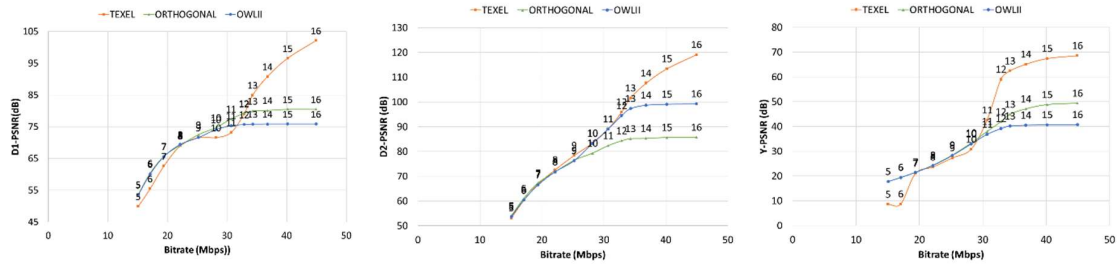

Figure 5: Comparison of metric performance

All three surface sampling strategies provide metrics that improve as the quantization parameter increases. The texel back-projection sampling strategy has issues with the geometry, since when the UV parameters are heavily quantized, the texels of the texture map does not belong to any triangle, therefore reducing the number of points as well.

The orthogonal projection has the advantage that the samples generated are situated in integer positions, so no floating-point value is used for the comparison. In order to provide more details on the objective metrics, we show in Figure 6 the behavior of the metric, when we change the quantization parameters independently for vertex position and texture mapping.

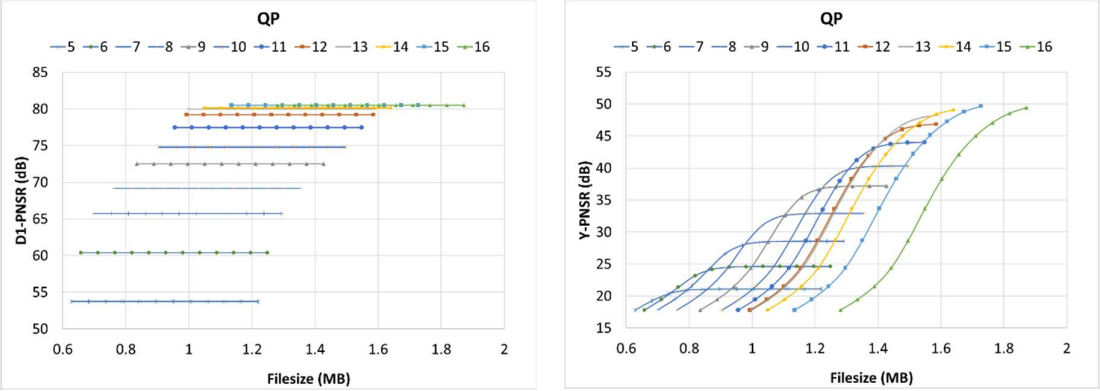

Figure 6: Independent QP variation

As expected, when the quantization parameter is fixed for the vertex position, the metric related to geometry does not change. However, the metric related to the attribute does change with the variation in quantization step for the texture coordinate. Furthermore, the distortion in geometry will influence the attribute metric as well. For instance, for high distortion in geometry, even the increase in the quantization step for the texture coordinate does not improve the measured quality. However, for low geometry distortion, the quantization parameter for texture coordinate increases the overall attribute quality. This makes sense because with bad geometry, the search for a match between the points of two point clouds results in errors (two uncorrelated points become a match), so the RGB value of the reference might always be wrong, regardless of the RGB value of the compressed point cloud. In the case that the geometry is high-quality, the attribute quality is only affected by the quantization parameters, which may result in a wrong RGB value for the compressed point cloud, even thought the corresponding point in the reference cloud is the correct one.

Conclusion

We presented a modification to the pcc_error software, in order to calculate D1/D2/Y-PSNR metrics for meshes. Internally, the software samples the surface of the input mesh, creating a point cloud, and performs the metric as before. Three methods were implemented, and from our studies, we recommend using orthogonal projections as the sampling method of choice. We believe that there is still room for improvement in the metric, by creating more points in areas that are not well sampled, due to the angle between the triangle surface and the projection direction. Furthermore, neighboring triangles with different orientations may have issues in the shared edges. Strategies as the ones proposed in [5], e.g. to extend the primitives, could be applied to reduce the number of holes. However, color value of the added voxels should also be carefully selected.

Bibliography

[1] Marvie, J.-E., “[V-PCC][requirements] An analysis of metrics for dynamic meshes”, input document m55717, January 2021
[2] MPEG PCC pcc_error command from MPEG 123, November 9, 2018, v0.12.3.
[3] Cao, K., Xu, Y., Lu, Y., and Wen, Z., “Owlii Dynamic Human Textured Mesh Sequence Dataset”, input document m42816, April 2018
[4] MPEG SVN repository, http://wg11.sc29.org/svn/repos/MPEG-04/Part16-Animation_Framework_eXtension_(AFX)/trunk/3Dgraphics/SC3DMC/trunk/
[5] NVidia Blog, “Basic of GPU voxelization”, https://developer.nvidia.com/content/basics-gpu-voxelization , accessed 01/04/2021