 ISO/IEC JTC 1/SC 29/WG 7 N231

**ISO/IEC JTC 1/SC 29/WG 7  
MPEG 3D Graphics Coding   
Convenorship: AFNOR (France)**

**Document type:** Output Document

**Title:** CfP for Dynamic Mesh Coding

**Status:** Approved

**Date of document:** 2021-11-08

**Source:** ISO/IEC JTC 1/SC 29/WG 7

**Expected action:** None

**Action due date:** None

**No. of pages:** 38 (with cover page)

**Email of Convenor:** marius.preda @ imt . fr

**Committee URL:** [https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg7](https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg3)

**INTERNATIONAL ORGANIZATION FOR STANDARDIZATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 7 MPEG 3D Graphics Coding**

**ISO/IEC JTC 1/SC 29/WG 7 N231**

**October 2021, Virtual**

|  |  |
| --- | --- |
| **Title** | **CfP for Dynamic Mesh Coding** |
| **Source** | **WG 7, MPEG 3D Graphics Coding** |
| **Status** | **Approved** |
| **Serial Number** | **MDS21000** |

# Abstract

This document is the Call for Proposals (CfP) for Dynamic Mesh coding technology, targeting an efficient representation of dynamic objects and real-time acquisition environments.

# Introduction

The advances in 3D capture, modeling, and rendering have promoted the ubiquitous presence of 3D content across several platforms and devices. Nowadays, it is possible to capture a baby’s first step in one continent and allow the grandparents to see (and maybe interact with) and enjoy a full immersive experience with the child in another continent. Nevertheless, in order to achieve such realism, models are becoming ever more sophisticated, and a significant amount of data is linked to the creation and consumption of those models. 3D meshes are widely used to represent such immersive content. A mesh is composed of several polygons that describe the boundary surface of a volumetric object. Each polygon is defined by its vertices in 3D space and the information on how the vertices are connected, referred to as connectivity information. Optionally, vertex attributes, such as colors, normals, etc., could be associated with the mesh vertices. Attributes could also be associated with the surface of the mesh by exploiting mapping information that describes a parameterization of the mesh onto 2D regions of the plane. Such mapping is usually described by a set of parametric coordinates, referred to as UV coordinates or texture coordinates, associated with the mesh vertices. 2D attribute maps are used to store high resolution attribute information such as texture, normals, displacements etc. Such information could be used for various purposes such as texture mapping and shading.

A dynamic mesh sequence may require a large amount of data since it may consist of a significant amount of information changing in time. Therefore, efficient compression technologies are required to store and transmit such content. Mesh compression standards IC, MESHGRID, FAMC [4] were previously developed by MPEG to address dynamic meshes with constant connectivity and time varying geometry and vertex attributes. However, these standards do not take into account time varying attribute maps and connectivity information. DCC (Digital Content Creation) tools usually generate such dynamic meshes. In counterpart, it is challenging for volumetric acquisition techniques, such as [3], to generate a constant connectivity dynamic mesh, especially under real time constraints. This type of content is not supported by the existing standards. MPEG is planning to develop a new mesh compression standard to directly handle **dynamic meshes with time varying connectivity information** and optionally time varying attribute maps. This standard targets lossy, and lossless compression for various applications, such as real-time communications, storage, free viewpoint video, AR and VR. Functionalities such as random access and scalable/progressive coding are also considered.

Companies and organizations are invited to submit proposals in response to this Call for Proposals. Dynamic mesh coding technologies will be evaluated based upon objective and subjective metrics. Results of these tests will be made public, taking into account that no direct identification of any of the contributors will be made (unless it is specifically requested or authorized by a contributor to be explicitly identified). Prior to having evaluated the results of the tests, no commitment to any course of action regarding the dynamic mesh coding technology can be made. In addition, subjective evaluation of proposals will be performed.

Descriptions of proposals shall be registered as input documents to the proposal evaluation meeting in April 2022 (see timeline in next section). Proponents are required to attend that meeting to present their proposals. For those organizations and individuals that are not accredited members of the MPEG working group, further information about logistical steps to attend the meeting can be obtained from the contact persons listed in Chapter 10.

# Timeline

## Timeline of the calls, deadlines and evaluation of the responses:

|  |  |  |
| --- | --- | --- |
| **Action** | **Date** | **Remarks** |
| Release of the CfP | 2021.10.29 |  |
| Declaration of intention of answering to the CfP. | 2022.01.14 | An email should be sent to the contact addresses listed in chapter 10. An individual FTP account and a proponent number for subjective testing will be provided to each proponent by 2022.01.17 as confirmation.  Test labs will issue an invoice to those who have declared the intention to answer the CfP. |
| Submission of testing material for subjective evaluation | 2022.03.18 | Testing material to be uploaded on the FTP site. Windows, MacOS or Linux decoder and the compressed bitstreams must be provided |
| Submission of the completed objective evaluation spreadsheets | 2022.03.25 | To be uploaded on the individual FTP site |
| Submission deadline for proposals documentation | 2022.03.25 | To be uploaded as input to the MPEG document management system |
| Compilation of submitted data in a unique spreadsheet and submission as an input contribution (without decoding time) | 2022.03.28 | Action performed by 3DG Chair |
| Feedback that MD5 checksums for decoded bitstreams are matched | 2022.03.28 | Action performed by 3DG Chair |
| Compilation of submitted data in a unique spreadsheet and submission as an input contribution (with decoding time) | 2022.04.04 | Action performed by 3DG Chair |
| Subjective evaluation with naive viewers | 2022.04.06  -  2022.04.18 | Test coordinator: Mathias Wien |
| Evaluation of responses | 2022.04.25  2022.04.29 | Action performed during the MPEG meeting week. Proponents are required to present their proposals in person. |
| Nominate the best proposal(s) | 2022.04.29 | Preference is to select one best proposal. If for any reason, there is no agreement on the best proposal, then the group will nominate a reduced number of candidates (2 or 3). Contributors would work in combining candidates and present in July a combined proposal. In July the combined proposal will be benchmarked against the initial best proposals. |
| Submission deadline of source code for encoder/decoder of selected best proposal or the selected candidates | 2022.05.06 | Source code shall allow to reproduce test results in bit exact manner and source code shall match with submitted documentation |
| Establishment of the first test model based on the best proposal(s) including the combined proposal | 2022.07.22 |  |

Table Timeline

## Preliminary Development Plan:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Year | Month | Day | MPEG meeting | City | Country | Stage |
| 2021 | 10 | 29 | 136 |  |  | Release of the CfP |
| 2022 | 01 |  | 137 |  |  |  |
|  | 04 |  | 138 |  |  | Nominate the best proposal(s) |
|  | 07 |  | 139 |  |  | Establishment of the first test model based on the best proposal(s) including the combined proposal |
|  | 10 |  | 140 |  |  | Approval of WD 1.0 |
| 2023 | 01 |  | 141 |  |  | Approval of WD 2.0 |
|  | 04 |  | 142 |  |  | Approval of CD |
|  | 07 |  | 143 |  |  |  |
|  | 10 |  | 144 |  |  |  |
| 2024 | 01 |  | 145 |  |  | Approval of DIS (prefer stable text) |
|  | 04 |  | 146 |  |  |  |
|  | 07 |  | 147 |  |  |  |
|  | 10 |  | 148 |  |  | Approval of FDIS |

Table : Preliminary Development Plan

# Definitions

**Mesh:** A mesh consists of the following components:

* Connectivity information
* Geometry information
* Mapping information
* Vertex attributes, and
* Attribute maps

**Dynamic Mesh**: Is a mesh where at least one of the five components is varying in time.

**Animated Mesh**: Is a dynamic mesh with constant connectivity.

**Connectivity Information:** A setof vertex indices describing how to connect the mesh vertices to create a 3D surface.

Note: Geometry and all the attributes share the same unique connectivity information.

**Geometry Information:** The geometry information is described by a set of 3D positions associated with the mesh vertices. The (x,y,z) coordinates describing the positions should have finite precision and dynamic range.

**Mapping Information:**Describes how to map the mesh surface to 2D regions of the plane. Such mapping is described by a set of UV parametric/texture coordinates associated with the mesh vertices together with the connectivity information. The texture coordinates should have finite precision and dynamic range.

**Vertex Attributes:** Vertex attributes are scalar or vector attribute values associated with the mesh vertices. The attributes values should have finite precision and dynamic range.

**Attribute Maps:** Attribute maps are attributes associated with the mesh surface and stored as 2D images/videos. The mapping between the videos (i.e., parametric space) and the surface is defined by the mapping information.

**Lossless Mesh Compression:** A mesh is losslessly encoded if and only if its connectivity information, geometry information, mapping information, its vertex attributes and attribute maps are compressed in a lossless manner.

**Lossless Connectivity Compression:** The connectivity information is encoded losslesslyif and only if the reconstructed connectivity is the same as the input up to a permutation of the vertex indices and triangles/polygons. Note: Any permutation of the indices of a triangle or polygon that preserve the surface orientation is allowed. For instance, a triangle (a,b,c) could also be represented as (b,c,a) and (c,a,b). The triangle (b,a,c) has a different orientation and is considered to be different.

**Lossless Geometry Compression:** The geometry information is losslessly compressed if and only if the reconstructed vertex positions are identical to the input vertex positions after applying the same vertex permutation applied to the connectivity information.

**Lossless Attributes Compression:** The vertex attributes are losslessly compressed if and only if the reconstructed vertex attributes are identical to the input vertex attributes after applying the same vertex permutation applied to the connectivity information.

**Lossless 2D Mapping Information Compression:** The 2D mapping information is losslessly encoded if and only if the reconstructed UV coordinates are identical to the input UV coordinates after applying the same vertex permutation applied to the connectivity information.

**Lossless Compression of Attribute Maps:** An attribute map is losslessly encoded if and only if the reconstructed attribute map is identical to the input attribute map.

**Lossy Mesh Compression:** Lossy mesh compression permits different tradeoffs between bitrate and quality. Such trade-offs could be achieved by encoding in lossy manner any components of the input mesh (i.e., connectivity, geometry, 2D mapping information, vertex attributes or attribute maps) either through usage of existing video coding architecture or through additional steps such as remeshing.

**Post-Processing:** Any process that is not in the decoding loop (does not have any influence on the decoding process).

**Pre-Processing:** Any process that is not in the encoding loop (could be applied independently of the encoding process).

# Requirements

This section presents the requirements for MPEG dynamic mesh coding. The requirements capture the use cases defined in document [1] but are not limited to these use cases.

## Conventions

The following conventions are used in this document for drafting requirements:

|  |  |
| --- | --- |
| **Convention** | **Meaning** |
| **shall enable** | The functionality shall be specified but its support is optional. |
| **shall support** | The functionality shall be specified and its support is mandatory. |
| **should enable** | The functionality is recommended to be specified and its support is optional. |
| **should support** | The functionality is recommended to be specified and supported. |
| **may enable** | The functionality may be specified and if it is, then its support is optional and it shall not have any weight in the selection or exclusion of any particular solution. |
| **may support** | The functionality may be specified and if it is, then its support is recommended but it shall not have any weight in the selection or exclusion of any particular solution. |
| **shall not preclude** | The functionality shall not be prevented. |
| **should not preclude** | It is recommended not to prevent the functionality. |

## Requirements for Dynamic Mesh compression

The next section describes the 3D mesh representation requirements, section 4.2.2 describes the compression requirements, and finally section 4.2.3 describes the video-based requirements.

### 3D Mesh Representation

a) The 3D mesh representation shall provide means to support static or dynamic meshes.

b) The 3D mesh representation shall support vertices and their 3D positions (X, Y, Z) with a specification of the precision and dynamic range of the 3D positions.

c) The 3D mesh representation shall support triangle connectivity information.

d) The 3D mesh representation should support parameterization information mapping the mesh surface on 2D regions of the plane.

e) The 3D mesh representation should support multiple vertex attributes associated with each vertex including colour, reflectance, normal vectors, and transparency or other generic attributes (e.g. user-defined vertex attributes).

f) The 3D mesh representation should support attribute maps, such as texture maps, and optionally normal maps, bump maps, or other generic maps (e.g. user-defined attribute maps).

### 3D Mesh Compression

a) The proposed solution shall support means for efficient compression of static and dynamic 3D meshes.

b) The proposed solution shall support at the minimum the encoding of meshes with 3D positions and triangle connectivity information.

c) The proposed solution shall support fixed-point input and output values for geometry, texture coordinates, vertex attributes and attribute maps. Floating point data shall be first converted to a fixed-point representation.

d) Lossy compression: The proposed solution shall support efficient (bitrate targets from 2 to 32 Mbit/s for acceptable to high visual quality) lossy compression permitting different tradeoffs between bitrate and subjective and/or objective quality.

f) The proposed solution shall support lossless compression, which guarantees that the reconstructed mesh is mathematically identical to the original one.

g) The proposed solution shall support temporal variations of mesh connectivity, mapping information, vertex positions, attribute maps and other vertex attributes.

h) The proposed solution shall support temporal random access.

i) Progressive and/or scalable coding: The proposed solution should support spatial (i.e., mesh resolution) and quality (i.e., precision on vertex positions and attribute values) scalability.

j) The proposed solution should enable error resilience to cope with packet loss without having to retransmit the entire mesh in the context of lossy compression.

k) The proposed solution should support tools that enable low latency and/or real-time implementation.

l) The proposed solutions complexity should support feasible implementation of encoding and decoding within the constraints of the available technology at the expected time of usage.

Note: Complexity includes: Power consumption, computational power, memory bandwidth etc.

m) The proposed solution should support parallel encoding and decoding.

### Video based requirements

a.) All proposals are strongly encouraged to be based on the V3C [6] framework and specification.

b.) The HEVC standard and it's HM 16.21-SCM 8.8 reference software implementation shall be used for all video encoding. No modification of the HM reference software is permitted.

# Test Materials, Categories and Conditions

This CfP addresses a subset of dynamic meshes categorized as “Dynamic Objects with Texture Mapping”. For this subset of dynamic meshes, vertex attributes are associated with the surface of the mesh through a mapping process to a set of 2D attribute maps. Per vertex attributes are not supported.

## Test Material Datasets

Below is a list of the mesh test material datasets to be used, organized based on the test class. The test class is an indicator of how complex a mesh is to encode, where A is the lowest and C the highest complexity. All test material datasets are available in the MPEG Content repository accessible under the following URL:

<https://mpegfs.int-evry.fr/mpegcontent/ws-mpegcontent/CfP/Mesh/CONTENT/voxelized>

Note: Downloaded test material datasets should be verified with MD5 checksums. Each zip file of a test material dataset contains an MD5 file (with the corresponding md5 sums for each file in the archive). A README.md file is also available with further details of the files in the MPEG repository.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category** | **Test**  **Class** | **Test material**  **dataset filename** | **# Frames** | **# Vertices** | **# Faces** | **Geometry**  **Precision** | **Texture Coord.**  **Precision** | **Texture**  **Map Size** | **Color Attribute** | **Sequence**  **Number** |
| Dynamic Objects with Texture Mapping | A | longdress | 300 | 22k | 40k | 10 bits | 12 bits | 2k x 2k | NA | 1 |
| soldier | 300 | 22k | 40k | 10 bits | 12 bits | 2k x 2k | NA | 2 |
| B | basketball\_player | 300 | 20k | 40k | 12 bits | 12 bits | 2k x 2k | NA | 3 |
| dancer | 300 | 20k | 40k | 12 bits | 12 bits | 2k x 2k | NA | 4 |
| C | mitch | 300 | 16k | 30k | 12 bits | 13 bits | 4k x 4k | NA | 5 |
| thomas | 300 | 16k | 30k | 12 bits | 13 bits | 4k x 4k | NA | 6 |
| football | 300 | 25k | 40k | 12 bits | 13 bits | 4k x 4k | NA | 7 |
| levi | 150 | 20k | 40k | 12 bits | 13 bits | 4k x 4k | NA | 8 |

Table Test material datasets (number (#) of vertices and faces are expressed per frame).

Notes:

1. The order of the vertices as they are stored in the file is not necessary to be maintained in the decoded versions.
2. The order of the connectivity of the vertices as they are stored in the file is not necessary to be maintained in the decoded versions, but for lossless compression condition, the orientation of the triangles should be maintained (see Lossless Connectivity Compression definition).
3. The datasets contain one PNG, one MTL and one OBJ file for each mesh frame, whereby the OBJ naming convention includes a suffix that indicates the precision for geometry and texture coordinates integer values, which are also indicated in Table 3. The PNG corresponds to the texture map, the MTL file corresponds to the wavefront material template library file that links the mesh with the corresponding texture image, and the OBJ corresponds to the voxelized mesh. The voxelized meshes available in the dataset contain only one unique connectivity information for both vertex positions and texture coordinates (notice that this could lead to vertices with duplicate 3D positions but different texture coordinates).

## Test Conditions and Parameters

### Test Conditions including target bit rates

The mesh test material will be tested under the following conditions:

* Lossless Mesh Compression
* Lossy Mesh Compression

#### Test Conditions for Lossless Mesh Compression

Compression rates for all test material datasets shall be provided according to an all-intra encoding configuration.

#### Test Conditions for Lossy Mesh Compression

The target bit rates under this condition in terms of Mbit/s are indicated in Table 4 and Table 5. These rates are inclusive of connectivity, geometry, mapping information and attribute map. The data submitted to this test condition will undergo both objective and subjective evaluation, with the sequences selected for subjective evaluation indicated in Annex D. For the subjective evaluation, it is permitted to use temporal prediction with a maximum intra period of 32 frames. For the objective evaluation, both all-intra and random access (with a maximum period of 32 frames) bit rates and distortion values should be provided.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Dataset | R1 | R2 | R3 | R4 | R5 |
| Longdress | 4 | 8 | 11 | 14 | 21 |
| Soldier | 4 | 8 | 11 | 14 | 21 |
| Basketball\_player | 3 | 5 | 10 | 14 | 21 |
| Dancer | 3 | 5 | 10 | 14 | 21 |
| Mitch | 3 | 4 | 6 | 8 | 12 |
| Thomas | 3 | 4 | 6 | 8 | 12 |
| Football | 4 | 8 | 12 | 17 | 25 |
| Levi | 4\* | 8 | 12 | 17 | 25 |

Table Target bitrates in Mbit/s for Random Access

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Dataset | R1 | R2 | R3 | R4 | R5 |
| Longdress | 5 | 9 | 12 | 15 | 22 |
| Soldier | 5 | 9 | 12 | 15 | 22 |
| Basketball\_player | 3 | 5 | 10 | 14 | 21 |
| Dancer | 3 | 5 | 10 | 14 | 21 |
| Mitch | 4 | 6 | 11 | 16 | 24 |
| Thomas | 4 | 6 | 11 | 16 | 24 |
| Football | 4 | 8 | 12 | 17 | 25 |
| Levi | 4\* | 8 | 12 | 17 | 25 |

Table Target bitrates in Mbit/s for All-Intra

Note: bit rates marked with a \* are not covered by anchors. When anchors are not available, the subjective evaluation will be performed only between technologies provided by proponents.

Note: Bitrates will be fully confirmed during MPEG 137.

Proposals shall submit bitstreams not exceeding the target rate points. Additional rate points beyond the designated rate, especially lower rates, are encouraged.

### Summary of all Test Conditions

The following table summarizes all test conditions and assigns codes that shall be used for naming the submitted bitstreams:

|  |  |
| --- | --- |
| Test Sub-Condition | Code for bitstream naming convention |
| Lossless mesh coding / All Intra | 0 |
| Lossy mesh coding / All Intra | 1 |
| Lossy mesh coding / Random Access | 2 |

Table Test Sub-Conditions

## Restrictions for all test categories and conditions

Point cloud compression technologies shall obey the following additional constraints:

1. Pre-and post-processing is only allowed in the lossy test conditions. If any pre/post-processing is done, the comparison shall be done with the original data (models before any pre-processing). Proponents are encouraged to describe in detail any pre- or post-processing procedure applied.
2. Low-level programming optimizations, such as assembly code/intrinsics and external compression libraries, are discouraged. If any such optimization is implemented, then the rationale for, and extent of, the optimization shall be described.
3. Optimization of encoding parameters using non-automatic means is discouraged. In case that optimization is done, then it needs to be described in detail.
4. The coding test set shall not be used as the training set for training large entropy coding tables, VQ codebooks, etc. If any processes in the encoder or decoder are designed using training data, then the coding test set shall not be used as part of the training set.
5. For all bitstreams for which subjective evaluation is conducted, submitted bitstreams must not exceed the target rate points.
6. For all testing categories, the intra period shall be less than or equal to 32 frames.

# Anchors

Anchors were generated by coding the meshes using state-of-the-art codecs for static mesh compression, like Draco and SC3DMC, while the texture maps represented by a sequence of PNG files were first converted to video format and then encoded with HEVC HM 16.21 with Screen Content Coding Extension (SCC) 8.8. Annex C contains information on how to regenerate the anchors.

The compressed anchor bitstreams for different settings are available at:

* <https://mpegfs.int-evry.fr/mpegcontent/ws-mpegcontent/CfP/Mesh/ANCHOR/bitstreams>

The decoded anchor meshes can be found under the following URL and files:

* <https://mpegfs.int-evry.fr/mpegcontent/ws-mpegcontent/CfP/Mesh/ANCHOR/decoded_meshes>

Each zip file of a test material dataset contains an MD5 file (with the corresponding md5 sums for each file in the archive). Results of the anchor are available in the Excel sheet provided with this CfP. The distortion is calculated according to Section 7.

For non-MPEG members, the software and the anchors will be available upon request to the contact persons.

# Evaluation Procedure

The evaluation procedure will be conducted to select a baseline set of technologies. Subsequent core experiments will be used to obtain the best technical solution to fulfill the requirements.

## Objective Evaluation of lossy condition

The distortion metrics compare the original meshes with the decoded meshes and provide PSNR values describing geometry and attribute distortion.

Proposals will be evaluated using a set of objective metrics, including PSNR of an image-based geometric error (GEO\_PSNR) and point-based geometric errors (D1 and D2) for geometry, as well as PSNR of color attributes (Y\_PSNR, U\_PSNR and V\_PSNR for point-based, and Y-PSNR for image-based metrics). The evaluation will be made based on the rate-distortion (RD) performance, and RD curves shall be plotted using PSNR as the quality measure. The rates shall be reported as Mbit/s.

### Geometric Distortions

Geometry distortion shall be reported with GEO\_PSNR, D1, and D2 PSNR values as specified in Annex B. The reported distortion measures shall be averaged over all frames. In addition, proponents shall report the MSE and PSNR per frame. The metric software provides means to compute and export these results.

### Attributes Distortion

The color distortion, described in Annex B, is measured in YUV space, and is reported as three PSNR values one for each channel: Y, U, and V. The PSNRs for each component will be individually used to compare methods. The reported distortion measures shall be averaged over all frames. In addition, proponents shall report the MSE and PSNR per frame. The metric software provides means to compute and export these results.

### Software & Usage

The metric software for performing the objective evaluation can be found in the MPEG GIT:

<http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-pcc-mmetric/-/tree/0_1_13>

Note: it is required for Linux systems to compile the metric software with gcc version 9.3.0.

Usage of the software is described in the readme.md file at the top level page.

Additional details about the metrics to be reported and the usage of the metric software is described in Annex B.

### Reporting of Results

See chapter 8.2 for information regarding evaluation spreadsheets.

## Objective Evaluation of lossless chain

Proposals will be evaluated, for each frame, using set of rules defined in Section 3:

1. *Lossless Mesh Compression*. (…)
2. *Lossless Connectivity Compression. (…)*
3. *Lossless Geometry Compression.* (…)
4. *Lossless Attributes Compression.* (…)
5. *Lossless 2D Mapping Information Compression*. (…)
6. *Lossless Compression of Attribute Maps*. (…)

The metric software provides a systematic means to validate these constraints returning a binary result: “equal” or “different”. Evaluation of proposals will run these tests and discard solutions for which equality is not obtained. Contributors can either use the software to validate their proposal, or any other means, there is no specific metric to be reported.

## Subjective Evaluation

Subjective evaluation will only be conducted for selected test material datasets and for the lossy test conditions RA and AI. The following subset of test material listed in Table 3 will be used for subjective evaluation: longdress, basketball\_player, mitch, football and levi. This selection contains test material from all test classes.

The bitrates for the subjective evaluation shall be the same as for the objective evaluations which are listed in Table 4 and Table 5, so the same bitstreams shall be used for both objective and subjective tests. To reduce subjective testing efforts and as responses are expected to to perform better than the anchor, only the four rate points R1 to R4 shall be used.

The anchors shall be the same as for the objective evaluation for the same test condition.

The test material will be rendered using the Mesh renderer selected by MPEG and the usage of the renderer is described in [2].

The Mesh renderer will be configured for using a uniform background color and a floor

The rendering view-point/camera path will follow a pre-defined path selected by the MPEG Test Coordinator and the WG7 chair for each test material, which is not known to the proponents. This prevents any possibility of optimizing a codec for these tests. The output of the camera path will be stored as uncompressed Full HD 1920x1080p30 video sequences of a length as close as possible to 10s. These video sequences will be viewed and evaluated by naïve test subjects using method Degradation Category Rating (DCR) Rec. [ITU-T P.910](https://www.itu.int/rec/T-REC-P.910-200804-I/en) [14]. Videos should be uploaded to the FTP site so that proponents can cross-check if the rendering is correct. Once that the bitstreams are submitted, the camera path will be shared.

### Subjective evaluation test logistics

The test coordinator for the subjective evaluation is:

Mathias Wien

RWTH Aachen University

Lehrstuhl für Bildverarbeitung

52056 Aachen

Germany

Email: wien@lfb.rwth-aachen.de

Tel.: +49-241-80-27867

The test coordinator will select 2 ITU-R BT.500 [13] compliant evaluation laboratories. Subjective evaluation shall be in line with guidelines described in [5].

### Reporting of results

The data extracted from the scoring sheets will be collected by the test labs in spreadsheets and the test coordinator will provide a consolidated test report. Mean Opinion Score (MOS) and Confidence Interval (CI) data will be computed. Also, a subjects’ post-screening will be applied to verify the level of reliability of their answers (data from subjects with a correlation index lower than 0.75 will be excluded from the MOS and CI values computation). Graphs reporting, for example, MOS on the y-axis and the bitrate on the x-axis, will be drafted.

Additional details on the subjective evaluation can be found in Annex D.

# Submission Requirements

Proposals are strongly encouraged to be based on the V3C [6] framework and specification and should include submissions to at least one of the three test conditions defined in this document. Proponents are required to present their proposals in person at MPEG 138. In the following sections, details on the coded test material and documentation that form a complete proposal are provided.

## Coded test material

The following material must be made available by proponents of technologies:

1. Bitstreams for all datasets in target test Categories, which follow the associated test conditions and satisfy the rate constraints specified in this document. Proponents shall use the following naming scheme:

PnnSmmCxxRyy.zip (.zip is a collection of the compressed datasets)

nn = Proponent number; to the Anchor is typically assigned the code P00

mm = Sequence number from Table 3

xx = a code identifying the test sub-condition which is defined in Table 6 Test Sub-Conditions

yy = the rate number, in our case typically from 1 to 5. The number 0 is used for lossless

1. Binary decoder executable (Windows, MacOS or Linux, configuration files (if required for decoder usage), and usage documentation allowing for decoding of the bitstreams. The decoder is expected to generate one OBJ file per frame and one texture image file in PNG format per frame.

## Evaluation spreadsheets

Complete submissions shall include results of the objective tests that shall be reported by using the Excel spreadsheet that has been obtained as part of this CfP. The spreadsheet itself shall be the reference for information to be reported.

## Documentation

Complete submissions shall include the following elements:

1. An information form must be submitted within each proposal. This form can be found in Annex A of this document.
2. A technical description for full conceptual understanding and generation of equivalent performance results by experts. This description should include all data processing paths and individual data processing components used to generate the bitstreams. It does not need to include complete bitstream format or implementation details, although as much detail as possible is desired.
3. The technical description shall state how the proposed technology behaves in terms of random access to any frame within the sequence. For example, a description of the GOP (Group of Pictures), such as structure and the maximum number of frames that must be decoded to access any frame, could be given.
4. The technical description shall specify the expected encoding and decoding delay characteristics of the technology, including structural delay, e.g., due to the amount of frame reordering and buffering and the degree by which the delay can be minimized by parallel processing.
5. The technical description shall contain information suitable to assess the complexity of the implementation of the technology, including the following:

* A complete complexity analysis for encoder and decoder is not required to be provided by contributors, but compression/decompression time should be reported per test material dataset (a relative time with respect to ALL anchors from the category/test condition(s) they target for encode/decode process should be provided). Time for input/output operations to storage systems are excluded in the calculations.
* Expected memory usage of encoder and decoder.
* Complexity of encoder and decoder, in terms of number of operations, dependencies that may affect throughput, etc.
* Degree of capability for parallel processing.
* Degree to which bitstreams can be considered progressive.

## Source code

* Proponents are encouraged (but not required) to allow other committee participants to have access, on a temporary or permanent basis, to their encoded bit streams and binary executables or source code.
* Proponents are encouraged to submit a statement about the programming language in which the software is written, e.g. C/C++, and the platform(s) on which the binaries were compiled.

Proponents are advised that, upon acceptance for further evaluation, it will be required that certain parts of any proposed technology be made available in source code format to participants in the core experiments process and for potential inclusion in the prospective standard as reference software. When a particular technology is a candidate for further evaluation, commitment to provide such software is a condition of participation. The software shall produce identical results to those submitted to the test on the same platform that the technology was submitted. Additionally, submission of improvements (bug fixes, etc.) is strongly encouraged.

## File package structure for the delivery of submissions

Proponents shall deliver their submission in the form described in this chapter. Note that submissions deviating from that format cannot be subjectively tested.

### Delivery structure

The directory shall contain six folders under the root of the FTP site described in chapter 2 of the CfP

* “app”: containing the binary decoder executable (Windows 64 bits, MacOS or Linux), the configuration file, if required, and the usage documentation allowing for decoding of the bitstreams.
* “enc”: containing all encoded files stored in sub-folders described later in this document.
* “cfg”: containing all scripts, like decoding scripts described later in this document.
* “doc”: containing the documents required by chapter 8.3 of the CfP.
* “src”: place holder for the potential delivery of source code as described in the timeline.
* “dec”: place holder for all decoded files. These will not be delivered by the proponents, but generated by 3DG Chair / Test Coordinator.

Note that the delivery of the material follows the timeline as described in chapter 2 of the CfP.

### Sub-Folder description

For each sequence and each bitrate, a specific directory shall contain the encoded (one or several by sequences). They must have the same prefix as its parent directory.

The sub-folders and files shall follow the following rules:

* enc/PxxSxxCxxRxx.zip

### Decoding scripts

To facilitate the evaluation and the decoding processes, each contributor shall add a bash script decode.sh (without parameters) in the cfg/ directory to decode the encoded bitstreams and create/recreate the decoded files.

The decoding script shall create folders “dec/PxxSxxCxxRxx/” and store there decoded objects PxxSxxCxxRxx\_%04d.obj/.png/.mtl.

Before proceeding with the evaluation, the MD5 checksums shall be checked.

### Examples

For example, for each proponent, the folder of each contribution would have the following structure:

P05/

app/

…

enc/

P05S01C02R01.zip

P05S01C02R02.zip

…

cfg/

decode.sh

…

doc/

…

src/

…

dec/

PxxSxxCxxRxx/PxxSxxCxxRxx\_%04d.obj/.png/.mtl

# IPR

Proponents are advised that this call is being made subject to the patent policy of ISO/IEC and other established policies of the standardization organization. The persons named below as contacts can assist potential submitters in identifying the relevant policy information.

# Contacts

Contact persons:

Dr. Marius Preda

MPEG 3D Graphics Chair

Institut Mines-Télécom

37-39 rue Dareau 75014 Paris, France

Email: [marius.preda@it-sudparis.eu](mailto:marius.preda@it-sudparis.eu)

Dr. Igor Curcio

MPEG Requirements Chair

Nokia Technologies

Hatanpäan Valtatie 30

33100 Tampere

Finland

Email: [igor.curcio@nokia.com](mailto:igor.curcio@nokia.com)

# References

1. Use cases for Mesh Coding, ISO/IEC JTC1/SC29 WG7 Doc. N00112, Online, April 2021
2. 3D graphics renderer for the CfP on dynamic mesh compression, ISO/IEC JTC1/SC29 WG7 Doc. N00194, Online, July 2021
3. A. Collet, M. Chuang, P. Sweeney, D. Gillett, D. Evseev, D. Calabrese, H. Hoppe, A. Kirk and S. Sullivan, "High-quality streamable free-viewpoint video," in ACM Transaction on Graphics (SIGGRAPH), 2015
4. Information technology - Coding of audio-visual objects - Part 16: Animation Framework eXtension (AFX), ISO/IEC 14496-16
5. Guidelines for Verification Testing of Visual Media Specifications (draft 2), ISO/IEC JTC 1/SC 29/AG 5 Doc. N00030, Online, July 2021
6. Information technology - Coded representation of immersive media - Part 5: Visual volumetric video-based coding (V3C) and video-based point cloud compression (V-PCC), ISO/IEC 23090-5:2021
7. Image based mesh metric proposal, ISO/IEC JTC 1/SC 29/WG 7 Doc. m57389, Online, July 2021
8. Mesh metric software v0.1.11 updates and review, ISO/IEC JTC 1/SC 29/WG 7 Doc. m57390, Online, July 2021
9. Rasterization: a Practical Implementation, <https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation/visibility-problem-depth-buffer-depth-interpolation>
10. Face Culling, https://www.khronos.org/opengl/wiki/Face\_Culling
11. Fibonacci sphere lattice, http://extremelearning.com.au/evenly-distributing-points-on-a-sphere/
12. LookAt function from OpenGL, <https://www.khronos.org/registry/OpenGL-Refpages/gl2.1/xhtml/gluLookAt.xml>
13. Recommendation ITU-R BT.500-14 (2019), Methodologies for the subjective assessment of the quality of television images
14. Recommendation ITU-T P.910 (2008), Subjective video quality assessment methods for multimedia applications.
15. Renderer Software, ISO/IEC JTC 1/SC 29/WG 7 Doc. m58240, Online, October 2021

# Annex A: Information Form

1. Title of the proposal
2. Organization (name of proposing company, name and email of contact person)
3. Which test conditions are covered by your proposal? Could your proposal be extended to support the remaining test conditions?
4. Proponents shall provide information on how the requirements for Dynamic Mesh Coding in chapter 4 are fulfilled, in the form of a table. This table should list the requirement, the fulfillment (yes/no) and give a short rationale why the requirement is fulfilled.

|  |  |  |
| --- | --- | --- |
| Requirements on Dynamic Mesh Coding | Fulfillment (yes/no) | Rationale for fulfillment |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table Fulfillment of requirements for Dynamic Mesh Coding

# Annex B: Object Evaluation Metrics & Usage of Metric Software

In this annex we specify the primary metric to be used for the CfP evaluation, referred to as the point-based metric and described in section B.1. Additionally, an informative image-based metric is described in section B.2. Proponents are required to report both metrics.

## B.1 Point-based metric (primary)

The point-based metric converts the reference and distorted meshes into two point clouds by applying the sampling procedure described in section B.1.1. The two point clouds are then compared by applying the point cloud metric described in sections B.1.2, B.1.3, B.1.4, B.1.5, and B1.6.

### B.1.1 Mesh sampling

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 normal of the triangle is computed as the cross product of its two edges and normalized to have a unit length. All the points obtained by sampling the triangle inherit its normal vector.

### B.1.2 Geometric Distortion

Let and denote the original and the compressed point cloud obtained from the sampling procedure described in B.1.1, respectively. Consider evaluating the compression errors, denoted as in point cloud relative to reference point cloud. The steps to compute both point-to-point error (D1) and point-to-plane error (D2) for geometric errors are summarized in the following and illustrated in the below figure.

For each point in point cloud, i.e., the black point in the figure, identify a corresponding point in point cloud, i.e. the red point in the figure. Nearest neighbor is used to locate the corresponding point. In particular, a KD-tree search is used to perform the nearest neighbor search in order to reduce the computation complexity.

### B.1.3 Computing D1

Determine an error vector by connecting the identified point in reference point cloud  to point in point cloud . The length of the error vector is the point-to-point error, i.e.,

Based on the point-to-point distances for all points , the point-to-point error (D1) for the whole point cloud, withas the number of points in point cloud, is defined as:

### B.1.4 Computing D2

Project the error vector along the normal direction and get a new error vector. In this way, the point-to-plane error is computed as,

The point-to-plane error (D2) for the whole point cloud is then defined as,

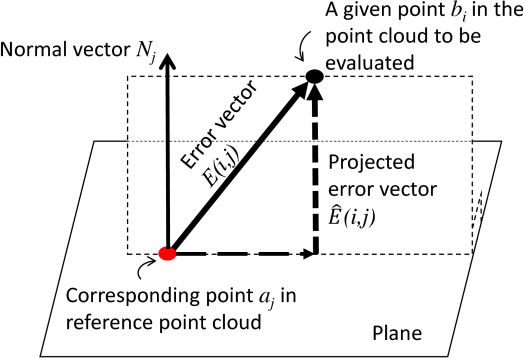


Figure : **Illustration of point-to-point distance (D1) and point-to-plane distance (D2)**

### B.1.5 Geometric PSNR Calculation

The geometric PSNR value is computed as:

where is the maximum length of the sequence bounding box (maxBBLength) as specified in Table 10, and is the symmetric mean squared point-to-point () or point-to-plane () error, which are obtained by considering the maximum distortions and computed as follows:

and .

For dynamic content, the peak value is unchanged over the frames of a sequence.

### B.1.6 Attribute PSNR Calculation

The attribute PSNR value is computed as:

For color attributes, the MSE for each of the three color components is calculated. A conversion from RGB space to YUV space is conducted using ITU-R BT.709, 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. Since the color attributes for all test data have a bit depth of 8 bits per point, the peak value for PSNR calculation is 255.

## B.2 Image-based metric (informative)

The computation of the lossy metric is based on image MSE/PSNR processing [7][8]. An overview of the approach is given in Figure 21. For each frame, the reference and the distorted models are rendered for several view directions , using an orthographic projection (see section B.2.1). The images obtained from the rendering of reference and distorted models are then compared using some adapted image MSE/PSNR metrics (see section B.2.3). The results are averaged over a set of view directions for the frame and over the frames of the sequence.

Diagram

Description automatically generated

Figure . Overview of the image-based metric. Ref/DisColor are the color images/buffers. Ref/DisMask are binary images where pixel[i,j]=1 if there exist a projection in associated color buffer, and 0 otherwise. Ref/DisDepth are the depth buffers. All the buffers have same dimensions of 2048x2048 pixels.

The number of views is fixed to 16 and the resolution of image, depth and mask buffers is set to 2048x2048 pixels. The Figure 32 gives an example of rendered image buffers for the 16 view directions.

Graphical user interface, application

Description automatically generated

Figure . Example of color images generated for the basketball player using 16 views.

## B.2.1 Rendering of one view

Graphical user interface, application

Description automatically generated

Figure . Rendering of image buffers for a given view direction vdi.

The rendering is part of the metric and is implemented inside the metric software.

The rendering of one view is illustrated in Figure 43. The pixels of the image could be obtained by ray tracing or rasterization of the mesh. It is decided to use rasterization for performance reasons. Please refer to [9] for an excellent overview of rasterization and use of depth buffer for solving the visibility problem. Our rasterizer is somewhat simpler that the one presented in [9] since it implements only orthographic projection, hence does not require perspective correction steps.

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 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 is the plane tangent to the bounding sphere and perpendicular to the view direction vector.

The mesh is rendered using *clockwise back-face culling* which suits all the models of the anchor in terms of visual rendering. See [khronos Face Culling page](https://www.khronos.org/opengl/wiki/Face_Culling) [10] for a good explanation of what is back-face culling. This technique is used in every game, AR, VR systems to accelerate the rendering. The idea is to skip the rasterization of triangles for which the face normal is not facing toward the camera. The face normal is calculated by cross product of two edges and of the triangle. Therefore, depending if we use the edges clockwise or counter clockwise , the normal may be oriented on one side or the other.

Note: if the provided answer does not preserve faces orientations based on clockwise orientations it will get low MSE/PSNR scores for the metric.

The rendering step generates the color, the mask, and the depth buffers.

* The color buffer contains for each pixel i,j the RGB value, of the nearest projected triangle.
  + In case of a textured mesh, 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.
* 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.
* The depth buffer contains for each pixel i,j the distance from the projection plane to the 3D surface in 3D space.

## B.2.2 Positioning of views

The positioning of the 16 views, that is to say the set of orientations, is obtained by using a [Fibonacci sphere lattice](http://extremelearning.com.au/evenly-distributing-points-on-a-sphere/) [11]. This distribution aims at generating points over a sphere in an evenly spaced manner. Once one has the points , the directions are the vectors passing through these points and pointing toward the center of the sphere. The Figure 2 gives an example of generated images for 16 views using this method.

The Fibonacci sphere samples are computed as follows:

const double pi = std::atan(1.0) \* 4;

// golden angle in radians

Const float phi = (float)(pi \* (3. - std::sqrt(5.)));

// glm::vec3 a vector of three floats, each entry a direction

std::vector<glm::vec3> camDir;

for (size\_t i = 0; i < targetNbSamples; ++i) {

float y = 1 - (i / float(targetNbSamples - 1)) \* 2; // y goes from 1 to - 1

float radius = std::sqrt(1 - y \* y); // radius at y

float theta = phi \* i; // golden angle increment

float x = std::cos(theta) \* radius;

float z = std::sin(theta) \* radius;

camDir.push\_back(glm::vec3(x, y, z));

}

A special attention is also given to generate the up vectors (see Figure 3). The up vector determines, for a given direction , 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](https://www.khronos.org/registry/OpenGL-Refpages/gl2.1/xhtml/gluLookAt.xml) [12].

For a given view direction viewDir (), the viewUp () vector is defined as follows:

if (glm::abs(viewDir) == glm::vec3(0, 1, 0))

viewUp = glm::vec3(0, 0, 1);

else

viewUp = glm::vec3(0, 1, 0);

In other terms, whenever view direction is not a north or south pole, we use y axis vector as the up vector , otherwise we use the z axis vector as the up vector .

Those values were selected so most of the 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 .

## B.2.3 distortion for one view

The calculus of the distortion is based on the general Mean Squared Error (MSE) formula. Let be a sample of an original image, a sample of a distorted image and the width of both buffers in pixels. The for the two images is calculated as follows:

In the rest of this document, we will decline this formula to our needs, especially restricting processing to parts where the mask generated from the reference and distorted models is equal to one. Let be a matrix of same size of and (Each mask buffer contains for each pixel i,j a binary value set to 1 if a projection for this pixel exists and 0 otherwise. See section B.2.1).

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

A conversion from RGB space to YUV space is performed by using ITU-R BT.709, since YUV space correlates better with human perception.

In the following the color image/buffer is considered YUV.

Let be a sample of the color image/buffer of the reference model for view direction , a sample of the color image/buffer of the distorted model for view direction . The color MSE for the YUV images, noted , is calculated as follows on each component in :

Let be a sample of the depth image/buffer of the reference model for view direction , a sample of the depth image/buffer of the distorted model for view direction . Let be the dynamic range of the depth signal initialized with the maximum between the diagonals of the bounding boxes of both models. The depth MSE for the depth images/buffers, noted , is calculated as follows:

This depth renormalization to 255 is used to get geometric MSEs and PSNRs comparable to the color ones.

The final metric results per frame are the followings:

And the respective associated with are computed with the following formula:

The metric tool reports also the ratio of unmatched samples defined as follows:

where is given by:

Additionally, the metric tool reports also the ratio () of the bounding box diagonal of the reference and the diagonal of the bounding box of the distorted mesh as follows:

## B.3 Calculation of MSE and PSNR for a sequence

Metrics for a sequence are obtained by computing the **Minimum**, the **Maximum**, the **Mean**, the **Variance,** and the **Standard deviation** of the metric evaluated on each frame of the sequence.

## B.4 Software usage for computing the point cloud and image-based metrics

The “equ” and “ibsm” metrics are symmetrical, but the “pcc” is not. To ensure the numerical stability of the results among participants it is required that:

* the **--modelA and --MapA** are used for the **reference** model**.**
* the **--modelB and --mapB** are used for the **distorted** model**.**

## B.4.1 Lossless condition

Lossless condition with log to console:

./mm compare \

--mode equ \

--inputModelA ref.obj –inputMapA ref.png \

--inputModelB dis.obj –inputMapB dis.png > summary.txt

## B.4.2 Lossy condition

Lossy condition on dequantized models using the point-based metric with log to console and statistics reporting into files:

./mm \

reindex --sort oriented -i ref.obj -o ID:ref\_reordered END

sample --mode grid --gridSize 512 \

--useNormal --useFixedPoint --minPos "$globalMinPos" --maxPos "$globalMaxPos" \

--bilinear -i ID:ref\_reordered -m ref.png -o ID:pcRef END\

reindex --sort oriented -i dis.obj -o ID:dis\_reordered END

sample --mode grid --gridSize 512 \

--useNormal --useFixedPoint --minPos "$globalMinPos" --maxPos "$globalMaxPos" \

--bilinear -i ID:dis\_reordered -m dis.png -o ID:pcDis END\

compare --mode pcc \

--inputModelA ID:pcRef --inputModelB ID:pcDis \

--outputCsv perFrame.csv > summary.txt

Lossy condition on dequantized models using the image-based metric with log to console and statistics reporting into files:

./mm compare --mode ibsm \

--inputModelA ref.obj --inputMapA ref.png \

--inputModelB dis.obj --inputMapB dis.png \

--outputCsv perFrame.csv > summary.txt

**Attention:** Lossy condition shall be applied on dequantized models.

One can use the dequantized models provided in the anchor CONTENT/dequantized model as a reference.

Alternatively, one can dequantize the anchor CONTENT/voxelized model on the fly while computing the metric. The values for the quantization parameters can be obtained from Table 3 (geometry precision, QP, and texture coordinate precision, QT). Quantization ranges (minPos, maxPos) are provided for each model in the anchor dataset analysis files (the files can be found in <http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-vmesh-anchor/-/tree/master/scripts/statistics_anchor> and also in Table 7).Those values can be loaded in one call from the analysis file using sh/bash as follows:

source $seqName\_statistics\_all.log

Dequantizing in one call is then performed as follow for textured meshes, assuming distorted model voxDis.obj is also quantized (remove the texture/UV related for per vertex color meshes):

./mm

dequantize --inputModel voxRef.obj --outputModel ID:deqRef \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0 0” --maxUv ”1.0 1.0” END \

reindex --sort oriented -i ID:deqRef -o ID:ref\_reordered END

sample --mode grid --gridSize 512 \

--useNormal --useFixedPoint --minPos "$globalMinPos" --maxPos "$globalMaxPos" \

--bilinear -i ID:ref\_reordered -m ref.png -o ID:pcRef END\

dequantize --inputModel voxDis.obj --outputModel ID:deqDis \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0,0” --maxUv ”1.0,1.0” END \

reindex --sort oriented -i ID:deqDis -o ID:ref\_reordered END

sample --mode grid --gridSize 512 \

--useNormal --useFixedPoint --minPos "$globalMinPos" --maxPos "$globalMaxPos" \

--bilinear -i ID:ref\_reordered -m dis.png -o ID:pcDis END\

compare --mode pcc \

--inputModelA ID:pcRef --inputModelB ID:pcDis \

--outputCsv perFrame.csv > summary.txt

./mm dequantize --inputModel voxRef.obj --outputModel ID:deqRef \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0 0” --maxUv ”1.0 1.0” END \

dequantize --inputModel voxDis.obj --outputModel ID:deqDis \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0 0” --maxUv ”1.0 1.0” END \

compare --mode ibsm \

--inputModelA ID:deqRef --inputMapA ref.png \

--inputModelB ID:deqDis --inputMapB dis.png \

--outputCsv perFrame.csv> summary.txt

## B.5.3 Sequence processing

Following sample demonstrates how to execute commands on a numerated sequence of objects ranging from 00150 to 00165 included. The "%3d" part of the file names will be replaced by the frame number ranging from firstFrame to lastFrame, coded on 3 digits.:

./mm \

sequence --firstFrame 150 --lastFrame 165 END\

dequantize --inputModel voxRef\_00%3d.obj --outputModel ID:deqRef \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0 0” --maxUv ”1.0 1.0” END \

reindex --sort oriented -i ID:deqRef -o ID:ref\_reordered END

sample --mode grid --gridSize 512 \

--useNormal --useFixedPoint --minPos "$globalMinPos" --maxPos "$globalMaxPos" \

--bilinear -i ID:ref\_reordered -m ref\_00%3d.png -o ID:pcRef END\

dequantize --inputModel voxDis\_00%3d.obj --outputModel ID:deqDis \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0,0” --maxUv ”1.0,1.0” END \

reindex --sort oriented -i ID:deqDis -o ID:ref\_reordered END

sample --mode grid --gridSize 512 \

--useNormal --useFixedPoint --minPos "$globalMinPos" --maxPos "$globalMaxPos" \

--bilinear -i ID:ref\_reordered -m dis\_00%3d.png -o ID:pcDis END\

compare --mode pcc \

--inputModelA ID:pcRef --inputModelB ID:pcDis \

--outputCsv perFrame.csv > summary.txt

./mm sequence --firstFrame 150 --lastFrame 165 END\

dequantize --inputModel voxRef\_00%3d.obj --outputModel ID:deqRef \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0 0” --maxUv ”1.0 1.0” END \

dequantize --inputModel voxDis\_00%3d.obj --outputModel ID:deqDis \

--qp $QP --minPos ”$globalMinPos” --maxPos ”$globalMaxPos”

--qt $QT --minUv ”0 0” --maxUv ”1.0 1.0” END \

compare --mode ibsm \

--inputModelA ID:deqRef --inputMapA ref\_00%3d.png \

--inputModelB ID:deqDis --inputMapB dis\_00%3d.png \

--outputCsv perFrame.csv > summary.txt

On the file summary.txt, the **Maximum**, the **Minimum**, the **Mean**, the **Variance,** and the **Standard deviation** of the metric evaluated on each frame of the sequence can also be found.

# Annex C: Anchor tool usage

Anchors were generated in the following manner.

Texture maps represented by a sequence of PNG files were first converted to GBR444 or YUV420 video formats using HDRTools, and subsequently encoded with HEVC HM 16.21 with Screen Content Coding Extension (SCC) 8.8.

The HDRTools software can be downloaded from the following link:

* <https://gitlab.com/standards/HDRTools.git>

The configuration files used for conversion can be found at:

* <http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-vmesh-anchor/-/tree/master/scripts/cfg/hdrconvert>

The required version of the HEVC test model can be obtained using the following command:

* git clone --branch HM-16.21+SCM-8.8 <https://vcgit.hhi.fraunhofer.de/jvet/HM.git>

The HM configuration files used for all-intra and random-access conditions can be found in the following link:

* <http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-vmesh-anchor/-/tree/master/scripts/cfg/hm>

For the lossless case, the mesh geometry of the dataset was coded using a modified version of SC3DMC, the MPEG reference software for static mesh compression, which can be found in the following link:

* <http://mpegx.int-evry.fr/software/MPEG/PCC/tfan_mesh_anchor.git>

And the configuration file used for lossless compression is available at:

* <http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-vmesh-anchor/-/tree/master/scripts/cfg/afx>

For the lossy condition, a mesh compression tool from Google, Draco, was used, and available in the following link:

* <https://google.github.io/draco/>

To achieve certain bitrates, additionally to mesh compression, mesh decimation and texture sub-sampling were also applied to the dataset. For texture sub-sampling, HDRTools was used, while for mesh decimation, Meshlab software was used. The Meshlab software can be downloaded from here:

* git clone <https://github.com/cnr-isti-vclab/meshlab/releases/tag/Meshlab-2020.03>

And the configuration files used to create the decimated meshes are available here:

* <http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-vmesh-anchor/-/tree/master/scripts/cfg/meshlab>

Since mesh decimation consistent results across platforms could not be guaranteed, the decimated meshes used for the anchor generation are also available in the MPEG repository:

* <https://mpegfs.int-evry.fr/mpegcontent/ws-mpegcontent/CfP/Mesh/CONTENT/simplified>

The compressed anchor bitstreams for different settings and MD5 checksums are available at:

* <https://mpegfs.int-evry.fr/mpegcontent/ws-mpegcontent/CfP/Mesh/ANCHOR/bitstreams>

In order to decode the provided bitstreams and reconstruct the meshes, the following procedure was done:

Texture map files encoded with HM, the binary files can be decoded with the respective HM encoder with the following command:

**TAppDecoderStatic** \

--BitstreamFile=${inBin} --ReconFile=${outYUV} \

--OutputBitDepth=8 --OutputBitDepthC=8

If a sequence of texture map images per frame is required, the output of the video decoder can be further transformed to a sequence of PNG files by using HDRTools with the following commands:

**HDRConvert** \

-f ${config} \

-p SourceFile=${input\_yuv\_video} \

-p OutputFile=${output\_rgb\_video} \

-p SourceWidth=${inWidth} -p SourceHeight=${inHeight}\

-p NumberOfFrames=${numFr} \

-p OuputWidth=${outWidth} -p OutputHeight=${outHeight}

The meshes for the lossy condition can be decoded using the Draco decoder with the following command:

**draco\_decoder** -i ${compMesh}.drc -o ${compMesh}.obj

The meshes for the lossless condition can be decoded using the SC3DMC decoder with the following command:

**SC3DMCDecoder** ${compMesh} ${compMesh}

The decoded anchor meshes can be found under the following URL for each of the categories and files:

* <https://mpegfs.int-evry.fr/mpegcontent/ws-mpegcontent/CfP/Mesh/ANCHOR/decoded_meshes>

Results of the anchor are available in the Excel sheet provided with this CfP. The distortion is calculated according to Section 7.

Each zip file of a test material dataset contains an MD5 file (with the corresponding md5 sums for each file in the archive).

The anchor can be generated with software previously indicated, which may be compiled on a diverse number of platforms, such as MacOS, Linux and/or Windows. The following scripts to generate the anchor results are the following (also made available in the github - <http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-vmesh-anchor/-/tree/master/scripts>):

* **lossy\_anchor\_generate.sh**: script to generate lossy anchor results
* **lossy\_anchor\_verification.sh**: script to verify the lossy anchor results
* **lossless\_anchor\_generate.sh**: script to generate lossless anchor results
* **lossless\_anchor.sh\_verification**: script to verify the lossless anchor results

Note that the scripts were created for the Linux platform only. All scripts contain an initialize\_script() function that will create local directories and check for the presence of the appropriate software, please adjust the paths for your particular system.

Lossless anchor

Texture Coding

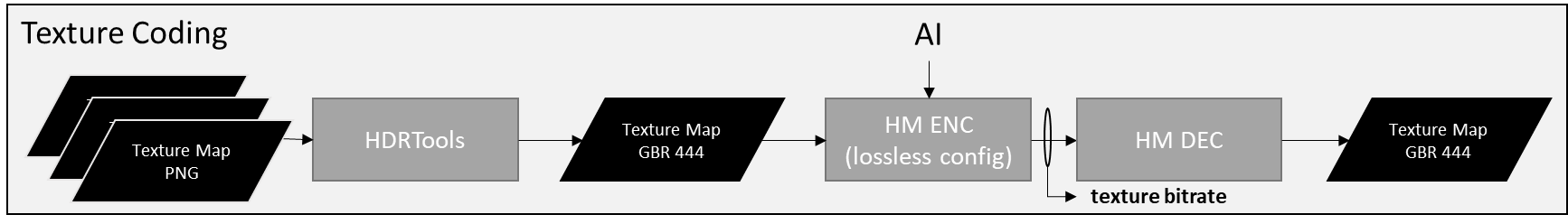
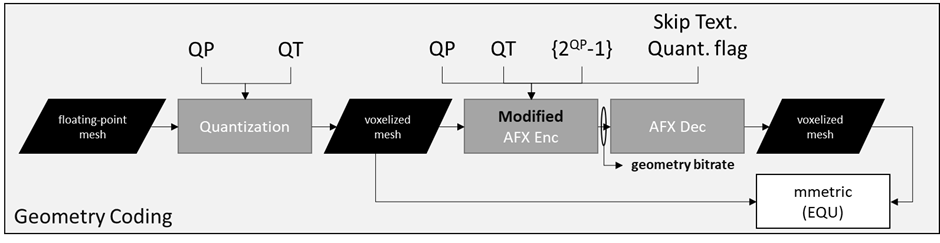


Figure : Lossless texture map coding

For lossless compression of texture maps, the sequence of PNG files is first converted to a GBR 444 video format, which is used as the input for the HM encoder. The necessary configuration files for all-intra condition are present in the configuration folder.

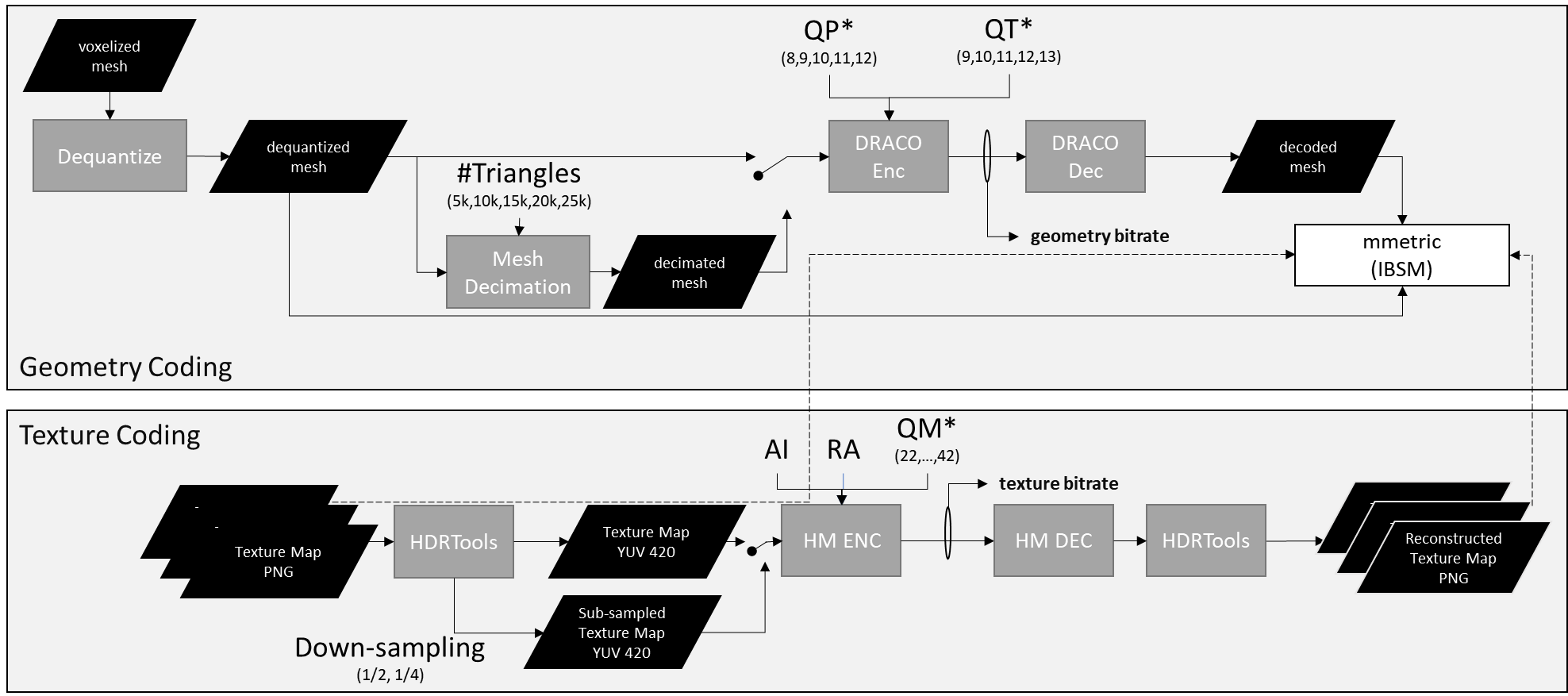
Geometry Coding

Figure : Lossless geometry coding

In order to lossless encode the integer-valued mesh sequences, a modification to skip the quantization procedure for texture coordinates in the AFX software was necessary. The input parameter to skip quantization of the texture coordinates was added to the software available in the MPEG git repository. Both input and output meshes are compared for equality, to guarantee the preservation of vertex attributes, even if the order of vertices and connectivity is changed.

Lossy anchor

Lossy texture and mesh coding

Figure : Block diagram for lossy mesh compression

For texture coding, the sequence of PNG images is converted into a YUV420 video sequence for subsequent encoding with HM. For the geometry part, Draco is used to encode the mesh connectivity, the vertices 3D position and texture coordinates. In order to achieve the desired target bitrates, in addition to coding the meshes with HM and Draco, a combination of different quantization parameters and pre-processing techniques, such as mesh decimation and texture sub-sampling, was used. A search for the parameter combination was conducted and the selected parameters can be found in Table 8 and Table 9.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category** | **Test**  **Class** | **Test material**  **dataset filename** | **Rate** | **Draco**  **QP** | **Draco**  **QT** | **Mesh Resolution** | **HM**  **QP** | **Texture**  **Resolution** |
| Dynamic Objects with Texture Mapping | A | Longdress | R1 | 7 | 7 | 5 | 51 | 1 |
| R2 | 10 | 9 | 15 | 44 | 1 |
| R3 | 11 | 10 | 20 | 42 | 1 |
| R4 | 12 | 9 | 25 | 40 | 1 |
| R5 | 12 | 10 | original | 36 | 1 |
| Soldier | R1 | 7 | 7 | 10 | 51 | 1 |
| R2 | 10 | 8 | 20 | 44 | 1 |
| R3 | 11 | 9 | 25 | 42 | 1 |
| R4 | 12 | 9 | 25 | 36 | 1 |
| R5 | 12 | 10 | original | 32 | 1 |
| B | Basketball\_player | R1 | 7 | 8 | 5 | 51 | 1 |
| R2 | 10 | 8 | 10 | 48 | 1 |
| R3 | 12 | 9 | 20 | 42 | 1 |
| R4 | 12 | 10 | 25 | 36 | 1 |
| R5 | 12 | 11 | original | 32 | 1 |
| Dancer | R1 | 7 | 7 | 5 | 48 | 1 |
| R2 | 9 | 9 | 10 | 44 | 1 |
| R3 | 12 | 7 | 20 | 40 | 1 |
| R4 | 12 | 10 | 25 | 36 | 1 |
| R5 | 12 | 10 | original | 32 | 1 |
| C | Mitch | R1 | 10 | 8 | 5 | 48 | 1 |
| R2 | 12 | 9 | 5 | 44 | 1 |
| R3 | 12 | 8 | 10 | 42 | 1 |
| R4 | 11 | 9 | 15 | 40 | 1 |
| R5 | 12 | 10 | 20 | 38 | 1 |
| Thomas | R1 | 12 | 8 | 5 | 51 | 1 |
| R2 | 10 | 9 | 10 | 48 | 1 |
| R3 | 11 | 9 | 15 | 44 | 1 |
| R4 | 12 | 11 | 15 | 40 | 1 |
| R5 | 12 | 11 | 25 | 38 | 1 |
| Football | R1 | 6 | 7 | 5 | 51 | 4 |
| R2 | 10 | 9 | 15 | 48 | 4 |
| R3 | 12 | 8 | 25 | 44 | 4 |
| R4 | 12 | 10 | 25 | 40 | 2 |
| R5 | 12 | 11 | original | 44 | 1 |
| Levi | R1 | -1 | -1 | -1 | -1 | -1 |
| R2 | 7 | 2 | 25 | 51 | 4 |
| R3 | 11 | 2 | 25 | 48 | 4 |
| R4 | 10 | 7 | 25 | 42 | 4 |
| R5 | 12 | 9 | original | 36 | 4 |

Table : Coding parameters for random access lossy anchor generation

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Category** | **Test**  **Class** | **Test material**  **dataset filename** | **Rate** | **Draco**  **QP** | **Draco**  **QT** | **Mesh Resolution** | **HM**  **QP** | **Texture**  **Resolution** |
| Dynamic Objects with Texture Mapping | A | Longdress | R1 | 11 | 8 | 5 | 51 | 1 |
| R2 | 10 | 9 | 15 | 48 | 1 |
| R3 | 12 | 10 | 10 | 44 | 1 |
| R4 | 12 | 9 | 15 | 42 | 1 |
| R5 | 12 | 11 | 25 | 40 | 1 |
| Soldier | R1 | 9 | 8 | 10 | 51 | 1 |
| R2 | 10 | 9 | 20 | 48 | 1 |
| R3 | 12 | 8 | 20 | 44 | 1 |
| R4 | 12 | 9 | 25 | 42 | 1 |
| R5 | 12 | 11 | original | 38 | 1 |
| B | Basketball\_player | R1 | 8 | 7 | 5 | 51 | 1 |
| R2 | 9 | 8 | 10 | 48 | 1 |
| R3 | 12 | 10 | 15 | 44 | 1 |
| R4 | 12 | 10 | 25 | 42 | 1 |
| R5 | 12 | 10 | original | 38 | 1 |
| Dancer | R1 | 8 | 7 | 5 | 51 | 1 |
| R2 | 9 | 7 | 10 | 48 | 1 |
| R3 | 11 | 8 | 20 | 44 | 1 |
| R4 | 12 | 11 | 20 | 40 | 1 |
| R5 | 12 | 10 | original | 38 | 1 |
| C | Mitch | R1 | 7 | 8 | 5 | 51 | 1 |
| R2 | 10 | 8 | 5 | 48 | 1 |
| R3 | 10 | 9 | 10 | 44 | 1 |
| R4 | 11 | 9 | 15 | 42 | 1 |
| R5 | 12 | 11 | 20 | 40 | 1 |
| Thomas | R1 | 11 | 9 | 5 | 51 | 1 |
| R2 | 11 | 7 | 10 | 48 | 1 |
| R3 | 12 | 11 | 15 | 44 | 1 |
| R4 | 12 | 11 | 25 | 42 | 1 |
| R5 | 12 | 12 | original | 38 | 1 |
| Football | R1 | 6 | 7 | 5 | 51 | 4 |
| R2 | 9 | 8 | 15 | 48 | 4 |
| R3 | 11 | 9 | 20 | 51 | 2 |
| R4 | 12 | 11 | 25 | 48 | 2 |
| R5 | 12 | 11 | original | 44 | 2 |
| Levi | R1 | -1 | -1 | -1 | -1 | -1 |
| R2 | 6 | 2 | 25 | 51 | 4 |
| R3 | 9 | 3 | 25 | 48 | 4 |
| R4 | 9 | 6 | 25 | 51 | 2 |
| R5 | 12 | 7 | original | 44 | 4 |

Table : Coding parameters for all-intra lossy anchor generation

Notice that in the case of Levi dataset, Meshlab could not perform mesh decimation due to the characteristics of the input mesh (the data contain several small texture map islands, so the mesh decimation that preserves the texture island boundaries could not achieve the target triangle count, since most of the vertices belong to a border and could not be modified).

Annex D: Details on subjective testing

## D.1 Generation of video sequences

The Mesh renderer[2] will be configured to output video sequences with the following video parameters:

* Video resolution: progressive uncompressed 1920x1080p Full HD format
* Frame rate: The frame rate is fixed to 30 frames per second
* Color space: ITU-R BT.709
* Sub-sampling: 4:2:0 YUV 10 bit little endian

The background color shall be set to a medium grey (RGB 128/128/128) and the floor color shall be set to a slightly darker grey such as (RGB 95/95/95).

The Mesh renderer software can be found in the MPEG GIT:

<http://mpegx.int-evry.fr/software/MPEG/PCC/mpeg-pcc-renderer.git>

For producing the videos, the first step is to generate a camera path for each sequence. The generation of camera paths is described in [15] section 3.2.3 and the resulting path is stored in a .txt file.

The second step is the generation of the videos with default parameters as described above by using the example command line:

./mpeg-pcc-renderer/scripts/renderer.sh -i ./dec/PxxSxxCxxRxx/ --camera=camerapath\_xxxx.txt

Note that the renderer performs the dequantization procedure for decoded videos as decribed in Annex E

## D.2 Delivery of submissions for subjective testing

The encoded bitstreams shall be submitted to the FTP site provided to each proponent. Proponents shall provide: decoder executable (including everything that is needed for using the decoder, such as decoding scripts), compressed bitstreams, and corresponding MD5 checksums. Decoded bitstreams have to be provided on request for verification needs. Due to the large file size of decoded bitstreams, only their MD5 checksums are required to be provided to the FTP site in a text file by the response date.

## D.3 Laboratory setup

The test coordinator will select the test laboratories and make sure that these comply with MPEG guidelines [5] and ITU-R BT. 500[13]. Test coordinator and test lab will agree on handling of scoring tool (e.g. paper sheet or electronic). Test coordinator will deliver a consolidated document with all test results to WG7.

## D.4 Selection and training of test subjects

Test subjects will be naive viewers and each subject will be carefully screened for visual acuity (Snellen Chart) and color blindness (Ishihara tables). The test coordinator and test lab will make sure that the test panel has a state-of-the-art number of test subjects.

Before participating in a test experiment, all subjects will participate in a training activity during which a detailed explanation of the test scope, the test method, the voting procedure, and the kind of assessment they are expected to do, will be provided.

The training activity will include a short demonstration session conducted after the training explanation, to let the subjects practice with the scoring procedure and to allow them to familiarize themselves with the video sequences; some explanations will be provided about the kind of impairments to look for in the videos that they will see.

The video sequences included in the demonstration session will be additional sequences that are not used for the formal test.

## D.5 Related costs

The testing fee when responding to the CfP is set to EUR 3000 per candidate codec. The fee is used to compensate the efforts of the recruited test subjects.

Annex E: Visualization of voxelized mesh content

For the dataset content, the vertices attributes (3D position and texture coordinate) are provided in integer format. For correct visualization of these voxelized meshes, a dequantization procedure that converts the integer values to floating point values should be conducted. The mmetric software provides the capability to dequantize the voxelized meshes by applying the following command line:

**mm** \

**dequantize** --inputModel ${voxelizedMesh} --outputModel ${outMesh} \

--qp ${qp} --qt ${qt} \

--minPos ${globalPosMin} --maxPos ${globalMaxPos} \

--minUV “0.0 0.0” --maxUV “1.0 1.0” \

--useFixedPoint

The values for qp (quantization parameter for vertex 3D coordinate) and qt (quantization parameters for texture coordinate) is equivalent to the options of Geometry Precision and Texture Coordinate Precision respectively, and are indicated in Table 3. The values for globalPosMin and globalPosMax are provided in Table 10.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Test material**  **dataset filename** | **globalPosMin** | | | **globalMaxPos** | | | **maxBBLength** |
| **(x)** | **(y)** | **(z)** | **(x)** | **(y)** | **(z)** |
| Longdress | -0.475553989 | -1.4576 | -0.284981996 | 481.324005 | 1023.37 | 659.137024 | 1024.8276 |
| Soldier | -0.366236001 | 1.10722005 | 0.224947006 | 508.764008 | 1023.37 | 637.421997 | 1022.26277995 |
| Basketball\_player | -725.812988 | -483.908997 | -586.02002 | 1252.02002 | 1411.98999 | 1025.34998 | 1977.833008 |
| Dancer | -902.244995 | -486.196991 | -670.518005 | 621.093994 | 1576.04004 | 738.028992 | 2062.237031 |
| Mitch | -588.255981 | 5.80515003 | -469.799011 | 734.567993 | 1829.69995 | 697.179016 | 1823.89479997 |
| Thomas | -265.006989 | -4.04448986 | -248.710999 | 320.546997 | 1820.93005 | 400.225006 | 1824.97453986 |
| Football | -0.000159517003 | 3.32326999e-06 | 0.000132931003 | 1024 | 980.619995 | 966.692993 | 1023.96268540018 |
| Levi | -0.780686975 | -0.0424938016 | -0.594317973 | 0.857237995 | 1.90897 | 0.687259018 | 1.9514638016 |

Table Conversion parameters for the Mesh sequences

The voxelized content has been dequantized using the procedure described above, the results can be found in the following URL:

* <https://mpegfs.int-evry.fr/mpegcontent/ws-mpegcontent/CfP/Mesh/CONTENT/dequantized>

Note: Downloaded test material datasets should be verified with MD5 checksums. Each zip file of a test material dataset contains an MD5 file (with the corresponding md5 sums for each file in the archive). Only the meshes are provided, the texture map is the same as the ones in the voxelized folder.