 ISO/IEC JTC 1/SC 29/WG 04 N0309

**ISO/IEC JTC 1/SC 29/WG 04  
MPEG Video Coding   
Convenorship: CN**

**Document type:** Output Document

**Title:** Call for MPEG immersive video test materials

**Status:** Approved

**Date of document:** 2023-02-10

**Source:** ISO/IEC JTC 1/SC 29/WG 04

**Expected action:** None

**Action due date:** None

**No. of pages:** 8 (without cover page)

**Email of Convenor:** yul@zju.edu.cn

**Committee URL:** <https://sd.iso.org/documents/ui/#!/browse/iso/iso-iec-jtc-1/iso-iec-jtc-1-sc-29/iso-iec-jtc-1-sc-29-wg-4>

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION

ORGANISATION INTERNATIONALE DE NORMALISATION

ISO/IEC JTC 1/SC 29/WG 04 MPEG VIDEO CODING

**ISO/IEC JTC 1/SC 29/WG 04 N** **0309**

**January 2023, Online**

|  |  |
| --- | --- |
| Title | Call for MPEG Immersive Video test materials |
| Source | WG 04, MPEG Video Coding |
| Status | Approved |
| Serial Number | 22396 |

# Introduction

The Motion Pictures Experts Group (MPEG), during its process of standardization, routinely conducts experiments to explore new technologies and to verify technologies that are introduced into a standard. The Video sub-group (WG04) also follows the same process. Experiments are performed on suitable video test material relevant to the experiment, where the test material are sourced from MPEG members and non-members alike. Without good test material, the outcomes of experiments could lead to inaccurate conclusions.

Currently, WG04, as part of its exploration activities, is performing experimentation on 6-DoF immersive video coding. In this activity, a three dimensional scene that is captured by numerous cameras (camera rig) is analysed and encoded. The arrangment of the cameras that capture the scene may have any arragement (regular/irregular). There is also no limitation placed on the inter-camera baseline distances; it may be narrow or wide. Each camera in the camera rig provides its captured videos in a known projection format, e.g. perspective or equirectangular.

While conducting exploration experiments on the above-listed topics, WG04 has realized that there is a dire need for more and appropriate test materials, especially omnidirectional (360 degree) 6DoF video content, as well as camera rig video sequences (aka multiview) of non-Lambertian scenes (e.g. scenes with mirror reflections and/or transparent objects) in preparing the second edition of MIV (MPEG Immersive Video). There is also an immediate need for content that is captured/synthesized using non-planar rigs. Therefore, WG04 solicits new test materials from contributors. In the following sections, the characteristics of test materials solicited, their formatting, and the process of contribution are described.

# Characteristics of test materials

## 6DoF immersive video coding experiments

For experiments on 6DoF immersive video coding, a typical test content material is a multi-view video of some 5 to 10 seconds at a frame-rate greater than or equal to 30 frames per second. The test materials are either captured using a real physical camera rig or synthesised using computer graphics editing softwares such as Blender, Unreal, or Unity. The characteristics of test content that are sought are as follows:

* The cameras in the rig are either:
  + inward-looking, where the cameras are all pointed towards a central region of a sphere,
  + outward-looking, where the cameras are all pointing towards the horizon in world space, or
  + a combination of both inward and outward looking camera constellation.
* The cameras in the capture rig can have temporally varying intrinsics and extrinsic parameters. The current source input format does not support dynamic camera parameter change. Therefore, we also solicit input contributions for extending the current source format to support dynamic camera parameter changes, and also support for scenarios where cameras can be added or dropped during capture.
* The cameras in the camera rig can use any projection type that corresponds to a physical camera, e.g. perspective or fisheye. If the camera captured video stream requires correction for distortion, it would be desirable that the distortion parameters and/or the undistorted video is also included in the contribution. Additionally, equirectangular projection can also be used. Currently, there is a lack of fully omini-directional test content for experiments, and any added content that is fully omni-directional would be highly recommended.
* The capture rig should be such that the viewing space (the 3D space from within which a viewport of the scene can be rendered without significant rendering artefacts) be larger than a simple head-scale 6DoF motion. There is no specific limitations on how large the viewing space should be, but ideally, the viewing space should be large enough that a viewer is able to walk around.
* High-quality depth maps corresponding to the captured texture videos should be provided. If available, a depth confidence map can also be provided. The depth-maps can either use a single colour channel representation or a colourized depth representation, where multiple colour channels (RGB or YUV) are used for the depth representation. Furthermore, the camera parameters, both intrinsics (e.g. auto-focus) and extrinsics, of the capture depth video need not be aligned with cameras that capture other attributes (e.g. texture).
* It is also desirable to have good quality object segmentation maps corresponding to the captured videos. Each pixel in the object map shall have a unique non-negative integer identifier that enumerates the segmented object it belongs to.
* In general, the test material should be complex enough to represent the state of the art use-cases used by the industry. Test material of both indoor or outdoor scenery of the natural world are highly encouraged. Other properties of the scenes (some properties often referred to as non-Lambertian) needed for experimentation are:
* scenes with fine geometry (thin surfaces such as hair/fur, grass and alike).
* complex light interactions (global illumination, Fresnel effects and alike).
* scenes with light scattering medium such as fog, cloud, water.
* scenes with biological entities (for e.g., humans, animals, and vegetation).
* scenes where in addition to texture and geometry, data such as bi-directional distribution functions (BRDFs), materials characteristics, and transparency are also made available.
* scenes with transparency values contributed as RGBA videos.

# Recommended formats for the contribution of test materials

## Default YUV input format

In this format, geometry and attributes like texture are delivered separately in YUV format. A handy and common way of naming the file is the following:

v{*i*}\_{*t*}\_{*w*}x{*h*}\_yuv{*f*}p{b}le.yuv

where,

|  |  |
| --- | --- |
| *i* | A unique positive integer index used to identify the camera that captured the video sequence. The indexing should preferably start from zero. |
| *t* | denotes the property of the video that is represented by the video stream. The following types are allowed: (a) texture (b) depth (c) entitymap |
| *w* | width (total number of pixels in a row) of a frame in the Y (luma) channel of the video sequence. |
| *h* | height (total number of pixels in a column) of a frame in the Y (luma) channel of the video sequence. |
| *f* | The yuv sub-sampling format used for the video sequence. e.g. 420 |
| *b* | The bits per channel of the YUV sequence. |

Examples:

* v0\_depth\_2048x1088\_yuv420p16le.yuv
* v2\_texture\_2048x1088\_yuv420p10le.yuv
* v3\_entity\_2048x1088\_yuv420p.yuv

It is possible to have content with uncomplete views by using a specific coding value of the geometry.

The default input format is completed by a JSON file described in Annex A of this document describing all necessary information like camera intrinsics and extrinsics.

If other formats are provided, it would be highly desirable that a converter to convert from the proposed format to a YUV format also be provided.

## MPI input format

Additionally, we are also soliciting video content in the form of Multi-Plane images (MPI) [2]. This representation consists of an ordered set of layers, each layer representing texture and transparency with a constant depth. The depth is itself ideally represented as a normalized disparity value. It is also possible to provide multiple MPI sequences for the same volumetric scene, for example MPIs generated from different viewing positions and orientations. The MPI sequences can be provided as either one of three alternates:

* as YUV texture plus transparency format, or
* a much more compact .pcs format, as described in [4]
* as files in OpenEXR 2.0 [3] format.

The YUV texture plus transparency format consists of one yuv420p10le YUV sequence file for texture and another yuv400p8 YUV sequence file for transparency. Each frame of the MPI sequence (both texture and transparency) is a concatenated sequence of MPI layers ordered by decreasing depth (furthest first to closest last). An MPI sequence, for both texture and transparency, is a concatenation of MPI frames in presentation order. A JSON file should include all necessary information, and a document describing the process of the depth value recovery of each layer should also be provided.

As an example, the JSON file can include the total number of layers *n*, the parameters of the camera used for the MPI including the minimum disparity value, *1/zmin*, and maximum disparity value, *1/zmax*, the number of bits, *b*, used for coding the depth values.

Disparity, *d*, is then computed by the equation

*d* = *l* + (*h* – *l*) \* *b* / (2\*\**n* – 1),

where *d* denotes disparity of a layer, *h* denotes the maximum disparity, *l* denotes the minimum disparity, *b* denotes the layer index, and *n* denotes the number of bits for coding the disparity.

## OpenEXR format

Another option to contribute test materials is to use the OpenEXR 2.0 format. The OpenEXR 2.0 format allows specifying a variable list of samples per pixels, to generate deep images. Multiple values at different depths could be stored for each pixel in the sequence. Additionally, each file can contain several separate, but related, data parts. This enables storing of multiple MPIs in a single file, if needed. The compression of the OpenEXR files should be chosen such that artefacts are avoided.

## Recommendations for contribution of photorealistic synthetic content

In the case of photorealistic synthetic content, it is recommended that in addition to the rendered image data, the 3D model itself, including scripts that are required to generate the dataset is supplied. This would enable other MPEG experts to render variants of the same scene. For instance, it would be possible to simulate a virtual camera array by placing multiple cameras in the scene. Preferably, scenes should include a timeline that allows for rendering of short movies with some dynamics. Due to the open source and easy accessibility of the Blender software, a blend file would be the most preferred format for this type of content. Furthermore, it is recommended that contributors also provide at least three ground-truth pose-traces rendered from a moving camera with in the scene. The generated pose-traces should be smooth, slow, and comfortably viewable on a normal 2D display. Such a pose trace could be rendered with a perspective viewport with 1920x1080 resolution and a 60° field-of-view.

The format of the 3D model should be in a suitable interchange format. Typical representations of 3D models are:

* Texture, mesh, and lighting information,
* Procedural scenes (e.g. algorithms).

Content providers are nevertheless asked to render their 3D scene to create multiview image-based representations. The depth maps should also be generated.

## Mesh format

Mesh content featuring multiple objects or complex scenes are solicited for the MPEG joint exploration experiments on Mesh-MIV coding. Mesh contents should be contributed as a sequence of frames, where each frame contains at least an OBJ file and a PNG file. Additionally, an MTL file per frame can also be provided. The naming convention is as follows:

SequenceName\_f%06d.obj / .png / .mtl

The frames have to be sequential and use 0-based indexing. Cleaning and decimation tools such as Meshlab can be used to improve the quality of the mesh content. When filtering is done, care should be taken to ensure that the texture map is also updated to reflect the refined geometry, while keeping the best fidelity possible. Contributors are also encouraged to submit raw camera captures (in the default YUV input format) along with their camera parameters so that the same content can be used as the multi-view variant in the joint exploration activity. Documentation of how the content was generated is highly recommended.

# Process for contribution of new test materials

To ensure that the contributed test content is in proper order, and is consistent with the requirements of experiments to be performed, it is requested that all details of the test materials be introduced to the group as an input contribution, at least a month in advance of a scheduled MPEG WG04 meeting. Upon receiving such contributions, the chairs of the AhG will organize time in a call, for the contribution to be discussed.

# Information about copyrights

Content owners should provide a copyright notice along with the dataset to inform MPEG about copyright and usage restrictions. It is recommended that the copyrights notice and license statement be suitable for the content to be used for experimentation in MPEG, academic research, and standard promotion purposes.

# Contact information

For further information/clarification on any aspect regarding the creation, formatting, or the process of contributing test materials, the following people can be contacted:

* Gauthier Lafruit *(gauthier.lafruit (at) ulb.be)*
* Jun Young Jeong (*jyj0120 (at) etri.re.kr*)

# Annex-A

JSON format describing content

Metadata shall be provided in the form of a JSON file that lists the following properties, in arbitrary order, for each video. The properties are the same for all frames, and listed below.

There is a block of general information:

* Version: A version number in x.x format that is modified when required after consensus in the group.
* Content\_name: A common name linking this file to the given content
* Fps: The frame rate of the captured source content
* Frames\_number: The total number of frames
* BoundingBox\_center: The center of the bounding box, expressed in OMAF coordinate system [1]
* lengthsInMeters: A JSON boolean indicating if the length parameters in the JSON file (e.g. depth range) are measured in meters.
* sourceCameraNames: a list containing the camera names used for capturing the scene.
* sourceCameraIds: a ordered list containing indices, which are expressed as non-negative integer values, associating the camera names in sourceCameraNames.
* Additionally, optional information that could be useful for understanding the context of the scene can be added. However, this information will not be used by the test model software.

For each camera:

* Name: Camera Name of the file, as used in the file names as described in 3.1
* Position: position of the center of the camera as three values [x, y, z] in meters in OMAF coordinate system, as explained in figures 5.3 & 5.4 of [1],
* Rotation: orientation of the related camera [yaw, pitch, roll] expressed in degree and in OMAF coordinate system, as explained in figures 5.3 & 5.4 of [1],
* Depthmap: if the view has a depth map or not (Boolean 1: true, 0: false),
* Background: if the view is background or not (Boolean 1:true, 0:false)
* HasInvalidDepth: if the view is uncomplete (true), or complete (false). Non-existent pixels are identified by a depth value of 0.
* Depth\_Range: [Rnear, Rfar]. This “R” denomination should be understood here as generic: it is either a radius value if the projection format is equirectangular, or a z value if the projection format is perspective. The Rfar value is permitted to be infinite. When the Rfar value is meant to be infinite, it will be arbitrarily written as the 1000.0 value.
* Resolution: image/video resolution [width x height]
* Projection: “Perspective” or “Equirectangular”
* Hor\_range and Ver\_range (in case of equirectangular projection): image/video horizontal and vertical range [phimin, phimax] x [thetamin, thetamax]. Full Field of View (FoV) is [-180, 180] x [-90, 90]. These ranges are expressed in the camera coordinate system.
* Focal (in case of perspective projection): focal expressed in pixels; this field is only valid for linear perspective camera. When this field is present, the camera is understood to be in linear perspective (pinhole camera).
* Principal point (in case of perspective projection)
* BitDepthColor and BitDepthDepth: number of bits for texture and depth respectively
* Video optional field (0: image, 1: video). When this field is not present, textures and depth inputs for that camera are made of images
* ColorSpace and DepthColorSpace: YUV420 by default

An optional Viewport view may be described to recommend a viewport.

Format of real numbers is eee.ffff where eee and ffff are respectively integer and fractional part of any length.

Examples of input JSON format can be found at <https://gitlab.com/mpeg-i-visual/tmiv/-/tree/v11.1/config/ctc/sequences>.

# References

1. ISO/IEC 23090-2:2021 Information technology — Coded representation of immersive media — Part 2: Omnidirectional media format
2. Immersive Light Field Video with a layered mesh representation, Broxton et all, SIGGRAPH 2020 Technical Paper, <https://augmentedperception.github.io/deepviewvideo/>
3. <https://www.openexr.com/>
4. m55709 Proposal for an optimized input format for MPI, MPEG133