 ISO/IEC JTC 1/SC 29/WG 04 N0144

**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:** 2021-10-15

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

**Expected action:** None

**Action due date:** None

**No. of pages:** 9 (including cover page)

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

**Committee URL:** <https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg4><https://isotc.iso.org/livelink/livelink/open/jtc1sc29wg3>

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** **0144**

**July 2021, Online**

|  |  |
| --- | --- |
| Title | Call for MPEG Immersive Video test materials |
| Source | WG 04, MPEG Video Coding |
| Status | Approved |
| Serial Number | 20925 |

# 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 [1]. In this activity, a three dimensional scene that is captured by numerous cameras (camera rig) is analysed and encoded. The arragment of the cameras that capture the scene may have any arragement (reqular/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 ERP.

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. 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 their contribution are described.

# Characteristics of test materials

## 6-DoF immersive video coding experiments

For experiments on 6-DoF 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 camera in the camera rig can use any projection type that corresponds to a physical camera, e.g. perspective or fisheye. 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.
* 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 indoor or outdoor scenery of the natural world are highly encouraged. Other properties of the scenes that is interesting 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 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* | Type denotes the property of the video that is represented by the video stream. The following types are allowed: (a) texture (b) depth |
| *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

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 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) [4]. 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 [6]
* as files in OpenEXR 2.0 [5] 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 number of bits for coding the disparity, and *n* denotes the total number of layers.

## 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 any scripts that is 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.

The format of the 3D model should be in a suitable interchange format [2]. 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.

# 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:

* [Vinod Kumar Malamal Vadakital](mailto:vinod.malamalvadakital@nokia.com) (vinod.malamalvadakital (at) nokia.com)
* [Renaud Dore](mailto:Renaud.Dore@InterDigital.com) (renaud.dore (at) interdigital.com)

# Annex-A

JSON format describing content

Metadata shall be provided under the form of a JSON file that enlists in any order the following properties per video, the properties being the same for all frames of the content, and listed here below

There is a block of general information:

* A general name linking this file to a given content
* The fps of the content (30 / 60 / 90)
* The total number of frames
* The center of the bounding box, expressed in OMAF coordinate system [3]
* Additionally, optional information that could be useful for understanding the context of the scene can also be added, but will not used by the MIV encoder.

For each camera

* Name: Camera Name of the file, as used in the file names described here above
* 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 [3],
* 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 [3],
* 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 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 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 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.

Example an input JSON file format provided below:

{

"Content\_name": "Fan",

"BoundingBox\_center": [1.65,2.50,1.45],

"Fps": 30,

"Frames\_number": 5,

"Informative":

{

"Object\_range": [0.35, 12.50],

"Cameras\_number": 15,

"RigRadius": 0.40

},

"cameras":

[

{

"Name":"viewport",

"Position": [-2.5000, 1.6500, 1.4500],

"Rotation": [-33.9126, 7.7346, -0.0000],

"Depthmap": 1,

"Background": 1,

"HasInvalidDepth": false,

"Depth\_range": [0.35, 12.50],

"Resolution": [1920,1080],

"Projection": "Perspective",

"Focal": [2058.73, 2058.73],

"Principle\_point":[960,540],

"BitDepthColor": 10,

"BitDepthDepth": 16,

"ColorSpace": "YUV420",

"DepthColorSpace": "YUV420"

},

{

"Name":"v0",

"RowCol":"v00",

"Position": [-2.6004, 1.4765, 1.5491],

"Rotation": [-33.9126, 7.7346, -0.0000],

"Depthmap": 1,

"Background": 1,

"HasInvalidDepth": false,

"Depth\_range": [0.35, 12.50],

"Resolution": [1920,1080],

"Projection": "Perspective",

"Focal": [2058.73, 2058.73],

"Principle\_point":[960,540],

"BitDepthColor": 10,

"BitDepthDepth": 16,

"ColorSpace": "YUV420",

"DepthColorSpace": "YUV420"

},

{

"Name":"v1",

"RowCol":"v01",

"Position": [-2.5446, 1.5595, 1.5491],

"Rotation": [-33.9126, 7.7346, -0.0000],

"Depthmap": 1,

"Background": 1,

"HasInvalidDepth": false,

"Depth\_range": [0.35, 12.50],

"Resolution": [1920,1080],

"Projection": "Perspective",

"Focal": [2058.73, 2058.73],

"Principle\_point":[960,540],

"BitDepthColor": 10,

"BitDepthDepth": 16,

"ColorSpace": "YUV420",

"DepthColorSpace": "YUV420"

},

…continue the loop on the views…

{

"Name":"v14",

"RowCol":"v24",

"Position": [-2.3996, 1.8235, 1.3509],

"Rotation": [-33.9126, 7.7346, -0.0000],

"Depthmap": 1,

"Background": 1,

"HasInvalidDepth": false,

"Depth\_range": [0.35, 12.50],

"Resolution": [1920,1080],

"Projection": "Perspective",

"Focal": [2058.73, 2058.73],

"Principle\_point":[960,540],

"BitDepthColor": 10,

"BitDepthDepth": 16,

"ColorSpace": "YUV420",

"DepthColorSpace": "YUV420"

}

]

}

# References

1. Exploration Experiments on Future MPEG Immersive Video, ISO/IEC JTC1/SC29/WG04 MPEG2021/ N0145, Online, October 2021.
2. Interchangeable Formats in VFX Industry (v1), ISO/IEC JTC1/SC29/WG11 MPEG2018/N17252, Macau, Oct 2018.
3. Revised text of ISO/IEC FDIS 23090-2 Omnidirectional Media Format, ISO/IEC JTC1/SC29/WG11 MPEG2018/N17399, Gwangju, January 2018.
4. Immersive Light Field Video with a layered mesh representation, Broxton et all, SIGGRAPH 2020 Technical Paper, <https://augmentedperception.github.io/deepviewvideo/>
5. <https://www.openexr.com/>
6. m55709 Proposal for an optimized input format for MPI, MPEG133