 ISO/IEC JTC 1/SC 29/WG 04 N0186

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

**Document type:** Output Document

**Title:** Call for MPEG-I Visual Test Materials

**Status:** Approved

**Date of document:** 2022-02-04

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

**Expected action:** None

**Action due date:** None

**No. of pages:** 7

**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** **0186**

**January 2022, Online**

|  |  |
| --- | --- |
| Title | Call for Test Materials for Lenslet Video Coding |
| Source | WG 04, MPEG Video Coding |
| Status | Approved |
| Serial Number | 21306 |

# 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 the “Lenslet Video Coding for Compression of Dense Light Field contents”.

The Dense Light Field activity aims to compare the coding performance of different potential representations of dense light field content (lenslet, multiview, multiview lenslet, etc), and explore new application scenarios for Lenslet Video Coding (LVC).

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 lenslet and multiview lenslet video contents. There is also an immediate need for content that is captured/synthesized. 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 for Lenslet Video Coding

Dense Light Fields for lenslet video coding can either be captured by dense multi-camera arrays or lenslet cameras (e.g. Lytro or Raytrix). Currently, two formats: multiview and lenslet, are commonly used. These data formats can be converted from one to the other. However, such a conversion may or may not be invertible, depending on whether a plenoptic 1.0 or a plenoptic 2.0 camera model is used.

Currently, there are more test sequence in dense multiview format than those using plenoptic camera. We solicit, new lenslet video content in plenoptic 1.0 and 2.0 format, with a priority for the latter.

Furthermore, MPEG has initiated exploration on new applications using dense light field videos obtained from plenoptic camera arrays for immersive video. Therefore, we solicit new test content as input to the Ad Hoc Group on Lenslet Video Coding (LVC), following the description below.

* Lenslet video captured by plenotpic cameras, specially having both camera and objects motion.
* Multiview lenslet video captured by an array of plenoptic cameras,
* Dense multiview video corresponding to each lenslet video, and/or the conversion tool to synthesize such dense multiview video,
* Multiview depth map video corresponding to the dense multiview video, and/or the tool to estimate such depth maps, and
* Camera parameters of the plenoptic cameras, and corresponding virtual cameras of the dense multiview video.

# 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.

## OpenEXR format

Another option to contribute test materials is to use the OpenEXR 2.0 format [3]. 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 (Multi-Plane images (MPI) [2]) 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~~.~~ 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:

* [Mehrdad](mailto:vinod.malamalvadakital@nokia.com) Teratani (*mehrdad.teratani (at) ulb.be*)
* [Xin](mailto:Renaud.Dore@InterDigital.com) Jin (*jin.xin (at) sz.tsinghua.edu.cn*)

# 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 consenses 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 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.

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

# Annex-B

Formate conversion tools

A set of tools are provided in [4] under Reference Plenoptic Virtual camera Convertor (RPVC) that can convert among several formats (json, exr, colmap [5]) as follows. These tools can handle scenes with several cameras’ models.

* Conversion from colmap coordinate format to json
* Conversion from json coordinate format to colmap
* Conversion from colmap depth map format to exr

# 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. Reference Plenoptic Virtual camera Convertor (RPVC): <https://gitlab.com/mpeg-dense-light-field/RPVC>
5. <https://github.com/colmap/colmap>