Text

Description automatically generatedISO/IEC JTC 1/SC 29/WG 03 N0859

**ISO/IEC JTC 1/SC 29/WG 03  
MPEG Systems   
Convenorship: KATS (Korea, Republic of)**

**Document type:** Input Contribution

**Title:** Text of ISO/IEC 23090-14 DAM 1 Support for immersive media codecs in scene description

**Date of document:** 2023-04-28

**Source:** WG03

**No. of pages:** 19 (with cover page)

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

================ First Change =======================

*In clause 2 on Normative references, add the following references:*

*ISO/IEC 23090-10,, Information technology — Coded representation of immersive media — Part 10: Carriage of visual volumetric video-based coding data*

*ISO/IEC DIS 23090-5/DAmd 1, 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 DIS 23090-12/DAmd.1, Information technology — Coded representation of immersive media — Part 12: MPEG immersive video — Amendment 1: V3C extension mechanism*

*Khronos, Vulkan 1.3.221 – A Specification (with all registered Vulkan extensions),* [*https://registry.khronos.org/vulkan/specs/1.3-extensions/html/vkspec.html*](https://registry.khronos.org/vulkan/specs/1.3-extensions/html/vkspec.html)

================ Next Change =======================

*Replace figure 1 in clause 4.2 by the following figure*

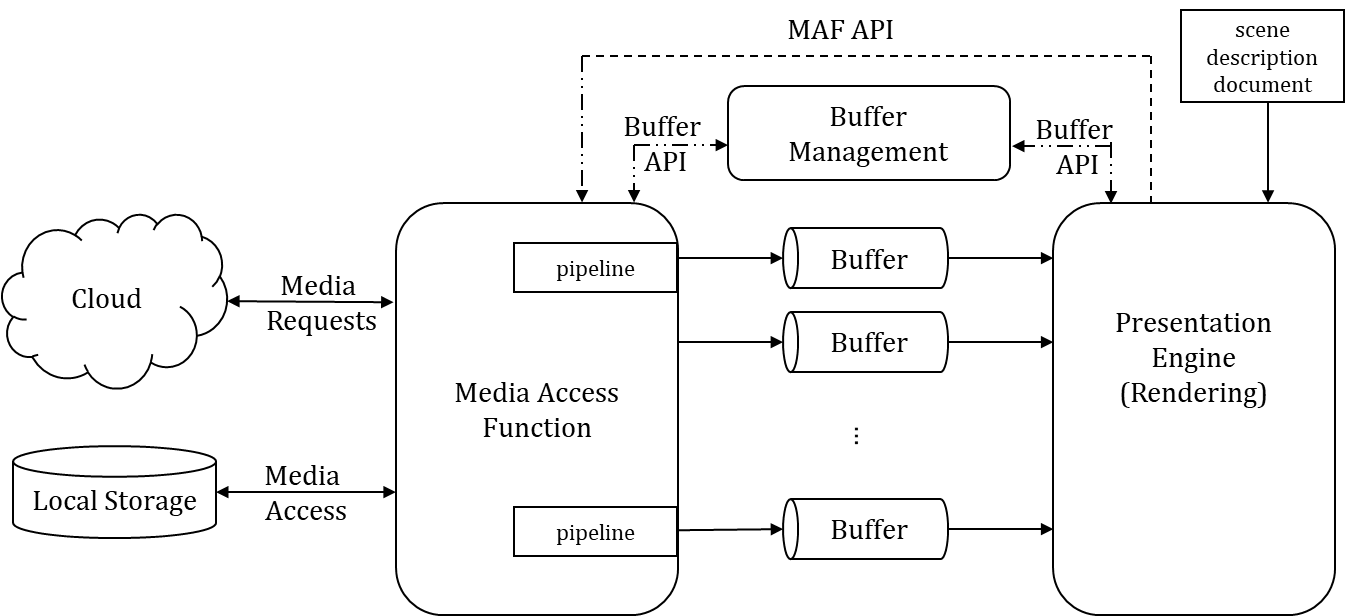
**

Figure 1 – Scene description reference architecture

================ Next Change =======================

*Add the following abbreviation to clause 3.2:*

MIV MPEG immersive video

ERP Equirectangular projection

PLR Point Local Reconstruction

EOM Enhanced Occupancy Mode

================ Next Change =======================

## *Add the following entries to Table B.1. in Annex B: Attribute registry*

Table B.1 MPEG attribute registry

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Accessor type(s)** | **Component type(s)** | **Description** | **Reference and example shader program** |
| \_MPEG\_V3C\_ATTR\_REFLECTANCE | scalar | 5123 | indicates the reflectance information that is associated with each point in a volumetric frame |  |
| \_MPEG\_V3C\_ATTR\_MATERIAL\_ID | scalar | 5123 | indicates a supplemental information that identifies material type of a point in a volumetric frame |  |
| \_MPEG\_V3C\_ATTR\_TRANSPARENCY | scalar | 5123 | indicates the transparency information that is associated with each point in a volumetric frame |  |

================ Next Change =======================

*Add the following clauses to Annex F:*

F.10 MPEG\_primitive\_V3C

In the example downloadable from https://standards.iso.org/iso-iec/23090/-14/ed-1/en/example\_MPEG\_primitive\_V3C, a usage of the MPEG\_primitive\_V3C is presented.

F.11 MPEG\_sampler\_YCbCr

In the example downloadable from https://standards.iso.org/iso-iec/23090-14/ed-1/en/example\_MPEG\_sampler\_YCbCr, a usage of the MPEG\_sampler\_YCbCr extension is presented.

================ Next Change =======================

*Add Annex G with the following content*

1. Support for MPEG-I Media

## MPEG\_primitive\_V3C extension

### General

In order to support V3C compressed objects in MPEG-I scene description, the MPEG\_media extension is used to refer to V3C compressed bitstreams.

The presentation engine may support the operations to perform the 3D reconstruction of decoded V3C components as indicated in the Figure 2. The presentation engine accesses the decoded V3C data through buffers.

The syntax of the V3C object is provided as an extension to mesh.primitive in a scene description format. The extension refers to the decoded data of a V3C object. Each decoded V3C component is signalled using properties defined in the MPEG\_primitive\_V3C extension. The extension is specific to objects coded with a V3C compression scheme (i.e., ISO/IEC 23090-5 [2] or ISO/IEC 23090-12 [3]).

Usage of the extension must be listed in the *extensionsUsed* top-level glTF property.

"*extensionsUsed*": [

"MPEG\_primitive\_V3C"

]

Figure G.1 depicts the structure of the V3C mesh compression extension:



Figure G.1 Example structure of V3C compressed primitive

If the Presentation Engine does not support the MPEG\_primtive\_V3C extension, it shall request the reconstructed raw data as described by the primitive attributes.

### Semantics

An MPEG\_primitive\_V3C extension refers to several V3C components, containing the decoded projected maps and metadata necessary such as atlas data for the 3D reconstruction process.

Table G.1 provides a list of the possible components and their description:

Table G.1 MPEG\_primitive\_V3C properties

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Default** | **Usage** | **Description** |
| \_MPEG\_V3C\_CONFIG | integer | N/A | M | This component provides a reference to a timed accessor that contains configuration information that is applicable to a sequence of frames of the V3C decoded mesh primitive. The binary format of the configuration buffer is provided in clause G.1.3. |
| \_MPEG\_V3C\_AD | object | N/A | M | this component shall reference a timed accessor that provides the V3C atlas data buffer. The atlas buffer format is defined in clause G.1.4. Future specifications of the atlas data buffer format shall use a different version.  Exactly one atlas component shall be present, irrespective of the version. |
| \_MPEG\_V3C\_GVD\_MAPS | array(integer) | N/A | M | this component shall provide an array of video texture references, each of which corresponds to one map of the decoded geometry video data. |
| \_MPEG\_V3C\_OVD\_MAP | integer | N/A | O | this component shall provide a video texture reference, which corresponds to the decoded occupancy video data map. |
| \_MPEG\_V3C\_AVD | array(object) | N/A | O | this component shall provide an array of objects, each of which describing an attribute component of the V3C compressed mesh primitive. The properties of the components are described in Table G.2. |
| \_MPEG\_V3C\_CAD | object | N/A | CM | This object lists different properties described for the Common Atlas Data in ISO/IEC 23090-5. |
| Legend:  For attributes: M=mandatory, O=optional, OD=optional with default value, CM=conditionally mandatory. | | | | |

The \_MPEG\_V3C\_AD object shall have the structure as describe in Table G.2:

Table G.2 Properties of \_MPEG\_V3C\_AD object

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Default** | **Usage** | **Description** |
| bufferFormat | string | "baseline” | O | provides an identifier of the associated atlas data buffer format. A list of supported atlas data buffer formats is provided in Table G.4. |
| accessor | integer | N/A | M | This provides the index of the timed accessor that provides access to the atlas data buffer. |

The \_MPEG\_V3C\_AVD object shall have the following structure:

Table G.3 Properties of \_MPEG\_V3C\_AVD object

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Default** | **Usage** | **Description** |
| type | uint8 | 0 | O | provides the type of the attribute as defined by the “V3C attribute types” in ISO/IEC 23090-5. |
| maps | array(integer) | N/A | M | This array shall provide a list of video texture references, each of which corresponds to one map of the decoded attribute video data. |

Each mesh primitive shall reference exactly one atlas data buffer.

Different buffer formats with unique string identifier for the atlas data are defined in Table G.4.

Table G.4 List of atlas data buffer formats

|  |  |
| --- | --- |
| **buffer\_format** | **Description** |
| baseline | The configuration data is defined in Table G.5 and the corresponding atlas data buffer format is defined in Table G.6. |
| extended | Atlas data with common atlas parameters and and PROJECTED patch type application-specific data with PLR information, EOM patch type application-specific data, and RAW patch type application-specific data. The configuration data format is defined in G.5 and the corresponding atlas data buffer format is defined in Table G.7. |
| miv | Atlas data with common atlas parameters and PROJECTED patch type application-specific parameters for MIV. The configuration data format is defined in G.5 and the corresponding atlas data buffer format is defined in Table G.8. |

### Configuration Data Buffer Format

The configuration data buffer is binary formatted data that provides static configuration data that is applicable for the V3C compressed primitive. The data shall comply to the following format:

Table G.5 Configuration data buffer format

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| **frame\_width** | uint16 | indicates the frame width in luma samples of the atlas and all other associated V3C components. |
| **frame\_height** | uint16 | indicates the frame height in luma samples of the atlas and all other associated V3C components. |
| **map\_count** | uint8 | indicates the number of maps used for encoding the geometry and attribute data for the current atlas. |
| **patch\_packing\_block\_size** | uint8 | specifies the value of the variable PatchPackingBlockSize in ISO/IEC 23090-5, that is used for the horizontal and vertical placement of the patches within the current atlas. |

### Atlas Data Buffer Format

The atlas data buffer is binary formatted data that shall comply to the following formats in Table G.6, Table G.7 and Table G.8 depending on the buffer format for the atlas data. The atlas buffer format for “buffer\_format” value “baseline” is described in Table G.7.

Table G.6 Atlas data buffer format for buffer\_format:baseline

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| **patch\_count** | uint16 | provides the total number of patches. |
| for( i=0;i<patch\_count;i++ ) { |  |  |
| **2d\_pos\_x** | float | specifies the x-coordinate of the top-left corner of the patch bounding box for the current patch. |
| **2d\_pos\_y** | float | specifies the y-coordinate of the top-left corner of the patch bounding box for the current patch. |
| **2d\_size\_x** | float | specifies the width of the current patch. |
| **2d\_size\_y** | float | specifies the height of the current patch. |
| **3d\_offset\_u** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the tangent axis. |
| **3d\_offset\_v** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the bi-tangent axis. |
| **3d\_offset\_d** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the normal axis. |
| **patch\_projection\_id** | uint8 | specifies the identifier of the projection mode and the index of the normal to the projection plane of the current patch. |
| **patch\_orientation** | uint8 | specifies the index of the patch orientation of the current patch. |
| **lod\_scale\_x** | uint16 | specifies the LOD scaling factor to be applied to the tangent axis of the current patch. |
| **lod\_scale\_y** | uint16 | specifies the LOD scaling factor to be applied to the bi-tangent axis of the current patch. |
| } |  |  |

The atlas buffer format for “buffer\_format” value “extended” is described in Table G.7.

Table G.7 Atlas data buffer format for buffer\_format:extended

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| **patch\_count** | uint16 | provides the total number of patches. |
| **for ( i=0;i<patch\_count;i++ ) {** |  |  |
| **patch\_type** | uint8 | specifies the type of patch |
| **2d\_pos\_x** | float | specifies the x-coordinate of the top-left corner of the patch bounding box for the current patch. |
| **2d\_pos\_y** | float | specifies the y-coordinate of the top-left corner of the patch bounding box for the current patch. |
| **2d\_size\_x** | float | specifies the width of the current patch. |
| **2d\_size\_y** | float | specifies the height of the current patch. |
| **3d\_offset\_u** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the tangent axis. |
| **3d\_offset\_v** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the bi-tangent axis. |
| **3d\_offset\_d** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the normal axis. |
| **patch\_projection\_id** | uint8 | specifies the identifier of the projection mode and the index of the normal to the projection plane of the current patch. |
| **patch\_orientation** | uint8 | specifies the index of the patch orientation of the current patch. |
| **lod\_scale\_x** | uint16 | specifies the LOD scaling factor to be applied to the tangent axis of the current patch. |
| **lod\_scale\_y** | uint16 | specifies the LOD scaling factor to be applied to the bi-tangent axis of the current patch. |
| **if (patch\_type == PROJECTED) {** |  |  |
| **plri\_map\_present** | bool | specifies if the plr information is present |
| **if (plri\_map\_present) {** |  |  |
| **plrd\_level** | bool | specifies the level of PLR data for a patch |
| **if (plr\_level == 0 ) {** |  |  |
| **for ( b = 0; b < blockcount < b++) {** |  |  |
| **plrd\_present\_block\_flag** | bool | specifies whether the PLR data is present for a block |
| **if (plrd\_present\_block\_flag == 1) {** |  |  |
| **plrd\_block\_mode** | uint8 | specifies the mode of PLR data for a block |
| **}** |  |  |
| **}** |  |  |
| **} else {** |  |  |
| **plrd\_present\_patch\_flag** | bool | specifies whether the PLR data is present for the patch |
| **if (plrd\_present\_patch\_flag) {** |  |  |
| **plrd\_patch\_mode** | uint8 | specifies the mode of the PLR data for the patch |
| **}** |  |  |
| **}** |  |  |
| **}** |  |  |
| **else if (patch\_type == EOM)** |  |  |
| **eom\_patch\_count** | uint8 | specifies the number of patches that may be associated with the current patch |
| **eom\_points** | uint8 | specifies the number of EOM coded points in the patch associated with the current patch |
| **associated\_patch\_index** | uint8 | specifies the index of the i-th patch associated with the current patch |
| **}** |  |  |
| **else if (patch\_type == RAW)** |  |  |
| **raw\_points** | uint8 | specifies the number of RAW coded points in the current patch |
| **}** |  |  |
| **}** |  |  |

Note: The calculation of blockCount is specified in function BlockCnt (xSize, ySize) in clause 8.4.7.9. of ISO/IEC 23090-5. The arguments to the function are patch dimensions, i.e., 2d\_size\_x, and 2d\_size\_y.

The atlas buffer format for “buffer\_format” value “miv” is described in Table G.8.

Table G.8 Atlas data buffer format for buffer\_format:miv

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| **patch\_count** | uint16 | provides the total number of patches. |
| **for( i=0;i<patch\_count;i++ ) {** |  |  |
| **2d\_pos\_x** | float | specifies the x-coordinate of the top-left corner of the patch bounding box for the current patch. |
| **2d\_pos\_y** | float | specifies the y-coordinate of the top-left corner of the patch bounding box for the current patch. |
| **2d\_size\_x** | float | specifies the width of the current patch. |
| **2d\_size\_y** | float | specifies the height of the current patch. |
| **3d\_offset\_u** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the tangent axis. |
| **3d\_offset\_v** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the bi-tangent axis. |
| **3d\_offset\_d** | float | specifies the shift to be applied to the reconstructed patch points in the current patch along the normal axis. |
| **patch\_projection\_id** | uint8 | specifies the identifier of the projection mode and the index of the normal to the projection plane of the current patch. |
| **patch\_orientation** | uint8 | specifies the index of the patch orientation of the current patch. |
| **lod\_scale\_x** | uint16 | specifies the LOD scaling factor to be applied to the tangent axis of the current patch. |
| **lod\_scale\_y** | uint16 | specifies the LOD scaling factor to be applied to the bi-tangent axis of the current patch. |
| **patch\_view\_index** | uint8 | specifies the index in the buffer format for the view parameter |
| **patch\_entity\_id** | uint8 | specifies the patch entity ID for the current patch |
| **patch\_depth\_occ\_threshold** | uint8 | specifies the threshold below with the occupancy value is defined to be unoccupied for the current patch |
| **tile\_patch\_texture\_offset\_1** | uint8 | specifies the offset applied to the first component sample values of the attribute for the current patch |
| **tile\_patch\_texture\_offset\_2** | uint8 | specifies the offset applied to the second component sample values of the attribute for the current patch |
| **tile\_patch\_texture\_offset\_3** | uint8 | specifies the offset applied to the third component sample values of the attribute for the current patch |
| **}** |  |  |

### Common atlas data

#### G.1.5.1 Overview

The common atlas data is common to all atlases and shall correspond to the Common Atlas Data in ISO/IEC 23090-5.

#### G.1.5.2 MIV extension to CAD

Some of the common atlas data information which is common for the atlases in a V3C bitstream is specified in ISO/IEC 23090-12 such as view parameters. The syntax for the MIV extension to common atlas data is specified ISO/IEC 23090-12. It includes a list of view parameters which can be used during the rendering process (Annex H.1 in ISO/IEC 23090-12). An MIV\_view\_parameters property is defined for the \_MPEG\_V3C\_CAD object as shown in Table G.9.

The \_MPEG\_V3C\_CAD object can be extended to describe additional properties that may be introduced in future iterations of ISO/IEC 23090-5 or in extensions to that specification.

Table G.9 Definition of properties defined in \_MPEG\_V3C\_CAD in MPEG\_V3C extension

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Default** | **Usage** | **Description** |
| mivViewParameters | integer | N/A | O | This component provides a reference to a timed-accessor that contains the view parameters stored in the common atlas data that is applicable to a sequence of frames of the V3C decoded mesh primitive. The buffer format for the view parameters is described in Table G.10. |
| Legend:  For attributes: M=mandatory, O=optional, OD=optional with default value, CM=conditionally mandatory. | | | | |

#### G.1.5.3. Buffer format for MIV view parameters

Table G.10 describes the binary buffer format for view parameters.

Table G.10 Buffer format for view parameters

|  |  |  |
| --- | --- | --- |
| **Field** | **Type** | **Description** |
| **num\_views** | uint16 | number of views |
| **for ( int p = 0; p < num\_views ; p++) {** |  |  |
| **view\_id\_to\_index** | uint8 | mapping of the id associated with each view |
| **view\_in\_paint\_flag** | bool | specifies if the view is an inpaint view |
| **view\_pos\_x** | uint8 | specifies in scene units the x-coordinate of the location of the view with view index equal to v. |
| **view\_pos\_y** | uint8 | specifies in scene units the y-coordinate of the location of the view with view index equal to v. |
| **view\_pos\_z** | uint8 | specifies in scene units the z-coordinate of the location of the view with view index equal to v. |
| **view\_quat\_x** | uint8 | specifies the x components for the rotation of the view with view index equal to v using the quaternion representation |
| **view\_quat\_y** | uint8 | specifies the y components for the rotation of the view with view index equal to v using the quaternion representation |
| **view\_quat\_z** | uint8 | specifies the z components for the rotation of the view with view index equal to v using the quaternion representation |
| **view\_quat\_w** | uint8 | specifies the w components for the rotation of the view with view index equal to v using the quaternion representation |
| **view\_type** | uint8 | specifies the projection method of the view |
| **projection\_plane\_width** | uint8 | specifies the horizontal resolution of projection plane |
| **projection\_plane\_height** | uint8 | specifies the vertical resolution of the projection plane |
| **if (view\_type == 0) {** |  | equirectangular projection |
| **erp\_phi\_min** | float32 | specifies the minimum longitude range for an ERP projection in units of degrees |
| **erp\_phi\_max** | float32 | specifies the maximum longitude range for an ERP projection in units of degrees |
| **erp\_theta\_min** | float32 | specifies the minimum latitude range for an ERP projection in units of degrees |
| **erp\_theta\_max** | float32 | specifies the maximum latitude range for an ERP projection in units of degrees |
| **} else if (view\_type == 1) {** |  | perspective projection |
| **perspective\_focal\_hor** | float32 | specifies in luma samples position units the horizontal components of the focal of a perspective projection of the view with view index |
| **perspective\_focal\_ver** | float32 | specifies in luma samples position units the vertical components of the focal of a perspective projection of the view with view index |
| **perspective\_principal\_point\_hor** | float32 | specifies in luma sample positions the horizontal coordinates of the principal point of a perspective projection of the view |
| **perspective\_principal\_point\_ver** | float32 | specifies in luma sample positions the vertical coordinates of the principal point of a perspective projection of the view |
| **} else if ( view\_type -== 2) {** |  | orthographic projection |
| **ortho\_height** | float32 | specifies in scene units the vertical dimensions of the captured part of the volumetric frame |
| **ortho\_width** | float32 | specifies in scene units the horizontal dimensions of the captured part of the volumetric frame |
| **}** |  |  |
| **quantization\_law** | uint8 | specifies the type of depth quantization method of the view |
| **if ( quantization\_law == 0) {** |  |  |
| **norm\_dis\_low** | uint8 | specifies the normalized disparity of the lowest signalled geometry value |
| **norm\_dis\_high** | uint8 | specifies the normalized disparity of the highest signalled geometry value |
| **}** |  |  |
| **depth\_occ\_threshold** | uint8 | specifies the default occupancy threshold used in the occupancy value extraction process |
| **pp\_root** | bool | specifies whether the view has a parent in the pruning graph at the encoder stage |
| **if (!pp\_root) {** |  |  |
| **pp\_num\_parents** | uint8 | specifies the number of parents of the view in the pruning graph at the encoder stage |
| **for ( int i = 0; i < pp\_num\_parents; i++) {** |  |  |
| **pp\_view\_parent\_idx** | uint8 | specifies the index of the i-th parent view in the pruning graph at the encoder stage. |
| **}** |  |  |
| **}** |  |  |
| **}** |  |  |

### Processing Model

#### G.1.6.1 General

The Presentation Engine is equipped with a graphics processing unit (GPU). The loader in the Presentation Engine will parse the MPEG-I scene description file. If the loader supports the reconstruction of V3C objects and accepts the MPEG\_primitive\_V3C extension, then the loader in the presentation engine will process the MPEG\_primitive\_V3C extension for a mesh element that contains the extension. The presentation engine will then request the MAF to supply the decoded V3C data indicated by the extension in the associated buffers. The decoded V3C data provided by the properties specified by the MPEG\_primitive\_V3C extension are then loaded to the GPU memory.

An implementation (e.g., a shader implementation) is run on the decoded V3C data to generate the final 3D reconstructed object. The logic of 3D reconstruction is facilitated by using the V3C information such as atlas, geometry, and occupancy. The 3D object can be further textured using the texture information with different V3C attributes.

Since the MPEG\_primitive\_V3C extension is expressed at the mesh-level, a node referencing a mesh with the MPEG\_primitive\_V3C extension will position the object in the scene graph for rendering.

The transformation parameters in the V3C bitstream as defined in H.8.3.6.3.3 shall be ignored.

#### G.1.6.2 MIV support

MIV is a special case of the V3C representation in MPEG-I scene description as meshes. The MPEG-I scene description author will supply the viewing space boundaries in the accessor referred to by the “POSITION” attribute of a mesh. The viewing space boundaries are conveyed by the MIV bitstream. The texture of the MIV content is directly passed to the renderer based on different camera views in the viewing space. The information for the camera views is stored in the atlas property of the MPEG\_primitive\_V3C extension.

## MPEG\_sampler\_YCbCr extension

### General

A sampler-level extension is described to sample a video texture natively in parallel processing devices such as GPUs. This extension shall be present if the format of the referencing video texture is set to YCbCr.

A texture object in the textures array may use a sampler with the “MPEG\_sampler\_YCbCr” sampler extension to provide information to the Presentation Engine to sample the video texture when the texture format is a chroma format such as YCbCr.

### Semantics

Table G.11 provides a description of the properties defined in the MPEG\_sampler\_YCbCr sampler extension.

Table G.11 MPEG\_sampler\_YCbCr semantic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Default** | **Usage** | **Description** |
| ycbcrModel | integer | 1 | O | Describes the color matrix for conversion between color models. The supported formats are defined by the VkSamplerYcbcrModelConversion enumeration in clause 13.1 of Vulkan 1.3. |
| ycbcrRange | integer | 0 | O | Describes whether the encoded values have headroom and foot room, or whether the encoding uses the full numerical range. |
| chromaFilter | integer | 0 | O | Describes the filter for chroma reconstruction. |
| components | array(integer) | [0,0,0,0] | O | Applies a swizzle to the [r,g,b,a] components based on [VkComponentSwizzle](https://registry.khronos.org/vulkan/specs/1.3-extensions/html/vkspec.html" \l "VkComponentSwizzle) enums prior to range expansion and color model conversion. If present, the array shall include 4 values, each of which corresponding to the r,g,b,a components in order of appearance. |
| xChromaOffset | integer | 0 | O | Describes the [sample location](https://registry.khronos.org/vulkan/specs/1.3-extensions/html/vkspec.html#textures-chroma-reconstruction) associated with downsampled chroma components in the x dimension. xChromaOffset has no effect for formats in which chroma components are not downsampled horizontally. |
| yChromaOffset | integer | 0 | O | Describes the [sample location](https://registry.khronos.org/vulkan/specs/1.3-extensions/html/vkspec.html#textures-chroma-reconstruction) associated with downsampled chroma components in the y dimension. yChromaOffset has no effect for formats in which the chroma components are not downsampled vertically. |

### Processing Model

The MPEG\_sampler\_YCbCr extension provides relevant configuration information for the native YCbCr device extensions or shader compiler to read and sample a YCbCr texture.

================ Next Change =======================

*In section 5.3.1.2, Table 11, change the Description of the format attribute as follows:*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| format | string | RGB | O | Indicates the format of the pixel data for this video texture. The allowed values are: RED, GREEN, BLUE, RG, RGB, RGBA, BGR, BGRA, DEPTH\_COMPONENT. The semantics of these values are defined in Table 8.3 of OpenGL specification [2].  Additionally, YCbCr formats are supported. The semantics for the YCbCr formats are defined in Table 76 in Vulkan specification [Vulkan 1.3]. A sampler with the MPEG\_sampler\_YCbCr extension shall be linked to a YCbCr texture.  Note that the number of components shall match the type indicated by the referenced accessor. Normalization of the pixel data shall be indicated by the normalized attribute of the accessor. |

*In section 5.2.1.2, Table 6, change the Description of the track attribute as follows:*

Table 6 – Definitions of items in the tracks array of MPEG\_media.alternative extension

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Type** | **Default** | **Usage** | **Description** |
| track | string | N/A | M | URL fragment to access the track within the media alternative.  The URL structure is defined for the following formats:  DASH: Using MPD Anchors (URL fragments) as defined in ISO/IEC 23009-1:2019:Annex C (Table C.1).  ISOBMFF: URL fragments as specified in ISO/IEC 14496-12:2020:Annex C.  SDP: stream identifier of the media stream as defined in Annex C.  When V3C data is referenced in the scene description document as in item in MPEG\_media.alternative.tracks and the referenced item corresponds to an ISBOBMFF track, the following applies:   * For single-track encapsulated V3C data, the referenced track in MPEG\_media shall be the V3C bitstream track. * For multi-track encapsulated V3C data, the referenced track in MPEG\_media shall be the V3C atlas track.   When G-PCC data is referenced by the scene description file as an item in MPEG\_media.alternative.tracks and the referenced item complies with the provisions of track in ISOBMFF, the following applies:   * For single-track encapsulated G-PCC data, the track referenced in MPEG\_media shall be the G-PCC bitstream track； * For multi-track encapsulated G-PCC data, the track referenced in MPEG\_media shall be the G-PCC geometry bitstream track. |
| codecs | string | N/A | M | The codecs parameter, as defined in IETF RFC 6381, of the media included in the track.  When the track includes different types of codecs (e.g. the AdaptationSet includes Representations with different codecs), the codecs parameter may be signaled by comma-separated list of values of the codecs. |