ISO/IEC JTC 1/SC 29/WG 03 N1429

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

**Document type:** Output Document

**Title:** Technologies under Consideration for ISO/IEC 23001-10

**Status:** Approved

**Date of document:** 2025-02-20

**Source:** ISO/IEC JTC 1/SC 29/WG 03

**No. of pages:** 1 (with cover page)

**Email of Convenor:** young.L @ samsung . com

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

**INTERNATIONAL ORGANIZATION FOR STANDARDIZATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 03 MPEG SYSTEMS**

**ISO/IEC JTC 1/SC 29/WG 03 N1429**

**January 2025, Geneva CH**

|  |  |
| --- | --- |
| **Title** | **Technologies under Consideration for ISO/IEC 23001-10** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **24729** |

**Abstract**

The document contains following technologies under consideration for the Carriage of Timed Metadata Metrics in ISO Base Media File Format (ISO/IEC 23001-10):

[1. CAMBI 1](#_Toc184915423)

[2. Timed Metadata for Spatial Relationships of Immersive Media 3](#_Toc184915424)

*Clause 4.3*

*Add the following subclauses at the end of the clause.*

# CAMBI

#### Definition

CAMBI is a banding detector. In CAMBI’s full-reference mode, it takes a (distorted) video and a reference video as inputs and produces a banding score as the output. The algorithm extracts pixel-level maps at multiple scales for frames of the encoded video. Subsequently, it combines these maps into a single index motivated by the human contrast sensitivity function (CSF). A value of 0 indicates no banding is present. The maximum CAMBI value with default parameters is 31. Typically, banding starts to be noticeable around a CAMBI of 5, it is annoying around a CAMBI of 10, and 15 or above corresponds to very annoying banding artifacts. The full-reference mode is computed by combining the no-reference CAMBI of its two inputs as *max(0, cambi\_no\_ref\_distorted - cambi\_no\_ref\_source)*. The calculation of the no-reference CAMBI metric is performed as follows.

CAMBI is applied to 8 or 10-bit video. It consists of three main steps: A) pre-processing, B) Multiscale Banding Confidence, and C) spatio-temporal pooling.

## Pre-processing

Each input frame goes through several pre-processing steps. First, the luma component is extracted. Then, if needed, the frame is converted to 10-bit via left-shift. Finally, if the input bit depth is less than 10-bit, a 2 × 2 averaging low-pass filter with ¼ weights is applied.

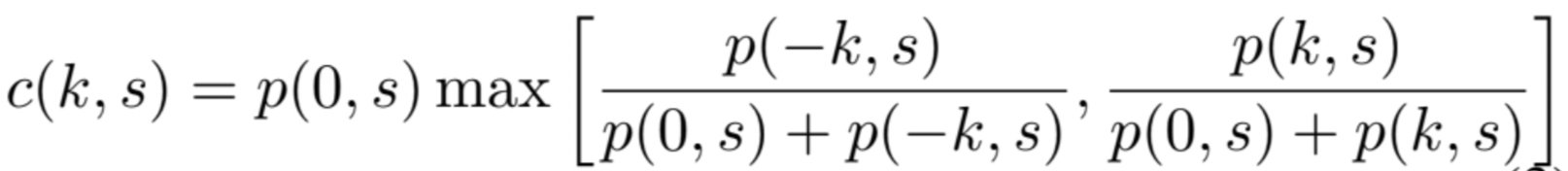
## Multiscale Banding Confidence

A spatial mask for the input is obtained at the input resolution. A (luma) sample has zero derivative if it is equal to its right and bottom neighbors. The mask includes all the samples for which the number of zero derivatives in a *fs* x *fs* square around the sample is larger than a *mask\_index* that depends on the resolution:

mask\_index = (*fs\*fs* + 3 \* (log2( (input\_width >> 6) \* (input\_height >> 6)) - 11) - 1)>>1

where *Fs* = 7.

CAMBI uses 5 scales of the spatial mask and input from the pre-processing, each obtained by the decimation by 2 in each direction. A sample-wise banding confidence *c(k, s)* at each scale *s* ∈ {1, 2, 3, 4, 5} and contrast parameter k ∈ {1, 2, 3, 4} is computed by

, (1)

where *p(k, s)* is given by

A black text with a white background

Description automatically generated with medium confidence (2)

In the equation 2, (x, y) refers to a particular sample location, and *I*(*x, y*), *Ns*(*x, y*) and ∇(*x, y*) correspond to the intensity, neighborhood of a scale *s*, and the zero derivative value from the spatial mask at this particular sample, respectively. δ(., .) is an indicator for the samples belonging to the mask. Thus, *p*(*k, s*) corresponds to the fraction of samples, in a neighborhood around (*x, y*), with an intensity difference of *k* amongst the set of samples with a gradient magnitude smaller than the *mask\_index* τg.

With these default parameters, a total of twenty CAMBI maps are obtained per-frame capturing banding across 4 contrast-steps and 5 spatial-frequencies.

The metric implementation is available in [6].

## Spatio-temporal pooling

The CAMBI maps are spatio-temporally pooled to obtain the score. Spatial pooling of CAMBI maps is done weighting by contrast and including only the worst *κp* (default *p* = 60%) of the samples to get frame score:

A math equations and numbers

Description automatically generated with medium confidence

where 1/*v*◦ represents spatial-frequency at which banding is detected. The final CAMBI score for a sequence is obtained by the arithmetic average of the per-frame scores.

#### Metric code name

CAMBI quality metric values shall be provided under the ‘cmbi’ metric code name.

#### Sample storage format

Each CAMBI metric value shall be stored as an unsigned 16-bit integer value.

#### Decoding operation

Given stored 16-bit integer value x, the corresponding CAMBI value shall be derived as follows (expressed in floating point):

CAMBI = (real) (x) / 2048;

# Timed Metadata for Spatial Relationships of Immersive Media

This section defines metadata data structures for 2D and 3D spatial sources and regions, and their carriage in timed metadata tracks for signaling spatial relationship of immersive media. The metadata data structures are defined in the format that can be adopted into MPEG-I Part 7 [2]. In particular, the following MPEG-I reference coordinate system [2] is used as the 3D Cartesian coordinate system with 6DoF:



* 1. **2D and 3D Spatial Source and Region Metadata Data Structures**

In order to specify spatial relationships of 2D/3D regions within their respective 2D and 3D sources, some common metadata data structures are defined in this section.

* + 1. **Syntax** 
       1. ***2D and 3D Elements***

aligned(8) class 3DPositionStruct() {  
 signed int(32) centre\_x;  
 signed int(32) centre\_y;  
 singed int(32) centre\_z;  
}

aligned(8) class 3DLocationStruct() {  
 signed int(32) near\_top\_left\_x;  
 signed int(32) near\_top\_left\_y;  
 singed int(32) near\_top\_left\_z;  
}

aligned(8) class 3DRotationStruct() {  
 signed int(32) rotation\_yaw;  
 signed int(32) rotation\_pitch;  
 signed int(32) rotation\_roll;  
}

aligned(8) class 2DPositionStruct() {  
 signed int(32) centre\_x;  
 signed int(32) centre\_y;  
}

aligned(8) class 2DLocationStruct() {  
 signed int(32) top\_left\_x;  
 signed int(32) top\_left\_y;  
}

aligned(8) class 2DRotationStruct() {  
 signed int(32) rotation\_angle;  
}

aligned(8) class 3DOrietationStruct() {  
 signed int(32) centre\_azimuth;  
 signed int(32) centre\_elevation;  
 singed int(32) centre\_tilt;  
}

aligned(8) class 2DRangeStruct(shape\_type) {  
 if (shape\_type == 0) { // 2D rectangle   
 unsigned int(32) range\_width;  
 unsigned int(32) range\_height;  
 }  
 if (shape\_type == 1) { // 2D circle  
 unsigned int(32) range\_radius;  
 }  
 // other values of shape\_type are reserved  
}  
aligned(8) class 3DRangeStruct(shape\_type) {  
 2DRangeStruct(shape\_type); // including 2D shape types  
 if (shape\_type == 2) { // 3D tile  
 unsigned int(32) range\_width;  
 unsigned int(32) range\_height;  
 unsinged int(32) range\_depth;  
 }  
 if (shape\_type == 3) { // 3D spherical region  
 unsigned int(32) range\_width;  
 unsigned int(32) range\_height;  
 unsinged int(32) range\_depth;  
 }  
 if (shape\_type == 4) { // 3D sphere  
 unsigned int(32) range\_radius;  
 }  
 // other values of shape\_type are reserved  
}

* + - 1. ***2D and 3D Sources***

aligned(8) class SpatialRelationship2DSourceStruct(  
 location\_included\_flag,   
 rotation\_included\_flag,  
 range\_included\_flag,  
 shape\_type)   
{  
 if (location\_included\_flag)   
 2DLocationStruct();  
 if (rotation\_included\_flag)   
 2DRotationStruct();   
 if (range\_included\_flag)  
 2DRangeStruct(shape\_type);  
 unsigned int(32) source\_id;  
}

aligned(8) class SpatialRelationship3DSourceStruct(  
 location\_included\_flag,   
 rotation\_included\_flag,  
 range\_included\_flag,  
 shape\_type)  
{  
 if (location\_included\_flag)   
 3DLocationStruct();  
 if (rotation\_included\_flag)   
 3DRotationStruct();   
 if (range\_included\_flag)  
 3DRangeStruct(shape\_type);  
 unsigned int(32) source\_id;  
}

* + - 1. ***Regions with 2DoF and 6DoFs***

aligned(8) RegionWith2DoFStruct(  
 location\_included\_flag,   
 rotation\_included\_flag,  
 range\_included\_flag,   
 shape\_type,  
 interpolate\_included\_flag)   
{  
 if (location\_included\_flag)   
 2DLocationStruct();  
 if (orientation\_included\_flag)   
 2DRotationStruct();   
 if (range\_included\_flag)  
 2DRangeStruct(shape\_type);   
 if (interpolate\_included\_flag) {  
 unsigned int(1) interpolate;  
 bit(7) reserved = 0;  
 }  
}

aligned(8) RegionWith6DoFStruct(  
 location\_included\_flag,   
 orientation\_included\_flag,  
 range\_included\_flag,   
 shape\_type,  
 interpolate\_included\_flag)   
{  
 if (location\_included\_flag)   
 3DLocationStruct();  
 if (orientation\_included\_flag)   
 3DRotationStruct();   
 if (range\_included\_flag)  
 3DRangeStruct(shape\_type);   
 if (interpolate\_included\_flag) {  
 unsigned int(1) interpolate;  
 bit(7) reserved = 0;  
 }  
}

* + - 1. ***Viewports with 3DoF and 6DoFs***

aligned(8) ViewportWith3DoFStruct(  
 orientation\_included\_flag,  
 range\_included\_flag,   
 shape\_type,  
 interpolate\_included\_flag)   
{  
 if (orientation\_included\_flag)   
 3DOrientationStruct();   
 if (range\_included\_flag)  
 2DRangeStruct(shape\_type);   
 if (interpolate\_included\_flag) {  
 unsigned int(1) interpolate;  
 bit(7) reserved = 0;  
 }  
}

aligned(8) ViewportWith6DoFStruct(  
 position\_included\_flag,   
 orientation\_included\_flag,  
 range\_included\_flag,   
 shape\_type,  
 interpolate\_included\_flag)   
{  
 if (position\_included\_flag)   
 3DPositionStruct();  
 if (orientation\_included\_flag)   
 3DOrientationStruct();   
 if (range\_included\_flag)  
 2DRangeStruct(shape\_type);   
 if (interpolate\_included\_flag) {  
 unsigned int(1) interpolate;  
 bit(7) reserved = 0;  
 }  
}

* + 1. **Semantics**

centre\_x, centre\_y and centre\_z specify the x, y and z axis values, respectively, of the centre of the sphere region, with respect to the origin of the underlying coordinate system.

near\_top\_left\_x, near\_top\_left\_y, and near\_top\_left\_z specify the x, y, and z axis values, respectively, of the near-top-left corner of the 3D rectagular region, with respect to the origin of the underlying 3D coordinate system.

rotation\_yaw, rotation\_pitch, and rotation\_roll specify the yaw, pitch, and roll angles, respectively, of the rotation that is applied to the unit sphere of each spherical region associated in the spatial relationship to convert the local coordinate axes of the spherical region to the global coordinate axes, in units of 2−16 degrees, relative to the global coordinate axes. rotation\_yaw shall be in the range of −180 \* 216 to 180 \*216 − 1, inclusive. rotation\_pitch shall be in the range of −90 \* 216 to 90 \* 216, inclusive. rotation\_roll shall be in the range of −180 \* 216 to 180 \* 216 − 1, inclusive.

top\_left\_x, and top\_left\_y specify the x, and y axis values, respectively, of the top-left corner of the rectagular region, with respect to the origin of the underlying coordinate system.

rotation\_angle specifies the angle of the counter-clock rotation that is applied to each of the 2D regions associated in the spatial relationship to convert the local coordinate axes of the 2D region to the global coordinate axes, in units of 2−16 degrees, relative to the global coordinate axes. rotation\_angle shall be in the range of −180 \* 216 to 180 \*216 − 1, inclusive.

centre\_azimuth and centre\_elevation specify the azimuth and elevation values, respectively, of the centre of the sphere region in units of 2−16 degrees. centre\_azimuth shall be in the range of −180 \* 216 to 180 \* 216 – 1, inclusive. centre\_elevation shall be in the range of −90 \* 216 to 90 \* 216, inclusive.

centre\_tilt specifies the tilt angle of the sphere region in units of 2−16 degrees. centre\_tilt shall be in the range of −180 \* 216 to 180 \* 216 – 1, inclusive.

range\_width and range\_height specify the width and height ranges, respectively, of a 2D or 3D rectangular region. They specify the ranges through a reference point of the rectangular region, which could be either the top left point or centre point, inferred as specified in the semantics of the structure containing the instances of these metadata.

range\_depth specifies the depth range of a 3D rectangular region. It specifies the ranges through the centre point of the region.

range\_radius specifies the radius range of a circluar region.

range\_azimuth and range\_elevation specify the azimuth and elevation ranges, respectively, of the sphere region in units of 2−16 degrees. range\_azimuth and range\_elevation specify the ranges through the centre point of the sphere region. range\_azimuth shall be in the range of 0 to 360 \* 216, inclusive. range\_elevation shall be in the range of 0 to 180 \* 216, inclusive.

shape\_type specifies a shape type of a 2D or 3D region, according to the following table:

|  |  |
| --- | --- |
| **Value** | **Description** |
| 0 | 2D rectangle |
| 1 | 2D circle |
| 2 | 3D tile |
| 3 | 3D sphere region |
| 4 | 3D sphere |
| others | Reserved |

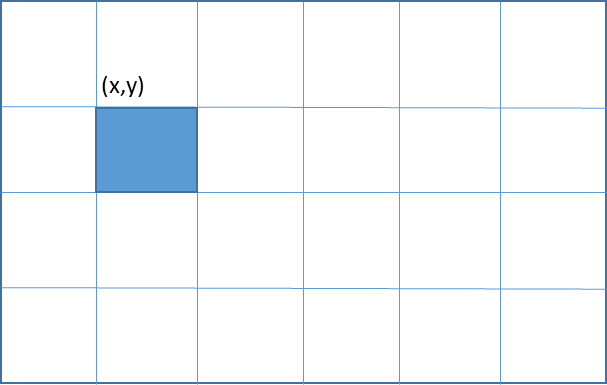
The semantics of interpolate are specified by the semantics of the structure containing this instance of it.

When any of the location, rotatioin, orientation, range, shape and interoperate metadata are not present in an instance of 2D and 3D source and region data structures, they are inferred as specified in the semantics of the structure containing the instance.

* 1. **Signalling of Spatial Relationship of Spatial Regions in Timed Metadata Tracks**

This section shows signalling of the following spatial relationships within timed metadata tracks using the 2D and 3D spatial source and region metadata data structures defined above, when individual tracks carry visual content of spatial regions:

1. 2D Planar Regions with 2DoF (for Sub-picture tracks)
2. 3D Spherical Regions with 6DoF
3. 3D Planar Regions with 6DoF
4. 3D Tile Regions with 6DoF (for PCC 3D Tile Tracks)
   * 1. **2D Planar Regions with 2DoF (for Sub-pictures in 2D space)**



* + - 1. ***Sample Entry***

Sample Entry Type: '2dpr'  
Container: Sample Description Box (‘stsd’)  
Mandatory: No  
Quantity: 0 or 1

aligned(8) class SpatialRelationship2DPlanarRegionsSampleEntry  
 extends MetadataSampleEntry (‘2dpr’) {  
 bit(5) reserved = 0;  
 unsigned int(1) source\_location\_included\_flag;   
 unsigned int(1) source\_rotation\_included\_flag;   
 unsigned int(1) source\_range\_included\_flag;   
 unsigned int(8) source\_shape\_type = 0; // for 2D planar region  
 SpatialRelationship2DSourceStruct(source\_location\_included\_flag,   
 source\_rotation\_included\_flag,   
 source\_range\_included\_flag,   
 source\_shape\_type);   
 bit(4) reserved = 0;  
 unsigned int(1) region\_location\_included\_flag;   
 unsigned int(1) region\_rotation\_included\_flag;   
 unsigned int(1) region\_range\_included\_flag;   
 unsigned int(1) region\_interpolate\_included\_flag;   
 unsigned int(8) region\_shape\_type = 0; // for 2D planar (sub)-region  
 RegionWith2DoFStruct(region\_location\_included\_flag,   
 region\_rotation\_included\_flag,   
 region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);

}

* + - 1. ***Sample format***

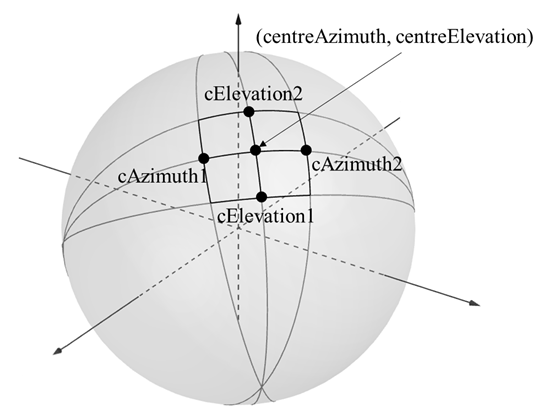
aligned(8) class SpatialRelationship2DPlanarRegionsSample(){  
 RegionWith2DoFStruct(!region\_location\_included\_flag,   
 !region\_rotation\_included\_flag,   
 !region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);  
}

interpolate indicates the continuity in time of the successive samples. When true, the application may linearly interpolate values of the ROI coordinates between the previous sample and the current sample. When false, there shall not be any interpolation of values between the previous and the current samples.

NOTE When using interpolation, it is expected that the interpolated samples match the presentation time of the samples in the referenced track. For instance, for each video sample of a video track, one interpolated 2D Cartesian coordinate sample is calculated.

Sync samples for region metadata tracks are samples for which the interpolate value is 0.

* + 1. **3D Spherical Regions with 6DoF (for 3D spherical regions in 3D space)**

* + - 1. ***Sample Entry***

Sample Entry Type: '6dsr'  
Container: Sample Description Box (‘stsd’)  
Mandatory: No  
Quantity: 0 or 1

aligned(8) class SpatialRelationship3DSphereRegionsSampleEntry  
 extends MetadataSampleEntry (‘6dsr’) {  
 bit(5) reserved = 0;  
 unsigned int(1) source\_location\_included\_flag;   
 unsigned int(1) source\_rotation\_included\_flag;   
 unsigned int(1) source\_range\_included\_flag;   
 unsigned int(8) source\_shape\_type; // 2 or 3 for 3D bounding box or sphere  
 SpatialRelationship3DSourceStruct(source\_location\_included\_flag,   
 source\_rotation\_included\_flag,   
 source\_range\_included\_flag,   
 source\_shape\_type);   
 bit(4) reserved = 0;  
 unsigned int(1) region\_location\_included\_flag;   
 unsigned int(1) region\_rotation\_included\_flag;   
 unsigned int(1) region\_range\_included\_flag;   
 unsigned int(1) region\_interpolate\_included\_flag;   
 unsigned int(8) region\_shape\_type = 1; // for 3D spherical region   
 RegionWith6DoFStruct(region\_location\_included\_flag,   
 region\_rotation\_included\_flag,   
 region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);  
}

* + - 1. ***Sample format***

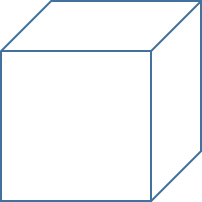
aligned(8) class SpatialRelationship3DSphereRegionsSample(){  
 RegionWith6DoFStruct(!region\_location\_included\_flag,   
 !region\_rotation\_included\_flag,   
 !region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);  
}

interpolate indicates the continuity in time of the successive samples. When true, the application may linearly interpolate values of the ROI coordinates between the previous sample and the current sample. When false, there shall not be any interpolation of values between the previous and the current samples.

NOTE When using interpolation, it is expected that the interpolated samples match the presentation time of the samples in the referenced track. For instance, for each video sample of a video track, one interpolated 2D Cartesian coordinate sample is calculated.

Sync samples for region metadata tracks are samples for which the interpolate value is 0.

* + 1. **3D Planar Regions with 6DoF (for 2D faces/tiles in 3D space)**

* + - 1. ***Sample Entry***

Sample Entry Type: '6dpr'  
Container: Sample Description Box (‘stsd’)  
Mandatory: No  
Quantity: 0 or 1

aligned(8) class SpatialRelationship3DPlanarRegionsSampleEntry  
 extends MetadataSampleEntry (‘6dpr’) {  
 bit(5) reserved = 0;  
 unsigned int(1) source\_location\_included\_flag;   
 unsigned int(1) source\_rotation\_included\_flag;   
 unsigned int(1) source\_range\_included\_flag;   
 unsigned int(8) source\_shape\_type; // 2 or 3 for 3D bounding box or sphere  
 SpatialRelationship3DSourceStruct(source\_location\_included\_flag,   
 source\_rotation\_included\_flag,   
 source\_range\_included\_flag,   
 source\_shape\_type);   
 bit(4) reserved = 0;  
 unsigned int(1) region\_location\_included\_flag;   
 unsigned int(1) region\_rotation\_included\_flag;   
 unsigned int(1) region\_range\_included\_flag;   
 unsigned int(1) region\_interpolate\_included\_flag;   
 unsigned int(8) region\_shape\_type = 0; // for 2D planar region  
 RegionWith6DoFStruct(region\_location\_included\_flag,   
 region\_rotation\_included\_flag,   
 region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);  
}

* + - 1. ***Sample format***

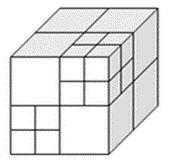
aligned(8) class SpatialRelationship3DPlanarRegionsSample(){  
 RegionWith6DoFStruct(!region\_location\_included\_flag,   
 !region\_rotation\_included\_flag,   
 !region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);  
}

interpolate indicates the continuity in time of the successive samples. When true, the application may linearly interpolate values of the ROI coordinates between the previous sample and the current sample. When false, there shall not be any interpolation of values between the previous and the current samples.

NOTE When using interpolation, it is expected that the interpolated samples match the presentation time of the samples in the referenced track. For instance, for each video sample of a video track, one interpolated 2D Cartesian coordinate sample is calculated.

Sync samples for region metadata tracks are samples for which the interpolate value is 0.

* + 1. **3D Tile Regions with 6DoF (for PCC 3D Tiles)**



* + - 1. ***Sample Entry***

Sample Entry Type: '6dtr'  
Container: Sample Description Box (‘stsd’)  
Mandatory: No  
Quantity: 0 or 1

aligned(8) class SpatialRelationship3DTileRegionsSampleEntry  
 extends MetadataSampleEntry (‘6dtr’) {  
 bit(5) reserved = 0;  
 unsigned int(1) source\_location\_included\_flag;   
 unsigned int(1) source\_rotation\_included\_flag;   
 unsigned int(1) source\_range\_included\_flag;   
 unsigned int(8) source\_shape\_type = 2; // 3D bounding box  
 SpatialRelationship3DSourceStruct(source\_location\_included\_flag,   
 source\_rotation\_included\_flag,   
 source\_range\_included\_flag,   
 source\_shape\_type);   
 bit(4) reserved = 0;  
 unsigned int(1) region\_location\_included\_flag;   
 unsigned int(1) region\_rotation\_included\_flag;   
 unsigned int(1) region\_range\_included\_flag;   
 unsigned int(1) region\_interpolate\_included\_flag;   
 unsigned int(8) region\_shape\_type = 2; // for 3D (sub-)bounding box (tile)  
 RegionWith6DoFStruct(region\_location\_included\_flag,   
 region\_rotation\_included\_flag,   
 region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);  
}

* + - 1. ***Sample format***

aligned(8) class SpatialRelationship3DTileRegionsSample(){  
 RegionWith6DoFStruct(!region\_location\_included\_flag,   
 !region\_rotation\_included\_flag,   
 !region\_range\_included\_flag,  
 region\_shape\_type,  
 region\_interpolate\_included\_flag);  
}

interpolate indicates the continuity in time of the successive samples. When true, the application may linearly interpolate values of the ROI coordinates between the previous sample and the current sample. When false, there shall not be any interpolation of values between the previous and the current samples.

NOTE When using interpolation, it is expected that the interpolated samples match the presentation time of the samples in the referenced track. For instance, for each video sample of a video track, one interpolated 2D Cartesian coordinate sample is calculated.

Sync samples for region metadata tracks are samples for which the interpolate value is 0

* + 1. **Viewports with 3DoF (for 2D faces/tiles in 3D space)**

* + - 1. ***Sample Entry***

Sample Entry Type: '6dvp'  
Container: Sample Description Box (‘stsd’)  
Mandatory: No  
Quantity: 0 or 1

aligned(8) class 6DoFViewportSampleEntry  
 extends MetadataSampleEntry (‘6dvp’) {  
 bit(4) reserved = 0;  
 unsigned int(1) position\_included\_flag;   
 unsigned int(1) orientation\_included\_flag;   
 unsigned int(1) range\_included\_flag;   
 unsigned int(1) interpolate\_included\_flag;   
 unsigned int(8) shape\_type; // 2 or 3 for 3D bounding box or sphere   
 ViewportWith6DoFStruct(position\_included\_flag,   
 orientation\_included\_flag,   
 range\_included\_flag,   
 shape\_type);   
 interpolate\_included\_flag);  
}

* + - 1. ***Sample format***

aligned(8) class 6DoFViewportSample(){  
ViewportWith6DoFStruct(!position\_included\_flag,   
 !orientation\_included\_flag,   
 !range\_included\_flag,   
 !shape\_type);   
 !interpolate\_included\_flag);   
}

interpolate indicates the continuity in time of the successive samples. When true, the application may linearly interpolate values of the ROI coordinates between the previous sample and the current sample. When false, there shall not be any interpolation of values between the previous and the current samples.

NOTE When using interpolation, it is expected that the interpolated samples match the presentation time of the samples in the referenced track. For instance, for each video sample of a video track, one interpolated 2D Cartesian coordinate sample is calculated.

Sync samples for region metadata tracks are samples for which the interpolate value is 0.

* 1. **Timed Metadata for Spatial Relationships of Immersive Media**

1. * 1. **General**

This clause defines metadata data structures for spatial sources (e.g., 3D bounding boxes) and regions within the sources of immersive media, especially visual volumetric video-based coding data (V3C), and their carriage in timed metadata tracks for signaling spatial relationship of the sources and regions. Figure 1 shows the reference coordinate system used as the 3D Cartesian coordinate system with 6DoF.



**Figure 1 — Reference coordinate system**

* + 1. **3D Spatial Region Metadata Data Structures**
       1. ***General***

In order to specify spatial relationships of regions within their respective sources, some common metadata data structures are defined in this subsection.

* + - 1. ***Syntax***

aligned(8) class 3DPoint() {  
 unsigned int(16) x;  
 unsigned int(16) y;  
 unsigned int(16) z;  
}

aligned(8) class CuboidRegionStruct() {  
 unsigned int(16) cuboid\_dx;   
 unsigned int(16) cuboid\_dy;   
 unsigned int(16) cuboid\_dz;  
}

aligned(8) class 3DSpatialRegionStruct(dimensions\_included\_flag) {  
 unsigned int(16) 3d\_region\_id;  
 3DPoint anchor;  
 if (dimensions\_included\_flag) {  
 CuboidRegionStruct();  
 }  
}

aligned(8) class 3DBoundingBoxStruct() {  
 unsigned int(16) bb\_dx;   
 unsigned int(16) bb\_dy;   
 unsigned int(16) bb\_dz;  
}

aligned(8) class V3CSpatialRegionsBox extends FullBox('v3sr',0,0) {  
 unsigned int(16) num\_regions;  
 for (i = 0; i < num\_regions; i++) {  
 3DSpatialRegionStruct(1);  
 unsigned int(8) num\_track\_groups;  
 for (j=0; j<num\_ track\_groups; j++) {  
 unsigned int(32) track\_group\_id;  
 }  
 }  
}

* + - 1. ***Semantics***

3d\_region\_id is an identifier for the spatial region.

x, y, and z specify the x, y, and z coordinate values, respectively, of a 3D point in the Cartesian coordinate system.

cuboid\_dx, cuboid\_dey, and cuboid\_dz indicate the dimensions of the cuboid sub-region in the Cartesian coordinates along the x, y, and z axes, respectively, relative to an anchor point.

bb\_dx, bb\_dy, and bb\_dz indicate the extension of the 3D bounding box of the entire volumetric media in the Cartesian coordinates along the x, y, and z axes, respectively, relative to the origin (0,0,0).

dimensions\_included\_flag is a flag that indicates whether the dimensions of the spatial region are signalled.

num\_regions indicates the number of 3D spatial regions in the volumetric media.

num\_track\_groups indicates the number of track groups associated with a 3D spatial region.

track\_group\_id identifies the track group for the tracks which carry the V3C components for the associated 3D spatial region.

* + 1. **Signalling of Spatial Relationship of Spatial Regions in Timed Metadata Tracks**

1. * + 1. ***General***

This subclause shows signalling of the following spatial relationships within timed metadata tracks using the spatial source and region metadata data structures in subclause 8.2, when individual tracks carry visual content of spatial regions:

* Dynamic spatial region metadata (for partial access of 3D volumetric media data)

Sample Entry Type: 'dysr'  
Container: Sample Description Box ('stsd')  
Mandatory: No  
Quantity: 0 or 1

If an immersive media track (e.g., V3C track) has an associated timed-metadata track with a sample entry type 'dysr', 3D spatial regions defined for the volumetric media stream carried by the immersive media (e.g., V3C) track are considered as dynamic regions (i.e., the spatial region information may dynamically change over time).

The associated timed-metadata track shall contain a 'cdsc' track reference to the immersive media track.

* + - 1. ***Syntax***

aligned(8) class DynamicSpatialRegionSampleEntry extends MetaDataSampleEntry('dysr') {  
 V3CSpatialRegionsBox();  
}

aligned(8) DynamicSpatialRegionSample() {  
 unsigned int(16) num\_regions;  
 for (i = 0; i < num\_regions; i++) {  
 3DSpatialRegionStruct(dimensions\_included\_flag);  
 }  
}

* + - 1. ***Semantics***

3DSpatialRegionStruct() is specified in subclause 8.2.

num\_regions indicates the number of 3D spatial regions signalled in the sample. This may not necessarily be equal to the total number of available regions. Only spatial regions whose position and/or dimensions are being updated are present in the sample.

num\_track\_groups indicates the number of track groups associated with a 3D spatial region.

track\_group\_id identifies the track group for the tracks which carry the immersive media (e.g., V3C) components for the associated 3D spatial region.

3DSpatialRegionStruct() is specified in subclause 8.2.

dimensions\_included\_flag is equal to 0 that indicates that the dimensions are not signalled and implies that they have been previously signalled for the same region (i.e., a previous instance of a 3DSpatialRegionStruct with the same 3d\_region\_id signalled the dimensions).

1. **References**
2. m47497. “Timed Metadata of Improved 2dcc and New 6dsc and 6dcc Types”. March 2019. Geneva, CH.
3. N18396. “WD 5 of ISO/IEC 23090-7 Metadata for Immersive Media (Systems)”. March 2019. Geneva, CH.
4. m47307. “MPEG#126 OMAF agenda and minutes”. March 2019. Geneva, CH.
5. m49351. “Carriage of MPEG-I Metadata of Improved 2dcc and New 6dsc and 6dcc Types in Timed Metadata Tracks”. July 2019. Gothenburg, SE.
6. m49352. “Signaling of MPEG-I Metadata of Improved 2dcc and New 6dsc and 6dcc Types in Track Groups”. July 2019. Gothenburg, SE.
7. CAMBI metrics implementation: <https://github.com/Netflix/vmaf>