ISO/IEC JTC 1/SC 29/WG 03 N00967

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

**Document type:** Output Document

**Title:** Text of ISO\_IEC 23090-7 CAMD Common Metadata for Immersive Media

**Status:** Approved

**Date of document:** 2023-08-09

**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 N00967**

**August 2023, Geneva CH**

|  |  |
| --- | --- |
| **Title** | **Text of ISO\_IEC 23090-7 CAMD Common Metadata for Immersive Media** |
| **Source** | **WG 03, MPEG Systems** |
| **Status** | **Approved** |
| **Serial Number** | **22961** |

**ISO/IEC 23090-7:2023/AMD1**

ISO/IEC TC JTC 1/SC 29/WG3

Secretariat: JISC

**Information technology** — **Coded representation of immersive media (MPEG-I) — Part 7: Metadata for Immersive Media — Amendment 1: Common Metadata for Immersive Media**

CD stage

**Warning for WDs and CDs**

This document is not an ISO International Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an International Standard.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

*To help you, this guide on writing standards was produced by the ISO/TMB and is available at* [*https://www.iso.org/iso/how-to-write-standards.pdf*](https://www.iso.org/iso/how-to-write-standards.pdf)

*A model manuscript of a draft International Standard (known as “The Rice Model”) is available at* [*https://www.iso.org/iso/model\_document-rice\_model.pdf*](https://www.iso.org/iso/model_document-rice_model.pdf)

© ISO 2023

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO’s member body in the country of the requester.

ISO copyright office

CP 401 • Ch. de Blandonnet 8

CH-1214 Vernier, Geneva

Phone: +41 22 749 01 11

Email: copyright@iso.org

Website: www.iso.org

Published in Switzerland

# *Change 1: Normative References*

*Add the following reference to the end of the reference list*

# IEEE STD 754-2019. *IEEE Standard for Floating-Point Arithmetic.* IEEE Computer Society (2019-07-22).

# *Change 2: Terms and Definitions*

*Rename the following term*

### **viewing orientation**

*with the following term (with the same definition)*

### **viewing orientation - spherical**

*Rename the following term*

### **viewport**

*with the following term (with the same definition)*

### **viewport - spherical**

*Add the following the terms and definitions list in Clause 3*

### **3.1.x viewing position**

triple of x, y, z characterizing the position in the global reference coordinate system of a user who is consuming the audio-visual content; in case of image or video, characterizing the position of the viewport.

### **3.1.xx viewing orientation – quaternionic**

triple of quat\_x, quat\_y, quat\_z characterizing the quaternion representation of the orientation that a user is consuming the audio-visual content; in case of image or video, characterizing the orientation of the viewport

### **3.1.xxx viewport – 3D**

projection of texture onto a planar surface of a field of view of an omnidirectional or 3D image or video suitable for display and viewing by the user with a particular viewing orientation and viewing position.

### **3.1.xxxx 3D spatial region**

a region in the 3D Cartesian space using an anchor point and a scale along the three axes

### **3.1.xxxxx spatial region**

a sphere region (3.1.24) or a 3D spatial region (3.1.xxxx)

# *Change 3: Organization of this document*

*Replace the following document organization list*

* Clause 5 describes common metadata applicable to immersive media. This includes reference co-ordinate system related metadata and other common metadata syntax and semantics.
* Clause 6 describes metadata that applies to video and images. This includes projection formats and region-wise packing metadata metadata which applies to video and images.

*with the following document organization list*

* Clause 5 describes common metadata applicable to immersive media, with omnidirectional media in particular. This includes reference co-ordinate system related metadata and other common metadata syntax and semantics.
* Clause 6 describes metadata that applies to video and images, with omnidirectional media in particular. This includes projection formats and region-wise packing metadata metadata which applies to video and images.
* Clause 7 describes common metadata applicable to immersive media, with Visual Volumetric Video-based Coding (V3C) and Video-based Point Cloud Compression (V-PCC) in particular. This includes extrinsic camera information, intrinsic camera information, and other 3D common metadata syntax and semantics.
* Appendices A and B on Annotations of non-timed visual volumetric data and G-PCC data.

# *Add the following paragraphs after the document organization list*

# This document follows the following guiding principles:

# Common metadata and their data structures shall be defined for both 3DoF and 6DoF immersive content, separately as well as jointly, in order to be used for applications that are either specific to separate 3DoF and 6DoF immersive content or general to mixed 3DoF and 6DoF immersive content.

# Basic and common data structures are defined for simple metadata, and extend and enhanced data structures are defined as extensions of basic and common metadata (e.g., viewport is an extension of viewpoint)

# Metadata structures shall be defined in a way to allow their encapsulation in ISOBMFF:

# Static: extension of containing boxes

# Dynamic: timed metadata tracks

# *Change 4:**Common Metadata for Immersive Media*

*Add the following clause 7 after clause 6.*

# Common Metadata for Immersive Media

## **Vector3**

Dimensions, positions, sizes for 3D immersive media can be defined using the following 3D vector data structure.

## **Syntax**

aligned(8) class Vector3(unsigned char precision\_bytes\_minus1) {  
 signed int((precision\_bytes\_minus1+1)\*8) x;  
 signed int((precision\_bytes\_minus1+1)\*8) y;  
 signed int((precision\_bytes\_minus1+1)\*8) z;  
}

## **Semantics**

precision\_bytes\_minus1: Plus 1, specifies the precision of Vector3 components in bytes. Valid values are in the range from [0, 3].

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

## **Scaling**

Scaling in 3-dimension is defined using the following data structure:

## **Syntax**

aligned(8) class 3DScaling (unsigned char precision\_bytes\_minus1) {  
 Vector3 scale(precision\_bytes\_minus1);   
}

## **Semantics**

precision\_bytes\_minus1: Plus 1, specifies the precision of scale components in bytes. Valid values are in the range from [0, 3].

scale.x, scale.y, and scale.z indicate the scaling extension in the Cartesian coordinates along the x, y, and z axes, respectively, relative to the origin (0,0,0).

## **Extrinsic Camera Information**

Extrinsic camera information is defined using the following data structure.

## **Syntax**

class CameraExtrinsics(unsigned char abs\_flag, unsigned char mode, unsigned char pos\_bytes\_minus1, unsigned char pos\_unit, unsigned char quat\_bytes\_minus1, unsigned char quat\_den\_bits\_minus1) {  
 if(mode & 0x1) {  
 signed int((pos\_bytes\_minus1+1)\*8) pos\_x;  
 }  
 if(mode & 0x2) {  
 signed int((pos\_bytes\_minus1+1)\*8) pos\_y;  
 }  
 if(mode & 0x4) {  
 signed int((pos\_bytes\_minus1+1)\*8) pos\_z;  
 }  
 if(mode & 0x8) {  
 Vector3 quat(quat\_bytes\_minus1);   
 }  
}

## **Semantics**

abs\_flag: If 1, absolute position and orientation is specified. If 0, the specified values are added relative to the previously coded position and orientation.

mode: Signalling mode; Valid values are:

[1, 7]: Only the position is signalled.

8: Only the orientation is signalled.

[9, 15]: Both, orientation and position are signalled.

pos\_bytes\_minus1: Plus 1 indicates the number of bytes to be read for pos\_x, pos\_y and pos\_z**.** Valid values are in the range from [0, 3].

pos**\_unit:** Unit of pos\_x, pos\_y and pos\_z. Valid values are in the range from [0, 2], where

0: µm

1: mm

2: m

quat\_bytes\_minus1: Plus 1 indicates the number of bytes to be read for quat.x, quat.y, quat.z**.** Valid values are in the range from [0, 1].

quat\_den\_bits\_minus1**:** Specifies the denominator of quat.x, quat.y and quat.z. Valid values for quat\_den\_bits\_minus1 are in the range from [0, 13]. The denominator is computed as follows:

denominator = 2**quat\_den\_bits\_minus1** + 1

pos\_x: Specifies the x-coordinate of the location of the camera in units specified by pos\_unit. When not present, its value shall be inferred to be 0 if abs\_flag is 1.

pos\_y: Specifies the y-coordinate of the location of the camera in units specified by pos\_unit. When not present, its value shall be inferred to be 0 if abs\_flag is 1.

pos\_z: Specifies the z-coordinate of the location of the camera in units specified by pos\_unit. When not present, its value shall be inferred to be 0 if abs\_flag is 1.

quat.x: Specifies the x component, qX, for the rotation of the camera using the quaternion representation. The range of quat\_x shall be in the range of -2quat\_den\_bits\_minus1+1 to 2quat\_den\_bits\_minus1+1, inclusive. When not present, its value shall be inferred to be 0 if abs\_flag is set to 1.

quat.y: Specifies the y component, qY, for the rotation of the camera using the quaternion representation. The range of quat\_y shall be in the range of -2quat\_den\_bits\_minus1+1 to 2quat\_den\_bits\_minus1+1, inclusive. When not present, its value shall be inferred to be 0 if abs\_flag is set to 1.

quat.z: Specifies the z component, qZ, for the rotation of the camera using the quaternion representation. The range of quat\_z shall be in the range of -2quat\_den\_bits\_minus1+1 to 2quat\_den\_bits\_minus1+1, inclusive. When not present, its value shall be inferred to be 0 if abs\_flag is set to 1.

The values of the quaternion representation are computed as follows:

qX = quat.x / denominator  
qY = quat.y / denominator  
qZ = quat.z / denominator

It is a requirement of bitstream conformance that:

qX2 + qY2 +qZ2 <= 1

The fourth component of the quaternion representation, qW, is computed as follows:

qW = Sqrt( 1 – ( qX2 + qY2 + qZ2 ) )

The point (w, x, y, z) represents a rotation around the axis directed by the vector (x, y, z) by an angle 2\*cos ^{-1}(w)=2\*sin ^{-1}(sqrt(x^{2}+y^{2}+z^{2})).

NOTE. As aligned ISO/IEC FDIS 23090-5, qW is always positive. If a negative qW is desired, one can signal all three syntax elements, cam\_quat\_x, cam\_quat\_y, and cam\_quat\_z with an opposite sign, which is equivalent.

## **Intrinsic Camera Information**

Intrinsic camera information is defined using the following data structure.

## **Syntax**

aligned(8) class IntCameraInfo (unsigned char precision\_bytes\_minus1) {  
 unsigned int(10) camera\_id;   
 bit(3) reserved = 0;   
 unsigned int(3) camera\_type;   
 if (camera\_type == 0) {  
 signed int((precision\_bytes\_minus1+1)\*8) erp\_horizontal\_fov;   
 signed int((precision\_bytes\_minus1+1)\*8) erp\_vertical\_fov;   
 }  
 if (camera\_type == 1) {  
 signed int((precision\_bytes\_minus1+1)\*8) perspective\_horizontal\_fov;   
 unsigned int(8)[4] perspective\_aspect\_ratio;   
 }  
 if (camera\_type == 2) {  
 unsigned int(8)[4] ortho\_aspect\_ratio;   
 unsigned int(8)[4] ortho\_horizontal\_size;   
 }  
 unsigned int(8)[4] clipping\_near\_plane;   
 unsigned int(8)[4] clipping\_far\_plane;   
}

## **Semantics**

camera\_id is an identifier number that is used to identify a given viewport camera parameters.

camera\_type indicates the projection method of the viewport camera. The value 0 specifies ERP projection. The value 1 specifies a perspective projection. The value 2 specifies an orthographic projection. Values in the range 3 to 255 are reserved for future use by ISO/IEC.

precision\_bytes\_minus1: Plus 1 indicates the number of bytes to be read for erp\_horizontal\_fov, erp\_vertical\_fov and perspective\_horizontal\_fov**.** Valid values are in the range from [0, 3].

erp\_horizontal\_fov specifies the longitude range for an ERP projection corresponding to the horizontal size of the viewport region, in units of radians. The value shall be in the range 0 to 2π.

erp\_vertical\_fov specifies the latitude range for an ERP projection corresponding to the vertical size of the viewport region, in units of radians. The value shall be in the range 0 to π.

perspective\_horizontal\_fov specifies the horizontal field of view for perspective projection in radians. The value of shall be in the range of 0 and π.

perspective\_aspect\_ratio specifies the relative aspect ratio of viewport for perspective projection (horizontal/vertical). The value shall be expressed in 32-bit binary floating-point format with the 4 bytes in big-endian order and with the parsing process as specified in IEEE 754.

ortho\_aspect\_ratio specifies the relative aspect ratio of viewport for orthogonal projection (horizontal/vertical). The value shall be expressed in 32-bit binary floating-point format with the 4 bytes in big-endian order and with the parsing process as specified in IEEE 754.

ortho\_horizontal\_size specifies the horizontal size of the orthogonal in metres. The value shall be expressed in 32-bit binary floating-point format with the 4 bytes in big-endian order and with the parsing process as specified in IEEE 754.

clipping\_near\_plane and clipping\_far\_plane indicate the near and far depths (or distances) based on the near and far clipping planes of the viewport in metres. The values shall be expressed in 32-bit binary floating-point format with the 4 bytes in big-endian order and with the parsing process as specified in IEEE 754.

## **Viewing Spaces**

A cuboid viewing space is defined using the following data structure.

## **Syntax**

aligned(8) class ViewingSpace(unsigned char precision\_bytes\_minus1) {  
 Vector3 anchor((precision\_bytes\_minus1+1)\*8);  
 Vector3 dimensions((precision\_bytes\_minus1+1)\*8);   
}

## **Semantics**

precision\_bytes\_minus1: Plus 1 indicates the number of bytes to be read for anchor, and dimensions**.** Valid values are in the range from [0, 3].

anchor.x, anchor.y, and anchor.z indicate the x, y, z position values of the anchor point of the viewing space, respectively, relative to the origin (0,0,0).

dimensions.x, dimensions.y, and dimensions.z indicate the dimensions (or ranges) in the Cartesian coordinates along the x, y, and z axes, respectively, from the anchor (anchor.x, anchor.y, anchor.z).

## **Cuboid Regions**

A cuboid region is defined using the following data structure.

## **Syntax**

aligned(8) class CuboidRegion (  
 unsigned int(1) anchor\_included,  
 unsigned int(1) scale\_included,  
 unsigned char precision\_bytes\_minus1) {  
 unsigned int(16) id;  
 if (anchor\_included) { // when anchor is not 0,0,0  
 Vector3 anchor((precision\_bytes\_minus1+1)\*8);  
 }  
 if (scale\_included) { // when scale is not (1,1,1)  
 Vector3 scale((precision\_bytes\_minus1+1)\*8);  
 }  
 Vector3 dimensions((precision\_bytes\_minus1+1)\*8);  
}

## **Semantics**

id: is an identifier of the region.

precision\_bytes\_minus1: Plus 1 indicates the number of bytes to be read for anchor, scale and dimensions**.** Valid values are in the range from [0, 3].

anchor.x, anchor.y, and anchor.z indicate the x, y, z position values of the anchor point of the viewing space, respectively, relative to the origin (0,0,0). The default is (0,0,0).

scale.x, scale.y, and scale.z indicate the scaling extension in the Cartesian coordinates along the x, y, and z axes, respectively, relative to the origin (0,0,0). The default is (1,1,1).

dimensions.x, dimensions.y, and dimensions.z indicate the dimensions (or ranges) in the Cartesian coordinates along the x, y, and z axes, respectively, from to the anchor (anchor.x, anchor.y, anchor.z).

## **3D Region Set**

## **Definition**

A 3D region set is a structure that defines a part of a volumetric media as a set of one or more regions. Each region of the set is specified as a geometry that defines the shape, position and size of this region inside a reference space that is mapped to the volumetric media with which the 3D region set is associated.

The geometry of a region described by the 3D region set can be represented either by:

* A point
* A polyline
* A plane
* A rectangular cuboid
* A value for an attribute of the volumetric media.

The part of the volumetric media that is contained in a region depends on the geometry of the region as follows:

* When the geometry of a region is represented by a point, the region contains the location of this point.
* When the geometry of a region is represented by a polyline, the region contains the lines that are part of the polyline.
* When the geometry of a region is represented by a plane, the region contains the surface of the plane included in the bounding box of the volumetric media.
* When the geometry of a region is represented by a rectangular cuboid, the region contains the volume inside this rectangular cuboid, including its faces.
* When the geometry of a region is represented by a value for an attribute of the volumetric media, the region contains all the points from the volumetric media whose attribute’s value is the same as the value specified for the region.

A region may be empty if it falls entirely outside the bounding box of the volumetric media with which it is associated. An empty region should be ignored.

## **Syntax**

aligned (8) class 3DRegionSet(version, flags) {  
 unsigned int precision = flags & 3;   
 unsigned int(8) region\_count;   
 for (r=0; r < region\_count; r++) {  
 unsigned int(8) geometry\_type;   
 if (geometry\_type == 0) {  
 // Use a 3D Point from Part-7.   
 Vector3 anchor(precision);   
 }  
 if (geometry\_type == 1) {  
 // polyline  
 unsigned int(field\_size) point\_count;   
 for (i=0; i < point\_count; i++) {  
 // Use a 3D Point from Part-7.   
 Vector3 point(precision);   
 }  
 }  
 if (geometry\_type == 2) {  
 // plane: to be defined in Part-7.   
 Vector3 anchor(precision);   
 Vector3 normal(precision);   
 }  
 if (geometry\_type == 3) {  
 // rectangular cuboid  
 // Defined in Part-7.   
 CuboidRegion cuboid(1, 0, precision);   
 QuaternionRotation rotation();  
 }  
 // … possibly other 3D shapes  
 if (geometry\_type == 5) {  
 // region defined by an attribute  
 // Bounding box of the region. Could be optional.   
 CuboidRegion cuboid(1, 0, precision)   
 unsigned int(8) region\_identifier\_value;   
 }  
}

## **Semantics**

version shall be equal to 0.

flags: the 2 least-significant bits define the precision used by the different geometry attributes of the 3D region set. The values of flags greater than 3 are reserved.

region\_count is the number of regions contained in this 3D region set.

geometry\_type specifies the type of the geometry of a region. The following values for geometry\_type are defined:

* 0: the region is described as a point
* 1: the region is described as a polyline
* 2: the region is described as a plane
* 3: the region is described as a rectangular cuboid
* 5: the region is described using an attribute of the volumetric media.
* Other values are reserved.

anchor specifies the position of the point composing a region when its geometry is a point. It specifies the position of a point contained in the plane composing a region when its geometry is a plane.

point specifies the position of a point belonging to the polyline composing a region when its geometry is a polyline.

normal specifies the normal vector of the plane composing a region when its geometry is a plane.

cuboid specifies the rectangular cuboid composing a region when its geometry is a rectangular cuboid. It specifies a bounding box for a region when its geometry is described using an attribute of the volumetric media.

rotation specifies the rotation applied to the cuboid composing a region when its geometry is a rectangular cuboid.

region\_identifier\_value specifies the value of the attribute defining a region when its geometry is described using an attribute of the volumetric media.

## **Volumetric region item**

## **Definition**

An item with an item\_type value of 'vran' is a volumetric region item that defines one or more regions of an item containing volumetric media.

A volumetric region item allows associating a same set of item properties or other items or both with each individual region it defines inside a volumetric media stored inside an item. Item properties should only be associated with a volumetric region item when the property value for the region differs from the matching (explicit or implied) property value for the whole volumetric media.

The volumetric region item is associated with the item inside which the regions are defined using an item reference of type 'cdsc' from the volumetric region item to the item containing the volumetric media.

The regions described by the volumetric region item defined in VolumetricRegionItem are specified in the data of the volumetric region item.

## **Syntax**

aligned (8) class VolumetricRegionItem {  
 unsigned int(8) version = 0;   
 unsigned int(8) flags;   
 3DRegionSet regions(version, flags);   
}

## **Semantics**

version shall be equal to 0. Readers shall not process an VolumetricRegionItem with an unrecognized version number.

flags is used to control the contents of the 3DRegionSet structure used for defining the content of the volumetric region item.

regions is the structure defining the regions contained in the volumetric region item using the 3DRegionSet defined in ISO/IEC 23090-7.

# *Change 5: Annex A and Annex B*

*Add the following Annex A and Annex V after the newly added clause 7.*

1. (normative)**Annotation of non-timed visual volumetric data** 
   1. **General**

This annex describes how a content creator shall associate annotations with spatial areas of non-timed visual volumetric data encoded and stored in a V3C item using a volumetric region item with item\_type value of 'vran' specified in ISO/IEC 23090-7.

* 1. **Region annotations of V3C data**

A region annotation, i.e., the annotation of a spatial area, of non-timed visual volumetric data stored in a V3C item may consist in:

* an item carrying metadata describing the region;
* an item property describing properties of the region; or
* an item carrying an image associated with the region.

NOTE For example, the image can be a 2D image carried in an image item compliant with ISO/IEC 23008-12, or a non-timed volumetric media carried in an item complying with this document.

Region annotations are associated with one or more spatial areas (or regions) of non-timed visual volumetric data by:

* defining and describing the geometry of these one of more regions in a volumetric region item as specified in ISO/IEC 23090-7;
* associating the volumetric region item with the V3C item it describes using a 'csdc' (content describes) item reference from the volumetric region item to the V3C item; and
* associating any or all of the following with the volumetric region item:
  + one or more descriptive item properties, using the ItemPropertyAssociationBox;

NOTE For instance, a region annotation can use a UserDescriptionProperty as specified in ISO/IEC 23008-12 to associate a description/tags with a region of the non-timed visual volumetric data.

* + one or more metadata items, using an item reference of type 'cdsc' from the metadata item to the volumetric region item.
  + one or more items carrying a two-dimensional image or non-timed volumetric media, using an item reference of type 'eroi' from the volumetric region item to the item.

The region annotation applies to each region described in the volumetric region item individually.

The same region annotations may be associated with several V3C items by associating the same volumetric region item with multiple V3C items.

1. (normative) **Annotation of non-timed G-PCC data** 
   1. **General**

This annex describes how a content creator shall associate annotations with spatial areas of non-timed G-PCC data encoded and stored in a G-PCC item using a volumetric region item with item\_type value of 'vran' specified in ISO/IEC 23090-7.

* 1. **Region annotations of G-PCC data**

A region annotation, i.e., the annotation of a spatial area, of non-timed G-PCC data stored in a G-PCC item may consist in:

* an item carrying metadata describing the region;
* an item property describing properties of the region; or,
* an item carrying an image associated with the region.

NOTE For example, the image can be a 2D image carried in an image item compliant with ISO/IEC 23008-12, or a non-timed volumetric media carried in an item complying with this document.

Region annotations are associated with one or more spatial areas (or regions) of non-timed G-PCC data by:

* defining and describing the geometry of these one of more regions in a volumetric region item as specified in ISO/IEC 23090-7;
* associating the volumetric region item with the G-PCC item it describes using a 'csdc' (content describes) item reference from the volumetric region item to the G-PCC item; and,
* associating any or all of the following with the volumetric region item:
  + one or more descriptive item properties, using the ItemPropertyAssociationBox;

NOTE For instance, a region annotation can use a UserDescriptionProperty as specified in ISO/IEC 23008-12 to associate a description/tags with a region of the non-timed visual volumetric data.

* + one or more metadata items, using an item reference of type 'cdsc' from the metadata item to the volumetric region item.
  + one or more items carrying a two-dimensional image or non-timed volumetric media, using an item reference of type 'eroi' from the volumetric region item to the item.

The region annotation applies to each region described in the volumetric region item individually.

The same region annotations may be associated with several G-PCC items by associating the same volumetric region item with multiple G-PCC items.