

Information technology — Coded representation of immersive media — Part 10: Carriage of visual volumetric video-based coding data

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.

© ISO/IEC 2025

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

Fax: +41 22 749 09 47

Email: copyright@iso.org

Website: www.iso.org

Published in Switzerland

Contents

Foreword.....	viii
Introduction	ix
1 Scope.....	1
2 Normative references	1
3 Terms and definitions.....	1
4 Abbreviated terms	2
5 Overview	3
5.1 General.....	3
5.2 Overall architecture for carriage of V3C data.....	3
5.3 Summary of referenceable code points	4
5.3.1 Brands	4
5.3.2 Uniform resource names	4
5.3.3 Restricted scheme types	5
5.3.4 Sample entry types	5
5.3.5 Box types	6
5.3.6 Track reference types	7
5.3.7 Track grouping types	8
5.3.8 Entity grouping types.....	8
5.3.9 Sample grouping types	8
6 Common definitions	8
6.1 General.....	8
6.2 Data structures.....	9
6.2.1 V3C decoder configuration record	9
6.2.2 Bashmesh configuration record.....	11
6.2.3 Arithmetic coded displacement configuration record	12
6.3 Entity grouping	13
6.3.1 Object switch alternatives	13
7 Carriage of timed visual volumetric video-based coding data.....	13
7.1 General.....	13
7.2 Common data structures and definitions.....	13
7.2.1 V3C configuration box	14
7.2.2 V3C unit header box	14
7.2.3 Decoder configuration mapping box	14
7.2.4 V3C atlas parameter set sample group	15
7.2.5 Alternate track grouping	16
7.2.6 Playout track grouping.....	16
7.3 Common track definitions.....	17
7.3.1 V3C bitstream track.....	17
7.3.2 V3C atlas and atlas tile track	18
7.3.3 V3C video component track.....	21
7.3.4 Track references.....	21
7.4 Timed V-PCC data storage in ISOBMFF	22
7.4.1 Single-track encapsulation	22
7.4.2 Multi-track encapsulation	23
7.5 Timed MIV data storage in ISOBMFF	27
7.5.1 Single-track encapsulation	27
7.5.2 Multi-track encapsulation	28
7.6 Timed V-DMC data storage in ISOBMFF	33

7.6.1	Data structures and definitions for V-DMC storage	33
7.6.2	Single-track encapsulation	35
7.6.3	Multi-track encapsulation	36
8	Carriage of non-timed visual volumetric video-based coding data	44
8.1	General.....	44
8.2	Common data structures and definitions.....	45
8.2.1	V3C configuration item property	45
8.2.2	V3C unit header item property	45
8.2.3	V3C atlas tile configuration item property	46
8.2.4	Sub-sample item property.....	46
8.2.5	Playout entity group.....	47
8.3	Common item definitions	47
8.3.1	V3C item.....	47
8.3.2	V3C atlas item	48
8.3.3	V3C atlas tile item.....	49
8.3.4	V3C image component item	49
8.3.5	Item references.....	49
8.4	Non-timed V-PCC data storage in ISOBMFF	49
8.4.1	Single-item encapsulation	50
8.4.2	Multi-item encapsulation	50
8.5	Non-timed MIV data storage in ISOBMFF	51
8.5.1	Single-item encapsulation	51
8.5.2	Multi-item encapsulation	51
8.6	Non-timed V-DMC data storage in ISOBMFF.....	53
8.6.1	Data structures and definitions.....	53
8.6.2	Single-item encapsulation	56
8.6.3	Multi-item encapsulation	56
9	Partial access of volumetric visual data	60
9.1	General.....	60
9.2	Common data structures	60
9.2.1	3D vector	60
9.2.2	Spatial region bounding box.....	61
9.2.3	Tile mapping.....	61
9.2.4	V3C object collection	62
9.3	Spatial region information structure	64
9.3.1	Definition	64
9.3.2	Syntax	64
9.3.3	Semantics	64
9.4	V3C tile video component track grouping	65
9.4.1	Definition	65
9.4.2	Syntax	65
9.4.3	Semantics	66
9.5	Volumetric media bounding box.....	66
9.5.1	Definition	66
9.5.2	Syntax	66
9.6	Static spatial region collection box	66
9.6.1	Definition	66
9.6.2	Syntax	67
9.6.3	Semantics	67
9.7	Dynamic spatial region information.....	67
9.7.1	General.....	67

9.7.2	Sample entry	67
9.7.3	Sample format	68
9.7.4	Sync samples	68
9.8	Storage of atlas tiles using NALUMapEntry	68
10	Viewport information	69
10.1	General.....	69
10.2	Structures.....	69
10.2.1	Extrinsic camera information	69
10.2.2	Intrinsic camera information.....	70
10.2.3	Viewport information	71
10.3	Viewport information timed-metadata track.....	72
10.3.1	General.....	72
10.3.2	Viewport information sample entry	72
10.3.3	Viewport information sample format	74
11	Encapsulation and signalling in MPEG-DASH	75
11.1	Single track mode.....	75
11.2	Multi-track mode.....	75
11.2.1	General.....	75
11.2.2	V3C preselections	76
11.2.3	V3C atlas tile preselections	77
11.3	DASH MPD descriptors for V3C content.....	77
11.3.1	XML namespace and schema	77
11.3.2	V3C video component descriptor.....	77
11.3.3	V3C non-video component descriptor	80
11.3.4	V3C descriptor	81
11.4	Supporting multiple versions of a V3C media	82
11.5	Switching codecs for V3C components.....	83
11.6	Signalling spatial regions for partial access.....	83
11.6.1	Static spatial regions	83
11.6.2	Dynamic spatial regions.....	86
11.7	Signalling recommended viewports	86
11.7.1	Static viewports	86
11.7.2	Dynamic viewports.....	88
12	Encapsulation and signalling in MMT.....	88
12.1	Introduction	88
12.2	MMT signalling descriptors for V3C content.....	89
12.2.1	Asset reference descriptor.....	89
12.2.2	V3C Asset descriptor	90
12.3	MMT signalling messages for V3C Content.....	91
12.3.1	General.....	91
12.3.2	V3C Asset Group message.....	92
12.3.3	V3C Selection message.....	93
12.3.4	V3C View Change Feedback message.....	95
12.4	Carriage of V3C content with MMT	96
12.5	Carriage of alternative information of V3C contents in MMT	99
Annex A (normative)	File format toolsets and brands	100
A.1	General.....	100
A.2	Single-track encapsulation of V3C data	100
A.3	Multi-track encapsulation of V3C data.....	100
A.3.1	Requirements on files.....	100

A.3.2	Requirements on readers.....	101
A.4	Encapsulation of non-timed V3C data	104
A.4.1	Requirements on files.....	104
A.4.2	Requirements on readers.....	104
Annex B (normative)	V3C DASH schema	107
Annex C (normative)	MIME types and sub-parameters	110
C.1	MIME types and sub-types.....	110
C.2	Sub-parameters for ‘codecs’ parameter	110
C.2.1	General.....	110
C.2.2	V3C family	110
Annex D (informative)	DASH MPD examples.....	111
D.1	Single track example	111
D.2	Multi-track example (using Preselection element)	112
D.3	Multi-track example (using preselection descriptor)	117
D.4	Multi-track example with multiple atlas tile tracks	120
D.5	Multi-track example with multiple atlas tile tracks and volumetric metadata	123
D.6	Alternative V3C content example	126
Annex E (informative)	Partial access support.....	131
E.1	Partial access utilizing V3C volumetric annotation SEI message family	131
E.1.1	General.....	131
E.1.2	Content stored in a single atlas with a single tile	131
E.1.3	Content stored in a single atlas with multiple tiles	131
E.1.4	Content stored in multiple atlases.....	132
E.2	Partial access using volumetric information timed-metadata tracks	134
E.2.1	General.....	134
E.2.2	Content stored in a single atlas with multiple tiles	134
E.2.3	Content stored in multiple atlases.....	134
E.2.4	Content with multiple levels of detail	134
E.3	Partial access for overlapping spatial subdivisions	135
E.3.1	General.....	135
E.3.2	Using viewport spatial regions	135
E.3.3	Using cuboid and viewport spatial regions	135
Annex F (informative)	Examples of using alternate groups	136
Annex G (informative)	Implementation examples of decoding all video components of V3C contents with single decoder instance.....	138
G.1	General.....	138

G.2	Using ISO/IEC 23090-13.....	138
G.3	Using bitstream reconstruction of HEVC tile	139
G.4	Using bitstream reconstruction of VVC subpicture	140
	Annex H (normative) Support of 2D snapshot images	143
H.1	Introduction	143
H.2	Single directional 2D snapshot image track.....	143
H.2.1	Overview	143
H.2.2	Restriction to the track.....	143
H.2.3	Track references.....	143
H.2.4	Indication of camera used for rendering snapshot images	144
H.3	Multi-directional 2D snapshot image track.....	144
	Bibliography.....	146

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see <https://patents.iec.ch>).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 29, *Coding of audio, picture, multimedia and hypermedia information*.

This second edition cancels and replaces the first edition (ISO/IEC 23090-10:2022), which has been technically revised.

The main changes are as follows:

- The support of packed video has been added.
- A single item encapsulation of a non-timed V3C data has been added.
- The document structure is changed to specify the encapsulation according to V3C applications. File format tools and brands are clarified.
- The carriage of coded media representations which comply with video-based dynamic mesh coding (specified in ISO/IEC 23090-29) has been added.

A list of all parts in the ISO/IEC 23090 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

This document addresses the storage of visual volumetric video-based coding data in files based on ISO/IEC 14496-12, reusing existing tools for storage of video-coded components. Another important aspect considered by this document is supporting flexible extraction of component streams at delivery or decoding time, or both.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of a patent.

ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO and IEC. Information may be obtained from the patent database available at www.iso.org/patents or patents.iec.ch.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those in the patent database. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Information technology — Coded representation of immersive media — Part 10: Carriage of visual volumetric video-based coding data

1 Scope

This document specifies carriage of coded media representations which comply with visual volumetric video content which is encoded using video-based coding and video-based point cloud compression (specified in ISO/IEC 23090-5) or its derived specification.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEEE 754-2019, *IEEE Standard for Floating-Point Arithmetic*

IETF RFC 6381, *The 'Codecs' and 'Profiles' Parameters for "Bucket" Media Types*

ISO/IEC 14496-1, *Information technology — Coding of audio-visual objects — Part 1: Systems*

ISO/IEC 14496-12:2022, *Information technology — Coding of audio-visual objects — Part 12: ISO base media file format*

ISO/IEC 14496-15, *Information technology — Coding of audio-visual objects — Part 15: Carriage of network abstraction layer (NAL) unit structured video in the ISO based media file format*

ISO/IEC 23008-1:2017, *Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 1: MPEG media transport (MMT)*

ISO/IEC 23009-1:2022, *Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats*

ISO/IEC 23090-5:2023, *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 23090-29, *Information technology — Coded representation of immersive media — Part 29: Video-based Dynamic Mesh Coding (V-DMC)*

W3C Recommendation, *XML schema part 1: Structures*

W3C Recommendation, *XML schema part 2: Datatypes*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 14496-1, ISO/IEC 14496-12, ISO/IEC 23090-5 and the following apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at <https://www.iso.org/obp>

— IEC Electropedia: available at <https://www.electropedia.org/>

3.1

atlas parameter sets

non-ACL NAL units that have nal_unit_type equal to NAL_ASPS, NAL_AAPS, or NAL_AFPS.

3.2

V3C content

volumetric media that is encoded using V3C or its derived specifications

Note 1 to entry: For the purposes of this document, the media shall be encoded using ISO/IEC 23090-5 or derived specifications.

3.3

volumetric visual track

track with a handler type reserved to describe volumetric visual track

3.4

V3C track

V3C bitstream track, V3C atlas track or V3C atlas tile track

3.5

V3C bitstream track

volumetric visual track containing V3C bitstream in case of single-track container

3.6

V3C atlas track

volumetric visual track containing V3C atlas bitstream in case of multi-track container

3.7

V3C atlas tile track

volumetric visual track containing portion of V3C atlas bitstream corresponding to one or more tiles in case of multi-track container

3.8

V3C video component track

video track which carries 2D video encoded data for any of the occupancy, geometry, or attribute component video bitstreams of the V3C bitstream

4 Abbreviated terms

2D	two-dimensional
3D	three-dimensional
CVS	coded V3C sequence
DASH	dynamic adaptive streaming over HTTP
HTTP	Hyper-text transfer protocol
IRAP	intra random access point
ISOBMFF	ISO base media file format
LoD	level of detail

MIV	MPEG immersive video
MMT	MPEG media transport
PCC	point cloud compression
SEI	supplemental enhancement information
V3C	visual volumetric video-based coding
VPS	V3C parameter set
V-DMC	video-based dynamic mesh coding
V-PCC	video-based point cloud compression

5 Overview

[Ed(SOH) : We need to update the following description to cover V3C and its derived specifications]

5.1 General

Visual volumetric video-based coding (V3C) provides mechanism for coding visual volumetric frames. Visual volumetric frames are coded by converting the 3D volumetric information into a collection of 2D images and associated data. The converted 2D images are coded using widely available video and image coding specifications and the associated data, i.e., atlas data, is coded according to ISO/IEC 23090-5. The coded images and the coded atlas data are multiplexed and form a V3C bitstream.

A V3C bitstream consists of one or more CVSs. A CVS starts with a VPS, included in at least one V3C unit or provided through external means, and contains one or more V3C units carrying V3C sub-bitstreams, with each V3C sub-bitstream associated with a V3C component, e.g., atlas, occupancy, geometry, or attribute.

5.2 Overall architecture for carriage of V3C data

Figure 1 shows a typical content flow process for V3C media.

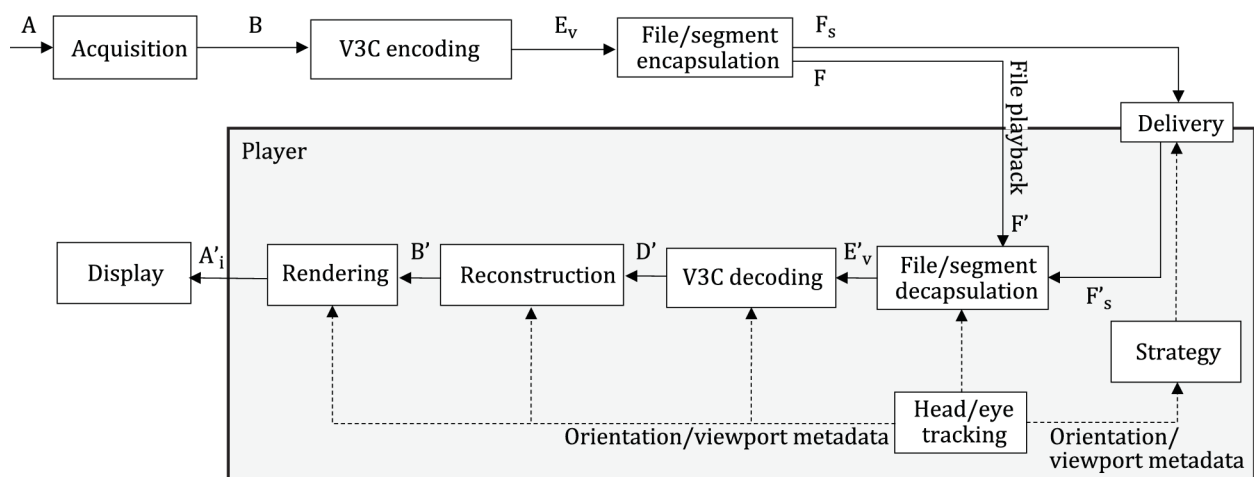


Figure 1 — Content flow process for V3C media

A real-world or synthetic visual scene (A) is captured by a set of cameras, a camera device with multiple lenses and sensors, or by virtual cameras. The acquisition results in source volumetric data (B). One or

multiple volumetric frames are encoded as a coded V3C bitstream including an atlas bitstream, at most one occupancy bitstream, a geometry bitstream, and zero or more attribute bitstreams (E_v). One or more coded bitstreams are then packaged into a media file for local playback (F) or a sequence of an initialization segment and media segments for streaming (F_s), according to a particular media container file format. In this document, the media container file format is the ISO Base Media File Format specified in ISO/IEC 14496-12. The file encapsulator may also include metadata into the file or the segments. The segments F_s are delivered using a delivery mechanism to a player.

The file that the file encapsulator outputs (F) is identical to the file that the file decapsulator takes as input (F'). A file decapsulator processes the file (F') or the received segments (F'_s) and extracts the coded bitstreams (E'_v) and parses the metadata. The V3C bitstream is then decoded into a decoded signal (D'). The decoded volumetric data (D') are reconstructed, rendered, and displayed onto the screen of a head-mounted display or any other display device based on the current viewing orientation or viewport. The current viewing orientation is determined by the head tracking and possibly also eye tracking functionality. In viewport-dependent delivery, the current viewing orientation is also passed to the strategy module, which determines the tracks to be received based on the viewing orientation.

The process described above is applicable to both live and on-demand use cases.

The following interfaces are normatively specified in this document:

- F/F' : media file including the specification of the track formats, which may contain constraints on the elementary streams contained within the samples of the tracks; see Clause 7 for timed V3C content and Clause 8 for non-timed V3C data.
- Clause 11 specifies the delivery related interfaces for DASH delivery.
- Clause 12 specifies the delivery related interfaces for MMT delivery.

5.3 Summary of referenceable code points

5.3.1 Brands

ISO/IEC 14496-12 defines the concept of brands, which may be indicated in the FileTypeBox. Brands are used in this document to indicate conformance to an encapsulation mode and a specific set of tools, as well as requirements on other specifications (e.g., ISO/IEC 14496-12).

The brands specified in this document are listed in Table 1 and defined in Annex A.

Table 1 — Brands specified in this document

Brand	Clause	Informative description
v3st	A.2	Single track encapsulation mode
v3mt	A.3	Multi-track encapsulation mode
v3mp	A.3	Multi-track encapsulation mode with partial access support
v3nt	A.4	Non-timed V3C data

5.3.2 Uniform resource names

The URNs specified in this document are listed in Table 2.

Table 2 — URNs specified in this document

URN	Clause	Informative description
-----	--------	-------------------------

urn:mpeg:mpegI:v3c:2020	11.3.1	Namespace for the XML elements and attributes specified in this document
urn:mpeg:mpegI:v3c:2020:videocomponent	11.3.2	Scheme identifier for the V3C video component DASH MPD descriptor
urn:mpeg:mpegI:v3c:2025:nonvideocomponent	11.3.3	Scheme identifier for the V3C non-video component DASH MPD descriptor
urn:mpeg:mpegI:v3c:2020:v3c	11.3.4	Scheme identifier for the V3C content DASH MPD descriptor
urn:mpeg:mpegI:v3c:2020:v3sr	11.6.1	Scheme identifier for the V3C static spatial region DASH MPD descriptor

5.3.3 Restricted scheme types

The restricted scheme types specified in this document are listed in Table 3.

Table 3 — Restricted scheme types specified in this document

Restricted scheme type	Clause	Informative description
v3vc	7.3.3	V3C component video

5.3.4 Sample entry types

The sample entry types specified in this document are listed in Table 4.

Table 4 — Sample entry types specified in this document

Sample entry type	Clause	Informative description
v3e1	7.4.1.2 7.5.1.2	For use with the single-track mode with all atlas parameter sets and SEI messages carried in decoder configuration record
v3eg	7.4.1.2 7.5.1.2	For use with the single-track mode with atlas parameter sets and SEI messages carried in decoder configuration record and in track samples
vdm1	7.6.2.2	For use with the single-track mode with all atlas parameter sets and SEI messages, basemesh parameter sets, and displacement parameter sets when arithmetic coded displacement component is present, carried in decoder configuration record
vdmg	7.6.2.2	For use with the single-track mode with all atlas parameter sets and SEI messages, basemesh parameter sets, and displacement parameter sets when arithmetic coded displacement component is present, carried in decoder configuration record and in track samples
v3c1	7.4.2.2.1 7.5.2.2.1	For use with the multi-track mode with a single atlas and all atlas parameter sets and SEI messages carried in decoder configuration record

v3cg	7.4.2.2.1 7.5.2.2.1	For use with the multi-track mode with a single atlas and atlas parameter sets and SEI messages carried in decoder configuration record and in track samples
v3cb	7.5.2.2.1	For use with a base track in the multi-track mode with multiple atlases
v3a1	7.5.2.2.1	For use with an atlas track in the multi-track mode with multiple atlases and all atlas parameter sets and SEI messages carried in decoder configuration record
v3ag	7.5.2.2.1	For use with an atlas track in multi-track mode with multiple atlases and atlas parameter sets and SEI messages carried in decoder configuration record and in track samples
v3t1	7.3.2.2	For use with an atlas tile track in the multi-track mode
dyvm	9.7.2	For use with a timed metadata track indicating the dynamic spatial regions that are dynamically changing over time
6vpt	10.3.2	For use with a timed metadata track indicating viewport information that are dynamically changing over time

5.3.5 Box types

The box types specified in this document are listed in bold in Table 5. Mandatory boxes are marked with an asterisk. Box types without a four-character code are marked with '-' in the structure.

Table 5 — Box types specified in this document

Box types, structure, and cross-reference (Informative)										
moov									*	ISOBMFF <i>container for all the metadata</i>
	trak								*	ISOBMFF <i>container for an individual track or stream</i>
		trgr								ISOBMFF <i>track grouping indication</i>
			potg							7.2.6 <i>playout track group box</i>
			vtcg							9.4 <i>atlas tile components track group box</i>
		mdia							*	ISOBMFF <i>container for the media information in a track</i>
			minf						*	ISOBMFF <i>media information container</i>
				stbl					*	ISOBMFF <i>sample table box, container for the time/space map</i>
					stsd				*	ISOBMFF <i>sample descriptions (codec types, initialization etc.)</i>
						-				ISOBMFF <i>sample entry or restricted sample entry</i>
							rinf			ISOBMFF <i>restricted scheme info box</i>
								frma		ISOBMFF <i>original format box</i>
								schm		ISOBMFF <i>scheme type box</i>
								schi		ISOBMFF <i>scheme information box</i>
									vunt	7.2.2 <i>V3C unit header box</i>
									mmvi	7.4.2.3.1 <i>Multimap video box</i>
						dyvm				9.7.2 <i>dynamic volumetric metadata sample entry</i>
						6vpt				10.3.2 <i>viewport information sample entry</i>

Box types, structure, and cross-reference (Informative)											
							6vpC			10.3.2	viewport information configuration box
					-					ISOBMFF	visual sample entry
					-					ISOBMFF	volumetric visual sample entry
							v3cC			7.2.1	V3C decoder configuration box
							vunt			7.2.2	V3C unit header box
							v3tC			7.3.2.2	V3C atlas tile configuration box
							v3tC			7.6.1.3	Instanced rendering box
							vpbb			9.5	
							v3sc			9.6	Static spatial region collection box
meta										ISOBMFF	Metadata
	grpl									ISOBMFF	group list box
		epl								8.2.5	layout entity group box
		swpc								6.3.1	object switch alternatives box
	iprp									ISOBMFF	item properties box
		ipco								ISOBMFF	item property container box
			v3cp							8.2.1	V3C configuration item property
			vutp							8.2.2	V3C unit header item property
			v3tp							8.2.3	V3C atlas tile configuration item property
			subs							8.2.4	Sub-sample item property
			vire							8.6.1.1	Instanced rendering item property

5.3.6 Track reference types

The track reference types specified in this document are listed in Table 6.

Table 6 — Track reference types specified in this document

Track reference type	Clause	Informative description
v3cs	7.5.2.4.1	Referenced track is a V3C atlas track
v3ct	7.3.4.2	Referenced track is a V3C atlas tile track
v3vo	7.3.4.3	Referenced track is a V3C video component track carrying occupancy data
v3vg	7.3.4.3	Referenced track is a V3C video component track carrying geometry data
v3va	7.3.4.3	Referenced track is a V3C video component track carrying attribute data
v3vp	7.3.4.3	Referenced track is a V3C video component track carrying packed video component data
vdmb	7.6.3.6.2	Referenced track is a V3C component track carrying basemesh data

vdmd	7.6.3.6.2	Referenced track is a V3C component track carrying arithmetic coded displacement data
bmcs	7.6.3.6.3	Referenced track is a submesh track

5.3.7 Track grouping types

The track grouping types specified in this document are listed in Table 7.

Table 7 — Track grouping types specified in this document

Track grouping type	Clause	Informative description
potg	7.2.6	Playout track grouping
vtcg	9.4	V3C tile components track grouping

5.3.8 Entity grouping types

The entity grouping types specified in this document are listed in Table 8.

Table 8 — Entity grouping type specified in this document

Entity grouping type	Clause	Informative description
swpc	6.3.1	Object switch alternatives
eplv	8.2.5	Playout entity grouping

5.3.9 Sample grouping types

The sample grouping types specified in this document are listed in Table 9.

Table 9 — Sample grouping types specified in this document

Sample grouping type	Clause	Informative description
vaps	7.2.3	V3C atlas parameter set sample grouping

6 Common definitions

6.1 General

This clause contains common data structures, and entity groups which are used for carriage of timed and non-timed V3C data in a file. The V3C data shall be encoded by ISO/IEC 23090-5 or its derived specifications.

6.2 Data structures

6.2.1 V3C decoder configuration record

6.2.1.1 Definition

The V3C decoder configuration record provides V3C bitstream's decoding specific information (i.e. parameter sets and SEI messages) for further configuration and initialization of the V3C decoder. This document sets the following restrictions for V3C content encapsulation:

- `num_of_v3c_parameter_sets` in `V3CDecoderConfigurationRecord` shall be equal to 1.
- NAL units of the same type and with the same `array_completeness` value should always be stored in a single setup unit array.
- For a given atlas the tile IDs shall remain unique for the duration of the sequence. Tile IDs shall not be re-used between atlas frame parameter sets.
- V3C data is naturally represented as variable bit rate in the file format and should be filled for transmission if needed. Filler Data NAL units and Filler Data SEI messages shall not be present in the file format stored stream when the sample entry does not also permit in-stream parameter sets.

If the version number is not supported or recognized by the reader, then it shall not attempt to decode this configuration record or the bitstreams to which it applies.

6.2.1.2 Syntax

```
aligned(8) class V3CDecoderConfigurationRecord(unsigned int version) {
    if(version == 0){
        unsigned int(8) ptl_profile_toolset_idc
        unsigned int(3) unit_size_precision_bytes_minus1;
        unsigned int(5) num_of_v3c_parameter_sets;
        for (int i=0; i < num_of_v3c_parameter_sets; i++) {
            unsigned int(16) v3c_parameter_set_length;
            bit(8) v3c_parameter_set[v3c_parameter_set_length];
        }
        unsigned int(8) num_of_setup_unit_arrays;
        for (int j=0; j < num_of_setup_unit_arrays; j++) {
            unsigned int(1) array_completeness;
            bit(1) reserved = 0;
            unsigned int(6) nal_unit_type;
            unsigned int(8) num_nal_units;
            for (int i=0; i < num_nal_units; i++) {
                unsigned int(16) setup_unit_length;
                bit(8) setup_unit[setup_unit_length];
            }
        }
    }
}
```

6.2.1.3 Semantics

`ptl_profile_toolset_idc` indicates the toolset combination profile component to which the V3C bitstream conforms as specified in Annex A of ISO/IEC 23090-5. Values in range 0-1 inclusive indicate that the V3C bitstream conforms to V-PCC (ISO/IEC 23090-5), values in range of 64-66

inclusive indicate that the V3C bitstream conforms to MIV (ISO/IEC 23090-12) and values in range 128-143 inclusive indicate that the V3C bitstream conforms to V-DMC (ISO/IEC 23090-29). Other values are reserved.

`unit_size_precision_bytes_minus1` plus 1 specifies the precision, in bytes, of the sample stream NAL unit or sample stream V3C unit to which this configuration record applies. The value of this field shall be conditional on the 4CC-code of the sample entry. For V3C atlas tracks `unit_size_precision_bytes_minus1` shall be equal to `ssnh_unit_size_precision_bytes_minus1` in `sample_stream_nal_header()`. For V3C bitstream tracks `unit_size_precision_bytes_minus1` shall be equal to `ssvh_unit_size_precision_bytes_minus1` in `sample_stream_v3c_header()`.

`num_of_v3c_parameter_sets` specifies the number of V3C parameter set units signalled in the decoder configuration record.

`v3c_parameter_set_length` indicates the size, in bytes, of the `v3c_parameter_set` array. The signalled value shall not be equal to 0.

NOTE `v3c_parameter_set_length` syntax element can be represented by up to 64 bits, as defined in ISO/IEC 23090-5. In this document, it limits the representation of the information to 16 bits as it suffices in practical implementations.

`v3c_parameter_set` is an array of data containing the entire `v3c_unit` of type `V3C_VPS`, as defined in ISO/IEC 23090-5.

`num_of_setup_unit_arrays` indicates the number of arrays of atlas NAL units of the indicated type(s).

`array_completeness` when equal to 1 indicates that all atlas NAL units of the given type are in the following array and none are in the stream; when equal to 0 indicates that additional atlas NAL units of the indicated type may be in the stream; the default and permitted values are constrained by the sample entry name.

`nal_unit_type` indicates the type of the atlas NAL units in the following array (which shall be all of that type); it takes a value as defined in ISO/IEC 23090-5; it is restricted to take one of the values indicating a `NAL_ASPS`, `NAL_AAPS`, `NAL_AFPS`, `NAL_PREFIX_ESEI`, `NAL_PREFIX_NSEI`, `NAL_SUFFIX_ESEI`, or `NAL_SUFFIX_NSEI` atlas NAL unit.

`num_nal_units` indicates the number of atlas NAL units of type `nal_unit_type` included in the configuration record for the stream to which this configuration record applies.

`setup_unit_length` indicates the size, in bytes, of the `setup_unit` array. The signalled value shall not be equal to 0.

`setup_unit` is an array of data containing the entire `nal_unit` as defined in ISO/IEC 23090-5. The contained NAL unit shall be of the same type as specified by `nal_unit_type`. When present in `setup_unit`, `NAL_PREFIX_ESEI`, `NAL_PREFIX_NSEI`, `NAL_SUFFIX_ESEI`, or `NAL_SUFFIX_NSEI` contain SEI messages of a 'declarative' nature, that is, those that provide information about the stream as a whole

6.2.2 Bashmesh configuration record

6.2.2.1 Definition

The basemesh decoder configuration record provides the basemesh decoding specific information for configuring and initialization of the basemesh decoder.

6.2.2.2 Syntax

```
aligned(8) class BaseMeshDecoderConfigurationRecord {
    bit(5) reserved=0;
    unsigned int(3) unit_size_precision_bytes_minus1;
    unsigned int(8) num_of_bmesh_setup_unit_arrays;
    for (int j=0; j < num_of_bmesh_setup_unit_arrays; j++) {
        unsigned int(1) array_completeness;
        bit(1) reserved = 0;
        unsigned int(6) bmesh_nal_unit_type;
        unsigned int(8) num_bmesh_nal_units;
        for (int i=0; i < num_bmesh_nal_units; i++) {
            unsigned int(16) bmesh_setup_unit_length;
            bmesh_nal_unit bmesh_setup_unit(bmesh_setup_unit_length);
        }
    }
    // additional fields
}
```

6.2.2.3 Semantics

`unit_size_precision_bytes_minus1` plus 1 specifies the precision, in bytes, of the sample stream NAL unit to which this configuration record applies. It shall be equal to `ssnh_unit_size_precision_bytes_minus1` in `sample_stream_nal_header()` for the basemesh component bitstream.

`num_of_bmesh_setup_unit_arrays` indicates the number of arrays of following basemesh NAL units of the indicated type(s) and shall be same or larger than one.

`array_completeness` when equal to 1 indicates that all basemesh NAL units of the given type are in the following array and none are in the stream; when equal to 0 indicates that additional base mesh NAL units of the indicated type may be in the stream; the default and permitted values are constrained by the sample entry name.

`bmesh_nal_unit_type` indicates the type of the basemesh NAL units in the following array. It takes a value as defined in ISO/IEC 23090-29 Annex H; it is restricted to take one of the values indicating a `BNAL_BMSPS`, `BNAL_BMFPS`, `BNAL_PREFIX_ESEI`, `NAL_PREFIX_NSEI`, `BNAL_SUFFIX_ESEI`, or `BNAL_SUFFIX_NSEI` base mesh NAL unit.

`num_bmesh_nal_units` indicates the number of basemesh NAL units of type indicated by `bmesh_nal_unit_type` in the following array.

`bmesh_setup_unit_length` indicates the size, in bytes, of the `bmesh_setup_unit` field. The length field includes the size of both the basemesh NAL unit header and the basemesh NAL unit payload but does not include the length field itself.

`bmesh_setup_unit` contains a basemesh NAL unit according to related `bmesh_nal_unit_type`.

6.2.3 Arithmetic coded displacement configuration record

6.2.3.1 Definition

Arithmetic coded displacement decoder configuration record provides the decoding specific information for arithmetic coded displacement sub-bitstream.

6.2.3.2 Syntax

```
aligned(8) class ACDisplacementDecoderConfigurationRecord {
    bit(5) reserved=0;
    unsigned int(3) unit_size_precision_bytes_minus1;
    unsigned int(8) num_of_displ_setup_unit_arrays;
    for (int j=0; j < num_of_displ_setup_unit_arrays; j++) {
        unsigned int(1) array_completeness;
        bit(1) reserved = 0;
        unsigned int(6) displ_nal_unit_type;
        unsigned int(8) num_displ_nal_units;
        for (int i=0; i < num_displ_nal_units; i++) {
            unsigned int(16) displ_setup_unit_length;
            displ_nal_unit displ_setup_unit(displ_setup_unit_length);
        }
    }
    // additional fields
}
```

6.2.3.3 Semantics

`unit_size_precision_bytes_minus1` plus 1 specifies the precision, in bytes, of the sample stream NAL unit to which this configuration record applies. It shall be equal to `ssnh_unit_size_precision_bytes_minus1` in `sample_stream_nal_header()` for the displacement component bitstream.

`num_of_displ_setup_unit_arrays` indicates the number of arrays of following displacement NAL units of the indicated type(s) and shall be same or larger than one.

`array_completeness` when equal to 1 indicates that all displacement NAL units of the given type are in the following array and none are in the stream; when equal to 0 indicates that additional displacement NAL units of the indicated type may be in the stream; the default and permitted values are constrained by the sample entry name.

`displ_nal_unit_type` indicates the type of the displacement NAL units in the following array. it takes a value as defined in ISO/IEC 23090-29 Annex J; it is restricted to take one of the values indicating a `DNAL_DSPS`, `DNAL_DFPS`, `DNAL_PREFIX_ESEI`, `DNAL_PREFIX_NSEI`, `DNAL_SUFFIX_ESEI`, or `DNAL_SUFFIX_NSEI` displacement NAL unit.

`num_displ_nal_units` indicates the number of displacement NAL units of type indicated by `displ_nal_unit_type` in the following array.

`displ_setup_unit_length` indicates the size, in bytes, of the `displ_setup_unit` field. The length field includes the size of both the displacement NAL unit header and the displacement NAL unit payload but does not include the length field itself.

`displ_setup_unit` contains a displacement NAL unit according to related `displ_nal_unit_type`.

6.3 Entity grouping

6.3.1 Object switch alternatives

6.3.1.1 Definition

Box Types:	'swpc'
Container:	Groups List Box ('grpl')
Mandatory:	No
Quantity:	Zero or more

EntityToGroupBox with grouping_type equal to 'swpc' specifies tracks and items that are associated with each other based on a logical context and are user switchable alternatives of each other.

If ObjectSwitchAlternativesBox is absent, there is no information on which items or tracks should be played out together. When ObjectSwitchAlternativesBox contains alternatives, only items or tracks from one point-cloud object within an alternate group should be played or streamed at any one time.

6.3.1.2 Syntax

```
aligned(8) class ObjectSwitchAlternativesBox extends EntityToGroupBox('swpc') {
}
```

7 Carriage of timed visual volumetric video-based coding data

7.1 General

This clause defines the storage for a V3C bitstream utilizing the existing capabilities of the ISO base media file format and defining extensions, when necessary. The clause is divided into subclauses where V3C application specific encapsulation is described. Only one of the below encapsulations for a same V3C bitstream shall be used at the same time.

- Single track encapsulation, where a V3C bitstream track contains the entire coded V3C bitstream.
- Multiple track encapsulation, where each V3C sub-bitstream is encapsulated into a separate track.

In this clause, the volumetric media related descriptions and definitions defined in ISO/IEC 14496-12:2022 apply. A volumetric visual track shall be identified by the volumetric visual media handler type 'volv' in the HandlerBox of the MediaBox and by a volumetric visual media header. Multiple volumetric visual tracks may be present in the file.

The following subclauses define application specific mechanism for encapsulating V3C bitstreams:

- Subclause 7.4 specifies the encapsulation of timed V-PCC data in ISOBMFF.
- Subclause 7.5 specifies the encapsulation of timed MIV data in ISOBMFF.
- Subclause 7.6 specifies the encapsulation of timed V-DMC data in ISOBMFF.

7.2 Common data structures and definitions

This clause contains common definitions that can be used by different applications of V3C.

7.2.1 V3C configuration box

7.2.1.1 Definition

A V3C decoder configuration box includes a `V3CDecoderConfigurationRecord` as defined in subclause 6.2.1. In this document, the value of `version` shall be equal to 0.

7.2.1.2 Syntax

```
class V3CConfigurationBox extends FullBox('v3cC', version = 0, 0) {  
    V3CDecoderConfigurationRecord v3c_config(version);  
}
```

7.2.1.3 Semantics

`v3c_config` contains a single instance of `V3CDecoderConfigurationRecord` defined in subclause 6.2.1.

7.2.2 V3C unit header box

7.2.2.1 Definition

Box Types: 'vunt'
Container: V3C atlas sample entry and `SchemeInformationBox`
Mandatory: Yes
Quantity: One or more

The `V3CUnitHeaderBox` contains the V3C unit header describing the data carried by the respective track.

7.2.2.2 Syntax

```
aligned(8) class V3CUnitHeaderBox extends FullBox('vunt', version = 0, 0){  
    bit(8) header[4];  
}
```

7.2.2.3 Semantics

`header` contains a single instance of the 32-bit V3C unit header as defined in ISO/IEC 23090-5.

7.2.3 Decoder configuration mapping box

7.2.3.1 Definition

Box Types: 'dssm'
Container: `V3CBitstreamSampleEntry` ('v3e1', 'v3eg', 'vdm1' or 'vdmg')
Mandatory: No
Quantity: zero or one (per sample entry)

The `DecoderConfigurationMappingBox` provides associations or mapping between decoder configuration information and sub-bitstreams to which a decoder configuration applies.

7.2.3.2 Syntax

```
aligned(8) class DecoderConfigurationMappingBox extends FullBox('dssm', version=0, flags=0)  
{
```



```

    unsigned int(8) nb_associations;
    for (i=0; i < nb_associations; i++) {
        unsigned int(8) decoder_cfg_index;
        unsigned int(8) num_subbitstreams;
        for (j=0; j < num_subbitstreams; j++)
            unsigned int(32) v3c_unit_header;
    }
}

```

7.2.3.3 Semantics

`nb_associations` indicates the number of associations declared in this box.

`decoder_cfg_index` provides an index of a declared list of decoder configuration boxes for the V3C bitstream. It is a 1-based index with values between 1 to the number of configuration boxes declared in the containing sample entry, minus 1 (the `V3CConfigurationBox` is not mapped).

`num_subbitstreams` indicates the number of sub-bitstreams that use the decoder configuration specified by the `decoder_cfg_index`.

`v3c_unit_header` is the V3C unit header as per ISO/IEC 23090-5 of the V3C unit applying to the sub-bitstream that is mapped to a decoder configuration information.

7.2.4 V3C atlas parameter set sample group

7.2.4.1 Definition

Box Types:	'vaps'
Container:	Sample Group Description Box ('sgpd')
Mandatory:	No
Quantity:	Zero or one

The use of 'vaps' for the `grouping_type` in sample grouping represents the assignment of samples in V3C atlas track or V3C bitstream track to the atlas parameter sets carried in this sample group.

When a `SampleToGroupBox` with `grouping_type` equal to 'vaps' is present, an accompanying `SampleGroupDescriptionBox` with the same grouping type shall be present and contains the ID of the group that the samples belong to.

Sample grouping type 'vaps' shall not be used with a track with a sample entry 'v3c1', 'v3e1' or 'v3a1'.

NOTE V3C atlas parameter set sample group can be used to improve random access of atlas tracks, by removing the need to replicate parameter sets and SEI messages for sync samples.

When sample grouping type 'vaps' is used, the sample shall not include atlas parameter sets, and if it is not used, a sample may include atlas parameter sets.

7.2.4.2 Syntax

```

aligned(8) class V3CAtlasParamSampleGroupDescriptionEntry()
    extends VolumetricVisualSampleGroupEntry('vaps') {
    unsigned int(8) num_of_setup_units;
    for (int i=0; i < num_of_setup_units; i++) {
        unsigned int(16) setup_unit_length;
    }
}

```

```

        // nal_unit(size) as defined in ISO/IEC 23090-5
        nal_unit_setup_unit(setup_unit_length);
    }
}

```

7.2.4.3 Semantics

`num_of_setup_units` specifies the number of setup units signalled in the sample group description.

`setup_unit_length` indicates the size, in bytes, of the `setup_unit` field. The length field includes the size of both the NAL unit header and the NAL unit payload but does not include the length field itself.

`setup_unit` is a NAL unit of type `NAL_ASPS`, `NAL_AAPS`, `NAL_AFPS`, `NAL_PREFIX_ESEI`, `NAL_PREFIX_NSEI`, `NAL_SUFFIX_ESEI`, or `NAL_SUFFIX_NSEI` carrying data associated with this group of samples.

7.2.5 Alternate track grouping

7.2.5.1 V3C content alternatives

V3C content may be encoded as different versions in a file. Different alternatives are indicated by the alternate track mechanism defined in ISO/IEC 14496-12 (i.e., `alternate_group` field of the `TrackHeaderBox`). V3C atlas tracks or V3C bitstream tracks which have the same `alternate_group` value shall be different versions of the same V3C content.

Annex F describes examples of using alternate V3C content.

7.2.5.2 V3C video component alternatives

V3C video component tracks may have alternatives. In such a case, only one of the V3C video component tracks that belong to an alternative group shall be referenced by the V3C atlas track or V3C atlas tile track. V3C video component tracks which are alternatives of each other should use the alternate grouping mechanism, as defined in ISO/IEC 14496-12.

Annex H describes examples of using alternate V3C video components.

7.2.6 Playout track grouping

7.2.6.1 General

When only some combination of tracks from the alternate versions of the V3C components should be played together, then playout group mechanism shall be used. Playout groups are signalled using `PlayoutTrackGroupBox`.

NOTE A track can be part of more than one playout group.

7.2.6.2 Definition

Box Types:	'potg'
Container:	<code>TrackGroupBox</code>
Mandatory:	No
Quantity:	Zero or more

Playout track groups are defined using the track group type `PlayoutTrackGroupBox`, which extends `TrackGroupBox` defined in ISO/IEC 14496-12. A `PlayoutTrackGroupBox` indicates that a track

belongs to a set of tracks constituting a playout group. Only tracks within the same playout group can be played out together. For each playout group that a track is a member of, a corresponding instance of `PlayoutTrackGroupBox` with the unique `track_group_id` for that playout group shall be present in the `TrackGroupBox` of that track.

7.2.6.3 Syntax

```
aligned(8) class PlayoutTrackGroupBox extends TrackGroupBox('potg') {
    // track_group_id is inherited from TrackGroupBox
}
```

7.3 Common track definitions

7.3.1 V3C bitstream track

7.3.1.1 V3C bitstream sample entry

7.3.1.1.1 Definition

A V3C bitstream track sample entry shall contain a `V3CConfigurationBox` and one or more 2D video configuration boxes, as defined in ISO/IEC 14496-15, can be present in the V3C bitstream sample entry to signal the 2D video decoder configuration and initialization information for the V3C video components. When a single 2D video configuration box is present, the corresponding 2D video decoder configuration and initialization information is applied to all V3C video components and a `DecoderConfigurationMappingBox` shall not be present.

An optional `BitRateBox` as defined in ISO/IEC 14496-12 may be present in the V3C bitstream sample entry to signal the bit rate information of the V3C bitstream track.

7.3.1.1.2 Syntax

```
aligned(8) class V3CBitstreamSampleEntry() extends VolumetricVisualSampleEntry (type) {
    V3CConfigurationBox config;
    //additional boxes
}
```

7.3.1.1.3 Semantics

`compressorname` in the base class `VolumetricVisualSampleEntry` indicates the name of the compressor used with the value "`\012V3C Coding`" being recommended; the first byte is a count of the remaining bytes, here represented by `\012`, which (being octal 12) is 10 (decimal), the number of bytes in the rest of the string.

7.3.1.2 V3C bitstream track sample format

7.3.1.2.1 Definition

A V3C bitstream sample shall contain one V3C composition unit which is a set of all sub-bitstream composition units that share the same presentation time and each sub-bitstream composition unit shall contain one or more V3C units which belong to a particular presentation time.

A sample may be self-contained (e.g., a sync sample) or decoding-wise dependent on other samples of V3C bitstream track.

7.3.1.2.2 Syntax

```
aligned(8) class V3CBitstreamSample {
    // sample_size size of sample from SampleSizeBox
    for (int i=0; i < sample_size; ) {
        unsigned int(v3c_config.unit_size_precision_bytes_minus1 + 1)*8) v3c_unit_size;
        bit(8) ss_v3c_unit[v3c_unit_size];
        i += v3c_unit_size + v3c_config.unit_size_precision_bytes_minus1 + 1;
    }
}
```

7.3.1.2.3 Semantics

`v3c_unit_size` specifies the size, in bytes, of the `ss_v3c_unit` array. The size is equivalent to the sample stream v3c unit size `ssvu_v3c_unit_size` as defined in ISO/IEC 23090-5, Annex C.

`ss_v3c_unit` contains a single V3C unit in V3C unit sample stream format as defined in ISO/IEC 23090-5:2021, Annex C.

7.3.1.3 V3C bitstream track sync sample

A V3C bitstream sync sample shall satisfy all the following conditions:

- It shall be independently decodable.
- None of the samples that come after the sync sample (in decoding order) have any decoding dependency on any sample prior to the sync sample.
- All samples that come after the sync sample (in decoding order) are successfully decodable.

7.3.1.4 V3C bitstream track sub-sample

A V3C bitstream track sub-sample is a V3C unit which is contained in a V3C bitstream track sample.

A V3C bitstream track may contain one `SubSampleInformationBox` in its `SampleTableBox`, or in the `TrackFragmentBox` of each of its `MovieFragmentBoxes`, which lists the V3C bitstream track sub-samples.

The 32-bit unit header of the V3C unit which represents the sub-sample shall be copied to the 32-bit `codec_specific_parameters` field of the sub-sample entry in the `SubSampleInformationBox`. The V3C unit type of each sub-sample shall be identified by parsing the `codec_specific_parameters` field of the sub-sample entry in the `SubSampleInformationBox`.

7.3.2 V3C atlas and atlas tile track

7.3.2.1 V3C atlas sample entry

7.3.2.1.1 Definition

`V3CAtlasSampleEntry` extends `VolumetricVisualSampleEntry`. An optional `BitRateBox` may be present in the V3C atlas sample entry to signal the bit rate information of the V3C atlas track.

7.3.2.1.2 Syntax

```
aligned(8) class V3CAtlasSampleEntry() extends VolumetricVisualSampleEntry (type) {
    V3CConfigurationBox config;
```

```

V3CUnitHeaderBox  unit_header;
V3CUnitHeaderBox  cad_unit_header; // optional
}

```

7.3.2.1.3 Semantics

`compressorname` in the base class `VolumetricVisualSampleEntry` indicates the name of the compressor used with the value "`\012V3C Coding`" being recommended; the first byte is a count of the remaining bytes, here represented by `\012`, which (being octal 12) is 10 (decimal), the number of bytes in the rest of the string.

`config` contains a single instance of `V3CConfigurationBox` as defined in subclause 7.2.1.

`unit_header` contains a single instance of `V3CUnitHeaderBox` representing the common atlas or atlas bitstream unit header as defined in subclause 7.2.2. The `vuh_unit_type` in the `V3CUnitHeaderBox` equals `V3C_AD`, or `V3C_CAD` when sample entry type '`v3cb`' is used.

`cad_unit_header` contains a single instance of `V3CUnitHeaderBox` representing the common atlas bitstream unit header as defined in subclause 7.2.2. The `vuh_unit_type` in the `V3CUnitHeaderBox` equals `V3C_CAD`. It is optionally present when both common atlas and atlas sub-bitstream are carried in the respective track.

7.3.2.2 V3C atlas tile sample entry

7.3.2.2.1 Definition

Sample Entry Type:	'v3t1'
Container:	SampleDescriptionBox
Mandatory:	Yes
Quantity:	One or more

A V3C atlas tile track uses `V3CAtlasTileSampleEntry` which extends `VolumetricVisualSampleEntry` with a sample entry type of '`v3t1`'.

A V3C atlas tile track shall only contain ACL NAL units corresponding to tiles, indicated by `tile_id` in `V3CAtlasTileConfigurationBox`.

The `V3CAtlasTileSampleEntry` shall not contain `V3CConfigurationBox` or `V3CUnitHeaderBox`. Information provided by these boxes is found in the V3C atlas track sample entry that references the V3C atlas tile track. Other optional boxes may be included.

7.3.2.2.2 Syntax

```

class V3CAtlasTileConfigurationBox extends FullBox('v3tC', version = 0, 0) {
    unsigned int(3) unit_size_precision_bytes_minus1;
    unsigned int(1) spatial_scalability_enabled_flag;
    bit(4) reserved = 0;
    if (spatial_scalability_enabled_flag) {
        unsigned int(8) lod_index;
    }
    unsigned int(16) num_tiles;
    for(int i=0; i < num_tiles; i++){
        unsigned int(16) tile_id;
    }
}

```

```
Shaligned(8) class V3CAtlasTileSampleEntry() extends VolumetricVisualSampleEntry ('v3t1')
{
    V3CAtlasTileConfigurationBox tile_info;
}
```

7.3.2.2.3 Semantics

`unit_size_precision_bytes_minus1` plus 1 specifies the precision, in bytes, of the sample stream NAL unit to which the sample entry containing this configuration box applies. The value of this field shall be equal to `ssnh_unit_size_precision_bytes_minus1` in `sample_stream_nal_header()` for the atlas component bitstream.

`spatial_scalability_enabled_flag` is a flag indicating whether the LoD-based scalability is supported by the carried V3C content.

`lod_index` indicates the LoD index value associated with the tiles carried by the atlas tile track. An atlas tile track with a certain LoD index (if present) should be selected with all atlas tile tracks with lower `lod_index` values carrying corresponding tiles. An LoD tile set associated with a lower `lod_index` value should be processed first.

`num_tiles` number of tiles contained in this track

`tile_id` specifies the tile ID of the tile present in the track. The value of `tile_id` is equal to value of `afti_tile_id` syntax element in atlas frame tile information, defined in ISO/IEC 23090-5.

`compressorname` in the base class `VolumetricVisualSampleEntry` indicates the name of the compressor used with the value "\017V3C Atlas Tiles" being recommended; the first byte is a count of the remaining bytes, here represented by \017, which (being octal 17) is 15 (decimal), the number of bytes in the rest of the string.

7.3.2.3 V3C atlas and V3C atlas tile sample format

7.3.2.3.1 Definition

Each sample in a V3C atlas track or V3C atlas tile track corresponds to a single coded atlas access unit.

NOTE When V3C atlas sample contains no reconstruction SEI message as defined in ISO/IEC 23090-5, it can be marked as non-output sample as defined in ISO/IEC 14496-12.

7.3.2.3.2 Syntax

```
aligned(8) class V3CAtlasSample {
    // sample_size value is the size of the sample from the SampleSizeBox
    int i = 0;
    while ( i < sample_size ) {
        unsigned int(v3c_config.unit_size_precision_bytes_minus1 + 1)*8) nal_size;
        bit(8) ss_nal_unit[nal_size];
        i += nal_size + v3c_config.unit_size_precision_bytes_minus1 + 1;
    }
}
```

7.3.2.3.3 Semantics

`nal_size` specifies the size, in bytes, of the `ss_nal_unit` array. This size is equivalent to the sample stream NAL unit size `ssnu_nal_unit_size` as defined in ISO/IEC 23090-5, Annex D.

`ss_nal_unit` is an array of data containing a single NAL unit as defined in ISO/IEC 23090-5.

NOTE Both, `nal_size` and `ss_nal_unit` replicate the sample stream NAL unit format `sample_stream_nal_unit` as defined in ISO/IEC 23090-5.

7.3.2.4 V3C atlas track and V3C atlas tile track sync sample

A sync sample in a V3C atlas track or V3C atlas tile track is a sample that contains an intra random access point (IRAP) coded atlas access unit as defined in ISO/IEC 23090-5.

NOTE Atlas parameter sets and SEI messages can be repeated, if needed, at a sync sample to allow for random access.

7.3.3 V3C video component track

A V3C video component track carries 2D video encoded data of V3C video component. The storage of V3C video component tracks utilizes the existing capabilities of the ISO base media file format and derived specifications, for example, ISO/IEC 14496-15 defines mechanisms for carriage of ISO/IEC 14496-10 and ISO/IEC 23008-2 coded V3C video components.

V3C video component tracks shall be represented in the file as restricted video and shall use a generic restricted sample entry '`resv`' with additional requirements:

- `SchemeTypeBox` is present in `RestrictedSchemeInfoBox` and `scheme_type` is set to '`vvvc`'.
- `SchemeInformationBox` is present in `RestrictedSchemeInfoBox` and contain a `V3CUnitHeaderBox`.
- In track header the `track_in_movie` flag is set to 0, to indicate that this track should not be presented alone.

NOTE There is no restriction on the video codec used for encoding V3C video components. Each V3C video component can be encoded using different video codecs.

7.3.4 Track references

7.3.4.1 General

To link a track with other tracks, the track reference tools defined in ISO/IEC 14496-12 shall be used.

7.3.4.2 Referencing V3C atlas tile tracks

To link a V3C atlas track to V3C atlas tile tracks with sample entry '`v3t1`', the 4CCs of the track reference type '`v3ct`' shall be used in the `TrackReferenceTypeBox` added to a `TrackReferenceBox` within the `TrackBox` of the V3C atlas track.

7.3.4.3 Referencing V3C video component tracks

To link a V3C atlas track or a V3C atlas tile track to video component tracks, one or more `TrackReferenceTypeBoxes` shall be added to a `TrackReferenceBox` within the `TrackBox` of the V3C atlas track or V3C atlas tile track, one for each component. The `TrackReferenceTypeBox` shall contain an array of `track_IDs` designating the video tracks which the V3C atlas track or V3C atlas tile track references. The `reference_type` of a `TrackReferenceTypeBox` identifies the type of the video component (i.e., occupancy, geometry, attribute, or packed). The 4CCs of these track reference types shall be:

- 'v3vo': the referenced track(s) contain the video-coded occupancy V3C component.
- 'v3vg': the referenced track(s) contain the video-coded geometry V3C component.
- 'v3va': the referenced track(s) contain the video-coded attribute V3C component.
- 'v3vp': the referenced track(s) contain the video-coded packed V3C component.

The type of the V3C component carried by the referenced restricted video track, and signalled in the `RestrictedSchemeInfoBox` of the track, shall match the reference type of the track reference from the V3C atlas track or V3C atlas tile track.

When 'v3ct' track reference is present in a V3C atlas track, 'v3va', 'v3vo', 'v3vg', 'v3vp' track references shall not be used in the V3C atlas track.

7.4 Timed V-PCC data storage in ISOBMFF

In this clause, a V3C bitstream refers to the bitstream which is coded with video-based point cloud coding (ISO/IEC 23090-5 Annex H). The V3C bitstream contains a set of video sub-bitstreams and atlas sub-bitstreams. This clause specifies the encapsulation of the V3C bitstream in tracks within a file.

7.4.1 Single-track encapsulation

7.4.1.1 General

A single-track encapsulation requires the V3C bitstream to be represented by a single-track declaration, referred to as a V3C bitstream track. The entire V3C bitstream shall be encapsulated into a single V3C bitstream track.

A single-track encapsulated V3C data could be provided to media workflows for further processing (e.g., multi-track file generation, transcoding, DASH segmentation, etc.).

More than one track for same V3C bitstream shall not be present. Any V3C video component tracks for the same V3C bitstream shall not be present.

7.4.1.2 V3C bitstream sample entry

7.4.1.2.1 Definition

Sample Entry Type:	'v3e1', 'v3eg'
Container:	SampleDescriptionBox
Mandatory:	A 'v3e1' or 'v3eg' sample entry is mandatory
Quantity:	One or more

The `V3CBitstreamSampleEntry` as defined in subclause 7.3.1.1 is used with with a sample entry type of 'v3e1' or 'v3eg'.

The following restrictions shall be applied:

- The value of `ptl_profile_toolset_idc` shall be in the range of 0 to 1, inclusive.
- under the 'v3e1' sample entry, the value of `array_completeness` shall be 1 for arrays containing atlas parameter sets.

- under the 'v3eg' sample entry, the value of `array_completeness` should be 0 for arrays containing atlas parameter sets.

Under the 'v3e1' sample entry, all atlas parameter sets and SEI messages, as defined in ISO/IEC 23090-5, shall be in the `setup_unit` array. Under the 'v3eg' sample entry, the atlas parameter sets and SEI messages may be present in the `setup_unit` array, or in the samples of the V3C bitstream track.

7.4.1.3 V3C bitstream track sample format

The `V3CBitstreamSample` as defined in subclause 7.3.1.2 is used.

7.4.1.4 V3C bitstream track sync sample

The V3C bitstream track sync sample as defined in subclause 7.3.1.3 is used.

7.4.1.5 V3C bitstream track sub-sample

The V3C bitstream track sub-sample as defined in subclause 7.3.1.4 is used.

7.4.2 Multi-track encapsulation

7.4.2.1 General

There may be three types of tracks in a multi-track encapsulated V3C data container: V3C atlas track, V3C atlas tile track, and V3C video component track. A multi-track encapsulated V3C data container shall include at least one V3C atlas track that references zero or more V3C atlas tile tracks or one or more V3C video component tracks. A V3C atlas tile track, when present, references one or more V3C video component tracks. The number of V3C video component tracks in a multi-track encapsulated V3C data container is dependent on the V3C toolset profile, defined in ISO/IEC 23090-5, that is used.

To indicate the association of V3C video component tracks to a V3C atlas track, or V3C atlas tile track, ISOBMFF track referencing is utilized, where the V3C atlas track, or V3C atlas tile track, contain track references to the V3C video component tracks.

Tracks belonging to the same CVS are time-aligned. Samples that contribute to the same volumetric frame across the different V3C video component tracks, V3C atlas track and V3C atlas tile tracks shall have the same composition time. Atlas parameter sets used for such samples shall have a decoding time equal or prior to the composition time of the volumetric frame. In addition, all tracks belonging to the same CVS shall have the same implied or explicit edit lists.

NOTE 1 Synchronization between the elementary streams in V3C atlas track, V3C atlas tile tracks and V3C video component tracks is handled by the ISOBMFF track timing structures (`stts`, `ctts`, and `cslg`), or equivalent mechanisms in movie fragments.

NOTE 2 The sync samples in the V3C atlas track, V3C atlas tile track and V3C video component tracks can be time-aligned. In the absence of time-alignment, random access can involve pre-rolling the various tracks from different sync start-times, to enable starting at the desired time. In the case of time-alignment (e.g., required by a V3C profile such as the V-PCC Basic toolset profile as defined in ISO/IEC 23090-5), the sync samples of the V3C atlas track can be considered as the random-access points for the V3C content, and random access can be done by only referencing the sync sample information of the V3C atlas track.

An example layout of a multi-track encapsulated V3C data container is shown in Figure 1. The boxes in the figure map to corresponding ISOBMFF boxes, as defined in ISO/IEC 14496-12. Payloads of V3C units of a V3C bitstream are mapped to individual tracks within the multi-track container file based on their types.

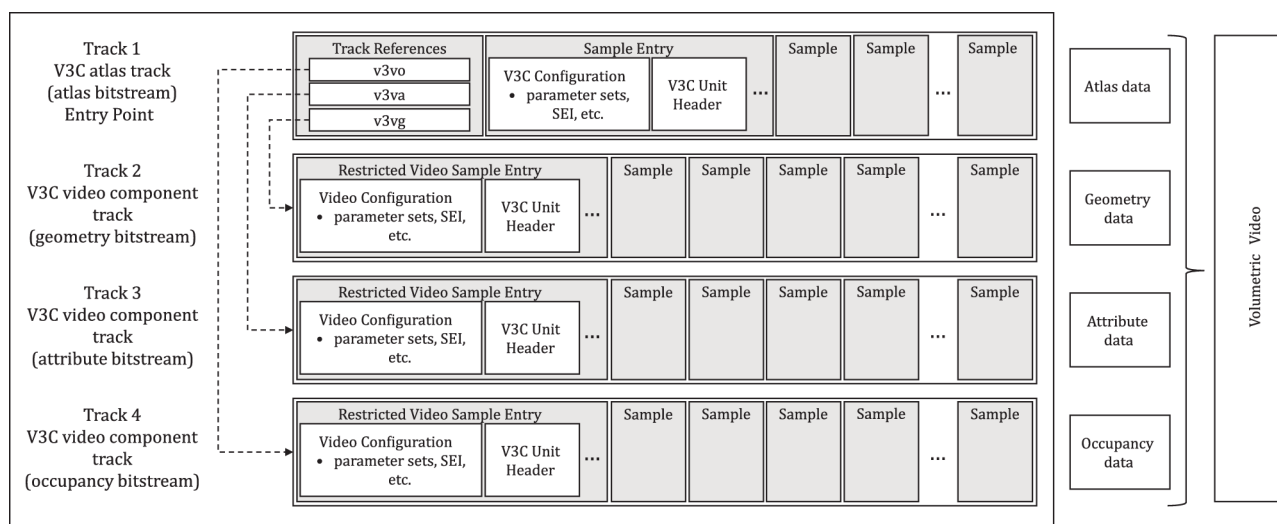


Figure 1 — Visualization of a multi-track encapsulation of V-PCC

Per content, a multi-track encapsulated V3C data container shall include the following:

- One V3C atlas track(s) which may include track references:
 - to other tracks carrying the payloads of video compressed V3C units (i.e., V3C unit types equal to V3C_OVD, V3C_GVD, V3C_AVD, or V3C_PVD as specified in ISO/IEC 23090-5), or
 - to V3C atlas tile tracks.
- Zero or more V3C video component tracks where the samples contain access units of a video-coded elementary stream for occupancy data (i.e., payloads of V3C units of type equal to V3C_OVD as specified in ISO/IEC 23090-5).
- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for geometry data (i.e., payloads of V3C units of type equal to V3C_GVD as specified in ISO/IEC 23090-5).
- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for attribute data (i.e., payloads of V3C units of type equal to V3C_AVD as specified in ISO/IEC 23090-5).
- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for packed data (i.e., payloads of V3C units of type equal to V3C_PVD as specified in ISO/IEC 23090-5).
- Zero or more V3C atlas tile tracks where the samples contain only ACL NAL units for a sub-set of atlas tiles. V3C atlas tile track may also contain track references to other tracks carrying the payloads of video compressed V3C units (i.e., V3C unit types equal to V3C_OVD, V3C_GVD, V3C_AVD, and V3C_PVD) for the indicated sub-set of atlas tiles.

7.4.2.2 V3C atlas and atlas tile track

7.4.2.2.1 V3C atlas sample entry

7.4.2.2.1.1 Definition

Sample Entry Type:	'v3c1', 'v3cg'
Container:	SampleDescriptionBox
Mandatory:	A 'v3c1', 'v3cg' sample entry is mandatory
Quantity:	One or more

A V3C atlas track uses the `V3CAtlasSampleEntry` as defined in subclause 7.3.2, with the following additional restrictions.

- Only sample entry types of 'v3c1' or 'v3cg' are used.
- A V3C atlas track sample entry contains a `V3CConfigurationBox`, as defined in subclause 7.2.1, and exactly one `V3CUnitHeaderBox`, as defined in subclause 7.2.2.
- The value of `ptl_profile_toolset_idc` in the `V3CConfigurationBox` shall be in the range of 0 to 1, inclusive.
- Each sample in a V3C atlas track corresponds to a single coded atlas access unit.
- `unit_header` contains a single instance of `V3CUnitHeaderBox` representing the atlas bitstream unit header as defined in subclause 7.2.2. The `vuh_unit_type` in the `V3CUnitHeaderBox` shall be equal to `V3C_AD`.
- V3C atlas tracks have the `track_in_movie` flag in track header is set to 1.

Depending on the V3C bitstream or sample entry type of the atlas track, following restrictions may be placed on V3C atlas tracks

- Under the 'v3c1' sample entry, the value of `array_completeness` shall be 1 for arrays containing atlas parameter sets.
- Under the 'v3cg' sample entry, the value of `array_completeness` should be 0 for arrays containing atlas parameter sets.

7.4.2.2.2 V3C atlas tile sample entry

A V3C atlas tile track uses `V3CAtlasTileSampleEntry` as defined in subclause 7.3.2.2.

7.4.2.2.3 V3C atlas sample format

`V3CAtlasSample` format is used as defined in subclause 7.3.2.3. Each sample in the V3C atlas track corresponds to a single coded atlas access unit associated with same `vuh_atlas_id` indicated in V3C unit header box in sample entry.

7.4.2.2.4 V3C atlas track and V3C atlas tile track sync sample

V3C atlas track and V3C atlas tile track sync samples as defined in subclause 7.3.2.4 are used.

7.4.2.3 V3C video component track

A V3C video component track as defined in subclause 7.3.3 is used with the following restrictions.

- The `vuh_unit_type` in the `V3CUnitHeaderBox` present in the `SchemeInformationBox` is equal to one of `V3C_OVD`, `V3C_GVD`, `V3C_AVD`, or `V3C_PVD`.

7.4.2.3.1 Multimap video box

7.4.2.3.1.1 Definition

Box Type:	'mmvi'
Container:	<code>SchemeInformationBox</code>
Mandatory:	No
Quantity:	Zero or one

The `MultiMapVideoBox` is used to indicate that decoded video frames contain two or more temporarily interleaved video frames that represent maps. This box shall not be present in the case of single-track encapsulation of V3C content.

When `MultiMapVideoBox` is present, it indicates that a temporal interleaving map packing arrangement is used. File parsers should implicitly set the composition time for `map_count_minus1 + 1` consecutive samples to be equal to that of the first sample in the group of samples in the interleaved map packing arrangement.

When temporal interleaving map packing arrangement is used, each sync sample and each SAP sample with `SAP_type` in the range of 1 to 3, inclusive, indicated by the stream access point sample group, represent an iterative arrangement of samples representing map 0 to `map_count_minus1`, in composition time order, up to but excluding the next sync sample or SAP sample with `SAP_type` in the range of 1 to 3, inclusive, indicated by the stream access point sample group.

7.4.2.3.1.2 Syntax

```
aligned(8) class MultiMapVideoBox extends FullBox('mmvi', version = 0, 0){
    bit(4) reserved = 0;
    unsigned int(4) map_count_minus1;
}
```

7.4.2.3.1.3 Semantics

`map_count_minus1` plus 1 indicates the number of maps present in the track as consecutive samples. This value shall not be equal to 0.

7.4.2.4 Track references

7.4.2.4.1 Referencing V3C atlas tile tracks

The '`v3ct`' track reference as defined in subclause 7.3.4.2 shall be used in a V3C atlas track with sample entries '`v3c1`' or '`v3cg`' to V3C atlas tile tracks with sample entry '`v3t1`'.

7.4.2.4.2 Referencing V3C video component tracks

The '`v3va`', '`v3vo`', '`v3vg`', or '`v3vp`' track references as defined in subclause 7.3.4.3 shall be used to link a V3C atlas track with sample entries '`v3c1`' or '`v3cg`', or a V3C atlas tile track with sample entry '`v3t1`' to video component tracks.

7.4.2.5 Summary

Table 10 provides a summary of the sample entry types for tracks carrying atlas data defined in this subclause.

Table 10 — Summary of sample entry types for V3C atlas tracks and atlas tile tracks

Sample entry type		'v3c1'	'v3cg'	'v3t1'
V3C atlas sample entry	VPS	yes	yes	N/A
	Atlas parameter sets	yes	may	N/A
	V3C unit header box	yes	yes	N/A
Atlas parameter set sample group		may	may	N/A
Track references		'v3vo' 'v3vg' 'v3va' 'v3vp' 'v3ct'	'v3vo' 'v3vg' 'v3va' 'v3vp' 'v3ct'	'v3vo' 'v3vg' 'v3va'
Sample		ACL + non-ACL ^a	ACL + non-ACL ^a	ACL
^a When 'v3t1' track is present, samples of 'v3c1' and 'v3cg' track only contains non-ACL NAL units. If 'v3t1' track is not present, samples shall also contain ACL NAL units. yes = mandatory, may = optional, N/A = not applicable				

7.5 Timed MIV data storage in ISOBMFF

In this clause, a V3C bitstream refers the bitstream which is coded with MPEG immersive video (ISO/IEC 23090-12). The V3C bitstream contains a set of video sub-bitstreams and atlas sub-bitstreams. This clause specifies the encapsulation of the V3C bitstream in tracks within a file.

7.5.1 Single-track encapsulation

7.5.1.1 General

A single-track encapsulation requires the V3C bitstream to be represented by a single-track declaration, referred to as a V3C bitstream track. The entire V3C bitstream shall be encapsulated into a single V3C bitstream track.

A single-track encapsulated V3C data could be provided to media workflows for further processing (e.g., multi-track file generation, transcoding, DASH segmentation, etc.).

More than one track for same V3C bitstream shall not be present. Any V3C video component tracks for the same V3C bitstream shall not be present.

7.5.1.2 V3C bitstream sample entry

7.5.1.2.1 Definition

Sample Entry Type:	'v3e1', 'v3eg'
Container:	SampleDescriptionBox
Mandatory:	A 'v3e1' or 'v3eg' sample entry is mandatory
Quantity:	One or more

The `V3CBitstreamSampleEntry` as defined in subclause 7.3.1.1 is used with a sample entry type of 'v3e1' or 'v3eg'.

The following restrictions shall be applied:

- The value of `ptl_profile_toolset_idc` shall be in the range of 64 to 66, inclusive.
- under the 'v3e1' sample entry, the value of `array_completeness` shall be 1 for arrays containing atlas parameter sets.
- under the 'v3eg' sample entry, the value of `array_completeness` should be 0 for arrays containing atlas parameter sets.

Under the 'v3e1' sample entry, all atlas parameter sets and SEI messages, as defined in ISO/IEC 23090-5, shall be in the `setup_unit` array. Under the 'v3eg' sample entry, the atlas parameter sets and SEI messages may be present in the `setup_unit` array, or in the samples of the V3C bitstream track.

7.5.1.3 V3C bitstream track sample format

The `V3CBitstreamSample` as defined in subclause 7.3.1.2 is used.

7.5.1.4 V3C bitstream track sync sample

The V3C bitstream track sync sample as defined in subclause 7.3.1.3 is used.

7.5.1.5 V3C bitstream track sub-sample

The V3C bitstream track sub-sample as defined in subclause 7.3.1.4 is used.

7.5.2 Multi-track encapsulation

7.5.2.1 General

There may be three types of tracks in a multi-track encapsulated V3C data container: V3C atlas track, V3C atlas tile track, and V3C video component track. A multi-track encapsulated V3C data container shall include at least one V3C atlas track that references zero or more V3C atlas tile tracks or one or more V3C video component tracks. A V3C atlas tile track, when present, references one or more V3C video component tracks. The number of V3C video component tracks in a multi-track encapsulated V3C data container is dependent on the V3C toolset profile, defined in ISO/IEC 23090-12, that is used.

To indicate the association of V3C video component tracks to a V3C atlas track, or V3C atlas tile track, ISOBMFF track referencing is utilized, where the V3C atlas track, or V3C atlas tile track, contain track references to the V3C video component tracks.

Tracks belonging to the same CVS are time-aligned. Samples that contribute to the same volumetric frame across the different V3C video component tracks, V3C atlas track and V3C atlas tile tracks shall have the

same composition time. Atlas parameter sets used for such samples shall have a decoding time equal or prior to the composition time of the volumetric frame. In addition, all tracks belonging to the same CVS shall have the same implied or explicit edit lists.

NOTE 1 Synchronization between the elementary streams in V3C atlas track, V3C atlas tile tracks and V3C video component tracks is handled by the ISOBMFF track timing structures (stts, cts, and cslg), or equivalent mechanisms in movie fragments.

NOTE 2 The sync samples in the V3C atlas track, V3C atlas tile track and V3C video component tracks can be time-aligned. In the absence of time-alignment, random access can involve pre-rolling the V3C video component tracks from different sync start-times, to enable starting at the desired time. In the case of time-alignment (e.g., required by a V3C profile such as the MIV Main toolset profile as defined in ISO/IEC 23090-12), the sync samples of the V3C atlas track can be considered as the random-access points for the V3C content.

An example layout of a multi-track encapsulated V3C data container is shown in Figure 2. The boxes in the figure map to corresponding ISOBMFF boxes, as defined in ISO/IEC 14496-12. Payloads of V3C units of a V3C bitstream are mapped to individual tracks within the multi-track container file based on their types.

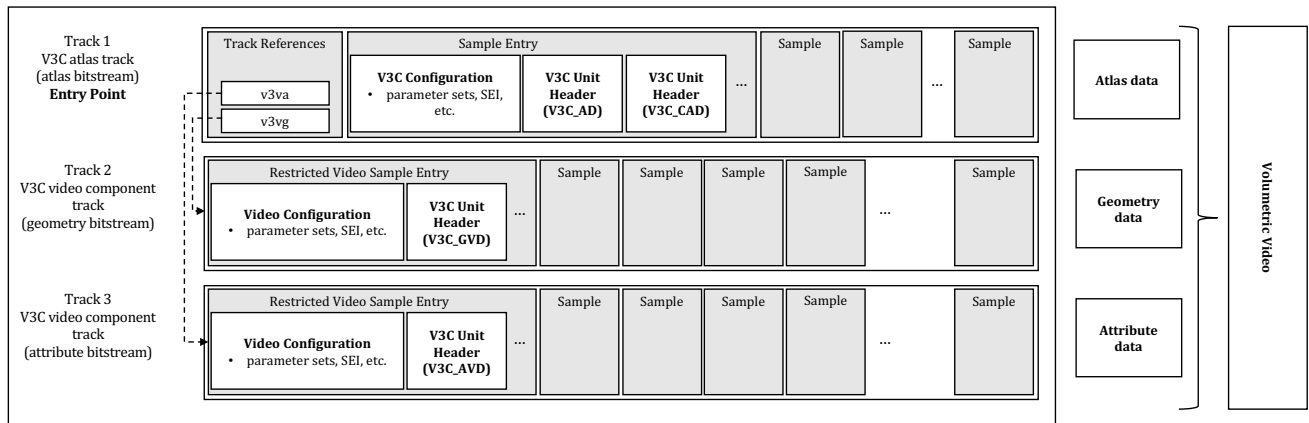


Figure 2 — Visualization of a multi-track encapsulation of MIV

Per content, a multi-track encapsulated V3C data container shall include the following:

- One V3C atlas track(s) which may include track references:
 - to other tracks carrying the payloads of video compressed V3C units (i.e., V3C unit types equal to `V3C_OVD`, `V3C_GVD`, `V3C_AVD`, or `V3C_PVD` as specified in ISO/IEC 23090-5)
 - to V3C atlas tile tracks
 - to other V3C atlas tracks when multiple atlases are present in the bitstream .
- zero or more V3C video component tracks where the samples contain access units of a video-coded elementary stream for occupancy data (i.e., payloads of V3C units of type equal to `V3C_OVD` as specified in ISO/IEC 23090-5).
- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for geometry data (i.e., payloads of V3C units of type equal to `V3C_GVD` as specified in ISO/IEC 23090-5).
- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for attribute data (i.e., payloads of V3C units of type equal to `V3C_AVD` as specified in ISO/IEC 23090-5).

- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for packed data (i.e., payloads of V3C units of type equal to V3C_PVD as specified in ISO/IEC 23090-5).
- Zero or more V3C atlas tile tracks where the samples contain only ACL NAL units for a sub-set of atlas tiles. V3C atlas tile track may also contain track references to other tracks carrying the payloads of video compressed V3C units (i.e., V3C unit types equal to V3C_OVD, V3C_GVD, V3C_AVD, and V3C_PVD) for the indicated sub-set of atlas tiles.

7.5.2.2 V3C atlas and atlas tile track

7.5.2.2.1 V3C atlas sample entry

7.5.2.2.1.1 Definition

Sample Entry Type:	'v3c1', 'v3cg', 'v3cb', 'v3a1', or 'v3ag'
Container:	SampleDescriptionBox
Mandatory:	A 'v3c1', 'v3cg', 'v3cb', 'v3a1', or 'v3ag' sample entry is mandatory
Quantity:	One or more

A V3C atlas track uses the `V3CAtlasSampleEntry` as defined in subclause 7.3.2, with the following additional restrictions:

- A V3C atlas track shall not carry ACL NAL units belonging to more than one atlas.
- A V3C atlas track sample entry contains one `V3CConfigurationBox`, as defined in subclause 7.2.1, and one or two `V3CUnitHeaderBox`, as defined in subclause 7.2.2.
- The value of `ptl_profile_toolset_idc` in the `V3CConfigurationBox` shall be in the range of 64 to 66, inclusive.
- When the V3C bitstream contains a single atlas, a V3C atlas track with sample entry 'v3c1' or 'v3cg' shall be used.
- When the V3C bitstream contains multiple atlases, each atlas bitstream shall be stored as a separate V3C atlas track with the sample entry type 'v3a1', or 'v3ag'. One additional track with the sample entry type 'v3cb' shall be present, which is the entry point track referencing the other V3C atlas tracks with the sample entry type 'v3a1', or 'v3ag'.

Depending on the V3C bitstream or sample entry type of the atlas track, following restrictions are placed on V3C atlas tracks:

- When the sample entry type is 'v3c1' or 'v3cg', the sample entry contains two instances of `V3CUnitHeaderBox`. Each sample in the track correspond to a single coded atlas access unit and/or coded common atlas access unit.
- When the sample entry type is 'v3a1' or 'v3ag', the sample entry contains one `V3CUnitHeaderBox`. Each sample in the track correspond to a single coded atlas access unit.
- When the sample entry type is 'v3cb', the sample entry contains one `V3CUnitHeaderBox`. Each sample in the track correspond to a single coded common atlas access unit.

- Under the 'v3a1' and 'v3ag' sample entry, the `num_of_v3c_parameter_sets` shall be equal to 0. The V3C parameter set shall be stored in the sample entry of the atlas track with 'v3cb'.
- The samples of a V3C atlas track with sample entry type 'v3cb' shall not include any ACL NAL units.
- Under the 'v3c1' and 'v3a1' sample entry, the value of `array_completeness` shall be 1 for arrays containing atlas parameter sets.
- Under the 'v3cg' and 'v3ag' sample entry, the value of `array_completeness` should be 0 for arrays containing atlas parameter sets.
- The parameter sets and SEI messages present in a V3C atlas track with 'v3cb' sample entry apply to all referenced V3C atlas tracks.
- For tracks with sample entry types 'v3c1', 'v3cg' or 'v3cb' the `track_in_movie` flag in track header is set to 1.
- For tracks with sample entry types 'v3a1', or 'v3ag', the `track_in_movie` flag in track header is set to 0.

NOTE V3C atlas tracks with sample entry type 'v3cb', 'v3a1' and 'v3ag' are relevant for carrying multiple atlases.

7.5.2.2.2 V3C atlas tile sample entry

A V3C atlas tile track uses the `V3CAtlasTileSampleEntry` as defined in subclause 7.3.2.2, with the following additional restrictions:

- All V3C atlas tile track samples shall contain only ACL NAL units, which belong to the same atlas.
- The `spatial_scalability_enabled_flag` is set to 0, to indicate that the LoD-based scalability is not supported.

7.5.2.2.3 V3C atlas sample format

The `V3CAtlasSample` format is used as defined in subclause 7.3.2.3 with the following clarifications:

- When 'v3cb' sample entry is used, each sample in the V3C atlas track corresponds to one or more non-ACL NAL units.
- When 'v3c1', 'v3cg', 'v3a1' or 'v3ag' sample entry is used, each sample in the V3C atlas track corresponds to a coded atlas access unit associated with same `juh_atlas_id` indicated in V3C unit header box in sample entry.

7.5.2.2.4 V3C atlas track and V3C atlas tile track sync sample

V3C atlas track and V3C atlas tile track sync samples as defined in subclause 7.3.2.4 are used.

7.5.2.3 V3C video component track

V3C video component track as defined in subclause 7.3.3 is used with the following restrictions.

- The `juh_unit_type` in the `V3CUnitHeaderBox` present in the `SchemeInformationBox` is equal to one of `V3C_OVD`, `V3C_GVD`, `V3C_AVD`, and `V3C_PVD`.

7.5.2.4 Track references

7.5.2.4.1 Referencing V3C atlas tracks

To link a V3C atlas track with sample entry 'v3cb' to V3C atlas tracks with sample entries 'v3a1' or 'v3ag', the 4CCs of the track reference type 'v3cs' shall be used in the `TrackReferenceTypeBox`.

7.5.2.4.2 Referencing V3C atlas tile tracks

The 'v3ct' track reference as defined in subclause 7.3.4.2 shall be used in a V3C atlas track with sample entries 'v3c1', 'v3cg', 'v3a1' or 'v3ag' to V3C atlas tile tracks with sample entry 'v3t1'.

7.5.2.5 Referencing V3C video component tracks

The 'v3va', 'v3vo', 'v3vg', or 'v3vp' track references as defined in subclause 7.3.4.3 shall be used to link a V3C atlas track with sample entries 'v3c1', 'v3cg', 'v3a1' or 'v3ag', or a V3C atlas tile track with sample entry 'v3t1' to video component tracks.

7.5.2.6 Summary

Table 11 provides a summary of the sample entry types for tracks carrying atlas data defined in this document.

Table 11 — Summary of sample entry types for V3C atlas tracks and atlas tile tracks

Sample entry type		'v3c1'	'v3cg'	'v3cb'	'v3a1'	'v3ag'	'v3t1'
V3C atlas sample entry	VPS	yes	yes	yes	no	no	N/A
	Atlas parameter sets	yes	may	yes	yes ^a	may ^a	N/A
	V3C unit header box	Two	Two	One	One	One	N/A
Atlas parameter set sample group		may	may	no	may (partially ^b)	may (partially ^b)	N/A
Track references		'v3vo' 'v3vg' 'v3va' 'v3vp' 'v3ct'	'v3vo' 'v3vg' 'v3va' 'v3vp' 'v3ct'	'v3cs'	'v3vo' 'v3vg' 'v3va' 'v3vp' 'v3ct'	'v3vo' 'v3vg' 'v3va' 'v3vp' 'v3ct'	'v3vo' 'v3vg' 'v3va'
Sample		ACL + non-ACL ^c	ACL + non-ACL ^c	non-ACL	ACL + non-ACL ^c	ACL + non-ACL ^c	ACL

^a It only includes atlas component sub-bitstream parameter sets associated with same `vuh_atlas_id` indicated in V3C unit header box.

^b Atlas component sub-bitstream parameter sets associated with same `vuh_atlas_id` indicated in V3C unit header box are included.

^c When 'v3t1' track is present, samples of 'v3c1', 'v3cg', 'v3a1' and 'v3ag' track only contains non-ACL NAL units. If 'v3t1' track is not present, samples shall also contain ACL NAL units.

yes = mandatory, may = optional, no = disallowed, N/A = not applicable

7.6 Timed V-DMC data storage in ISOBMFF

In this clause, a V3C bitstream refers to the bitstream which is coded with video-based dynamic mesh coding (ISO/IEC 23090-29). The V3C bitstream contains V3C component sub-bitstreams, such as atlas, basemesh, displacement, and attribute components. This clause specifies the encapsulation of the V3C bitstream in tracks within a file.

7.6.1 Data structures and definitions for V-DMC storage

7.6.1.1 Basemesh decoder configuration box

7.6.1.1.1 Definition

A basemesh decoder configuration box contains the basemesh decoder configuration record, as defined in subclause 6.2.2. It provides the basemesh decoding specific information for configuring and initialization of the basemesh decoder. At least one basemesh sequence parameter set NAL unit shall be present in the basemesh decoder configuration record.

7.6.1.1.2 Syntax

```
aligned(8) class BaseMeshConfigurationBox
    extends FullBox('vbmC', version = 0, 0) {
        BaseMeshDecoderConfigurationRecord bm_config;
    }
```

7.6.1.1.3 Semantics

`bm_config` contains a single instance of `BaseMeshDecoderConfigurationRecord` defined in subclause 6.2.2.

7.6.1.2 Arithmetic coded displacement decoder configuration box

7.6.1.2.1 Definition

An arithmetic coded displacement decoder configuration box contains the arithmetic coded displacement decoder configuration record, as defined in subclause 6.2.3. It provides the decoding specific information for arithmetic coded displacement sub-bitstream. At least one displacement sequence parameter set NAL unit shall be present in the arithmetic coded displacement decoder configuration record.

7.6.1.2.2 Syntax

```
aligned(8) class ACDisplacementConfigurationBox
    extends FullBox('vdcC', version = 0, 0) {
        ACDisplacementDecoderConfigurationRecord disp_config;
    }
```

7.6.1.2.3 Semantics

`disp_config` contains a single instance of `ACDisplacementDecoderConfigurationRecord` defined in subclause 6.2.3.

7.6.1.3 Instanced rendering box

7.6.1.3.1 Definition

Sample Entry Type:	'vire'
Container:	SampleEntry ('v3c1', 'v3cg')
Mandatory:	No
Quantity:	One

A single `InstancedRenderingBox` may present in the sample entry of a V3C atlas track to indicate that the mesh representation associated with the atlas track may be used for instanced rendering.

7.6.1.3.2 Syntax

```
aligned(8) class InstancedRenderingBox extends FullBox('vire', version = 0, 0){
    unsigned int(1) can_be_temporally_instanced;
    unsigned int(1) can_be_texture_instanced;
    unsigned int(6) reserved;
}
```

7.6.1.3.3 Semantics

`can_be_temporally_instanced`, when equal to 1, indicates that the mesh information associated with the atlas track may be used for temporal instancing.

`can_be_texture_instanced`, when equal to 1, indicates that the mesh information associated with the atlas track may be used for texture instancing.

7.6.1.4 Instanced attribute variant track grouping

7.6.1.4.1 General

For instanced rendering, multiple versions of an attribute track may exist that can be indexed using the same set of UV-coordinates and the same mesh. For such applications, instanced attribute track group offers the needed information to identify tracks that can be considered as alternative attributes for instanced drawing. An application may choose to use zero or more tracks from the group and the group does not mandate any rendering behavior. It merely informs that there is an option for instanced drawing with multiple different attribute variants.

7.6.1.4.2 Definition

Sample Entry Type:	'iavg'
Container:	TrackGroupBox
Mandatory:	No
Quantity:	Zero or more

Instanced attribute variant track groups are defined using `InstancedAttributeVariantTrackGroupBox`, which extends `TrackGroupBox` as defined in ISO/IEC 14496-12. The tracks belonging to the track group are considered as alternative attributes for instanced rendering and shall contain information that can be indexed using the same set of UV-coordinates. Zero or more tracks of the group may be used at any given time. All tracks within the track

group shall be time-aligned, meaning that the samples that contribute to the same volumetric frame across the tracks in the track group shall have the same composition time. This track group shall be present in the file, when `InstancedRenderingBox` is present in the sample entry of a V3C atlas track and, when `can_be_texture_instanced` equals 1 in the `InstancedRenderingBox`. All tracks in the track group shall have the same attribute type.

7.6.1.4.3 Syntax

```
aligned(8) class InstancedAttributeVariantTrackGroupBox extends TrackGroupBox('iavg')
{
    // track_group_id is inherited from TrackGroupBox
}
```

7.6.2 Single-track encapsulation

7.6.2.1 General

A single-track encapsulation of a V3C bitstream requires that the entire bitstream is carried by a single V3C bitstream track.

More than one track for same V3C bitstream shall not be present. Any V3C video component tracks for the same V3C bitstream shall not be present.

7.6.2.2 V3C bitstream sample entry

7.6.2.2.1 Definition

Sample Entry Type:	'vdm1', 'vdmg'
Container:	SampleDescriptionBox
Mandatory:	A 'vdm1' or 'vdmg' sample entry is mandatory
Quantity:	One or more

A V3C bitstream track shall use `VolumetricVisualSampleEntry` with a sample entry type of 'vdm1' or 'vdmg'.

The V3C bitstream sample entry with a sample entry type 'vdm1' and 'vdmg' shall contain a `V3CConfigurationBox` as defined in subclause 7.2.1 containing the V3C bitstream's decoding specific information, and a `BaseMeshConfigurationBox` containing base mesh decoder configuration and initialization information. A `ACDisplacementConfigurationBox` may be present.

When a V3C bitstream contains video sub-bitstream(s) of V3C components, the corresponding video decoder configuration box(es), as defined in ISO/IEC 14496-15, are present in the V3C bitstream sample entry to signal the 2D video decoder configuration and initialization information for decoding video-based V3C components.

The following restrictions shall apply to the V3C bitstream sample entry with a sample entry type 'vdm1' and 'vdmg':

- The value of `pt1_profile_toolset_idc` in the `V3CConfigurationBox` shall be in the range of 128 to 143, inclusive.
- under the 'vdm1' sample entry, the default and mandatory value of `array_completeness` in `V3CConfigurationBox` and `BaseMeshConfigurationBox` is 1 for arrays of all types of atlas

parameter sets and basemesh parameter sets, respectively, and 0 for arrays of all other types. When `ACDisplacementConfigurationBox` is present, the default and mandatory value of its `array_completeness` is 1 for arrays of displacement parameter sets and 0 for arrays for all other types.

- under the 'vdmg' sample entry, the default value of `array_completeness` in `V3CConfigurationBox` and `BaseMeshConfigurationBox` is 0 for all arrays. When `ACDisplacementConfigurationBox` is present, the default value of its `array_completeness` is 0 for all arrays.

7.6.2.2.2 Syntax

```
aligned(8) class V3CBitstreamSampleEntry()
    extends VolumetricVisualSampleEntry ('vdm1' or 'vdmg') {
        V3CConfigurationBox  config;
        BaseMeshConfigurationBox  bmesh_config;
        ACDisplacementConfigurationBox  displ_config; //optional
        //additional boxes
    }
```

7.6.2.2.3 Semantics

`config` provides the V3C bitstream's decoding specific information, as defined in subclause 7.2.1.

`bmesh_config` contains the basemesh decoder configuration and initialization information, as defined in subclause 7.6.1.1.

`displ_config` contains the decoding specific information for arithmetic coded displacement sub-bitstream, as defined in subclause 7.6.1.2.

7.6.2.3 V3C bitstream track sample format

The `V3CBitstreamSample` as defined in subclause 7.3.1.2 is used.

7.6.2.4 V3C bitsream track sync sample

The V3C bitstream track sync sample shall satisfy all conditions as defined in subclause 7.3.1.3.

7.6.2.5 V3C bitstream track sub-sample

The syntax and semantics of the V3C bitstream sub-sample as defined in subclause 7.3.1.4 is applied.

7.6.3 Multi-track encapsulation

7.6.3.1 General

There may be four types of tracks in a multi-track encapsulated V3C data container: V3C atlas track, V3C atlas tile track, V3C video component track, and V3C non-video component track.

A multi-track encapsulated V3C data container shall include at least one V3C atlas track that references zero or more V3C atlas tile tracks, zero or more V3C video component tracks, and zero or more non-video component (e.g., base mesh or arithmetic coded displacement) tracks. A V3C atlas tile track, when present, references zero or more V3C video component tracks and zero or more V3C non-video component tracks. The number of V3C video and non-video component tracks in a multi-track encapsulated V3C data container is dependent on the V3C toolset profile, defined in ISO/IEC 23090-5, that is used.

Tracks belonging to the same CVS are time-aligned. Samples that contribute to the same mesh frame across the different V3C video and non-video component tracks, V3C atlas track and V3C atlas tile tracks shall have the same composition time. Atlas parameter sets used for such samples shall have a decoding time equal or prior to the composition time of the mesh frame. In addition, all tracks belonging to the same CVS shall have the same implied or explicit edit lists.

A multi-track encapsulated V-DMC data container shall include the following:

- A V3C atlas track which contains a V3C parameter set and may contain atlas parameter sets in the sample entry and atlas component bitstream NAL units in the samples. A V3C atlas track may also include track references to other tracks carrying the payloads of video compressed V3C units (i.e., V3C unit types equal to V3C_GVD and V3C_AVD, or to non-video compressed V3C units (i.e., V3C unit types equal to V3C_BMD or V3C_ADD).
- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for geometry data (i.e., payloads of V3C units of type equal to V3C_GVD).
- Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for attribute data (i.e., payloads of V3C units of type equal to V3C_AVD). Zero or more V3C video component tracks where the samples contain access units of video-coded elementary streams for packed data (i.e., payloads of V3C units of type equal to V3C_PVD).
- Zero or more V3C non-video component tracks where the samples contain access units of elementary streams for non-video data (e.g., payloads of V3C units of type equal to V3C_BMD or V3C_ADD).
- Zero or more V3C atlas tile tracks where the samples contain only ACL NAL units for a sub-set of atlas tiles. V3C atlas tile track may also contain track references to other tracks carrying the payloads of video and non-video compressed V3C units (i.e., V3C unit types equal to V3C_GVD, V3C_AVD, V3C_BMD, and V3C_ADD) for the indicated sub-set of atlas tiles.

An overview of the structure for encapsulating timed V3C (V-DMC) data in a multi-track encapsulation is illustrated in Figure 3.

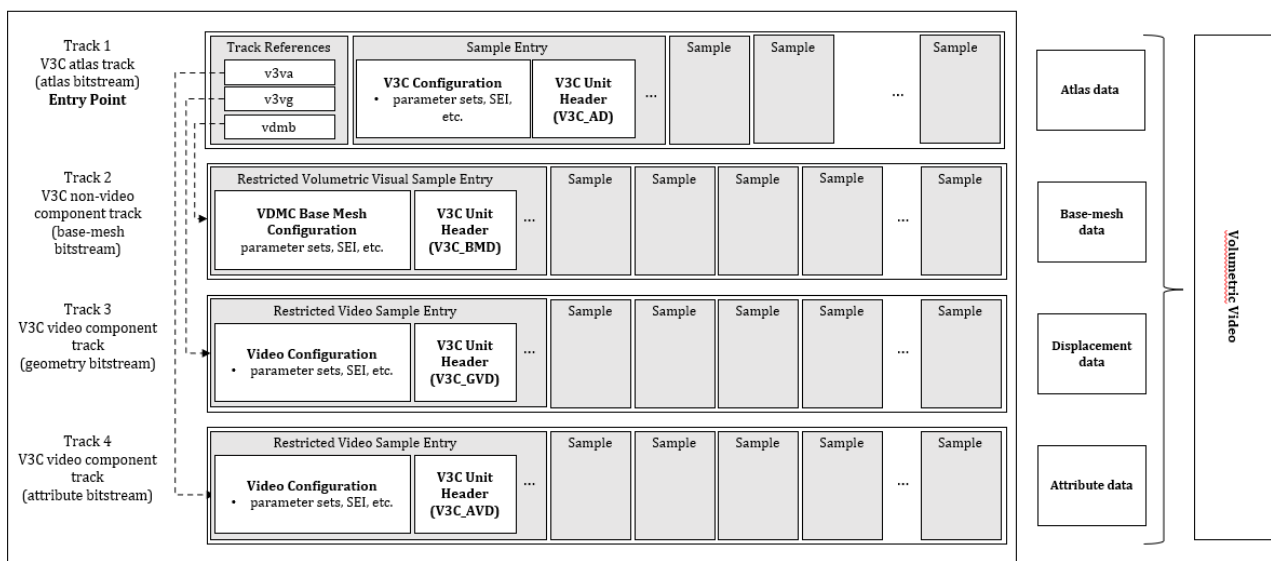


Figure 3 —Example layout of a multi-track encapsulation of timed V3C data containing V-DMC.**7.6.3.2 V3C atlas and atlas tile track****7.6.3.2.1 V3C atlas sample entry****7.6.3.2.1.1 Definition**

Sample Entry Type:	'v3c1', 'v3cg'
Container:	SampleDescriptionBox
Mandatory:	A 'v3c1', 'v3cg' sample entry is mandatory
Quantity:	One or more

A V3C atlas track uses the `V3CAtlasSampleEntry` as defined in subclause 7.3.2, with the following additional restrictions:— Only sample entry types of 'v3c1' or 'v3cg' are used.

- A V3C atlas track sample entry contains a `V3CConfigurationBox`, as defined in subclause 7.2.1, and exactly one `V3CUnitHeaderBox`, as defined in subclause 7.2.2.
- The value of `ptl_profile_toolset_idc` in the `V3CConfigurationBox` shall be in the range of 128 to 143, inclusive.
- Each sample in a V3C atlas track correspond to a single coded atlas access unit.
- `unit_header` contains a single instance of `V3CUnitHeaderBox` representing the atlas bitstream unit header as defined in subclause 6.2.3. The `vuh_unit_type` in the `V3CUnitHeaderBox` shall be equal to `V3C_AD`.
- V3C atlas tracks have the `track_in_movie` flag in track header is set to 1.

Depending on the V3C bitstream or sample entry type of the atlas track, following restrictions may be placed on V3C atlas tracks

- Under the 'v3c1' sample entry, the value of `array_completeness` shall be 1 for arrays containing atlas parameter sets.
- Under the 'v3cg' sample entry, the value of `array_completeness` should be 0 for arrays containing atlas parameter sets.

7.6.3.2.2 V3C atlas tile sample entry

V3C atlas tile tracks use `V3CAtlasTileSampleEntry` as defined in subclause 7.3.2.2.

7.6.3.2.3 V3C atlas sample format

`V3CAtlasSample` format is used as defined in subclause 7.3.2.3.

7.6.3.2.4 V3C atlas track and V3C atlas tile track sync sample

V3C atlas track and V3C atlas tile track sync samples as defined in subclause 7.3.2.4 are used.

7.6.3.3 V3C video component track

V3C video component track as defined in subclause 7.3.3 is used, with the following restrictions.

- The `vuh_unit_type` in the `V3CUnitHeaderBox` present in the `SchemeInformationBox` is equal to one of `V3C_GVD`, `V3C_AVD`, and `V3C_PVD`.

7.6.3.4 Basemesh and submesh track

Basemesh and submesh sub-bitstreams shall be represented as restricted volumetric visual tracks and shall use a generic restricted `VolumetricVisualSampleEntry` with the 4CC 'res3'. The restricted volumetric visual tracks follow the same transformation process as restricted video tracks with 'resv'. Restricted volumetric visual tracks include a `RestrictedSchemeInfoBox` with additional requirements:

- `SchemeTypeBox` is present in `RestrictedSchemeInfoBox` and `scheme_type` is set to 'vvbm'.
- In a basemesh track, `SchemeInformationBox` is present in `RestrictedSchemeInfoBox` and contains a `V3CUnitHeaderBox`.
- In the track header the `track_in_movie` flag is set to 0, to indicate that this track should not be presented alone.

7.6.3.4.1 Basemesh sample entry

7.6.3.4.1.1 Definition

Sample Entry Type:	'bmcb', 'bmcl' or 'bmcg'
Container:	<code>SampleDescriptionBox</code>
Mandatory:	A 'bmcb', 'bmcl' or 'bmcg' sample entry is mandatory
Quantity:	One or more

A basemesh track use `BaseMeshSampleEntry` which extends `VolumetricVisualSampleEntry` with a sample entry type of 'bmcb', 'bmcl' or 'bmcg'.

The basemesh track sample entry contains a `BaseMeshConfigurationBox`. A base-mesh track shall not carry BMCL NAL units belonging to more than one atlas. Following restrictions are set for basemesh tracks:

- When the basemesh bitstream is stored in a single track, a basemesh track with sample entry type 'bmcl' or 'bmcg' shall be used.
- When the basemesh bitstream contains multiple submeshes stored in multiple submesh tracks, a basemesh track with sample entry type 'bmcb' shall be used and submesh bitstreams shall be stored in one or more submesh tracks with sample entry type 'smcl'.
- Under the 'bmcb' sample entry, the basemesh track shall not include any BMCL NAL units.
- Under the 'bmcl' sample entry, all basemesh parameter sets shall be stored in the `setup_unit` array. SEI messages that apply to the stream as a whole may be stored in the `setup_unit` array as well.
- Under the 'bmcg' sample entry, the parameter sets and SEI messages may be present in the `setup_unit` array, or in the samples of basemesh track.

7.6.3.4.1.2 Syntax

```
aligned(8) class BaseMeshSampleEntry()
    extends VolumetricVisualSampleEntry ('bmcb', 'bmcl' or 'bmcg') {
```

```

BaseMeshConfigurationBox bmesh_config;
MPEG4ExtensionDescriptorsBox descr; // optional
}

```

7.6.3.4.1.3 Semantics

bmesh_config contains a single instance of BaseMeshConfigurationBox as defined in subclause 7.6.1.1.

7.6.3.4.2 Submesh sample entry

7.6.3.4.2.1 Definition

Sample Entry Type:	'smc1'
Container:	SampleDescriptionBox
Mandatory:	Yes
Quantity:	One or more

A basemesh can contain one or more submeshes. A submesh or a group of submeshes can be carried in a separate track that is a submesh track with sample entry type 'smc1'. A submesh track sample shall contain only BMCL NAL units, which belong to the same basemesh.

A track reference type 'bmcs' is used to indicate the relation from the basemesh track to the associated submesh tracks.

7.6.3.4.2.2 Syntax

```

aligned(8) class SubMeshSampleEntry
    extends VolumetricVisualSampleEntry('smc1') {
        SubMeshConfigurationBox sm_config;
    }
class SubMeshConfigurationBox extends FullBox('smcC', version = 0, 0) {
    unsigned int(6) num_submeshes_minus1;
    unsinged int(2) reserved;
    for(int i=0; i < num_submeshes_minus1; i++){
        unsigned int(24) submesh_id;
    }
}

```

7.6.3.4.2.3 Semantics

num_submeshes_minus1 indicates the number of submeshes contained in this track. This value shall be in range from 0 to 63, inclusive.

submesh_id is the identifier for the submesh present in this track. The value of submesh_id is equal to the value of the corresponding bmsi_submesh_id syntax element in bmesh_submesh_information(), defined in ISO/IEC 23090-29:Annex H. This value shall be in range from 0 to $2^{24}-1$, inclusive.

7.6.3.4.3 Basemesh sample format

7.6.3.4.3.1 Definition

Each sample in a basemesh track or submesh track corresponds to a single coded basemesh access unit. **[Ed(SOH) : more clarifications need to be specified according to sample entry type]**

7.6.3.4.3.2 Syntax

```
aligned(8) class BaseMeshSample {
    // sample_size value is the size of the sample from the SampleSizeBox.  int i = 0;
    while (i < sample_size) {
        unsigned int(bm_config.unit_size_precision_bytes_minus1 + 1)*8) nal_size;

        bit(8) ss_nal_unit[nal_size];
        i += nal_size + bm_config.unit_size_precision_bytes_minus1 + 1
    }
}
```

7.6.3.4.3.3 Semantics

`nal_size` specifies the size, in bytes, of the `ss_nal_unit` array. This size is equivalent to the sample stream NAL unit size `ssnu_nal_unit_size` as defined in ISO/IEC 23090-29 Annex D.

`ss_nal_unit` is an array of data containing a single BMCL or non-BMCL NAL unit as defined in 23090-29:Annex H.

7.6.3.4.3.4 Basemesh track and submesh track sync sample

A sync sample in a basemesh track is a sample that contains an intra random access point (IRAP) coded base-mesh access unit as defined in ISO/IEC 23090-29.

7.6.3.4.3.5 Basemesh track and submesh track sub-sample

In the case where more than one submesh is carried in the basemesh track or submesh track, a sample in a basemesh track or a submesh track includes a set of submeshes and each sub-sample contains one submesh. When more than one submesh is carried in the basemesh track or submesh track, one `SubSampleInformationBox` may be present in its `SampleTableBox`, or in the `TrackFragmentBox` of each of its `MovieFragmentBoxes` in the basemesh track or submesh track.

For the use of the `SubSampleInformationBox` in a basemesh track or a submesh track, a sub-sample is defined based on the value of the `flags` field of the `SubSampleInformationBox`. The `flags` specifies the type of sub-sample information given in this box as follows:

- If the value of the `flags` field is 0, each sub-sample contains one Submesh NAL unit in the group of submeshes carried in a sample of the Basemesh track.
- The values other than 0 are reserved for future use.

The `subsample_priority` field shall be set to a value in accordance with the specification of this field in ISO/IEC 14496-12.

The `discardable` field shall be set to 1 only if this sample is still decodable if this sub-sample is discarded.

When the `SubSampleInformationBox` is present in a basemesh track or a submesh track, the `codec_specific_parameters` field in the box shall have the semantics as follows:

```
if (flags == 0) {
    unsigned int(24) submesh_id;
    bit(8) reserved = 0;
}
```

The semantics of the above fields are:

`submesh_id` indicates the identifier for a submesh contained in this sub-sample of the basemesh track.
`submesh_id` shall be equal to the submesh identifier signaled in the syntax element `bmsi_submesh_id` syntax element. This value shall be in range from 0 to $2^{24}-1$, inclusive.

7.6.3.4.4 Submesh track group

7.6.3.4.4.1 General

Each submesh track can include one or more submeshes and these submeshes are related to one or more atlas tiles carried in one or more atlas tile tracks. To signal the relation between atlas tile tracks and associated submesh tracks, a track group `SubMeshTrackGroupBox` extends `TrackGroupTypeBox` defined in ISO/IEC 14496-12, is defined.

7.6.3.4.4.2 Syntax

```
aligned(8) class SubMeshTrackGroupBox extends TrackGroupTypeBox('smtg') {
    // track_group_id is inherited from TrackGroupTypeBox
    unsigned int(16) num_tiles;
    for(int i=0; i < num_tiles; i++) {
        unsigned int(16) tile_id;
        unsigned int(16) num_submeshes;
        for(int j=0; j < num_submeshes; j++) {
            unsigned int(16) submesh_id;
        }
    }
}
```

7.6.3.4.4.3 Semantics

`num_tiles` indicates the number of atlas tiles associated with this track group.

`tile_id` specifies the atlas tile ID of an atlas tile associated with the submeshes for this track group instance. The value of `tile_id` is equal to value of `afti_tile_id` syntax element in atlas frame tile information, defined in ISO/IEC FDIS 23090-5

`num_submeshes` indicates the number of submeshes associated with the the atlas tile.

`submesh_id` is the identifier for the submesh for this track group instance. The value of `submesh_id` is equal to the value of the corresponding `bmsi_submesh_id` syntax element in `bmesh_submesh_information()`, defined in ISO/IEC 23090-29:Annex H.

7.6.3.5 Displacement track

7.6.3.5.1 General

If the displacement component is coded using a 2D video codec, a track which carries the video-coded displacement sub-bitstream shall be defined as a V3C video component track defined in subclause 7.3.4.3.

If the displacement component is coded using an arithmetic codec, the track which carries the arithmetic coded displacement sub-bitstream shall be represented as a restricted volumetric visual track and shall use a generic restricted `VolumetricVisualSampleEntry` with the 4CC 'res3' which includes a `RestrictedSchemeInfoBox` with additional requirements:

- `SchemeTypeBox` is present in `RestrictedSchemeInfoBox` and `scheme_type` is set to 'vvbm'.

- `SchemeInformationBox` is present in `RestrictedSchemeInfoBox` and contain a `V3CUnitHeaderBox`.
- In track header the `track_in_movie` flag is set to 0, to indicate that this track should not be presented alone.

7.6.3.5.2 Arithmetic coded displacement sample entry

An arithmetic coded displacement track sample entry includes a `ACDisplacementConfigurationBox`.

7.6.3.5.3 Arithmetic coded displacement sample format

Each sample in an arithmetic coded displacement track corresponds to a single coded displacement access unit.

7.6.3.5.3.1 Syntax

```
aligned(8) class ACDisplacementSample {
    int i = 0;
    while(i < sample_size) {
        unsigned int(disp_config.unit_size_precision_bytes_minus1 + 1)*8) nal_size;
        bit(8)ss_nal_unit[nal_size];
        i += nal_size+disp_config.unit_size_precision_bytes_minus1 + 1
    }
}
```

7.6.3.5.3.2 Semantics

`nal_size` specifies the size, in bytes, of the `ss_nal_unit` array. This size is equivalent to the sample stream NAL unit size `ssnu_nal_unit_size` as defined in ISO/IEC 23090-29:Annex D.

7.6.3.6 `ss_nal_unit` is an array of data containing a single DCL or non-DCL NAL unit as defined in 23090-29:Annex J. Track references

7.6.3.6.1 Referencing V3C atlas tile tracks

The '`v3ct`' track reference as defined in subclause 7.3.4.2 shall be used in a V3C atlas track with sample entries '`v3c1`' or '`v3cg`' to V3C atlas tile tracks with sample entry '`v3t1`'.

7.6.3.6.2 Referencing V3C component tracks

To indicate the association of V3C component tracks to a V3C atlas track, or V3C atlas tile track, the `reference_type` of a `TrackReferenceTypeBox` identifies the type of the V-DMC component. The '`v3va`', '`v3vg`', or '`v3vp`' track references as defined in subclause 7.3.4.3 shall be used with the following additions.

- '`vdmb`': the referenced track(s) contain the basemesh component
- '`vdmd`': the referenced track(s) contain the arithmetic-coded displacement component.

This clause shall apply the following restriction.

- The '`v3vo`' 4CCs of track referency types shall not be present.

7.6.3.6.3 Referencing submesh tracks

To link a basemesh track with sample entry 'bmcb' to submesh tracks with sample entry 'smc1', the 4CCs of these track reference types 'bmcs' shall be used in the `TrackReferenceTypeBox` added to a `TrackReferenceBox` within the `TrackBox` of the basemesh track.

7.6.3.7 Summary

Table 12 provides a summary of the sample entry types for tracks carrying atlas data defined in this subclause.

Table 12 — Summary of sample entry types for V3C atlas tracks and atlas tile tracks

Sample entry type		'v3c1'	'v3cg'	'v3t1'
V3C atlas sample entry	VPS	yes	yes	N/A
	Atlas parameter sets	yes	may	N/A
	V3C unit header box	yes	yes	N/A
Atlas parameter set sample group		may	may	N/A
Track references		'v3vg' 'v3va' 'v3vp' 'v3ct' 'vdmb' 'vdmd'	'v3vg' 'v3va' 'v3vp' 'v3ct' 'vdmb' 'vdmd'	'v3vg' 'v3va'
Sample		ACL + non-ACL ^a	ACL + non-ACL ^a	ACL
^a When 'v3t1' track is present, samples of 'v3c1' and 'v3cg' track only contains non-ACL NAL units. If 'v3t1' track is not present, samples shall also contain ACL NAL units. yes = mandatory, may = optional, N/A = not applicable				

8 Carriage of non-timed visual volumetric video-based coding data

8.1 General

This clause specifies a format to encapsulate a non-timed V3C data, which contains a single volumetric frame, in a file as ISO/IEC 14496-12 items. The single V3C frame is encapsulated into one or more items as defined in this clause.

In this clause, one or more items corresponding to one V3C bitstream shall be stored in the same file-level Metabox.

8.2 Common data structures and definitions

8.2.1 V3C configuration item property

8.2.1.1 Definition

Box Types:	'v3cC'
Property type:	Descriptive item property
Container:	ItemPropertyContainerBox
Mandatory (per item):	specified by the mapping of V3C application to this document
Quantity (per item):	specified by the mapping of V3C application to this document

The V3C configuration item property provides V3C decoder configuration and initialization information. In this version of this document, only one V3C parameter set is present in the V3C configuration item property.

The V3C configuration item property is an essential since it is required to initialize the decoder. The corresponding essential flag in the `ItemPropertyAssociationBox` shall be set to 1 for a 'v3cC' item property.

8.2.1.2 Syntax

```
aligned(8) class V3CConfigurationProperty extends ItemProperty('v3cC', version=0, flags) {
    V3CDecoderConfigurationRecord v3c_config(version);
}
```

8.2.1.3 Semantics

`v3c_config` contains a single instance of `V3CDecoderConfigurationRecord` which is defined in subclause 6.2.1.

8.2.2 V3C unit header item property

8.2.2.1 Definition

Box Types:	'vutp'
Property type:	Descriptive item property
Container:	ItemPropertyContainerBox
Mandatory (per item):	Yes, for a V3C atlas item and a V3C component item
Quantity (per item):	One

The V3C unit header item property contains the V3C unit header describing the data carried by the associated item. It shall be associated with the V3C atlas items and the V3C component items.

The corresponding essential flag in the `ItemPropertyAssociationBox` shall be set to 1 for a 'vutp' item property.

8.2.2.2 Syntax

```
aligned(8) class V3CUnitHeaderProperty() extends ItemFullProperty('vutp', version=0, 0) {
    bit(8) header[4];
}
```

8.2.2.3 Semantics

`header` contains a single instance of the 32-bit V3C unit header syntax structure as defined in ISO/IEC 23090-5.

8.2.3 V3C atlas tile configuration item property

8.2.3.1 Definition

Box Types:	'v3tp'
Property type:	Descriptive item property
Container:	ItemPropertyContainerBox
Mandatory (per item):	Yes, for a V3C atlas tile item
Quantity (per item):	One

The V3C atlas tile configuration item property provides the number of atlas tiles and the identifier of tiles which are present in the associated atlas tile item.

The V3C atlas tile configuration item property is an essential property. The corresponding `essential` flag in the `ItemPropertyAssociationBox` shall be set to 1 for a 'v3tp' item property.

8.2.3.2 Syntax

```
aligned(8) class V3CAtlasTileConfigurationProperty() extends ItemFullProperty('v3tp',
version=0, 0) {
    unsigned int(16) num_tiles;
    for(int i=0; i < num_tiles; i++) {
        unsigned int(16) tile_id;
    }
}
```

8.2.3.3 Semantics

`num_tiles` indicates the number of tiles contained in related V3C atlas tile item.

`tile_id` indicates the tile ID of the tile contained in associated V3C atlas tile item. The value of `tile_id` is equal to value of `afti_tile_id` syntax element in atlas frame tile information, as defined in ISO/IEC 23090-5.

8.2.4 Sub-sample item property

8.2.4.1 Definition

Box type:	'subs'
Property type:	Descriptive item property
Container:	ItemPropertyContainerBox
Mandatory (per item):	No
Quantity (per item):	Zero or one for a V3C item

Sub-sample information for a coded V3C frame may be given using an associated item property that is exactly identical to `SubSampleInformationBox` as defined in ISO/IEC 14496-12.

The `entry_count` field of the `SubSampleInformationBox` shall be equal to 1, and the `sample_delta` field of the `SubSampleInformationBox` shall be equal to 0.

The 32-bit unit header of the V3C unit which represents the sub-sample shall be copied to the 32-bit `codec_specific_parameters` field of the sub-sample entry in the `SubSampleInformationBox`. The V3C unit type of each sub-sample is identified by parsing the `codec_specific_parameters` field of the sub-sample entry in the `SubSampleInformationBox`.

8.2.5 Playout entity group

8.2.5.1 General

When only some combination of items, from the alternate versions of the V3C component items should be played together, then playout group mechanism shall be used. Playout groups are signalled using `PlayoutEntityToGroupBox`.

NOTE An item can be part of more than one playout group.

8.2.5.2 Definition

Box type:	'eplv'
Container:	<code>GroupListBox</code>
Mandatory (per item):	No
Quantity (per item):	Zero or more

An `EntityToGroupBox` with `grouping_type` equal to 'eplv' specifies items that shall be played out together. An alternate group of entities consists of items that are mapped to the same entity group of type 'altr'. Only one item within an alternate group should be played or streamed at any one time. A `PlayoutEntityToGroupBox` is used to group non-timed (item) V3C data in the same group.

8.2.5.3 Syntax

```
aligned(8) class PlayoutEntityToGroupBox extends EntityToGroupBox('eplv') {
}
```

8.3 Common item definitions

8.3.1 V3C item

8.3.1.1 Definition

A V3C item is an item which represents a single volumetric frame and contains one V3C composition unit which is a set of all sub-bitstream composition units and each sub-bitstream composition unit contains one or more V3C units of the V3C frame.

8.3.1.2 Syntax

The `V3CItemData` is structurally identical to the syntax for V3C bitstream track sample format as defined in subclause 7.3.1.2.2

```
aligned(8) class V3CItemData {
    for (int i=0; i < item_size; ) { // derived from ItemLocationBox
        unsigned int(v3c_config.unit_size_precision_bytes_minus1 + 1)*8) v3c_unit_size;
        bit(8) ss_v3c_unit[v3c_unit_size];
        i += v3c_unit_size + v3c_config.unit_size_precision_bytes_minus1 + 1;
    }
}
```

8.3.1.3 Semantics

For `V3CItemData` the following applies:

- The value of `item_size` is equal to the sum of the `extent_length` values of each extent of the item, as specified in the `ItemLocationBox`.
- `v3c_config` indicates the record in the associated `V3CConfigurationProperty`.

`v3c_unit_size` specifies the size, in bytes, of the `ss_v3c_unit` array. This size is equivalent to the sample stream V3C unit size `ssnu_v3c_unit_size` as defined in ISO/IEC 23090-5, Annex C.

`ss_v3c_unit` contains a single V3C unit in V3C unit sample stream format as defined in ISO/IEC 23090-5, Annex C.

8.3.2 V3C atlas item

8.3.2.1 Definition

A V3C atlas item contains an independently decodable coded atlas access unit or coded common atlas access unit depending on item type. Item type 4CC codes 'v3c1', 'v3cb', and 'v3a1' identify V3C atlas items.

8.3.2.2 Syntax

`V3CAtlasItemData` is structurally identical to the syntax for an V3C atlas sample.

```
aligned(8) class V3CAtlasItemData
{
    for (i=0; i<item_size; ){ // derived from ItemLocationBox
        unsigned int(v3c_config.unit_size_precision_bytes_minus1 + 1)*8) nal_size;
        bit(8) ss_nal_unit[nal_size];
        i += nal_size + v3c_config.unit_size_precision_bytes_minus1 + 1;
    }
}
```

8.3.2.3 Semantics

In the syntax above, the following applies:

- The value of `item_size` is equal to the sum of the `extent_length` values of each extent of the item, as specified in the `ItemLocationBox`.
- `v3c_config` indicates the record in the associated V3C configuration property.

`nal_size` specifies the size, in bytes, of the `ss_nal_unit` array. This size is equivalent to the sample stream NAL unit size `ssnu_nal_unit_size` as defined in ISO/IEC 23090-5, Annex D.

`ss_nal_unit` is an array of data containing a single NAL unit as defined in ISO/IEC 23090-5 : —, Annex D.

NOTE Both, `nal_size` and `ss_nal_unit` replicate the sample stream NAL unit format `sample_stream_nal_unit` as defined in ISO/IEC 23090-5.

8.3.3 V3C atlas tile item

8.3.3.1 Definition

A V3C atlas tile item is an item of type 'v3t1' that contains one or more ACL NAL units which belong to the same atlas.

Each V3C atlas tile item shall be associated with one `V3CAtlasTileConfigurationProperty`. The `V3CAtlasTileConfigurationProperty` shall indicate the atlas tile IDs of tiles present in the V3C atlas tile item.

8.3.3.2 Syntax

The syntax of V3C atlas tile item data is defined as `V3CAtlasItemData` described in 8.3.2.2.

8.3.3.3 Semantics

The semantics of V3C atlas tile item data as defined in 8.3.2.3 are used.

8.3.4 V3C image component item

A V3C image component item is an item which represents visual V3C component data. A V3C component item shall store only one access unit of related video component data.

An item type 4CC code for a V3C component item shall be set depending on the codec used to encode corresponding video components. A V3C component item shall be associated with corresponding V3C unit header item property and codec specific configuration item property.

The V3C component items shall be marked as hidden items.

8.3.5 Item references

In order to indicate the relationship between V3C atlas items and V3C atlas tile items, a 'v3ct' item reference is used to link the V3C atlas item to V3C atlas tile items.

In order to indicate the association between a V3C atlas item and V3C component items or between V3C atlas tile item and V3C component items, item reference is defined “from” either a V3C atlas item or a V3C atlas tile item “to” the related V3C component items. The 4CC codes of item reference types shall be:

- 'v3vo': the referenced V3C component item(s) contain the occupancy data.
- 'v3vg': the referenced V3C component item(s) contain the geometry data.
- 'v3va': the referenced V3C component item(s) contain the attribute data.
- 'v3vp': the referenced V3C component item(s) contain the packed video data.

8.4 Non-timed V-PCC data storage in ISOBMFF

In this clause, non-timed V3C data refers a single access unit of V3C bitstream which is coded with video-based point cloud coding (ISO/IEC 23090-5 Annex H).

This clause defines the encapsulation of non-timed V3C data in items within a file. Two methods for encapsulation are defined 1) storage as a single item, where all sub-bitstreams of the V3C bitstream are stored in a single item, and 2) storage as multiple items, where each item stores a different sub-bitstream.

A handler type in the `HandlerBox` of the `MetaBox` shall be 4CC code 'volv'.

8.4.1 Single-item encapsulation

This clause specifies how to encapsulate a non-timed V3C data in a single item.

8.4.1.1 V3C item

The V3C item as defined in subclause 8.3.1 is used with the following restrictions.

- Only an item of type 'v3e1' shall be used.
- An item of type 'v3e1' shall have an associated `V3CConfigurationProperty`.
- Sub-sample information for V3C items may be given using an associated sub-sample item property.

If `PrimaryItemBox` exists, `item_ID` in this box shall be set to indicate an item of type 'v3e1'.

8.4.2 Multi-item encapsulation

This clause specifies how encapsulate non-timed V3C data into multiple items. Three item types called V3C atlas item, V3C atlas tile item and V3C component item are used.

An overview of the structure for encapsulating non-timed V3C data in a single atlas with a single atlas tile is illustrated in Figure 3.

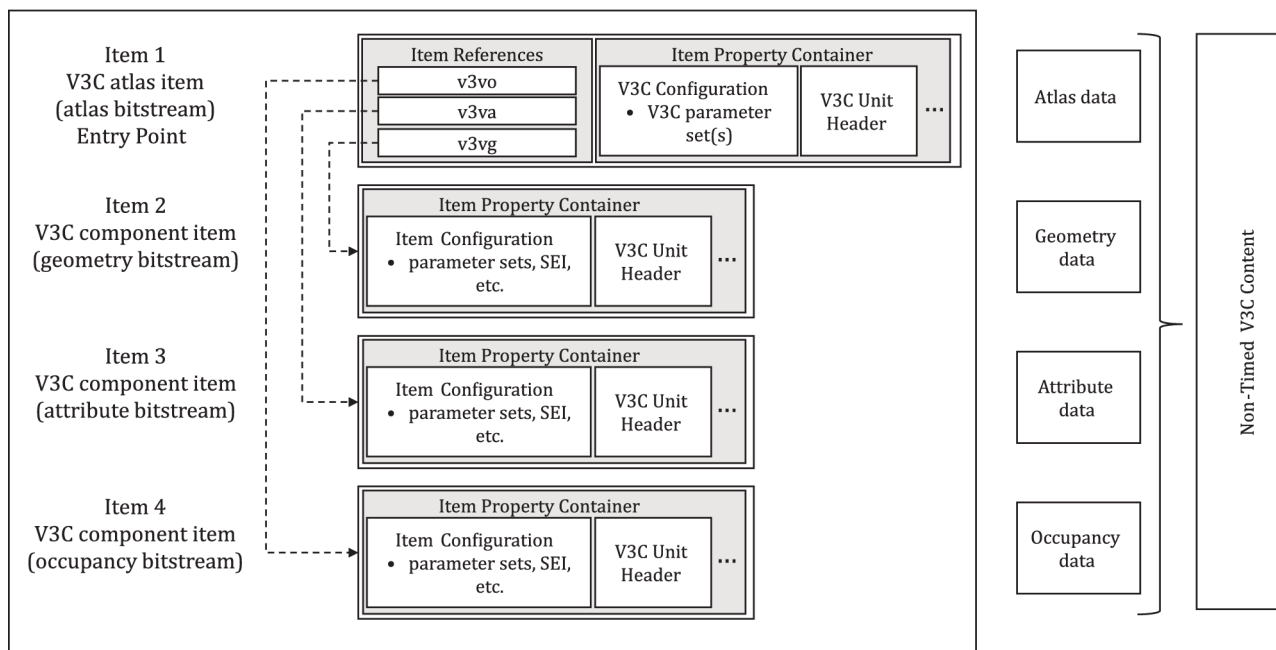


Figure 3 — Overview of structure for encapsulating non-timed V3C data

8.4.2.1 V3C atlas item

The V3C atlas item as defined in subclause 8.3.2 is used with the following restrictions:

- Only an item of type 'v3c1' shall be used.
- An item of type 'v3c1' shall have an associated `V3CUnitHeaderProperty`.
- An item of type 'v3c1' shall be associated with one `V3CConfigurationProperty`.

- If an item of type 'v3c1' is associated with V3C atlas tile items, the item of type 'v3c1' stores one or more non-ACL NAL units associated with the V3C atlas tile items. Otherwise, the item of type 'v3c1' stores one or more ACL or non-ACL NAL units.

If `PrimaryItemBox` exists, `item_ID` in this box shall be set to indicate an item of type 'v3c1'.

8.4.2.2 V3C atlas tile item

V3C atlas tile item as defined in subclause 8.3.3 is used.

8.4.2.3 V3C image component item

V3C image component item as defined in subclause 8.3.4 is used with the following restrictions.

- The V3C image component item stores V3C unit payload of V3C unit of type `V3C_OVD`, `V3C_GVD`, `V3C_AVD` and `V3C_PVD`.

8.4.2.4 Item references

The 'v3ct' item reference as defined in subclause 8.3.5 shall be used to link from the V3C atlas item to V3C atlas tile items.

The 'v3va', 'v3vo', 'v3vg', or 'v3vp' item references as defined in subclause 8.3.5. shall be used to link V3C atlas items or V3C atlas tile items to V3C image component items.

8.5 Non-timed MIV data storage in ISOBMFF

In this clause, non-timed V3C data refers a single access unit of V3C bitstream which is coded with MPEG Immersive Video (ISO/IEC 23090-12).

This clause defines the encapsulation of non-timed V3C data in items within a file. Two methods for encapsulation are defined 1) storage as a single item, where all sub-bitstreams of the V3C bitstream are stored in a single item, and 2) storage as multiple items, where each item stores a different sub-bitstream.

A handler type in the `HandlerBox` of the `MetaBox` shall be 4CC code 'volv'.

8.5.1 Single-item encapsulation

This clause specifies a format to encapsulate a non-timed V3C data in a single item.

8.5.1.1 V3C item

The V3C item as defined in subclause 8.3.1 is used with the following restrictions.

- Only an item of type 'v3e1' is used.
- An item of type 'v3e1' shall have an associated `V3CConfigurationProperty`.
- Sub-sample information for V3C items may be given using an associated sub-sample item property.

If `PrimaryItemBox` exists, `item_ID` in this box shall be set to indicate an item of type 'v3e1'.

8.5.2 Multi-item encapsulation

This clause specifies how encapsulate non-timed V3C data into multiple items. Three item types called V3C atlas item, V3C atlas tile item and V3C component item are used.

An overview of the structure for encapsulating non-timed V3C data which contains a single atlas, a geometry image, and an attribute image component is illustrated in Figure 4.

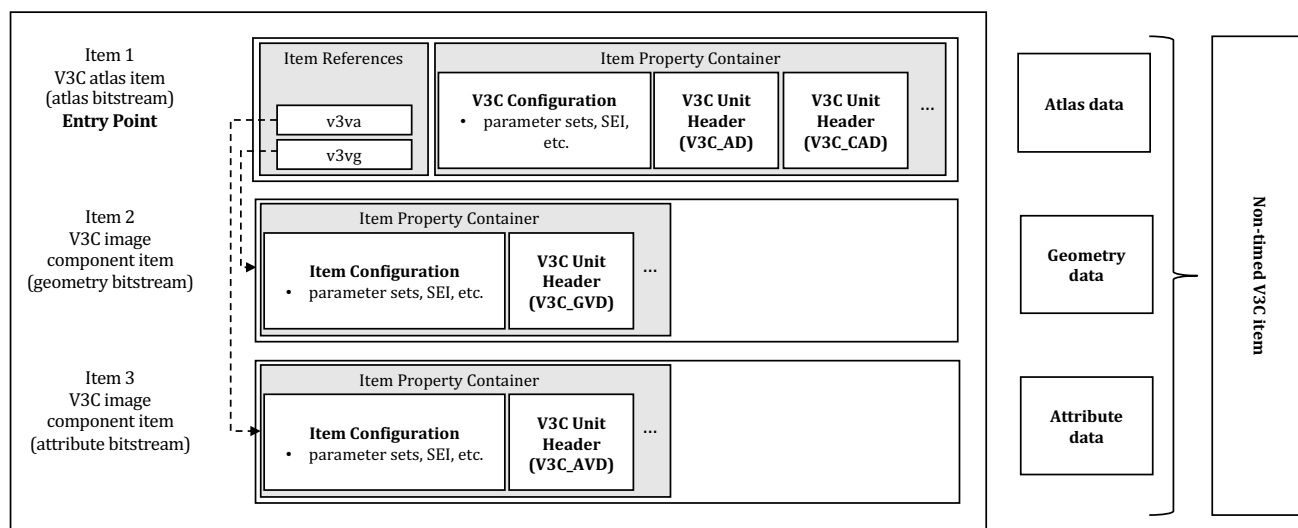


Figure 4. Overview of structure for encapsulating non-timed V3C data in a single atlas

8.5.2.1 V3C atlas item

V3C atlas item as defined in subclause 8.3.1 is used with the following restrictions:

- Items of type 'v3c1', 'v3cb', or 'v3a1' shall be used.
- Items of type 'v3c1' or 'v3cb' shall be associated with one `V3CConfigurationProperty`.
- When a single atlas exists, an item of type 'v3c1' shall be used. An item of type 'v3c1' shall be associated with two `V3CUnitHeaderProperty`. If an item of type 'v3c1' is associated with V3C atlas tile items, the item of type 'v3c1' item stores one or more non-ACL NAL units associated with the V3C atlas tile items. Otherwise, the item of type 'v3c1' stores one or more ACL or non-ACL NAL units.
- When multiple atlases exist, the common atlas data is stored in an item of type 'v3cb' and each atlas data shall be stored in separate item of type 'v3a1'. An item of type 'v3cb' shall not include any ACL NAL units. Items of type 'v3a1' shall be associated with one `V3CConfigurationProperty` and `num_of_v3c_parameter_sets` shall be equal to 0 in the associated `V3CConfigurationProperty`.
- If an item of type 'v3a1' is associated with V3C atlas tile items, the item of type 'v3a1' contains one or more non-ACL NAL units associated with same `juh_atlas_id` indicated in the `V3CUnitHeaderProperty`. Otherwise, it contains one or more ACL or non-ACL NAL units associated with same `juh_atlas_id` indicated in the `V3CUnitHeaderProperty`.

If `PrimaryItemBox` exists, `item_ID` in this box shall be set to indicate an item of type 'v3c1' or 'v3cb'.

8.5.2.2 V3C atlas tile item data

V3C atlas tile item as defined in subclause 8.3.3 is used.

8.5.2.3 V3C image component item

V3C image component item as defined in subclause 8.3.4 is used with the following restrictions.

- The V3C image component item stores V3C unit payload of V3C unit of type `V3C_OVD`, `V3C_GVD`, `V3C_AVD` and `V3C_PVD`.

8.5.2.4 Item references

The '`v3ct`' item reference as defined in subclause 8.3.5 shall be used to link from the V3C atlas item to V3C atlas tile items. Item references to link V3C atlas items or V3C atlas tile items to V3C image component items are used as defined in subclause 8.3.5.

In order to indicate the relationship between V3C atlas items, '`v3cs`' item reference is used to link the item of type '`v3cb`' to the item of type '`v3a1`'.

8.6 Non-timed V-DMC data storage in ISOBMFF

In this clause, non-timed V3C data refers a single access unit of V3C bitstream which is coded with video-based dynamic mesh coding (ISO/IEC 23090-29).

This clause defines the encapsulation of non-timed V3C data in items within a file. Two methods for encapsulation are defined 1) storage as a single item, where all sub-bitstreams of the V3C bitstream are stored in a single item, and 2) storage as multiple items, where each item stores a different sub-bitstream.

A handler type in the `HandlerBox` of the `MetaBox` shall be 4CC code '`volv`'.

8.6.1 Data structures and definitions

8.6.1.1 Basemesh configuration item property

8.6.1.1.1 Definition

Box Types:	<code>'bmcp'</code>
Property type:	Descriptive item property
Container:	<code>ItemPropertyContainerBox</code>
Mandatory (per item):	Yes, for a basemesh item of type ' <code>bmc1</code> ' or ' <code>bmcB</code> '
Quantity (per item):	One

`BaseMeshConfigurationProperty` provides the basemesh decoder configuration and initialization information.

The basemesh item configuration item property is an essential property. The corresponding essential flag in the `ItemPropertyAssociationBox` shall be set to 1 for a '`bmcp`' item property.

8.6.1.1.2 Syntax

```
aligned(8) class BaseMeshConfigurationProperty extends ItemFullProperty('bmcp', version =
0, 0) {
    BaseMeshDecoderConfigurationRecord bm_decoder_config;
}
```

8.6.1.1.3 Semantics

`bm_config` contains a single instance of `BaseMeshDecoderConfigurationRecord` which is defined in subclause 6.2.2.

8.6.1.2 Arithmetic coded displacement configuration item property

8.6.1.2.1 Definition

Box Types:	'adcp'
Property type:	Descriptive item property
Container:	ItemPropertyContainerBox
Mandatory (per item):	Yes, for a bashmesh item of type 'dpc1'
Quantity (per item):	One

`ACDisplacementConfigurationProperty` provides the decoding specific information for arithmetic coded displacement sub-bitstream.

8.6.1.2.2 Syntax

```
aligned(8) class ACDisplacementConfigurationProperty extends ItemFullProperty('adcp',
version = 0, 0) {
    ACDisplacementDecoderConfigurationRecord disp_config;
}
```

8.6.1.2.3 Semantics

`disp_config` contains a single instance of `ACDisplacementDecoderConfigurationRecord` which is defined in subclause 6.2.3.

8.6.1.3 Submesh configuration item property

8.6.1.3.1 Definition

Box Types:	'smcp'
Property type:	Descriptive item property
Container:	ItemPropertyContainerBox
Mandatory (per item):	Yes, for a bashmesh item of type 'smc1'
Quantity (per item):	One

The submesh item configuration item property provides the number of submeshes and the identifier of submeshes which are present in the associated submesh item.

The submesh item configuration item property is an essential property. The corresponding essential flag in the `ItemPropertyAssociationBox` shall be set to 1 for a 'smcp' item property.

8.6.1.3.2 Syntax

```
aligned(8) class SubMeshConfigurationProperty() extends ItemFullProperty('smcp',
version=0, 0) {
    unsigned int(6) num_submeshes_minus1;
    unsigned int(2) reserved;
    for(int i=0; i < num_submeshes_minus1; i++) {
        unsigned int(24) submesh_id;
    }
}
```



```
}
```

8.6.1.3.3 Semantics

`num_submeshes_minus1` indicates the number of submeshes contained in this item. The value of `num_submeshes_minus1` shall be in range from 0 to 63, inclusive.

`submesh_id` is the identifier for the submesh present in this item. The value of `submesh_id` is equal to the value of the corresponding `bmsi_submesh_id` syntax element in `bmesh_submesh_information()` as defined in ISO/IEC 23090-29:Annex H. The value of `submesh_id` ranges from 0 to $2^{24}-1$, inclusive.

8.6.1.1 Instanced rendering item property

8.6.1.1.1 Definition

Box Types:	'vire'
Property type:	Descriptive item property
Container:	ItemPropertyContainerBox
Mandatory (per item):	No
Quantity (per item):	One

A single `InstancedRenderingProperty` may be present in the `ItemPropertyContainerBox` of V3C atlas item indicating that the mesh representation associated with the atlas item may be used for instanced rendering.

8.6.1.1.2 Syntax

```
aligned(8) class InstancedRenderingProperty extends FullItemProperty('vire', version = 0,
0){
    unsigned int(1)    can_be_texture_instanced;
    unsigned int(7)    reserved;
}
```

8.6.1.1.3 Semantics

`can_be_temporally_instanced`, when equal to 1, indicates that the mesh information associated with the atlas track may be used for temporal instancing.

8.6.1.2 Instanced attribute variant entity group

8.6.1.2.1 General

For instanced rendering, multiple versions of an attribute item may exist that can be indexed using the same set of UV-coordinates and the same mesh. For such applications, instanced attribute entity group offers the needed information to identify items that can be considered as alternative attributes for instanced drawing. An application may choose to use zero or more items from the entity group and the entity group does not mandate any rendering behavior. It merely informs that there is an option for instanced drawing with different attribute variants.

8.6.1.2.2 Definition

Sample Entry Type:	'eiov'
Container:	GroupsListBox ('grpl')
Mandatory:	No
Quantity:	Zero or more

An EntityToGroupBox with `grouping_type` equal to 'eiov' indicates attribute items that shall be considered as variants for instanced rendering. Each item belonging to the entity group shall be indexable with the same UV-coordinates. An `InstancedAttributeVariantEntityToGroupBox` is used to group non-timed (item) V3C data in the same group. Zero or more items from the group may be used.

This entity group shall be present in the file, when `InstancedRenderingProperty` is present in the item property container of a V3C atlas item and, when `can_be_texture_instanced` equals 1 in the `InstancedRenderingProperty`. All items in the entity group shall have the same attribute type.

8.6.1.2.3 Syntax

```
aligned(8) class InstancedAttributeVariantEntityToGroupBox extends
EntityToGroupBox('eiov') {}
```

8.6.2 Single-item encapsulation

8.6.2.1 This clause specifies how to encapsulate a non-timed V3C data in a single item. V3C item

The V3C item as defined in subclause 8.3.1 is used with the following restrictions.

- Only an item of type 'vdm1' shall be used.
- An item of type 'vdm1' shall have an associated `V3CConfigurationProperty` and `BaseMeshConfigurationProperty`.
- If the displacement component is coded using an arithmetic codec, an item of type 'vdm1' shall have an associated `ACDisplacementConfigurationProperty`.
- Sub-sample information for V3C items may be given using an associated sub-sample item property.
- An item of type 'vdm1' shall not have an associated `InstancedRenderingProperty`

If `PrimaryItemBox` exists, `item_ID` in this box shall be set to indicate an item of type 'vdm1'.

8.6.3 Multi-item encapsulation

This clause specifies how to encapsulate a non-timed V3C data in multiple items. Four item types called V3C atlas item, V3C atlas tile item and V3C image component and V3C non-image component (basemesh, submesh and arithmetically coded displacements) item are used.

An overview of the structure for encapsulating non-timed V3C data which contains a single atlas, base mesh, a geometry image (visually coded displacement information), and an attribute image component is illustrated in **Figure 5**.

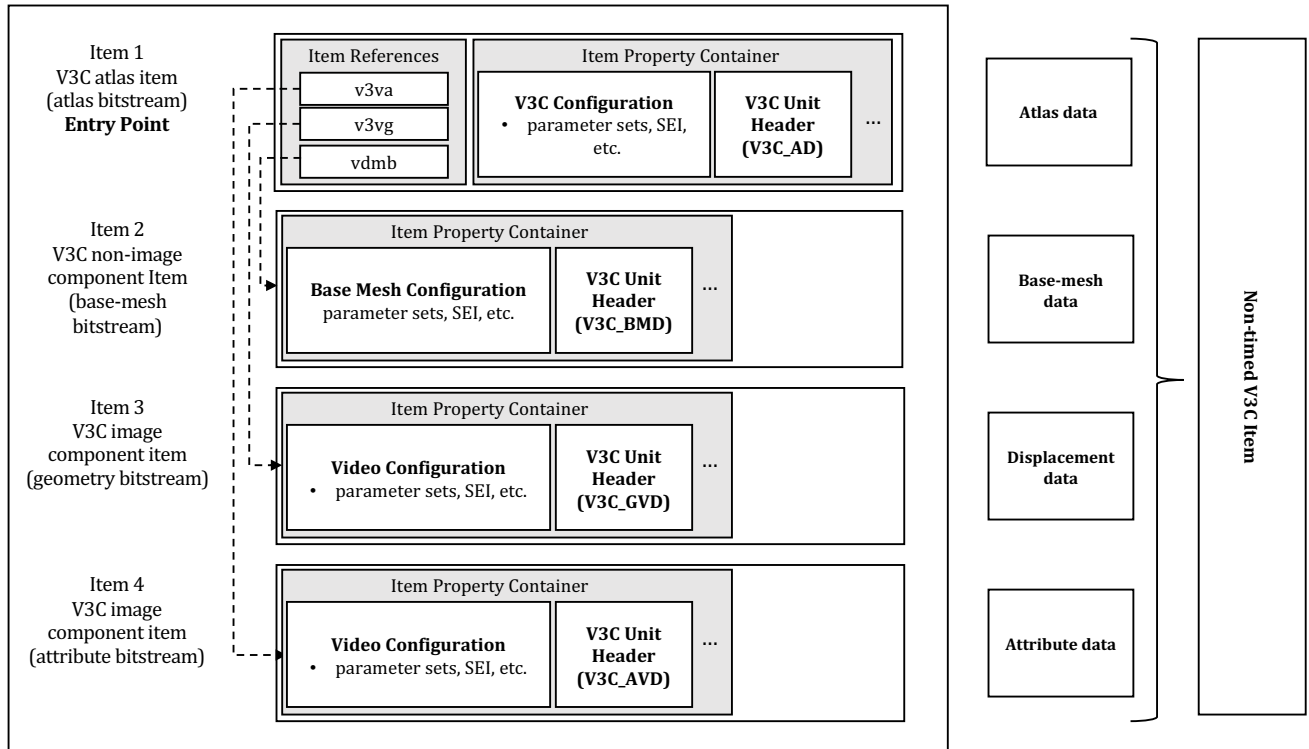


Figure 5. Overview of structure for encapsulating non-timed V3C data

8.6.3.1 V3C atlas item

The V3C atlas item as defined in subclause 8.3.2 is used with the following restrictions:

- Only an item of type 'v3c1' shall be used.
- An item of type 'v3c1' shall have an associated `V3CUnitHeaderProperty`.
- An item of type 'v3c1' shall be associated with one `V3CConfigurationProperty`.
- If an item of type 'v3c1' is associated with V3C atlas tile items, the item of type 'v3c1' stores one or more non-ACL NAL units associated with the V3C atlas tile items. Otherwise, the item of type 'v3c1' stores one or more ACL or non-ACL NAL units.

If `PrimaryItemBox` exists, `item_ID` in this box shall be set to indicate an item of type 'v3c1'.

8.6.3.2 V3C atlas tile item

V3C atlas tile item as defined in subclause 8.3.3 is used.

8.6.3.3 V3C image component item

V3C image component item as defined in subclause 8.3.4 is used with the following restrictions.

- The V3C image component item stores V3C unit payload of V3C unit of type `V3C_GVD`, `V3C_AVD` and `V3C_PVD`.

8.6.3.4 Basemesh item

8.6.3.4.1 Definition

The V3C basemesh item is defined as follows:

- A V3C basemesh item contains an independently decodable coded basemesh access unit.
- Item type 4CC code 'bmc1' or 'bmcb' identifies a V3C basemesh item.
- An item of type 'bmc1' or 'bmcb' shall be associated with one `V3CUnitHeaderProperty` as defined in subclause 8.2.2.
- Each V3C basemesh item shall be associated with one `BaseMeshConfigurationProperty` as defined in subclause 8.6.1.1.
- When the basemesh item contains multiple submeshes and an item of type 'bmcb' is used, submesh bitstreams shall be stored in one or more submesh items with an item of type 'smc1'. The item of type 'bmcb' shall not store any BMCL NAL units.
- The basemesh items shall be marked as hidden items.

8.6.3.4.2 Syntax

`BaseMeshItemData` is structurally identical to the syntax for a basemesh sample.

```
aligned(8) class BaseMeshItemData
{
    int i = 0;
    while ( i < item_size ) { // derived from ItemLocationBox
        unsigned int(bm_config.unit_size_precision_bytes_minus1 + 1)*8) nal_size;
        bit(8) ss_nal_unit[nal_size];
        i += nal_size + bm_config.unit_size_precision_bytes_minus1 + 1;
    }
}
```

8.6.3.4.3 Semantics

In the syntax above, the following applies:

- The value of `item_size` is equal to the sum of the `extent_length` values of each extent of the item, as specified in the `ItemLocationBox`.
- `bm_config` indicates the record in the associated Basemesh configuration property.

`nal_size` specifies the size, in bytes, of the `ss_nal_unit` array. This size is equivalent to the sample stream NAL unit size `ssnu_nal_unit_size` as defined in ISO/IEC 23090-29 Annex D.

`ss_nal_unit` is an array of data containing a single BMCL or non-BMCL NAL unit as defined in 23090-29:Annex H.

8.6.3.5 Submesh item

8.6.3.5.1 Definition

A submesh item is an item of type 'smc1' that contains one or more BMCL NAL units which belong to the same basemesh.

Each submesh item shall be associated with one `SubMeshConfigurationProperty`. The `SubMeshConfigurationProperty` shall indicate the submesh IDs of submeshes present in the submesh item. The submesh items shall be marked as hidden items.

8.6.3.5.2 Syntax

The syntax of submesh item data is defined as `BaseMeshItemData` described in subclause 8.6.3.4.2.

8.6.3.5.3 Semantics

The semantics of submesh item data as defined in subclause 8.6.3.4.2 are used.

8.6.3.6 Arithmetic coded displacement item

8.6.3.6.1 Definition

The arithmetic coded displacement item is defined as follows:

- An arithmetic coded displacement item contains an independently decodable coded arithmetic coded displacement access unit.
- Item type 4CC code 'acd1' identifies an arithmetic coded displacement item.
- An item of type 'acd1' shall be associated with one `V3CUnitHeaderProperty` as defined in subclause 8.2.2.
- Each arithmetic coded displacement item shall be associated with one `ACDisplacementConfigurationProperty` as defined in subclause 8.6.1.2.
- The arithmetic coded displacement items shall be marked as hidden items.

8.6.3.6.2 Syntax

`ACDisplacementItemData` is structurally identical to the syntax for an arithmetic coded displacement sample.

```
aligned(8) class ACDisplacementItemData
{
    int i = 0;
    while ( i < item_size ) { // derived from ItemLocationBox
        unsigned int(disp_config.unit_size_precision_bytes_minus1 + 1)*8) nal_size;
        bit(8) ss_nal_unit[nal_size];
        i += nal_size + disp_config.unit_size_precision_bytes_minus1 + 1;
    }
}
```

8.6.3.6.3 Semantics

In the syntax above, the following applies:

- The value of `item_size` is equal to the sum of the `extent_length` values of each extent of the item, as specified in the `ItemLocationBox`.
- `disp_config` indicates the record in the associated arithmetic coded displacement configuration property.

`nal_size` specifies the size, in bytes, of the `ss_nal_unit` array. This size is equivalent to the sample stream NAL unit size `ssnu_nal_unit_size` as defined in ISO/IEC 23090-29 Annex D.

`ss_nal_unit` is an array of data containing a single DCL or non-DCL NAL unit as defined in 23090-29:Annex J.

8.6.3.7 Item references

The '`v3ct`' item reference as defined in subclause 8.3.5 shall be used to link from the V3C atlas item to V3C atlas tile items.

The '`v3va`', '`v3vg`', or '`v3vp`' item references as defined in subclause 8.3.5. shall be used to link V3C atlas items or V3C atlas tile items to V3C image component items with the following restriction.

— '`v3vo`' item reference type shall not be used.

In order to indicate the association between a V3C atlas item and V3C non-image component items, i.e., basemesh and arithmetic-coded displacement items, two item reference types with 4CC codes '`vdmb`' and '`vdmd`' are defined. Item reference is defined “from” either a V3C atlas item “to” the related V3C non-image component items. The 4CC codes of item reference types shall be:

- '`vdmb`': the referenced item(s) contain the basemesh data.
- '`vdmd`': the referenced item(s) contain the arithmetic coded displacement data.

In order to indicate the relationship between a basemesh item to submesh items, item references with the 4CC code '`bmcs`' shall be used “from” a basemesh item “to” the submesh items.

9 Partial access of volumetric visual data

9.1 General

Signalling related to partial access functionality is defined in this this clause. Partial access relates to making available only a subset of V3C content. Alternative methods for signalling partial access related information using Volumetric annotation SEI message family, as defined in ISO/IEC 23090-5, Annex F, are discussed in Annex E.

NOTE This clause is only applicable to V-PCC and MIV applications.

9.2 Common data structures

9.2.1 3D vector

9.2.1.1 Syntax

```
aligned(8) class Vector3(int precision = 32) {
    int reserved_bits = 8 - (precision*3) % 8;
    if (reserved_bits != 8) {
        bit(reserved_bits) reserved = 0;
    }
    unsigned int(precision) x;
    unsigned int(precision) y;
    unsigned int(precision) z;
}
```

9.2.1.2 Semantics

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

9.2.2 Spatial region bounding box

This data structure defines a bounding box for a 3D region in Cartesian space using an anchor point and a scale along the three axes.

9.2.2.1 Syntax

```
aligned(8) class V3CBoundingBox (anchor_included, scale_included) {
    if (anchor_included) { // anchor is not 0,0,0
        unsigned int(8) bb_pos_precision;
        Vector3 bb_position(bb_pos_precision);
    }
    if (scale_included) {
        unsigned int(8) bb_scale_precision;
        Vector3 bb_scale(bb_scale_precision);
    }
}
```

9.2.2.2 Semantics

`bb_pos_precision` indicates the precision of `bb_position` in number of bits.

`bb_position.x`, `bb_position.y`, and `bb_position.z` indicate the position of the 3D bounding box in the Cartesian coordinates along the x , y , and z axes, respectively.

`bb_scale_precision` indicates the precision of `bb_scale` in number of bits.

`bb_scale.x`, `bb_scale.y`, and `bb_scale.z` 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) if `anchor_included` is set to 0 and relative to `bb_position` if `anchor_included` is set to 1.

9.2.3 Tile mapping

This data structure provides the mapping between a spatial region and the set of atlas tiles that contain patches associated with that spatial region. If spatial scalability is enabled, an instance of this data structure provides a separate tile mapping for each level-of-detail.

9.2.3.1 Syntax

```
aligned(8) class TileMapping (spatial_scalability_enabled) {
    if (spatial_scalability_enabled) {
        unsigned int(8) num_lod;
        for (int i=0; i < num_lod; i++) {
            // LoD to tiles mapping
            unsigned int(8) lod_index;
            unsigned int(8) lod_num_tiles;
            for (int j=0; j < lod_num_tiles; j++) {
                bit(2) reserved = 0;
                unsigned int(6) atlas_id;
            }
        }
    }
}
```

```

        unsigned int(16) lod_tile_id;
    }
} else {
    // spatial regions to tiles mapping
    unsigned int(8) num_tiles;
    for (j=0; j < num_tiles; j++) {
        bit(2) reserved = 0;
        unsigned int(6) atlas_id;
        unsigned int(16) tile_id;
    }
}
}

```

9.2.3.2 Semantics

`num_lod` indicates the number of LoDs available for an associated 3D spatial region.

`lod_index` indicates the ordering on the LoDs for an associated 3D spatial region. A set of atlas tiles with a certain `lod_index` should be selected with the sets of atlas tiles associated with all lower `lod_index` values. An LoD tile set associated with a lower `lod_index` value should be processed first.

`lod_num_tiles` indicates the number of atlas tiles associated with an LoD of a spatial region.

`atlas_id` indicates the atlas ID associated with the `lod_tile_id` or `tile_id` of a spatial region.

`lod_tile_id` is an ID for a V3C atlas tile associated with an LoD of the spatial region. The value of `lod_tile_id` is equal to value of `afti_tile_id` syntax element in atlas frame tile information, defined in ISO/IEC 23090-5.

`num_tiles` indicates the number of atlas tiles associated with a spatial region.

`tile_id` is an ID for a V3C atlas tile that is associated with the spatial region. The value of `tile_id` is equal to value of `afti_tile_id` syntax element in atlas frame tile information, defined in ISO/IEC 23090-5.

9.2.4 V3C object collection

9.2.4.1 Syntax

```

aligned(8) class V3CObject () {
    unsigned int(1) obj_cancel_flag;
    unsigned int(1) obj_priority_present_flag;
    unsigned int(1) obj_dependencies_present_flag;
    unsigned int(1) obj_bounding_box_present_flag;
    unsigned int(1) obj_spatial_scalability_enabled_flag;
    bit(1) reserved = 0;
    unsigned int(2) obj_idx_bytes_minus1;
    unsigned int((obj_idx_bytes_minus1 + 1)* 8) soi_object_idx;
    if (!obj_cancel_flag) {
        if (obj_priority_present_flag) {
            bit(4) reserved = 0;
            unsigned int(4) obj_priority_value;
        }
    }
}

```



```

    if (obj_dependencies_present_flag) {
        unsigned int(8) obj_num_dependencies;
        bit(6) reserved = 0;
        unsigned int(2) obj_dep_idx_bytes_minus1;
        for (i=0; i < obj_num_dependencies; i++)
            unsigned int((obj_dep_idx_bytes_minus1 + 1)* 8) soi_dep_object_idx[i];
    }
}
if (obj_bounding_box_present_flag) {
    V3CBoundingBox obj_bounding_box(1, 1);
}
TileMapping obj_tilemap(obj_spatial_scalability_enabled_flag);
}
}
aligned(8) class V3CObjectCollection {
    unsigned int(32) num_objects;
    for (int i=1; i<=num_objects; i++) {
        V3CObject obj;
    }
}

```

9.2.4.2 Semantics

`obj_cancel_flag` indicates that the object is cancelled.

`obj_priority_present_flag` indicates whether priority information is available for an object. Value 0 indicates that no object priority information is given. Value 1 indicates that object priority information is present.

`obj_dependencies_present_flag` indicates whether object dependency information is available for an object. Value 0 indicates that the object does not depend on other objects. Value 1 indicates that the object depends on one or more objects within the V3C content.

`obj_bounding_box_present_flag` indicates whether 3D bounding boxing information is available for an object. Value 0 indicates that no bounding box information is given. Value 1 indicates that that 3D bounding box information for the object is present.

`obj_spatial_scalability_enabled_flag` indicates whether the LoD-based scalability is supported by the carried V3C content. Value 1 indicates that LoD-based scalability is supported. Value 0 indicates that the carried V3C content does not include multiple LoDs.

`obj_idx_bytes_minus1` plus 1 specifies the length of the object index, in number of bytes, for an object in the signalled object list.

`soi_object_idx` indicates the value of an object index, as defined by the object scene information SEI message.

`obj_priority_value` indicates the priority value of an object in the object update list of the sample. The lower the priority value, the higher the priority.

`obj_num_dependencies` is the number of objects that an object in the object update list of the sample depends on.

`obj_dep_bytes_minus1` plus 1 specifies the length, in number of bytes, of the index of the dependent object.

`soi_dep_object_idx[i]` is the index of the *i*-th object that an object in the object update list of the sample depends on.

`obj_bounding_box` indicates the bounding box of the object.

`obj_tilemap` indicates how the V3C object is mapped to tiles and `LoDs.num_objects` indicates the number of objects in the object collection.

`obj` contains object-related partial access information.

9.3 Spatial region information structure

9.3.1 Definition

`V3CSpatialRegion` provides information of a spatial region or object based subdivision of the volumetric media and their mapping information to atlas tiles. It may include the *x, y, z* offset of the spatial region and the width, height, and depth of the region in 3D space, 3D bounding box information of the volumetric media or object based subdivision details.

9.3.2 Syntax

```
aligned(8) class V3CSpatialRegion {
    unsigned int(32) size;
    unsigned int(16) region_id;
    unsigned int(1) bb_anchor_present_flag;
    unsigned int(1) bb_scale_present_flag;
    unsigned int(1) tile_mapping_present_flag;
    unsigned int(1) tm_spatial_scalability_flag;
    unsigned int(1) object_collection_present_flag;
    bit(3) reserved = 0;
    if (bb_anchor_present_flag || bb_scale_present_flag) {
        V3CBoundingBox bounding_box(bb_anchor_present_flag, bb_scale_present_flag);
    }
    if (tile_mapping_present_flag) {
        TileMapping tile_map(tm_spatial_scalability_flag);
    }
    if (object_collection_present_flag) {
        V3CObjectCollection object_collection;
    }
}
```

9.3.3 Semantics

`size` is an integer that specifies the number of bytes in this element, including all its fields and contained elements.

`region_id` is an identifier for the spatial region.

`bb_anchor_present_flag` indicates the presence of the bounding box with an anchor field.

`bb_scale_present_flag` indicates the presence of the bounding box with the scale field.

`tile_mapping_present_flag` indicates the presence of tile mapping.

`tm_spatial_scalability_flag` indicates whether of the signalled tile mapping has multiple levels-of-detail. This flag shall be set to 0 if `tile_mapping_present_flag` is set to 0.

`object_collection_present_flag` indicates the presence of an object collection that lists objects in the spatial region.

9.4 V3C tile video component track grouping

9.4.1 Definition

A V3C tile video component track group is a track group that groups all the tracks carrying V3C video component information associated with a set of atlas tiles. This track group is used when an atlas contains more than one tile and all atlas component tiles for that atlas are carried in the corresponding V3C atlas track. An example of using V3C tile video component track grouping for one atlas track containing tiles 1 and 2 is shown in Figure 4.

The presence of a `TrackGroupTypeBox` with `track_group_type` equal to `'vtcg'` in a track indicates that this track belongs to a group of V3C video component tracks that correspond to a V3C tile video component group.

Tracks belonging to the same V3C tile video component group have the same value of `track_group_id` for `track_group_type` `'vtcg'`, and the `track_group_id` of tracks from one V3C tile video component track group differs from the `track_group_id` of tracks from any other V3C tile video component track group.

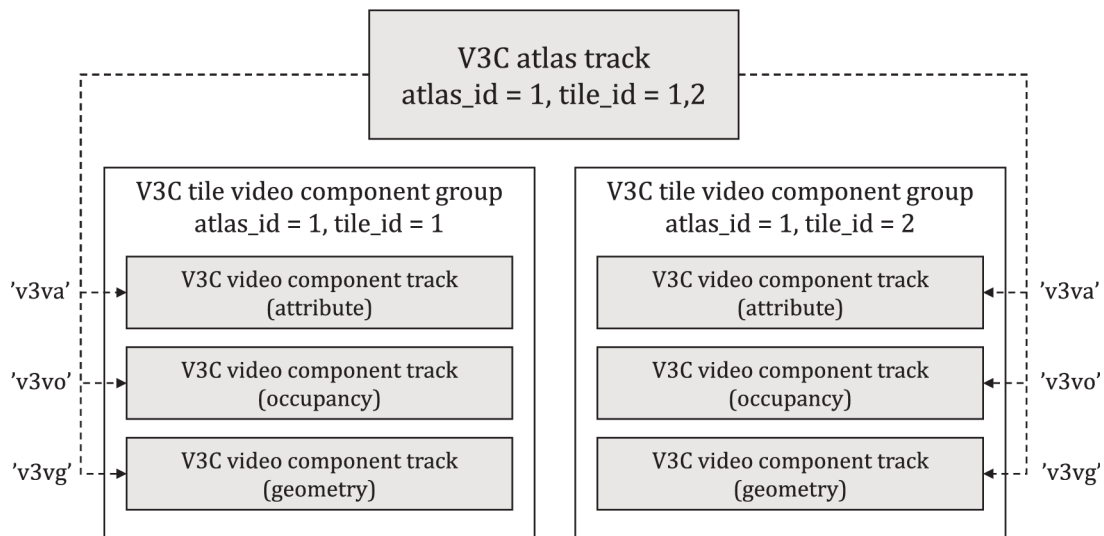


Figure 5 — Example of V3C tile video component track grouping

9.4.2 Syntax

```
aligned(8) class V3CTileVideoComponentGroupBox extends TrackGroupTypeBox('vtcg') {
    unsigned int(8) num_tiles;
    for (int i=0; i < num_tiles; i++) {
        unsigned int(6) atlas_id;
        bit(2) reserved = 0;
        unsigned int(16) tile_id;
```

```

    }
}

```

9.4.3 Semantics

`num_tiles` is the number of V3C atlas tiles associated with the track group.

`atlas_id` indicates the atlas ID associated with the `tile_id`.

`tile_id` is an id for a V3C atlas tile. The value of `tile_id` is equal to value of `afti_tile_id` syntax element in atlas frame tile information, defined in ISO/IEC 23090-5.

9.5 Volumetric media bounding box

9.5.1 Definition

Box Types: 'vpbb'
 Container: V3CBitstreamSampleEntry or V3CAtlasSampleEntry
 Mandatory: No
 Quantity: Zero or one

A V3CBoundingBox shall be present in the sample entry of a V3C atlas track when V3CSpatialRegionsBox is present.

When the atlas sequence parameter set of the atlas carried by the V3C atlas track includes VUI parameters, the values of `bb_position.x`, `bb_position.y`, and `bb_position.z` shall be identical to `vui_display_box_size[0]`, `vui_display_box_size [1]` and `vui_display_box_size [2]`, respectively, in the `vui_parameters()` syntax structure defined in ISO/IEC 23090-5, Annex G.

9.5.2 Syntax

```

aligned(8) class V3CBoundingBox extends FullBox('vpbb', version=0, 0) {
    V3CBoundingBox bounding_box(0, 1);
}

```

9.6 Static spatial region collection box

9.6.1 Definition

Box Types: 'v3sc'
 Container: V3CBitstreamSampleEntry, V3CAtlasSampleEntry or
 DynamicVolumetricMetadataSampleEntry ('dyvm')
 Mandatory: No
 Quantity: Zero or one

Static 3D spatial regions in V3C content and their respective associated tracks shall be signalled in a V3CSpatialRegionCollectionBox. 3D spatial regions may overlap each other. Example of partial access for overlapping spatial subdivisions is explained in Annex G.

If scene object information does not change over time, V3CSpatialRegionCollectionBox may be present in either V3CBitstreamSampleEntry or the V3CAtlasSampleEntry and include static mapping between partial access regions, i.e., spatial regions or objects, and associated tiles. When a V3CSpatialRegionCollectionBox is present, the information signalled in the box shall have

precedence over any information signalled in V3C Volumetric Annotation SEI messages in the V3C bitstream.

For all spatial regions in signalled in a `V3CSpatialRegionCollectionBox`, the following restrictions shall apply:

- `region.bounding_box` shall have the `bb_scale_present_flag` set.
- `region.object_collection` shall not include any objects with `obj_cancel_flag` set.

9.6.2 Syntax

```
aligned(8) class V3CSpatialRegionCollectionBox extends FullBox('v3sc', version=0, 0) {
    unsigned int(16) num_regions;
    for(int i=1; i<=num_regions; i++) {
        V3CSpatialRegion region;
    }
}
```

9.6.3 Semantics

`num_regions` indicates the number of 3D spatial regions in the V3C media.

`region` describes attributes related to 3D spatial region belonging to the V3C media.

9.7 Dynamic spatial region information

9.7.1 General

If the V3C atlas track has an associated timed-metadata track with a sample entry type '`dyvm`', 3D spatial regions defined for the volumetric media stream carried by the V3C atlas track are considered as dynamic regions (i.e., the spatial region information may dynamically change over time) and `V3CSpatialRegionCollectionBox` shall be present in the sample entry of the timed-metadata track, and not in the sample entry of the V3C track. If objects are added or removed in the middle of the bitstream by scene object information SEI messages, then at least one `V3CVolumetricMetadataSample` shall be present.

The associated timed-metadata track shall contain a '`cdsc`' track reference to the V3C atlas track or the V3C bitstream track.

9.7.2 Sample entry

9.7.2.1 Definition

Sample Entry Type:	' <code>dyvm</code> '
Container:	Sample Description Box (' <code>stsd</code> ')
Mandatory:	No
Quantity:	0 or 1

The sample entry for dynamic spatial region information associated to the V3C carriage format is defined by the `DynamicVolumetricMetadataSampleEntry`. This sample entry shall contain `V3CSpatialRegionCollectionBox`.

The codecs parameter value for this track as defined in RFC 6381 shall be set to '`dyvm`'.

9.7.2.2 Syntax

```
aligned(8) class DynamicVolumetricMetadataSampleEntry extends MetadataSampleEntry('dyvm')
{
    V3CSpatialRegionCollectionBox spatial_regions;
}
```

9.7.2.3 Semantics

`spatial_regions` contains initial mapping information between partial access regions, i.e., spatial regions or objects, defined for the volumetric media stream, and associated atlas tiles.

9.7.3 Sample format

9.7.3.1 General

Each sample is either:

- a) Exactly one empty sample with the `sample_size=0` (representing a period of non-zero duration in which there is no updates to spatial region information); or
- b) One or more `V3CSpatialRegion` elements that share the same start time and end time.

9.7.3.2 Syntax

```
aligned(8) class V3CVolumetricMetadataSample() {
    unsigned int(16) num_regions;
    for(int i=1; i<=num_regions; i++) {
        V3CSpatialRegion region;
    }
}
```

9.7.3.3 Semantics

`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.

`region` describes attributes related to 3D spatial region signalled in the sample.

9.7.4 Sync samples

All `V3CSpatialRegion` elements signalled in sync samples of the dynamic spatial region information timed-metadata track shall satisfy all the following conditions:

- `V3CBoundingBox` shall have both the `bb_anchor_present_flag` and the `bb_scale_present_flag` set, i.e., each bounding box shall include both the position and scale components.
- `V3CObjectCollection` shall not include any V3C objects with `obj_cancel_flag` set.

9.8 Storage of atlas tiles using `NALUMapEntry`

`NALUMapEntry` specified in ISO/IEC 14496-15 shall be present in the V3C atlas track when `V3CSpatialRegionCollectionBox` is present and no V3C atlas tile tracks are associated with the V3C

atlas track. This document uses `NALUMapEntry` as specified in ISO/IEC 14496-15 with the following additional requirements:

- The `NALUMapEntry`, when present, is used to assign an identifier, called `groupID`, to each atlas NAL units.
- For ACL NAL units, `groupID` shall be equal to `ath_id + 1`, where `ath_id` is specified in ISO/IEC 23090-5. For non-ACL NAL units, `groupID` shall be equal to 0, which implies that the atlas NAL unit is required for decoding any atlas tile in the same atlas frame to which the NAL unit belongs.
- The `NALUMapEntry`, when present, may or may not be linked to a sample group description setting the `grouping_type_parameter` of the `SampleToGroupBox` of type 'nalm'. Consequently, A `SampleToGroupBox` of type 'nalm' may or may not use version 0 of the box.

10 Viewport information

10.1 General

This clause specifies signalling of viewport information and associated intrinsic and extrinsic camera information for V3C content in container files. Viewport information may be conveyed through the defined viewport information structure that includes an extrinsic camera information structure that specifies the viewport's position and the rotation. In addition, the viewport information structure includes an intrinsic camera information structure. The receiver may render the V3C content based on the signalled viewport information at any point in time.

10.2 Structures

10.2.1 Extrinsic camera information

10.2.1.1 Syntax

```
aligned(8) class ExtCameraInfo () {
    unsigned int(8)[4] cam_pos_x;
    unsigned int(8)[4] cam_pos_y;
    unsigned int(8)[4] cam_pos_z;
    signed int(32) cam_quat_x;
    signed int(32) cam_quat_y;
    signed int(32) cam_quat_z;
}
```

10.2.1.2 Semantics

`cam_pos_x`, `cam_pos_y`, and `cam_pos_z`, respectively, indicate the x, y, and z coordinates of the position of the camera in metres in the global reference coordinate system. 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.

`cam_quat_x`, `cam_quat_y`, and `cam_quat_z`, indicate the x, y, and z components, respectively, of the rotation of the camera using the quaternion representation. The values shall be in the range of -2^{30} to 2^{30} , inclusive. When the component of rotation is not present, its value shall be inferred to be equal to 0. The value of rotation components may be calculated as follows:

$$qX = \text{cam_quat_x} \div 2^{30}, \quad qY = \text{cam_quat_y} \div 2^{30}, \quad qZ = \text{cam_quat_z} \div 2^{30}$$

The fourth component, q_W , for the rotation of the current camera model using the quaternion representation is calculated as follows:

$$q_W = \text{Sqrt}(1 - (q_X^2 + q_Y^2 + q_Z^2))$$

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

NOTE As aligned ISO/IEC 23090-5, q_W is always positive. If a negative q_W 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.

10.2.2 Intrinsic camera information

10.2.2.1 Syntax

```
aligned(8) class IntCameraInfo () {
    unsigned int(10) camera_id;
    bit(3) reserved = 0;
    unsigned int(3) camera_type;
    if (camera_type == 0) {
        signed int(32) erp_horizontal_fov;
        signed int(32) erp_vertical_fov;
    }
    if (camera_type == 1) {
        signed int(32) 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;
}
```

10.2.2.2 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.

`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 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.

10.2.3 Viewport information

10.2.3.1 Syntax

```
aligned(8) class ViewportInfo (ext_camera_flag, int_camera_flag) {
    if (ext_camera_flag == 1) {
        unsigned int(1) center_view_flag;
        bit(6) reserved = 0;
        if (center_view_flag == 0) {
            unsigned int(1) left_view_flag;
        } else {
            bit(1) reserved = 0;
        }
        ExtCameraInfo extCamInfo();
    }
    if (int_camera_flag == 1) {
        IntCameraInfo intCamInfo();
    }
}
```

10.2.3.2 Semantics

`center_view_flag` is a flag indicating whether the signalled viewport position corresponds to the centre of the viewport or to one of two stereo positions of the viewport. Value 1 indicates that the signalled viewport position corresponds to the centre of the viewport. Value 0 indicates that the signalled viewport position corresponds to one of two stereo positions of the viewport.

`left_view_flag` is a flag indicating whether the signalled viewport information correspond to the left stereo position or the right stereo position of the viewport. Value 1 indicates that the signalled viewport information corresponds to the left stereo position of the viewport. Value 0 indicates that the viewport information signalled correspond to the right stereo positions of the viewport.

`extCamInfo` is an instance of `ExtCameraInfo` defining the extrinsic camera parameters for the viewport.

`intCamInfo` is an instance of `IntCameraInfo` defining the intrinsic camera parameters for the viewport.

10.3 Viewport information timed-metadata track

10.3.1 General

This subclause describes the use of the timed metadata track to signal viewport information in V3C carriage format, composed of intrinsic and extrinsic camera parameters, including viewport position and rotation information as well as viewport camera parameters. To signal viewport information for a V3C content, the viewport information timed metadata track only references to related V3C atlas tracks, not directly to V3C video component tracks.

A viewport information timed metadata track containing a 'cdtg' track reference describes the referenced tracks and track groups collectively. When the timed metadata track is linked to one or more V3C atlas tracks with a 'cdsc' track reference, it describes each V3C atlas track individually.

NOTE The receiver can render the V3C content based on the signalled viewport information at any point in time, or it can recommend user to consume content based on the viewport information via other means. Viewport information track can also reference other non-V3C related video tracks stored in the same file. When viewport information track references both a V3C content and a video track, the referenced video track may contain the 2D rendering of the referenced V3C content.

10.3.2 Viewport information sample entry

10.3.2.1 Definition

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

The sample entry for viewport information associated to the V3C carriage format is defined by the `ViewportInfoSampleEntry`.

The viewport information sample entry shall contain a `ViewportInfoConfigurationBox`, describing the viewport type and, if applicable to all the samples of the track, the intrinsic and/or extrinsic camera parameters.

The codecs parameter value for this track as defined in RFC 6381 shall be set to '6vpt'.

10.3.2.2 Syntax

```
aligned(8) class ViewportInfoConfigurationBox
extends FullBox('6vpC', version = 0, 0) {
    unsigned int(7) viewport_type;
    bit(1) reserved = 0;
    string viewport_description;
    unsigned int(1) dynamic_int_camera_flag;
    unsigned int(1) dynamic_ext_camera_flag;
    bit(6) reserved = 0;
    if (dynamic_int_camera_flag == 0) {
        IntCameraInfo();
    }
    if (dynamic_ext_camera_flag == 0) {
        ExtCameraInfo();
    }
}
```

```
aligned(8) class ViewportInfoSampleEntry() extends MetadataSampleEntry ('6vpt') {
    ViewportInfoConfigurationBox();
}
```

10.3.2.3 Semantics

`viewport_type` specifies the type of the viewport as listed in

Table 13 for the *i*-th viewport parameter set for all samples referring to this sample entry.

Table 13 — Viewport Types

Value	Description
0	A recommended viewport per the director's cut, i.e., a viewport suggested according to the creative intent of the content author or content provider
1	A recommended viewport selected based on measurements of viewing statistics
2	A recommended viewport based on the selected viewport of another user
3	An initial viewport suggested to use when starting to play associated immersive media
4	A recommended viewport suggested for an associated spatial region
5..239	Reserved
240..255	Unspecified (for use by applications or external specifications)

`viewport_description` is null-terminated UTF-8 string that provides a textual description of the recommended viewport for the *i*-th viewport parameter set for all samples referring to this sample entry.

`dynamic_int_camera_flag` equal to 0 indicates that intrinsic camera parameters are fixed for all samples referring to this sample entry. If `dynamic_ext_camera_flag` is equal to 0, `dynamic_int_camera_flag` shall also be equal to 0.

`dynamic_ext_camera_flag` equal to 0 indicates that extrinsic camera parameters are fixed for all samples referring to this sample entry.

For `viewport_type` equal to 3, the timed metadata indicates the recommended initial viewport information, composed of the initial viewport positions and rotations, when playing the associated V3C media tracks. When the playback of a media track is intended to be started using another viewport than that indicated by initial viewport position (`cam_pos_x`, `cam_pos_y`, `cam_pos_z`) equal to (0, 0, 0) relative to the global coordinate axes and initial viewing rotation (`cam_quat_x`, `cam_quat_y`, `cam_quat_z`) equal to (0,0,0) relative to the global coordinate axes, this metadata track shall be present and associated with the media track. In the absence of this type of metadata, `cam_pos_x`, `cam_pos_y`, `cam_pos_z`, `cam_quat_x`, `cam_quat_y`, and `cam_quat_z` should all be inferred to be equal to 0 for the initial viewport.

10.3.3 Viewport information sample format

10.3.3.1 General

Each viewport sample carries an array of viewports of the type defined in the associated sample entry. The parameters of each viewport include the extrinsic and intrinsic camera information parameters described by `IntCameraInfo` and `ExtCameraInfo`. While extrinsic camera information parameters described by `ExtCameraInfo` are expected to be in each sample, intrinsic camera parameters described by `IntCameraInfo` are only present in a sample if the intrinsic camera parameters signalled in the earlier samples are no longer applicable.

Previously defined extrinsic or intrinsic camera parameters for a certain viewport from an earlier sample shall persist if not modified.

Any sample in a viewport information timed metadata track is allowed to be marked as a sync sample. For a particular sample in the timed metadata track, if at least one of the media samples in the referenced V3C atlas track having the same decoding time is a sync sample, the particular sample shall be marked as a sync sample, otherwise, that sample may or may not be marked as a sync sample.

If the viewport information timed-metadata track is present, extrinsic camera parameters expressed by `ExtCameraInfo()` shall be present in either sample entry or sample level. It is prohibited that both of the following concurrently happen; `dynamic_ext_camera_flag` in `ViewportInfoConfigurationBox` equals to 0 and `camera_extrinsic_flag[i]` equals to 0 for all samples.

10.3.3.2 Syntax

```
aligned(8) class ViewportInfoSample() {
    unsigned int(8) num_viewports;
    for (int i=1; i <= num_viewports; i++){
        unsigned int(7) viewport_id[i];
        unsigned int(1) viewport_cancel_flag[i];
        if (viewport_cancel_flag[i] == 0) {
            unsigned int(1) camera_extrinsic_flag[i];
            unsigned int(1) camera_intrinsic_flag[i];
            bit(6) reserved = 0;
            ViewportInfo (camera_extrinsic_flag[i], camera_intrinsic_flag[i]);
        }
    }
}
```

10.3.3.3 Semantics

`num_viewports` indicates the number of viewports signalled in the sample.

`viewport_id[i]` is an identifier number that is used to identify the *i*-th viewport.

`viewport_cancel_flag[i]` equals 1 indicates that the viewport with the id `viewport_id[i]` is cancelled. Indicates that viewport information for the *i*-th viewport follows.

`camera_intrinsic_flag[i]` equal to 1 indicates that the intrinsic camera parameters are present in the *i*-th viewport in the current sample. It shall be equal to 0 if `dynamic_int_camera_flag[i]` equals to 0. Moreover, it shall be set as 0 when `camera_extrinsic_flag[i]` equals to 0.

`camera_extrinsic_flag[i]` equal to 1 indicates that the extrinsic camera parameters are present in the *i*-th viewport in the current sample. It shall be equal to 0 if `dynamic_ext_camera_flag[i]` equals to 0.

11 Encapsulation and signalling in MPEG-DASH

11.1 Single track mode

The single-track mode in DASH enables streaming of V3C ISOBMFF files where V3C content is stored using single-track encapsulation. The single-track mode in DASH should be represented as one Adaptation Set with one or more Representations. Representations within the sole Adaptation Set shall use the same codec for the corresponding V3C components (e.g., the occupancy shall have the same codec in all Representations) but are not required to use the same codec for every V3C component (e.g., the occupancy could use one codec, e.g., ISO/IEC 14496-10, and geometry could be encoded by second codec, e.g., ISO/IEC 23008-2).

If a Representation consists of more than one Media Segment, an Initialization Media Segment shall be present. The Initialization Segment shall contain a `V3CDecoderConfigurationRecord` with the `v3c_parameter_set` syntax structure, as defined in (ISO/IEC 23090-5, Clause 7) and a Component Codec Mapping SEI Message, as defined in (ISO/IEC 23090-5, Annex E). The initialization segment shall also contain all necessary decoder configuration records for the other V3C components.

The first sample of a Media Segment shall have a Stream Access Point (SAP) of type 1 or 2. That means each sub-sample of the first sample shall have a Stream Access Point (SAP) of type 1 or 2.

The following restriction on some of the attributes shall be applied:

- The `@mimeType` parameter shall be 'application/mp4'
- The `@codecs` parameter shall be present on the adaptation set level and shall signal the maximum required capability to decode any Representation in the Adaptation Set. The `@codecs` parameter should be signalled on the representation level if different from the one on the adaptation set level.
- The `@codecs` parameter present on a representation level shall signal the required capability to decode any component in the Representation.
- When the 'codecs' parameter of a MIME type is used, sub-parameters are used as defined in Annex C.
- The `@maxWidth` and `@maxHeight` parameters shall not be signalled for any Adaptation Set.
- The `@frameRate` shall be signalled only in the **AdaptationSet** element, i.e., the value shall not be different for different Representations in one Adaptation Set.
- The `@width` and `@height` shall not be signalled for any Representation.

Examples of DASH signalling are provided in Annex D.

11.2 Multi-track mode

11.2.1 General

In the multi-track mode, each V3C video and non-video component shall be represented in the DASH manifest (MPD) file as a separate Adaptation Set. These Adaptation Sets are referred to as Video Component Adaptation Sets and Non-Video Component Adaptation Sets respectively and collectively as V3C Component Adaptation Sets. An additional Adaptation Set for atlas information serves as the Main

Adaptation Set for the V3C content. If a video-coded packed V3C component containing multiple V3C components is present, the packed V3C component is signaled by an AdaptationSet. If a geometry or attribute component has multiple maps, each map may be signalled using a separate **AdaptationSet** element.

The Main Adaptation Set shall have the @codecs attribute set to 'v3c1', 'v3cg' or 'v3cb' while the @codecs attribute for the V3C Component Adaptation is set based on the respective codec used for encoding the component. The value of @codecs shall be set to 'resv.vvvc.XXXX', where XXXX corresponds to the four-character code (4CC) of the codec from the original_format field in RestrictedSchemeInfoBox of Sample Entry (e.g., 'avc1', 'hvc1', or 'bmc1').

The Main Adaptation Set shall contain a single Initialization Segment at the adaptation set level. The Initialization Segment shall contain all parameter sets needed to initialize the V3C decoder, including V3C parameter sets as well as other parameter sets for component sub-bitstreams.

Media Segments for the Representation of the Main Adaptation Set shall contain one or more track fragments of the V3C atlas track. Media Segments for the Representations of V3C Component Adaptation Sets shall contain one or more track fragments of the corresponding video or non-video component track at the file format level. Examples of the DASH MPD signalling are provided in Annex D.

11.2.2 V3C preselections

The V3C preselection may either be signalled in MPD using a **PreSelection** element within the **Period** element or a Preselection descriptor at the Adaptation Set level. A V3C **PreSelection** element is signalled, as defined in ISO/IEC 23009-1, with an id list for the @preselectionComponents attribute including the id of the Main Adaptation Set for the volumetric media followed by the ids of the V3C Component Adaptation Sets. The @codecs attribute for the Preselection shall be set to 'v3c1', 'v3cg' or 'v3cb' indicating that the media represented by the Preselection is visual volumetric video-based coding media. When the media represented by the Preselection contains multiple atlases, 'v3cb' shall be used for the @codecs attribute.

Figure 6 illustrates an exemplary DASH configuration for grouping V3C components belonging to a single V3C content within an MPEG-DASH MPD file.

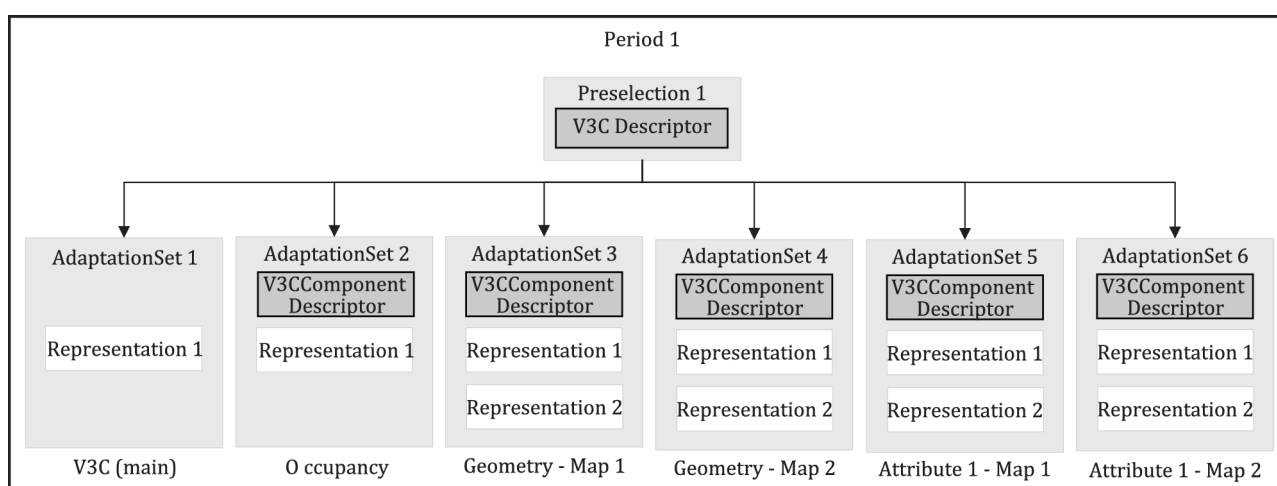


Figure 6 — Grouping V3C components in an MPD using Preselections

When multiple atlases are present in the V3C content, each atlas track shall be represented by a separate Adaptation Set considered as Atlas Adaptation Set. Atlas Adaptation Set shall have the @codecs attribute set to 'v3a1' or 'v3ag'. Representations of Atlas Adaptation Sets shall have a @dependencyId

attribute set to the id of the Representation of the Main Adaptation Set. Each Atlas Adaptation Set shall be part of a separate Preselection that includes the Atlas Adaptation Set, as the main Adaptation Set of the Preselection, and V3C Component Adaptation Sets for that atlas.

11.2.3 V3C atlas tile preselections

When V3C atlas tiles are carried in separate tracks these shall be represented by separate Adaptation Sets, considered as Atlas Tile Adaptation Sets, with the `@codecs` attribute for the Adaptation Sets set to `'v3t1'`. V3C component tracks associated with an atlas tile track shall also be carried in separate Adaptation Sets with the `@codecs` attribute set to `'resv.vvvc.XXXX'`, where XXXX corresponds to the four-character code (4CC) of the video codec (e.g., `'avc1'`, `'hvc1'`, or `'bmc1'`).

Atlas Tile Adaptation Sets and associated V3C Component Adaptation Sets shall be part of a single Atlas Tile Preselection in the MPD with the Atlas Tile Adaptation Set being the main Adaptation Set for that Preselection (i.e., the id of the Atlas Tile Adaptation Set is the first id in the list of Adaptation Sets of the `@preselectionComponents` attribute in the **Preselection** element or the `@value` attribute of the Preselection descriptor). Representations of the Atlas Tile Adaptation Set of an Atlas Tile Preselection shall have an `@dependencyId` attribute set to the id of a Representation in the corresponding Atlas Adaptation Set.

The concatenation of the Initialization Segment of the Main Adaptation Set, Atlas Adaptation Set and associated Atlas Tile Adaptation Set, in order, followed by subsegments of a Representation of the Main Adaptation Set, Atlas Adaptation Set and the Adaptation Sets associated with the Atlas Tile Preselection, in any order, results in an ISOBMFF file conforming to subclause 7.4.

11.3 DASH MPD descriptors for V3C content

11.3.1 XML namespace and schema

A number of XML elements and attributes are defined in subclause 11.3 and its subclauses. These XML elements are defined in separate namespaces `"urn:mpeg:mpegI:v3c:2020"` and `"urn:mpeg:mpegI:v3c:2025"`. Namespace designator `"v3c:"` is used to refer to these namespaces in this document. New XML elements and attributes are defined in XML schema documents in each subclause where a new MPD descriptor is specified. The namespace designator `"xs:"` shall correspond to namespace of XML Schema as defined in W3C Recommendation XML Schema Part 1. Some items in the "Data type" column of the tables of this clause use datatypes and meaning as defined in W3C Recommendation XML Schema Part 2 or in ISO/IEC 23009-1. Data types not defined in W3C Recommendation XML Schema Part 2 or ISO/IEC 23009-1 shall be as defined in Annex B.

11.3.2 V3C video component descriptor

To identify the type of Video Component Adaptation Set, a `V3CVideoComponent` descriptor shall be used. A `V3CVideoComponent` descriptor is an `EssentialProperty` descriptor with the `@schemeIdUri` set to `"urn:mpeg:mpegI:v3c:2020:videoComponent"`.

At Adaptation Set level, one `V3CVideoComponent` descriptor shall be signalled. The descriptor applies for all Representations of the Video Component Adaptation Set.

The `@value` of the `V3CVideoComponent` descriptor shall not be present. The `V3CVideoComponent` descriptor shall include elements and attributes as specified in Table 14.

Table 14 — Elements and attributes for the `V3CVideoComponent` descriptor

Elements and attributes	Use	Data type	Description
-------------------------	-----	-----------	-------------

videoComponent	0..N	v3c:VideoComponentType	An element whose attributes specify information for one of the V3C video components present in the Representation(s) of the Adaptation Set.
videoComponent @type	M	xs:string	Indicates the type of the V3C video component. Value 'geom' indicates a geometry component, 'occp' indicates an occupancy component, 'attr' indicates an attribute component, and 'pack' indicates a video-coded packed V3C component.
videoComponent @is_auxiliary	CM	xs:boolean	A flag indicating whether the V3C video component information represented by the Adaptation Set, is for auxiliary video. Value true indicates that the video is an auxiliary video and contains RAW and/or EOM patches. Equal to false indicates video may contain RAW and/or EOM patches. If not present, the default value is false.
videoComponent @map_index	CM	xs:integer	Indicates the index of one of the maps of the component represented by the Adaptation Set is present. Shall only be present if the presentation contains multiple maps which are stored in different Adaptation Sets and videoComponent @type has the value 'geom', 'attr', or 'pack'.
videoComponent @attribute_type	CM	xs:unsignedByte	Indicates the type of the attribute as defined in ISO/IEC 23090-5:2021 Table 3. Only values between 0 and 15, inclusive, are allowed. Shall only be present only when videoComponent @type has the value 'attr'.
videoComponent @attribute_index	CM	xs:unsignedByte	Indicates the index of the attribute. Shall be a value between 0 and 127, inclusive. Shall only be present when videoComponent @type has the value 'attr'.
videoComponent @attribute_dimension_partition_index	CM	xs:unsignedByte	Indicates the index of the dimension partition for the attribute carried in the Adaptation Set.

			Shall only be present when videoComponent @type has the value 'attr'. The default value is 1.
videoComponent @atlas_id	OD	xs:integer	Indicates the atlas id of the component represented by the Adaptation Set. If not present, the default value is 0.
videoComponent @tile_ids	0	xs:UIntVectorType	Specifies atlas tiles related to data contained in the Adaptation Set by providing a white-space separated list of tile ID values. If not present, the Adaptation Set will contain all tiles of associated with the @atlas_id.
<p>Key: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>..<maxOccurs> (N=unbounded)</p> <p>Elements are bold; attributes are non-bold and preceded with an @.</p>			

The data types for various elements and attributes shall be as defined in the XML schema. An XML schema for the V3CVideoComponent descriptor shall be as shown below. The schema shall be represented in an XML schema that has namespace `urn:mpeg:mpegI:v3c:2020` and is specified as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="urn:mpeg:mpegI:v3c:2020"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
  elementFormDefault="qualified">
  <xs:element name="videoComponent" type="v3c:VideoComponentType"/>

  <xs:complexType name="VideoComponentType">
    <xs:attribute name="type" type="xs:string" use="required" />
    <xs:attribute name="is_auxiliary" type="xs:boolean" default="false"/>
    <xs:attribute name="map_index" type="xs:integer" />
    <xs:attribute name="attribute_type" type="xs:unsignedByte" />
    <xs:attribute name="attribute_index" type="xs:unsignedByte" />
    <xs:attribute name="attribute_dim_partition_index"
type="xs:unsignedByte" />
    <xs:attribute name="atlas_id" type="xs:integer" use="optional"
default="0" />
    <xs:attribute name="tile_ids" type="UIntVectorType" use="optional" />
  </xs:complexType>
</xs:schema>
```

11.3.3 V3C non-video component descriptor

To identify the type of Non-Video Component Adaptation Set, a V3CNonVideoComponent descriptor shall be used. A V3CNonVideoComponent descriptor is an EssentialProperty descriptor with the @schemeIdUri set to "urn:mpeg:mpegI:v3c:2025:nonVideoComponent".

At Adaptation Set level, one V3CNonVideoComponent descriptor shall be signalled. The descriptor applies for all Representations of the Non-Video Component Adaptation Set.

The @value of the V3CNonVideoComponent descriptor shall not be present. The V3CNonVideoComponent descriptor shall include elements and attributes as specified in Table 15.

Table 15 — Elements and attributes for the V3CNonVideoComponent descriptor

Elements and attributes	Use	Data type	Description
nonVideoComponent	0..N	v3c: NonVideoComponentType	An element whose attributes specify information for one of the V3C non-video components present in the Representation(s) of the Adaptation Set.
nonVideoComponent @type	M	xs:string	Indicates the type of the V3C non-video component. The value of 'mesh' indicates a basemesh or submesh component and 'acdd' indicates arithmetically coded displacement component.
nonVideoComponent @tile_ids	O	xs:UIntVectorType	Specifies atlas tiles related to data contained in the Adaptation Set by providing a white-space separated list of tile ID values. If not present, the Adaptation Set will contain all tiles of associated with the @atlas_id.
nonVideoComponent @submesh_ids	O	xs:UIntVectorType	Specifies submesh ids related to the data contained in the Adaptation Set by providing a white-space separated list of submesh ID values. Shall only be present when component @type has the value 'mesh'. When the attribute is not present, the Adaptation Set will contain all submeshes.
Key: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>..<maxOccurs> (N=unbounded) Elements are bold ; attributes are non-bold and preceded with an @.			

The data types for various elements and attributes shall be as defined in the XML schema. An XML schema for the V3CNonVideoComponent descriptor shall be as shown below. The schema shall be represented in an XML schema that has namespace `urn:mpeg:mpegI:v3c:2025` and is specified as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="urn:mpeg:mpegI:v3c:2025"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2025"
  elementFormDefault="qualified">
  <xs:element name="nonVideoComponent" type="v3c:NonVideoComponentType"/>

  <xs:complexType name="NonVideoComponentType">
    <xs:attribute name="type" type="xs:string" use="required" />
    <xs:attribute name="tile_ids" type="UIntVectorType" use="optional" />
    <xs:attribute name="submesh_ids" type="UIntVectorType" use="optional"
  />
  </xs:complexType>
</xs:schema>
```

11.3.4 V3C descriptor

A `SupplementalProperty` element with a `@schemeIdUri` equal to `"urn:mpeg:mpegI:v3c:2020:v3c"` is referred to as a V3C descriptor. At most one V3C descriptor may be present in Main Adaptation Set, Atlas Adaptation Set, Atlas Tile Adaptation Set, V3C Preselection, or Atlas Tile Preselection.

The V3C descriptor shall contain the attributes defined in Table 16.

Table 16 — Attributes for the V3C descriptor

Attributes	Use	Data type	Description
<code>v3c:@vId</code>	CM	<code>xs:string</code>	An id for the volumetric media. This attribute shall be present if multiple versions of the same volumetric media are signalled in separate Adaptation Sets in the MPD.
<code>v3c:@atlas_id</code>	CM	<code>xs:integer</code>	Indicates the atlas id for the volumetric media information in the track(s) carried by the Adaptation Set. This attribute shall be present if the volumetric media contains more than one atlas which are not alternatives.
<code>v3c:@tile_ids</code>	O	<code>xs:UIntVectorType</code>	If present, indicates the atlas tile IDs carried in the Atlas Tile Adaptation Set. The value of the <code>@tile_ids</code> attribute is a whitespace separated list of atlas tile IDs. For ISOBMFF, this shall include all tile IDs listed in the

		V3CAtlasTileSampleEntry of the V3C atlas tile track.
Key: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>.. <maxoccurs> (n="unbounded)<br/"> Elements are bold; attributes are non-bold and preceded with an @. </maxoccurs>>		

The data types for the attributes shall be as defined in the XML schema. An XML schema for the V3C descriptor shall be as shown below. The schema shall be represented in an XML schema that has namespace `urn:mpeg:mpegI:v3c:2020` and is specified as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
  targetNamespace="urn:mpeg:mpegI:v3c:2020"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
  elementFormDefault="qualified">

  <xs:attribute name="vId" type="xs:string" />
  <xs:attribute name="atlas_id" type="xs:integer" use="optional" />
  <xs:attribute name="tile_ids" type="UIntVectorType" use="optional" />

</xs:schema>
```

When more than one Atlas Preselection for the same volumetric media content are present in the MPD, a V3C descriptor with the @atlas_id attribute shall be signalled in each Atlas Preselection.

When more than one Atlas Tile Preselection for the same volumetric media content are present in the MPD, a V3C descriptor with the @tile_ids attribute, and the @atlas_id attribute if the content contain more than one atlas, shall be signalled in each Atlas Tile Preselection. If the Atlas Tile Preselections belong to different atlases, a V3C descriptor with the @atlas_id attribute shall also be signalled in each atlas Adaptation Set.

When Preselection elements for Atlas Preselections or Atlas Tile Preselections are present in an MPD, the V3C video component descriptor in each of Video Component Adaptation Sets that are part of those Preslections shall not contain the @atlas_id or @tile_ids attributes.

11.4 Supporting multiple versions of a V3C media

Multiple versions of the same volumetric media shall be signalled using separate Preselections. Preselections that represent alternative versions of the same V3C content shall contain a V3C descriptor with the same @vId value. At most one V3C descriptor shall be present at the preselection level. These Preselections are therefore alternatives to each other and the id of the main Adaptation Set of the Preselection, first id in the list of Adaptation Set ids for the @preselectionComponents, may be different (where each version of the visual volumetric media has a separate Main Adaptation Set signalled in the MPD file).

11.5 Switching codecs for V3C components

If multiple versions of a V3C component are encoded using a different codec, each version shall be signalled in a separate Adaptation Set with the value of the `@codecs` attribute set according to the codec used. Moreover, each of these Adaptation Sets shall contain a SupplementalProperty descriptor with `@schemeIdUri` set to `urn:mpeg:dash:adaptation-set-switching:2016` and `@value` is a comma-separated list of Adaptation Set IDs corresponding to the other available versions to indicate that seamless switching between Representations across the Adaptation Sets of these versions is supported. Any additional rules for supporting switching across Adaptation Sets as defined by ISO/IEC 23009-1, subclause 5.3.3.5 shall apply.

11.6 Signalling spatial regions for partial access

NOTE This clause needs to be reviewed after partial access related clause 9 has been reviewed for VDMC support.

11.6.1 Static spatial regions

If the 3D spatial regions are static (i.e., the position and dimensions of each region do not change over the presentation time), the characteristics of the spatial regions and the mappings between those regions and V3C tiles shall be signalled using a V3C3DRegions descriptor. This descriptor is a SupplementalProperty element with a `@schemeIdUri` equal to `"urn:mpeg:mpegI:v3c:2020:v3sr"`. A single V3C3DRegions descriptor shall be present at the Adaptation Set level or the Representation level in the Main Adaptation Set, or at the Preselection level for the V3C content.

The `@value` of the V3C3DRegions descriptor shall not be present. The V3C3DRegions descriptor shall include elements and attributes as specified in Table 17.

Table 17 — Elements and attributes for the V3C3DRegions descriptor

Elements and attributes	Use	Data type	Description
v3sr	0..1	v3c:spatialRegionMapType	Container element whose attributes and elements specify a mapping between a 3D spatial region and V3C tiles.
v3sr.spatialRegion	1..N	v3c:spatialRegionType	An element whose attributes define a 3D spatial region and provide a mapping between the defined region and a number of V3C tiles.
v3sr.spatialRegion <code>@id</code>	M	xs:unsignedShort	An identifier for the 3D spatial region. The value of this attribute shall match the value of the <code>region_id</code> field signalled for the corresponding region in the ISOBMFF container.
v3sr.spatialRegion <code>@type</code>	OD	xs:unsignedByte	An attribute whose value indicates the type of the spatial region. Value 0 indicates a cuboid region. Value 1 indicates a region corresponding to viewport. The remaining values are reserved. If not present, the default value is 0.
v3sr.spatialRegion <code>.cuboid</code>	CM	v3c:spatialRegionCuboidType	An element specifying a cuboid extending from the reference point of the spatial

			region. This element shall be present only when the spatialRegion@type attribute is set to 0.
v3sr.spatialRegion .cuboid@anchor	M	UIntVectorType	An attribute containing a triplet of values describing x-, y- and z-components of the bb_position for the V3CBoundingBox signalled in the corresponding ISOBMFF container. . The values in the array are in said order and the length of array is three.
v3sr.spatialRegion .cuboid@dimensions	M	UIntVectorType	An attribute containing a triplet of values describing the x-, y- and z-dimensions of the bb_scale for the V3CBoundingBox signalled in the corresponding ISOBMFF container. The values in the array are in said order and the length of array is three.
v3sr.spatialRegion .viewport	CM	v3c.spatialRegionViewport Type	An element specifying a viewport corresponding to the spatial region. This element shall be present only when the spatialRegion@type attribute is set to 1.
v3sr.spatialRegion .viewport@rvIds	M	StringVectorType	A list of space separated identifiers corresponding to the values of the @viewport_id attribute for the RV descriptor indicating viewports corresponding to this region.
v3sr.spatialRegion @tile_ids	CM	xs:UIntVectorType	Indicates the atlas tile IDs mapped to this spatial region. The value of the @tile_ids attribute is a whitespace separated list of atlas tile IDs. This attribute shall be absent in the case of single-track encapsulation of the V3C content or when at least one lod element exists.
v3sr.spatialRegion .lod	0..N	v3c:lodType	Container element whose attributes provide a LoD information and corresponding V3C tiles for that LoD.
v3sr.spatialRegion .lod@idx	M	xs:unsignedByte	An identifier that indicates the ordering on the LoDs for an associated 3D spatial region. The value of this attribute shall match the value of the lod_index field signalled for the corresponding LoD in the ISOBMFF container.
v3sr.spatialRegion .lod@tile_ids	M	xs:UIntVectorType	A list of whitespace separated identifiers corresponding to the values of the atlas tile IDs mapped to this LoD.
Key: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>..<maxOccurs> (N=unbounded) Elements are bold ; attributes are non-bold and preceded with an @.			

The data types for the various elements and attributes of the V3C3DRegions descriptor shall be as defined in the XML schema that has the namespace 'urn:mpeg:mpegI:v3c:2020' and is specified as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
  targetNamespace="urn:mpeg:mpegI:v3c:2020"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
  elementFormDefault="qualified">

  <xs:element name="v3sr" type="v3c:spatialRegionMapType" />

  <xs:complexType name="spatialRegionMapType">
    <xs:element name="spatialRegion" type="v3c:spatialRegionType"
minOccurs="1"/>
  </xs:complexType>

  <xs:complexType name="spatialRegionType">
    <xs:attribute name="id" type="xs:unsignedShort" use="required" />
    <xs:attribute name="type" type="xs:unsignedByte" use="optional"
default="0" />
    <xs:attribute name="tile_ids" type="xs:UIntVectorType" />

    <xs:element name="cuboid" type="v3c:spatialRegionCuboidType"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="viewport" type="v3c:spatialRegionViewportType"
      minOccurs="0" maxOccurs="1"/>
    <xs:element name="lod" type="v3c:lodType" />
  </xs:complexType>

  <xs:complexType name="spatialRegionCuboidType">
    <xs:attribute name="anchor" type="UIntVectorType" use="required"
      minLength="3" maxLength="3" />
    <xs:attribute name="dimensions" type="UIntVectorType" use="required"
      minLength="3" maxLength="3"/>
  </xs:complexType>

  <xs:complexType name="spatialRegionViewportType">
    <xs:attribute name="rvIds" type="StringVectorType" use="required" />
  </xs:complexType>
```

```

<xs:complexType name="lodType">
  <xs:attribute name="idx" type="xs:unsignedByte" use="required" />
  <xs:attribute name="tile_ids" type="xs:UIntVectorType" use="required"
/>

</xs:complexType>
</xs:schema>

```

11.6.2 Dynamic spatial regions

When the 3D partitions are dynamic, a timed-metadata track for signalling the position and dimensions of each 3D region in the presentation timeline shall be used shall be carried in a separate Adaptation Set with a single Representation that is associated with a Representation in the Main Adaptation Set using the @associationId attribute, defined in ISO/IEC 23009-1, and an @associationType value that includes the 4CC 'cdsc'.

11.7 Signalling recommended viewports

11.7.1 Static viewports

A **SupplementalProperty** with a @schemeIdUri equal to "urn:mpeg:mpegI:v3c:2020:rv" is defined for the Recommended Viewport (RV) descriptor in order to signal the recommended viewports of the V3C content. This descriptor may be used by content providers to signal a set of viewport position and rotation parameters recommended for rendering the V3C content. The RV descriptor indicates that each Representation in the Adaptation Set (for the multi-track case, this includes the Representations in the Main Adaptation Set and other related Adaptation Sets for the corresponding V3C components) is recommended to be rendered based on the provided set of viewport position (@vp_pos) and rotation (@vp_quat).

One or more RV descriptors may be present in each Adaptation Set for the single-track DASH mode. In the case of the multi-track DASH mode, one or more RV descriptors, if present, shall only be placed in the Main Adaptation Set. No other RV descriptor shall be present at the MPD representation level or any other level in both single-track and multi-track DASH modes.

The RV descriptor shall include elements and attributes as specified in Table 18.

Table 18 — Elements and attributes for the RV descriptor

Elements and attributes	Use	Data type	Description
@viewport_id	0	xs:integer	An identifier for the viewport.
ViewportInfo	1	v3c:ViewportInfoType	Container element whose sub-elements and attributes provide information about the viewport.
ViewportInfo @vp_pos	M	v3c:FloatVectorType	Indicates the x-, y- and z-coordinates of the position of the viewport in metres in the global reference coordinate system. The values in the array are in said order and the length of array is three. If the viewport is dynamic, this attribute specifies the initial viewport position. Otherwise, this attribute specifies a static viewport's position.

ViewportInfo @vp_quat	M	v3c:IntVectorType	Indicates the x-, y- and z-components of the rotation of the viewport using the quaternion representation. The fourth component (w) may be calculated when other components are known. The integer values shall be mapped to range -1 and 1, inclusive. If the viewport is dynamic, this attribute specifies the initial viewport rotation. Otherwise, this attribute specifies a static viewport's rotation.
ViewportInfo @vp_center_view_flag	0	xs:boolean	If equal to 1, this attribute indicates that the viewport position signalled corresponds to the centre of the viewport. If equal to 0, it indicates that the viewport position signalled corresponds to one of two stereo positions of the viewport.
ViewportInfo @vp_left_view_flag	0	xs:boolean	If equal to 1, this attribute indicates that the viewport information signalled correspond to the left stereo position of the viewport. If equal to 0, it indicates that the viewport information signalled correspond to the right stereo position of the viewport.
ViewportInfo @initialViewport	0	xs:boolean	If equal to "true", this attribute specifies that this viewport is the initial viewport that should be used out of all the recommended viewports in the current Period. If equal to "false", this attribute specifies that this viewport is not the initial viewport in the current Period. In a Period at most one viewport shall have ViewportInfo @initialViewport equal to "true".
ViewportInfo @viewport_description	0	xs:string	Null-terminated UTF-8 string describing the human-readable textual information associated with the viewport, e.g., "VIP Tribune View", "Marathon Tribune View", etc.
ViewportInfo @viewport_type	0	xs:integer	Type of the recommended viewport as listed in Table 13.
Legend: For attributes: M=Mandatory, O=Optional, OD=Optional with Default Value, CM=Conditionally Mandatory. For elements: <minOccurs>..<maxOccurs> (N=unbounded) Elements are bold ; attributes are non-bold and preceded with an @.			

The data types for the various elements and attributes of the RV descriptor shall be as defined in the XML schema that has the namespace 'urn:mpeg:mpegI:v3c:2020' and is specified as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs=http://www.w3.org/2001/XMLSchema
  targetNamespace="urn:mpeg:mpegI:v3c:2020"
  xmlns:omaf="urn:mpeg:mpegI:v3c:2020"
  elementFormDefault="qualified">

  <xs:attribute name="viewport_id" type="xs:integer" use="optional" />
```

```

<xs:element name="ViewportInfo" type="v3c:ViewportInfoType"/>

<xs:complexType name="ViewportInfoType">
  <xs:attribute name="vp_pos" type="FloatVectorType" use="required"
    minLength="3" maxLength="3"/>
  <xs:attribute name="vp_quat" type="IntVectorType" use="required"
    minLength="3" maxLength="3"/>
  <xs:attribute name="vp_center_view_flag" type="xs:boolean"
use="optional"/>
  <xs:attribute name="vp_left_view_flag" type="xs:boolean"
use="optional"/>
  <xs:attribute name="initialViewport" type="xs:boolean" use="optional"/>
  <xs:attribute name="viewport_description" type="xs:string"
use="optional"/>
  <xs:attribute name="viewport_type" type="xs:integer" use="optional"
default="0"/>
  <xs:anyAttribute processContents="skip"/>
</xs:complexType>

</xs:schema>

```

11.7.2 Dynamic viewports

For dynamic viewports where the viewport position and/or rotation change over time, a timed-metadata track with a sample entry of type '6vpt' as described in subclause 10.3 shall be used for signalling changes to the viewport position and rotation at different times in the presentation timeline. This timed-metadata track shall be carried in a separate Adaptation Set with a single Representation that is associated with a Representation in the Main Adaptation Set using the @associationId attribute, defined in ISO/IEC 23009-1, and an @associationType value that includes the 4CC 'cdsc'.

12 Encapsulation and signalling in MMT

12.1 Introduction

For the carriage of V3C content using MMT, two types of signaling information are provided. One is the signaling information provided per MMT Package or MMT Assets by using the general MMT signaling messages, tables, and descriptors which are agnostic to specific content types or media types. Such information is used by the MMT receiving entity to understand the structure of MMT Packages or the properties of MMT Assets. The other is signaling information specific to V3C content which provides the structure of the V3C content and properties specific to the V3C content or its components. As depicted in Figure 7, the general MMT signaling information is processed by the MMT Receiving Entity and the V3C content specific information is processed by the V3C content specific information processing entity. In some cases, feedback may be provided from the V3C content specific information processing entity to the MMT Receiving Entity according to the V3C content specific signaling information.

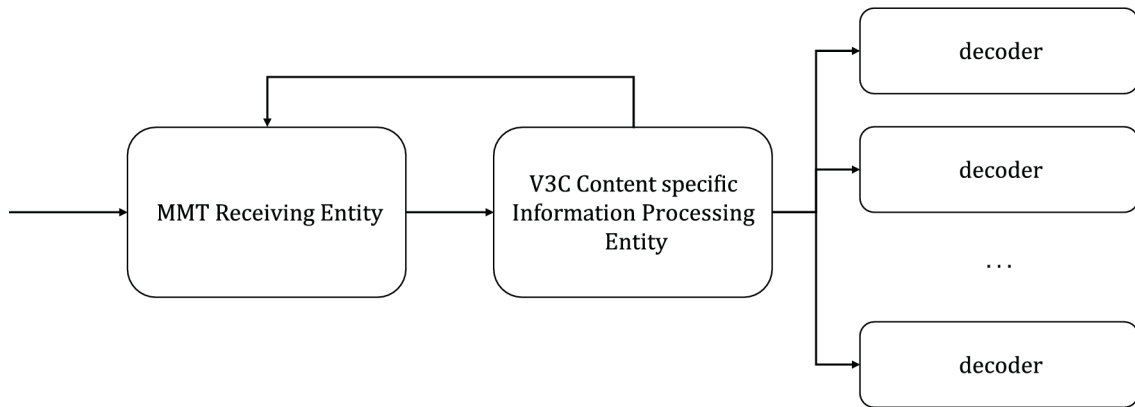


Figure 7 — Conceptual architecture of the client receiving V3C Content with MMT

12.2 MMT signalling descriptors for V3C content

12.2.1 Asset reference descriptor

12.2.1.1 General

This descriptor signals reference relationship information among Assets. The Asset Reference descriptor shall be added to the Asset descriptor loop of the signalling messages or signalling tables (e.g., MP table) of an Asset referencing other Assets.

For a V3C content, an Asset Reference descriptor is carried in the Asset descriptor loop of signalling messages or signalling tables of an Asset for the V3C atlas bitstream. The descriptor includes track references to the Assets carrying V3C video component bitstreams with the track reference types for each V3C video components.

12.2.1.2 Syntax

Table 19 shows the syntax of the Asset Reference descriptor.

Table 19 — Asset Reference descriptor

Syntax	Values	No. of bits	Mnemonic
<pre> Asset_Reference_Descriptor () { descriptor_tag descriptor_length group_identification number_of_reference for (i=0 ; i<N ; i++){ reference_type asset_id() } } </pre>	N	16 8 8 8 32	uimsbf uimsbf uimsbf uimsbf char

12.2.1.3 Semantics

`group_identification` specifies the group identification which identifies a group of Assets in reference relationship.

`number_of_reference` specifies the number of referenced Assets by the Asset this descriptor is associated with.

`reference_type` specifies the type of reference. This is described in four-character code ("4CC") type registered in MP4RA (<http://www.mp4ra.org>).

`asset_id()` provides the identifier of the Asset referenced by the Asset this descriptor is associated with, as defined in ISO/IEC 23008-1:2017, subclause 10.6.2.

12.2.2 V3C Asset descriptor

12.2.2.1 General

This Asset descriptor is used to inform the receiving entity and the consuming application about the content of an Asset that carries V3C content. This descriptor shall be added to the Asset descriptor loop of the signalling messages or signalling tables (e.g., MP table) of an Asset comprising a V3C content.

12.2.2.2 Syntax

Table 20 shows the syntax of the V3C Asset descriptor.

Table 20 — V3C Asset descriptor

Syntax	Values	No. of bits	Mnemonic
V3C_descriptor () {			
descriptor_tag		16	uimsbf
descriptor_length		16	uimsbf
data_type		8	uimsbf
all_tiles_present_flag		1	bslbf
reserved	'1111111'	7	bslbf
if (!all_tiles_present_flag) {			
num_tiles	N1	16	uimsbf
for (i=0; i<N1; i++) {			
tile_id		16	uimsbf
}			
}			
}			

12.2.2.3 Semantics

`descriptor_tag` indicates the type of a descriptor.

`descriptor_length` specifies the length in bytes counting from the next byte after this field to the last byte of the descriptor.

`data_type` indicates the type of V3C data present in this Asset group. Values for this field are listed in Table 21.

Table 21 — Values for `data_type`

Value	Description
0x00	All V3C components data (i.e., V3C atlas and V3C video components)
0x01	Atlas component data
0x02	Occupancy component data
0x03	Geometry component data
0x04	Attribute component data
0x05	Dynamic volumetric timed-metadata Information
0x06	Viewport timed-metadata information
0x07	Packed video component data
0x08	Mesh component data
0x09	Arithmetic Coded Displacement component data
0x0A-0xFF	Reserved

`all_tiles_present_flag` indicates whether all the tiles for the atlas component are part of an Asset or not. Value 1 indicates that data for all the atlas tiles are available in the Asset. Value 0 indicates that data for a sub-set of the atlas tiles are available in the Asset.

`num_tiles` indicate the number of tiles carried in this Asset.

`tile_id` indicates a unique identifier for a particular atlas tile.

12.3 MMT signalling messages for V3C Content

12.3.1 General

Several MMT signalling messages are defined for V3C content. The V3C content-specific messages shall have the value of the `message_id` field set to 0x0000 and the value of the `application_id` field set to urn:mpeg:mmt:app:v3c:2020 for identification. The type of the message is identified by the value of the `v3c_application_message_type` field as listed in Table 22.

Table 22 — Values for `v3c_application_message_type`

Application Message Type	Application Message Name
0x01	V3CAssetGroupMessage
0x02	V3CselectionMessage
0x03	V3CviewChangeFeedbackMessage
0x04-0xFF	Reserved

12.3.2 V3C Asset Group message

12.3.2.1 General

When sending V3C content via MMT, the V3CassetGroupMessage shall be used. This message provides the receiving entity with the information about the Assets associated with the V3C content. A receiving entity may then request a unique sub-set of these V3C Assets using the V3CSelectionMessage message. This message may also be used to inform the receiving entity about which of these Assets are currently being streamed to the receiving entity.

12.3.2.2 Syntax

Table 23 shows the syntax of the V3C Asset Group message.

Table 23 — V3C Asset Group message

Syntax	Values	No. of bits	Mnemonic
V3C_asset_group_message () { <i>message_id</i> <i>version</i> <i>length</i> message_payload { application_identifier() <i>v3c_application_message_type</i> <i>num_v3c_asset_groups</i> <i>start_time</i> for (i=0; i<N1; i++) { <i>v3c_asset_group_id</i> <i>pending_flag</i> <i>3D_spatial_region_info_flag</i> <i>reserved</i> if (<i>3D_spatial_region_info_flag</i>){ <i>num_regions</i> for (j=0; j<N2; j++) { <i>V3CSpatialRegion()</i> } } } } }	N1	16 8 8 16 8 1 1 6 16	uimbsf uimbsf uimbsf uimbsf bslbf bslbf bslbf uimbsf

12.3.2.3 Semantics

message_id indicates the identifier of the V3C application message.

version indicates the version of the V3C application message.

length indicates the length of the V3C application message in bytes, counting from the beginning of the next field to the last byte of the message. The value of this field shall not be equal to 0.

`application_identifier` indicates the application identifier as a URN that uniquely identifies the application to consume the contents of this message.

`v3c_application_message_type` indicates the type of the V3C application message.

`num_v3c_asset_groups` indicates the number of V3C Asset groups, where each group contains the Assets associated with a V3C component.

`start_time` indicates the presentation time of the V3C content from which the state of the Assets listed in this message are applicable.

`v3c_asset_group_id` indicates the value of the group identification field of the Asset Reference descriptor carried in the Asset descriptor loop of the Asset for the V3C atlas bitstream or V3C atlas tile bitstream.

`pending_flag` indicates if all the data components for an Asset group are ready for rendering. When set to 1, it indicates that the data is ready, otherwise the flag is 0.

`3D_spatial_region_info_flag` indicates whether 3D spatial region information is present for an Asset group or not. Value 0 indicates this 3D spatial region information is not provided. Value 1 indicates this 3D spatial region information is provided.

`num_regions` indicates the number of 3D spatial region information.

`V3CSpatialRegion()` is an instance of `V3CSpatialRegion` defined in subclause 9.3 of this document and carries the information of the 3D spatial regions covered by the Asset group.

`asset_id()` provides the Asset identifier of the Asset as defined in ISO/IEC 23008-1, subclause 10.6.2.

12.3.3 V3C Selection message

12.3.3.1 General

The client uses this feedback message to request the set of Assets to be streamed by the sending entity.

12.3.3.2 Syntax

Table 24 shows the syntax of the V3C Selection message.

Table 24 — V3C Selection Message

Syntax	Values	No. of bits	Mnemonic
--------	--------	-------------	----------

V3C_selection_message () { <i>message_id</i> <i>version</i> <i>length</i> message_payload { application_identifier() <i>v3c_application_message_type</i> <i>num_selected_asset_groups</i> for (i=0; i<N1; i++) { <i>v3c_asset_group_id</i> <i>reserved</i> <i>switching_mode</i> <i>num_assets</i> if (switching_mode == 0x01 0x02) { for (j=0; j<N2; j++) { <i>asset_id()</i> } } } }		16 8 16	uimsbf uimsbf uimsbf
	N1	8	uimsbf
	'1111'	8 4 4	uimsbf bslbf bslbf
	N2	16	uimsbf

12.3.3.3 Semantics

message_id indicates the identifier of the V3C application message.

version indicates the version of the V3C application message.

length indicates the length of the V3C application message in bytes, counting from the beginning of the next field to the last byte of the message. The value of this field shall not be equal to 0.

application_identifier indicates the application identifier as a URN that uniquely identifies the application to consume the contents of this message.

v3c_application_message_type indicates the type of the V3C application message.

num_selected_asset_groups indicates the number of Asset groups for which there is an associated state change request by the receiving entity.

v3c_asset_group_id is the value of the group identification field of the Asset_Reference_Descriptor carried in the asset descriptor loop of the Asset for the V3C Atlas bitstream or V3C Atlas tile bitstream

switching_mode indicates the switching mode used for the selection of assets as requested by the receiving entity. Switching modes are described in Table 25.

Table 25 — Switching modes and corresponding definition

Value	Switching mode	Definition of switching mode
-------	----------------	------------------------------

0x1	Refresh	For each asset listed as specified by its <i>asset_id</i> , its <i>state_flag</i> will be set to “1”, and the <i>state_flag</i> for all other non-listed assets of the same <i>asset_group_id</i> will be set to “0”. The states for assets of other non listed asset groups will remain unchanged.
0x2	Toggle	For each asset listed as specified by its <i>asset_id</i> , its <i>state_flag</i> will be changed (to “1”, if originally “0”, to “0” if originally “1”). The states for all non listed assets will remain unchanged
0x3	Send all	For the specified asset group, all associated assets within the group have their <i>state_flag</i> set to “1”.
0x4~0xF	Reserved	Reserved

num_assets indicates the number of Assets signalled for the state change according to the switching mode specified.

asset_id() provides the asset identifier of the Asset, as defined in ISO/IEC 23008-1:2017, subclause 10.6.2, for the state change according to the switching mode specified.

12.3.4 V3C View Change Feedback message

12.3.4.1 General

For view-dependent delivery of V3C content through MMT, the client may use the V3C View Change Feedback message to send its current viewport information to the server, after which the server can select and deliver the Assets corresponding to that viewport to the client.

12.3.4.2 Syntax

Table 26 shows the syntax of the V3C View Change Feedback message.

Table 26 — V3C View Change Feedback message

Syntax	Values	No. of bits	Mnemonic
V3C_view_change_feedback_message () {			
<i>message_id</i>		16	uimsbf
<i>version</i>		8	uimsbf
<i>length</i>		16	uimsbf
message_payload {			
application_identifier()			
<i>v3c_application_message_type</i>		8	uimsbf
<i>vp_pos_x</i>		32	uimsbf
<i>vp_pos_y</i>		32	uimsbf
<i>vp_pos_z</i>		32	uimsbf
<i>vp_quat_x</i>		32	uimsbf
<i>vp_quat_y</i>		32	uimsbf
<i>vp_quat_z</i>		32	uimsbf
<i>clipping_near_plane</i>		32	uimsbf
<i>clipping_far_plane</i>		32	uimsbf

<i>horizontal_fov</i>		32	uimsbf
<i>vertical_fov</i>		32	uimsbf
<i>last_processed_media_timestamp</i>		32	uimsbf
}			
}			

12.3.4.3 Semantics

`message_id` indicates the identifier of the V3C application message.

`version` indicates the version of the V3C application message.

`length` indicates the length of the V3C application message in bytes, counting from the beginning of the next field to the last byte of the message. The value of this field shall not be equal to 0.

`application_identifier` indicates the application identifier as a URN that uniquely identifies the application to consume the contents of this message.

`v3c_application_message_type` indicates the type of the V3C application message.

`vp_pos_x`, `vp_pos_y`, `vp_pos_z` respectively indicates the x, y and z coordinates of the position of the viewport in metres in the global reference coordinate system. The values are in units of 2^{-16} metres.

`vp_quat_x`, `vp_quat_y`, `vp_quat_z` indicates the x, y, and z components, respectively, of the rotation of the viewport region using the quaternion representation.

`clipping_near_plane`, `clipping_far_plane` indicates the near and far depths (or distances) based on the near and far clipping planes of the viewport in metres.

`horizontal_fov` specifies the longitude range corresponding to the horizontal size of the viewport region, in radians. The value is in the range 0 to 2π .

`vertical_fov` specifies the latitude range corresponding to the vertical size of the viewport region, in radians. The value is in the range 0 to π .

`last_processed_media_timestamp` indicates the presentation timestamp of the last media unit that has been appended to the decoder buffer. This field is used by the MMT sending entity to determine the next media unit from the new asset that is sent to the V3C player. The next media unit is the one with a timestamp or sequence number immediately following the indicated timestamp.

12.4 Carriage of V3C content with MMT

12.4.1 General

When a V3C bitstream is carried with MMT, entire V3C bitstream shall be completely included in a single MMT Package and each component bitstreams encapsulated in a single track or item shall be carried as an individual MMT Assets of the Package. MMT Package can carry a V3C bitstream coded with video-based point cloud coding (ISO/IEC 23090-5 Annex H), MPEG immersive video (ISO/IEC 23090-12) or video-based dynamic mesh coding (ISO/IEC 23090-29). An MMT Asset of MMT Package carrying V3C bitstream may contain a complete V3C bitstream composed of multiple components or a single component of V3C bitstream. Each MMT Assets consisting an MMT Package for V3C bitstream shall include V3C Asset descriptor. The MMT Assets carrying atlas component of V3C bitstream, the value of

the `data_type` field of the V3C Asset descriptor included in Asset descriptor loop of the Asset, shall include Asset Reference descriptor in Asset descriptor loop. The value of the `reference_type` field of such Asset Reference descriptor shall be one of the track reference type listed in the subclause 5.3.6.

For carriage of V3C bitstream, MPU as defined in subclause 7 of ISO/IEC 23008-1 shall be used. For carriage of timed V3C content timed MPU shall be used and non-timed MPU shall be used for carriage of non-timed V3C content. When an ISOBMFF file containing a single V3C bitstream includes only one file level meta box for multiple items, all MMT Assets for each item of such V3C bitstream shall include the same identical meta box. Other than that carriage of V3C content by MMT is agnostic to the type of the V3C content whether it is timed or non-timed.

12.4.2 Carriage of timed/non-timed V-PCC data with MMT

In this clause, a V3C bitstream refers to the bitstream which is coded with video-based point cloud coding (ISO/IEC 23090-5 Annex H).

12.4.2.1 Carriage of single-track/single-item encapsulated V-PCC data with MMT

When a V3C bitstream is represented by a single-track or single-item declaration, entire V3C bitstream is carried as a single MMT Asset. In this case, more than one MMT Asset for a same V3C bitstream shall not be present and any MMT Asset carrying V3C video component of the same V3C bitstream shall not be present.

When a V3C bitstream is carried as a single MMT Asset, the Asset descriptor loop of such Asset shall include a V3C Asset descriptor with the value of the `data_type` field equals to 0x00 and the value of `all_tiles_present_flag` equals to 1. In this case, Asset Reference descriptor shall not present.

12.4.2.2 Carriage of multi-track/multi-item encapsulated V-PCC data with MMT

When a V3C bitstream is encapsulated in multiple tracks or multiple items, the V3C bitstream is represented by a more than two MMT Assets. In this case, an MMT Package carrying a single V3C bitstream shall be consisted with at least one MMT Asset whose Asset descriptor loop includes V3C Asset descriptor with the value of the `data_type` field equal to 0x01 and at least one MMT Asset whose Asset descriptor loop includes V3C Asset descriptor with the value of `data_type` field equal to a number among 0x02, 0x03, 0x04 and 0x07.

When V3C atlas tiles are present in the V3C bitstream, the Asset descriptor loop of the Asset carrying V3C atlas bitstream shall include Asset Reference descriptor to the Assets carrying V3C atlas tile bitstream and the Asset descriptor loop of the Assets carrying V3C atlas tile bitstream shall include Asset Reference descriptor providing references to the Assets carrying V3C video component bitstreams. When a V3C bitstream has multiple V3C atlases, then the Asset descriptor loop of the Asset carrying common information applicable to all V3C atlases shall include Asset Reference descriptors to the Assets carrying V3C atlas bitstreams.

12.4.3 Carriage of timed/non-timed MIV data with MMT

In this clause, a V3C bitstream refers the bitstream which is coded with MPEG immersive video (ISO/IEC 23090-12).

12.4.3.1 Carriage of single-track/single-item encapsulated MIV data with MMT

When a V3C bitstream is represented by a single-track or single-item declaration, entire V3C bitstream is carried as a single MMT Asset. In this case, more than one MMT Asset for a same V3C bitstream shall not

be present and any MMT Asset carrying V3C video component of the same V3C bitstream shall not be present.

When a V3C bitstream is carried as a single MMT Asset, the Asset descriptor loop of such Asset shall include a V3C Asset descriptor with the value of the `data_type` field equals to 0x00 and the value of the `all_tiles_present_flag` equals to 1. In this case, Asset Reference descriptor shall not present.

12.4.3.2 Carriage of multi-track/multi-item encapsulated MIV data with MMT

When a V3C bitstream is encapsulated in multiple tracks or multiple items, the V3C bitstream is represented by a more than two MMT Assets. In this case, an MMT Package carrying a single V3C bitstream shall include one or more MMT Asset carrying V3C Asset descriptor with the value of the `data_type` field equal to 0x01 and at least one MMT Asset carrying V3C Asset descriptor with the value of `data_type` field equal to a number among 0x03, 0x04 and 0x07.

When V3C atlas tiles are present in the V3C bitstream, the Asset descriptor loop of the Asset carrying V3C atlas bitstream shall include Asset Reference descriptor to the Assets carrying V3C atlas tile bitstream and the Asset descriptor loop of the Assets carrying V3C atlas tile bitstream shall include Asset Reference descriptor providing references to the Assets carrying V3C video component bitstreams. When a V3C bitstream has multiple V3C atlases, then the Asset descriptor loop of the Asset carrying common information applicable to all V3C atlases shall include Asset Reference descriptors to the Assets carrying V3C atlas bitstreams.

12.4.4 Carriage of timed/non-timed V-DMC data with MMT

In this clause, a V3C bitstream refers to the bitstream which is coded with video-based dynamic mesh coding (ISO/IEC 23090-29).

12.4.4.1 Carriage of single-track/single-item encapsulated V-DMC data with MMT

When a V3C bitstream is represented by a single-track or single-item declaration, entire V3C bitstream is carried as a single MMT Asset. In this case, more than one MMT Asset for a same V3C bitstream shall not be present and any MMT Asset carrying V3C video component of the same V3C bitstream shall not be present.

When a V3C bitstream is carried as a single MMT Asset, the Asset descriptor loop of such Asset shall include a V3C Asset descriptor with the value of the `data_type` field equals to 0x00 and the value of the `all_tiles_present_flag` equals to 1. In this case, Asset Reference descriptor shall not present.

12.4.4.2 Carriage of multi-track/multi-item encapsulated V-DMC data with MMT

When a V3C bitstream is encapsulated in multiple tracks or multiple items, the V3C bitstream is represented by a more than two MMT Assets. In this case, an MMT Package carrying a single V3C bitstream shall include at least one MMT Asset carrying V3C Asset descriptor with the value of the `data_type` field equal to 0x01 and at least one MMT Asset carrying V3C Asset descriptor with the value of `data_type` field equal to a number among 0x03, 0x04, 0x07, 0x08 and 0x09. A MMT Asset carrying V3C Asset descriptor with the value of `data_type` field equal to 0x03 and 0x09 shall not be used for the same MMT Package.

When V3C atlas tiles are present in V3C bitstream, the Asset descriptor loop of the Asset carrying V3C atlas bitstream shall include Asset Reference descriptor to the Assets carrying V3C atlas tile bitstream and the Asset descriptor loop of the Assets carrying V3C atlas tile bitstream shall include Asset Reference descriptor providing references to the Assets carrying V3C video component bitstreams. When a V3C bitstream has multiple V3C atlases, then the Asset descriptor loop of the Asset carrying common information applicable to all V3C atlases shall include Asset Reference descriptors to the Assets carrying V3C atlas bitstreams.

When submeshes are present in a V3C bitstream, the MMT Asset carrying basemesh shall include an Asset Reference descriptor providing references to the MMT Asset carrying submeshes in Asset descriptor loop of such Asset.

12.5 Carriage of alternative information of V3C contents in MMT

V3C contents can have alternative relationship. When there are alternative V3C contents or alternative V3C components then such relationship is represented by the alternative track grouping mechanism as specified in subclause 7.2.4. When an ISOBMFF file containing V3C content with alternative V3C contents or alternative V3C components is carried with MMT, MMT alternative asset descriptor shall be included in the Asset descriptor loop of the Asset which has alternative relationship with other Assets.

When the value of the data_type field of the Asset whose Asset descriptor loop includes MMT alternative asset descriptor equals to 0x00, then each Assets listed in the MMT alternative asset descriptor carries an alternative V3C content. When the value of the data_type field of the Asset whose Asset descriptor loop includes MMT alternative asset descriptor equals to 0x01, then each Assets listed in the MMT alternative asset descriptor carries V3C atlas for an alternative V3C content. When the value of the data_type field of the Asset whose Asset descriptor loop includes MMT alternative asset descriptor equals to a number among 0x02, 0x03, 0x04, 0x07, 0x08 and 0x09, then each Assets listed in the MMT alternative asset descriptor carries alternative V3C component of same type.

Annex A (normative)

File format toolsets and brands

A.1 General

This annex defines what constitutes tools, for the purposes of branding files containing visual volumetric video-based coding content. A specific brand may require some or all of the tools indicated here. A brand should be chosen that indicates the full level of support required, including any requirements on other specifications (e.g., support for aspects of ISO/IEC 14496-12).

NOTE This clause is only applicable to V-PCC and MIV applications.

A.2 Single-track encapsulation of V3C data

The brand 'v3st' may be present among the `compatible_brands` list of the `FileTypeBox`. File readers conforming to the 'v3st' brand shall support single track encapsulation of V3C data specified in subclause 7.3.

A.3 Multi-track encapsulation of V3C data

A.3.1 Requirements on files

Files containing the brands 'v3mt' and 'v3mp' in the `compatible_brands` array of the `FileTypeBox` shall conform to the constraints defined in this subclause.

The boxes listed in Table A.1 are required in a file under the 'v3mt' and 'v3mp' brands. The Version column in the following table lists the versions of the boxes allowed by this brand. Other versions of the boxes shall not be present.

Table A.1 — Required boxes in a file under the 'v3mt' and 'v3mp' brands.

Hierarchy of boxes								Version	Box description
ftyp								-	file type and compatibility
moov									movie presentation
	trak								track
	mdia								media declaration
		hdlr							handler, declares the handler type for the track
		minf							media information
			vvhd					0	volumetric visual media header
			stbl						sample table
				stsd					sample description table
					-				sample entry

						v3cC		0	V3C decoder configuration
						vunt		0	V3C unit header information
					resv				restricted video sample entry
						rinf			restricted scheme information
							frma		original format
							schm		scheme type
							schI		scheme information

The following constraint applies in the case of the 'v3mp' brand:

- If volumetric annotation SEI messages are carried by the atlas sub-bitstream and object information in the scene object information SEI messages do not change over time, scene object information SEI message should be carried in the `setup_unit` arrays in the `V3CDecoderConfigurationRecord`.

A.3.2 Requirements on readers

Support for the boxes listed in Table A.2 is required under the 'v3mt' and 'v3mp' brands. The Version column in the following table specifies the versions of the boxes that shall be supported by the readers of the 'v3mt' and 'v3mp' brands.

Table A.2 — Boxes to be supported under the 'v3mt' and 'v3mp' brands.

Hierarchy of boxes								Version	Box description
ftyp								-	file type and compatibility
mdat								-	media data
free								-	free space
skip								-	
moov									movie presentation
	trak								track
	mdia								media declaration
		hdlr							handler, declares the handler type for the track
		minf							media information
			vvhd						volumetric visual media header
			stbl						sample table
				std					sample description table
					-				sample entry
						v3cC		0	V3C decoder configuration
						vunt		0	V3C unit header information
					resv				restricted video sample entry
						rinf			restricted scheme information

							frma		original format
							schm		scheme type
							schl		scheme information

In addition to boxes listed in Table Y, support for the boxes listed in Table A.3 is required under the 'v3mp' brand. The Version column in the table specifies the versions of the boxes that shall be supported by the readers of the 'v3mp' brand.

Table A.3 — Additional boxes to be supported under the 'v3mp' brand.

Hierarchy of boxes							Version	Box description	
						vpbb			volumetric media bounding box
						v3sc			static spatial regions collection
						6vpC			viewport information configuration
						v3tC			V3C atlas tile configuration

Readers shall recognize the sample entries in Table A.4 under the 'v3mt' brand:

Table A.4 — Sample entries to be recognized under the 'v3mt' brand.

Four-character code	Name of the sample entry
v3c1	multi-track with single atlas, all atlas parameter sets and SEI messages in decoder configuration record
v3cg	multi-track with single atlas, all atlas parameter sets and SEI messages in decoder configuration record or in samples
v3cb	base track in the multi-track mode with multiple atlases
v3a1	atlas track in multi-track mode with multiple atlases, atlas parameter sets and SEI messages carried in decoder configuration record
v3ag	atlas track in multi-track mode with multiple atlases, atlas parameter sets and SEI messages carried in decoder configuration record and in samples

Readers shall recognize the sample entries in Table A.5 under the 'v3mp' brand:

Table A. — Sample entries to be recognized under the 'v3mp' brand.

Four-character code	Name of the sample entry
v3c1	multi-track with single atlas, all atlas parameter sets and SEI messages in decoder configuration record
V3cg	multi-track with single atlas, all atlas parameter sets and SEI messages in decoder configuration record or in samples
v3cb	base track in the multi-track mode with multiple atlases

v3a1	atlas track in multi-track mode with multiple atlases, atlas parameter sets and SEI messages carried in decoder configuration record
v3ag	atlas track in multi-track mode with multiple atlases, atlas parameter sets and SEI messages carried in decoder configuration record and in samples
v3t1	atlas tile track in the multi-track mode
dyvm	timed metadata track indicating the dynamic spatial regions
6vpt	timed metadata track indicating viewport information

Readers shall recognize the track groups in Table A.6 under the 'v3mt' brand:

Table A.6 — Track groups to be recognized under the 'v3mt' brand.

Four-character code	Name of track group
potg	playout track group

Readers shall recognize the track groups in Table A.7 below under the 'v3mp' brand:

Table A.7 — Track groups to be recognized under the 'v3mp' brand.

Four-character code	Name of track group
potg	playout track group
vtcg	V3C tile components track group

Readers shall recognize the reference types in Table A.8 below under the 'v3mt' brand:

Table A.8 — Reference types to be recognized under the 'v3mt' brand.

Four-character code	Name of the reference type
v3cs	reference to track carry atlas data
v3vo	reference to track carrying occupancy data
v3vg	reference to track carrying geometry data
v3va	reference to track carrying attribute data

Readers shall recognize the reference types in Table A.9 below under the 'v3mp' brand:

Table A.9 — Reference types to be recognized under the 'v3mp' brand.

Four-character code	Name of the reference type
v3cs	reference to track carry atlas data
v3vo	reference to track carrying occupancy data
v3vg	reference to track carrying geometry data
v3va	reference to track carrying attribute data
v3ct	reference to track carrying V3C atlas tile

Readers shall recognize the sample groups in Table A.10 below under the 'v3mt' and 'v3mp' brands:

Table A.10 — Sample group types to be recognized under the 'v3mt' and 'v3mp' brands.

Four-character code	Name of the sample group type
vaps	sample group for atlas parameter sets

Readers shall recognize the restricted video schemes in Table A.11 below under the 'v3mt' and 'v3mp' brands:

Table A.11 — Restricted scheme types to be recognized under the 'v3mt' and 'v3mp' brands.

Four-character code	Name of the restricted scheme type
vVVC	V3C component video

A.4 Encapsulation of non-timed V3C data

A.4.1 Requirements on files

Files containing the brand 'v3nt' in the `compatible_brands` array of the `FileTypeBox` shall conform to the constraints defined in this subclause.

The boxes listed in Table A.12 may be required in a file under the 'v3nt' brand. The Version column in the following table lists the versions of the boxes allowed by this brand. Other versions of the boxes shall not be present.

Table A.12 — Required boxes in a file under the 'v3nt' brand.

Hierarchy of boxes			Version	Box description
ftyp			-	file type and compatibility
meta			0	metadata
	hdlr		0	handler, declares the metadata (handler) type
	iloc		0, 1, 2	item location
	iinf		0, 1	item information
		infe	2, 3	item information entry
	pitm		0, 1	primary item reference
	iprp		-	item properties
		v3cC	0	V3C configuration item property
		vutp	0	V3C unit header item property
	iref		0, 1	item reference box

Note that the brand 'v3nt' does not mandate a `MovieBox` ('moov') and therefore no brand from Annex E of 14496-12 is mandated.

A.4.2 Requirements on readers

Support for the boxes listed in Table A.13 is required under the 'v3nt' brand. The Version column in the following table specifies the versions of the boxes that shall be supported by the readers of the 'v3nt' brand.

Table A.13 — Boxes to be supported under the 'v3nt' brand.

Hierarchy of boxes			Version	Box description
ftyp			-	file type and compatibility
mdat			-	media data container
free			-	free space
skip			-	free space
meta			0	metadata
	hdlr		0	handler, declares the metadata (handler) type
	grpl			group list box
	dinf		-	data information box, container
		dref	0	data reference box, declares source(s) of items
	iloc		0, 1, 2	item location
	iinf		0, 1	item information
		infe	2, 3	item information entry
	iref		0, 1	item reference box
	pitm		0, 1	primary item reference
	idat		-	item data
	iprp		-	item properties

Readers shall support all the construction methods of the `ItemLocationBox`, and the construction of the data of items from multiple extents.

Readers shall recognize the item properties in Table A.14:

Table A.14 — Item properties to be recognized under the 'v3nt' brand.

Four-character code	Name of the property
v3cC	V3C configuration item property
vutp	V3C unit header item property
v3tp	V3C atlas tile configuration item property
hdlp	handler property

Readers shall recognize the entity group in Table A.15:

Table A.15 — Entity groups to be recognized under the 'v3nt' brand.

Four-character code	Name of the entity group
eply	playout entity group box

Readers shall recognize the reference types in Table A.16:

Table A4A — Reference types to be recognized under the 'v3nt' brand.

Four-character code	Name of the reference type
v3vo	reference to item carrying occupancy data
v3vg	reference to item carrying geometry data
v3va	reference to item carrying attribute data
v3ct	reference to item carrying V3C atlas tile

Annex B (normative)

V3C DASH schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="urn:mpeg:mpegI:v3c:2020"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
  elementFormDefault="qualified">

  <xs:element name="videoComponent" type="v3c:VideoComponentType" />
  <xs:complexType name="VideoComponentType">
    <xs:attribute name="type" type="xs:string" use="required" />
    <xs:attribute name="is_auxiliary" type="xs:boolean" default="false" />
    <xs:attribute name="map_index" type="xs:integer" />
    <xs:attribute name="attribute_type" type="xs:unsignedByte" />
    <xs:attribute name="attribute_index" type="xs:unsignedByte" />
    <xs:attribute name="attribute_dim_partition_index"
type="xs:unsignedByte" />
    <xs:attribute name="atlas_id" type="xs:integer" use="optional"
default="0" />
    <xs:attribute name="tile_ids" type="UIntVectorType" use="optional" />
  </xs:complexType>

  <xs:attribute name="vId" type="xs:string" />
  <xs:attribute name="atlas_id" type="xs:integer" use="optional" />
  <xs:attribute name="tile_ids" type="UIntVectorType" use="optional" />

  <xs:element name="v3sr" type="v3c:spatialRegionMapType" />
  <xs:complexType name="spatialRegionMapType">
    <xs:element name="spatialRegion" type="v3c:spatialRegionType"
minOccurs="1"/>
  </xs:complexType>

  <xs:complexType name="spatialRegionType">
    <xs:attribute name="id" type="xs:unsignedShort" use="required" />
    <xs:attribute name="type" type="xs:unsignedByte" use="optional"
default="0" />
```

```

        <xs:attribute name="tile_ids" type="UIntVectorType" />
        <xs:element name="cuboid" type="v3c:spatialRegionCuboidType"
minOccurs="0" maxOccurs="1"/>
        <xs:element name="lod" type="v3c:lodType" />
        <xs:element name="viewport" type="v3c:spatialRegionViewportType"
minOccurs="0" maxOccurs="1"/>
</xs:complexType>

<xs:complexType name="spatialRegionCuboidType">
    <xs:attribute name="anchor" type="UIntVectorType" use="required"
minLength="3" maxLength="3" />
    <xs:attribute name="dimensions" type="UIntVectorType" use="required"
minLength="3" maxLength="3"/>
</xs:complexType>

<xs:complexType name="spatialRegionViewportType">
    <xs:attribute name="rvIds" type="StringVectorType" use="required" />
</xs:complexType>

<xs:complexType name="lodType">
    <xs:attribute name="idx" type="xs:unsignedByte" use="required" />
    <xs:attribute name="tile_ids" type="UIntVectorType" use="required" />
</xs:complexType>

<!-- Added support for float and int vectors -->
<xs:simpleType name="FloatVectorType">
    <xs:list itemType="xs:float"/>
</xs:simpleType>
<xs:simpleType name="IntVectorType">
    <xs:list itemType="xs:integer"/>
</xs:simpleType>

<xs:attribute name="viewport_id" type="xs:integer" use="optional" />
<xs:element name="ViewportInfo" type="v3c:ViewportInfoType"/>

<xs:complexType name="ViewportInfoType">
    <xs:attribute name="vp_pos" type="FloatVectorType" use="required"
minLength="3" maxLength="3"/>
    <xs:attribute name="vp_quat" type="IntVectorType" use="required"

```

```

        minLength="3" maxLength="3"/>
        <xs:attribute name="vp_center_view_flag" type="xs:boolean"
use="optional"/>
        <xs:attribute name="vp_left_view_flag" type="xs:boolean"
use="optional"/>
        <xs:attribute name="initialViewport" type="xs:boolean" use="optional"/>
        <xs:attribute name="viewport_description" type="xs:string"
use="optional"/>
        <xs:attribute name="viewport_type" type="xs:integer" use="optional"
default="0"/>
        <xs:anyAttribute processContents="skip"/>
    </xs:complexType>
</xs:schema>

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"

    targetNamespace="urn:mpeg:mpegI:v3c:2025"
    xmlns:v3c="urn:mpeg:mpegI:v3c:2025"
    elementFormDefault="qualified">

    <xs:element name="nonVideoComponent" type="v3c:NonVideoComponentType" />
    <xs:complexType name="NonVideoComponentType">
        <xs:attribute name="type" type="xs:string" use="required" />
        <xs:attribute name="tile_ids" type="UIntVectorType" use="optional" />
        <xs:attribute name="submesh_ids" type="UIntVectorType" use="optional"
/>
    </xs:complexType>

</xs:schema>

```

Annex C (normative)

MIME types and sub-parameters

C.1 MIME types and sub-types

When MIME type is associated with V3C content as described in this document, MIME type of ‘application’ shall be used along with the sub-type ‘mp4’.

EXAMPLE Content-Type: application/mp4.

C.2 Sub-parameters for ‘codecs’ parameter

C.2.1 General

When the ‘codecs’ parameter of a MIME type is used, as defined in IETF RFC 6381, the sub-parameters in this annex apply when the MIME type identifies a file format of this family and the ‘codecs’ parameter starts with a sample-entry code from this document.

NOTE This clause is only applicable to V-PCC and MIV applications.

C.2.2 V3C family

When the first element of a value is a code indicating a codec from ISO/IEC 23090-5, as documented in subclause 7.3 (‘v3e1’ or ‘v3eg’) or in subclause 7.4 (‘v3c1’, ‘v3cg’, ‘v3cb’, ‘v3a1’, ‘v3ag’, or ‘v3t1’) and the respective track can be interpreted as containing an atlas sub-bitstream, the elements following are a series of values from `v3c_parameter_set` syntax structure, as defined in ISO/IEC 23090-5, contained in `v3c_parameter_set` of the V3C decoder configuration record, separated by period characters (“.”). In all numeric encodings, leading zeroes may be omitted.

- the `ptl_tier_flag`, encoded as ‘L’ (`ptl_tier_flag==0`) or ‘H’ (`ptl_tier_flag==1`), followed by the `ptl_level_idc`, encoded as a decimal number;
- the `ptl_profile_codec_group_idc` encoded as a hexadecimal number;
- the `ptl_profile_pcc_toolset_idc` encoded as a hexadecimal number
- the `ptl_profile_reconstruction_idc` encoded as a hexadecimal number

EXAMPLE `codecs=v3e1.L2.1.0.1`

Main tier, Level 2, video components are encoded with ISO/IEC 23008-2, V-PCC Basic toolset profile, Rec1 reconstruction profile.

Annex D (informative)

DASH MPD examples

D.1 Single track example

An example of MPD with a single-track mode is presented below.

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="urn:mpeg:dash:schema:mpd:2011"
  xsi:schemaLocation="urn:mpeg:dash:schema:mpd:2011 DASH-MPD.xsd"
  type="static"
  mediaPresentationDuration="PT3256S"
  minBufferTime="PT1.2S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

  <BaseURL>http://cdn1.example.com/</BaseURL>
  <BaseURL>http://cdn2.example.com/</BaseURL>

  <Period>
    <AdaptationSet
      mimeType="video/mp4" codecs="v3e1.L2.0.0.1, resv.vvvc.avc1.4D401E"
      frameRate="30">
      <SegmentList>
        <Initialization sourceURL="seg-m-init.mp4"/>
      </SegmentList>
      <Representation bandwidth="512000">
        <BaseURL>vpcc-512k.mp4</BaseURL>
      </Representation>
      <Representation bandwidth="1024000">
        <BaseURL>vpcc-1024k.mp4</BaseURL>
      </Representation>
      <Representation bandwidth="2048000">
        <BaseURL>vpcc-2048k.mp4</BaseURL>
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

D.2 Multi-track example (using Preselection element)

In this example, the video components of a V3C sequence are available in two different resolutions. Note that the resolution of the occupancy component may not necessarily be identical to that of the geometry and attribute components. For each resolution alternative of the geometry and attribute V3C video components, two bitrate alternatives are available.

Let O_A and O_B be the two occupancy resolutions. Similarly, let G_A and G_B be the two available geometry resolutions and A_A and A_B be the two attribute resolutions. Since each geometry and attribute resolution is available at two different bitrates, let us denote these as $G_{A,1}$, $G_{A,2}$, $G_{B,1}$, $G_{B,2}$, $A_{A,1}$, $A_{A,2}$, $A_{B,1}$, and $A_{B,2}$. Occupancy O_A is compatible with $G_{A,1}$, $G_{A,2}$, $A_{A,1}$, and $A_{A,2}$. While occupancy O_B is compatible with $G_{B,1}$, $G_{B,2}$, $A_{B,1}$, and $A_{B,2}$.

Each resolution of the geometry and attribute components can be signalled by a separate Adaptation Set with two Representations, one for each bitrate. Each occupancy resolution is also signalled using a separate Adaptation Set with a single Representation. Each Video Component Adaptation Set includes a V3CComponent descriptor with the `@type` set to the corresponding value. Finally, the V3C track which includes the atlas bitstream is signalled with an Adaptation Set containing a single Representation.

Compatible Video Component Adaptation Sets, along with the Main Adaptation Set, are grouped together in two Preselections in the MPD. To indicate that these Adaptation Sets are referenced in at least one Preselection, a Preselection descriptor without the `@value` is signalled in each Adaptation Set. Each Preselection includes a V3C descriptor that indicates at least the mandatory `@vId`. The values assigned to the `@vId` of the two Preselections are identical, indicating that both Preselections belong to the same point cloud content.

The Main Adaptation Set contains the Initialization Segment for the complete experience. Therefore, in the case of an ISOBMFF container, the Initialization Segment contains `TrackBoxes` for the V3C atlas track as well as the V3C video component tracks of all representations of the video components (all resolutions and bitrates).

Figure D.1 illustrates the different Adaptation Sets and their relation to the Preselections that represent the V3C content described in this example.

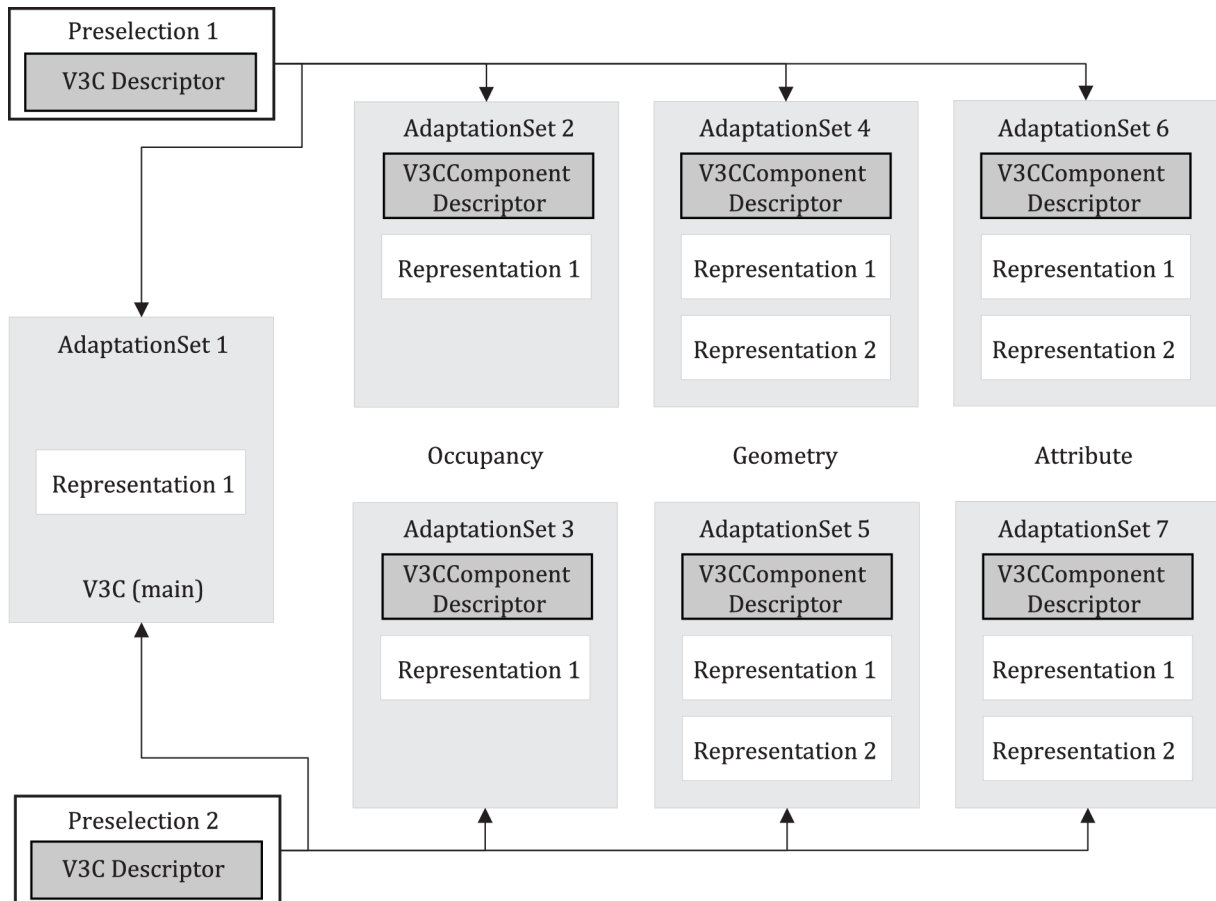


Figure D.1 — MPD layout for the multi-track V3C media example

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
  xmlns="urn:mpeg:dash:schema:mpd:2011"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
  type="static"
  mediaPresentationDuration="PT10S"
  minBufferTime="PT1S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

  <Period>
    <!-- Main V3C AdaptationSet -->
    <AdaptationSet id="1" codecs="v3c1">
      <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
      <Representation>
        ...
      </Representation>
    </AdaptationSet>
  </Period>
</MPD>
```

```

<!-- Occupancy -->
<AdaptationSet id="2" mimeType="video/mp4" codecs="resv.vvvc.hvcl">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
        <v3c:videoComponent type="occp" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<AdaptationSet id="3" mimeType="video/mp4" codecs="resv.vvvc.hvcl">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
        <v3c:videoComponent type="occp" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<!-- Geometry -->
<AdaptationSet id="4" mimeType="video/mp4" codecs="resv.vvvc.hvcl">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
        <v3c:videoComponent type="geom" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<AdaptationSet id="5" mimeType="video/mp4" codecs="resv.vvvc.hvcl">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
        <v3c:videoComponent type="geom" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

```

```

<!-- Attribute -->
<AdaptationSet id="6" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
        <v3c:videoComponent type="attr" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<AdaptationSet id="7" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
        <v3c:videoComponent type="attr" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<!-- Preselections -->
<Preselection id="1" tag="1" preselectionComponents="1 2 4 6" codecs="v3c1">
    <!--V3C Descriptor -->
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:vpc" vId="1" />
</Preselection>

<Preselection id="2" tag="2" preselectionComponents="1 3 5 7" codecs="v3c1">
    <!--V3C Descriptor -->
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:vpc" vId="1" />
</Preselection>

</Period>
</MPD>

```

In this example, the video-coded packed V3C component, which contains texture and transparency component, is present and two different resolutions are available. Each resolution of the video-coded packed V3C component can be signaled by a separate Adaptation Set with two Representations, one for each bitrate. Each Video Component Adaptation Set includes a V3CVideoComponent descriptor with the @type set to 'pack'. Finally, the V3C track which includes the atlas bitstream is signaled with an Adaptation Set containing a single Representation.

Compatible Video Component Adaptation Sets, along with the Main Adaptation Set, are grouped together in two Preselections in the MPD. To indicate that these Adaptation Sets are referenced in at least one Preselection, a Preselection descriptor without the @value is signaled in each Adaptation Set. Each Preselection includes a V3C descriptor that contains the same @vId value, indicating that both Preselections belong to the same V3C content.

Figure D.2 describes the relation between Adaptation Sets to the Preselections that represent the V3C content described in this example.

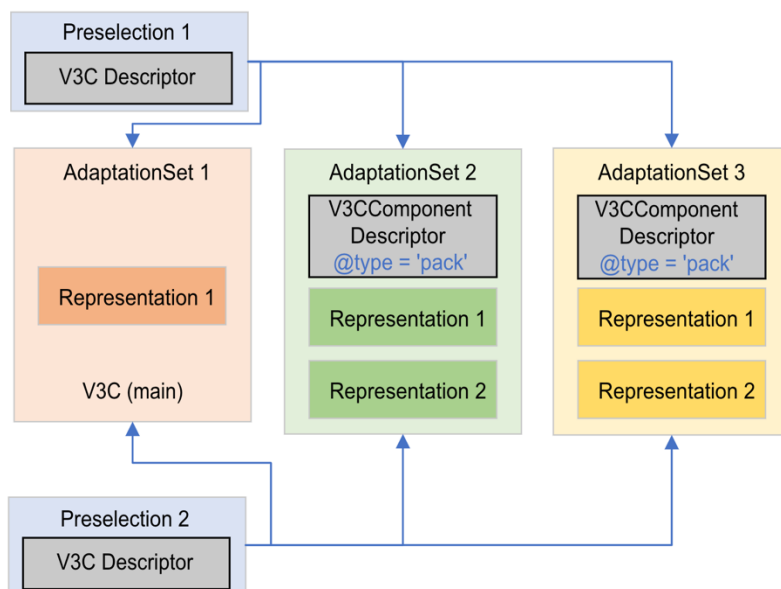


Figure D.2 – MPD layout for the multi-track V3C media containing the packed video components

```
<?xml version="1.0" encoding="UTF-8"?>
<MPD
  xmlns="urn:mpeg:dash:schema:mpd:2011"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
  type="static"
  mediaPresentationDuration="PT10S"
  minBufferTime="PT1S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

  <Period>
    <!-- Main V3C AdaptationSet -->
    <AdaptationSet id="1" codecs="v3c1">
      <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
      <Representation>
        ...
      </Representation>
    </AdaptationSet>

    <!-- Attribute -->
    <AdaptationSet id="2" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
      <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
```

```

        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
            <v3c:videoComponent type="pack" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <AdaptationSet id="3" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:component">
            <v3c:videoComponent type="pack" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <!-- Preselections -->
    <Preselection id="1" tag="1" preselectionComponents="1 2" codecs="v3c1">
        <!--V3C Descriptor -->
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:vpc" vId="1"
/>
    </Preselection>

    <Preselection id="2" tag="2" preselectionComponents="1 3" codecs="v3c1">
        <!--V3C Descriptor -->
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:vpc" vId="1"
/>
    </Preselection>

</Period>
</MPD>

```

D.3 Multi-track example (using preselection descriptor)

The following MPD example demonstrates how Preselection descriptors can be used for signalling the same V3C content described in subclause D.2.

```

<?xml version="1.0" encoding="UTF-8"?>
<MPD
    xmlns="urn:mpeg:dash:schema:mpd:2011"
    xmlns:v3c="urn:mpeg:mpegI:v3c:2020"

```

```

    type="static"
    mediaPresentationDuration="PT10S"
    minBufferTime="PT1S"
    profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

<Period>
  <!-- Main V3C AdaptationSet -->
  <AdaptationSet id="1" codecs="v3c1">
    <!-- V3C Descriptor -->
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1" />
    <!-- Preselection Descriptors -->
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" value="1,1
2 4 6" />
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" value="2,1
3 5 7" />
    <!-- Representation -->
    <Representation>
      ...
    </Representation>
  </AdaptationSet>

  <!-- Occupancy -->
  <AdaptationSet id="2" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent" >
      <v3c:videoComponent type="occp" />
    </EssentialProperty>
    <Representation>
      ...
    </Representation>
  </AdaptationSet>

  <AdaptationSet id="3" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
      <v3c:videoComponent type="occp" />
    </EssentialProperty>
    <Representation>
      ...
    </Representation>
  </AdaptationSet>

```



```

<!-- Geometry -->
<AdaptationSet id="4" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="geom" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<AdaptationSet id="5" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="geom" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<!-- Attribute -->
<AdaptationSet id="6" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="attr" attribute_type="1" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<AdaptationSet id="7" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="attr" attribute_type="1" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

```

</Period>

</MPD>

D.4 Multi-track example with multiple atlas tile tracks

In this MPD example, the V3C content is encapsulated using multi-track encapsulation and the container includes one atlas track, two atlas tile tracks, and six V3C video component tracks. Where each atlas tile track is associated with three V3C video component tracks carrying occupancy, geometry, and attribute information for the atlas tiles carried by the atlas tile track. The MPD file therefore contains nine Adaptation Sets and two Preselections, each Preselection grouping an Atlas Tile Adaptation Set with associated Video Component Adaptation Sets. And the Representation of each Atlas Tile Adaptation Set depends on the Representation of the Main Adaptation Set. The layout of this MPD is shown in Figure D.3.

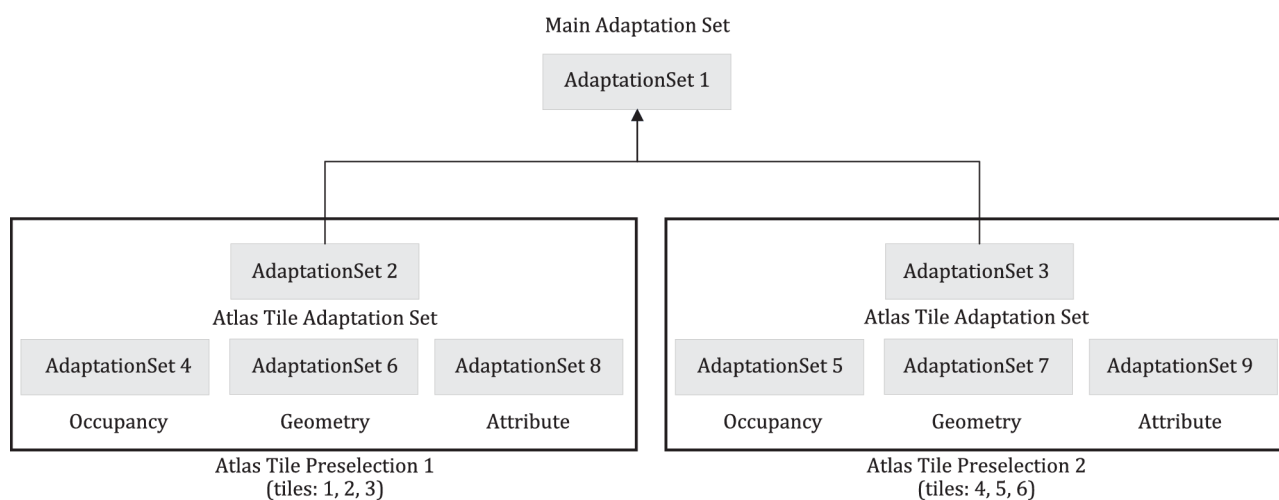


Figure D.3 — MPD layout for multi-track V3C media with two Atlas Tile Preselections

```

<?xml version="1.0" encoding="UTF-8"?>
<MPD
  xmlns="urn:mpeg:dash:schema:mpd:2011"
  xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
  type="static"
  mediaPresentationDuration="PT10S"
  minBufferTime="PT1S"
  profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

  <Period>
    <!-- Main V3C AdaptationSet -->
    <AdaptationSet id="1" codecs="v3c1">
      <!-- V3C Descriptor -->
      <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1" />
      <Representation id="1">
        ...

```

```

        </Representation>
    </AdaptationSet>

    <!-- Atlas Tile Adaptation Set 1 -->
    <AdaptationSet id="2" codecs="v3t1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" tile_ids="1
2 3" />
        <Representation dependencyId="1">
            ...
        </Representation>
    </AdaptationSet>

    <!-- Atlas Tile Adaptation Set 2 -->
    <AdaptationSet id="3" codecs="v3t1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" tile_ids="4
5 6" />
        <Representation dependencyId="1">
            ...
        </Representation>
    </AdaptationSet>

    <!-- Occupancy -->
    <AdaptationSet id="4" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="occp" tile_ids="1 2 3" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <AdaptationSet id="5" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="occp" tile_ids="4 5 6" />
        </EssentialProperty>
        <Representation>
            ...

```

```

        </Representation>
    </AdaptationSet>

    <!-- Geometry -->
    <AdaptationSet id="6" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="geom" tile_ids="1 2 3" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <AdaptationSet id="7" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="geom" tile_ids="4 5 6" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <!-- Attribute -->
    <AdaptationSet id="8" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="attr" attribute_type="1" tile_ids="1 2 3" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <AdaptationSet id="9" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="attr" attribute_type="1" tile_ids="4 5 6" />
        </EssentialProperty>
        <Representation>

```

```

        ...
    </Representation>
</AdaptationSet>

<!-- Atlas Tile Preselections -->
<Preselection id="1" tag="1" preselectionComponents="2 4 6 8" codecs="v3t1">
    <!--V3C Descriptor -->
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1" />
</Preselection>

<Preselection id="2" tag="2" preselectionComponents="3 5 7 9" codecs="v3t1">
    <!--V3C Descriptor -->
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1" />
</Preselection>

</Period>
</MPD>

```

D.5 Multi-track example with multiple atlas tile tracks and volumetric metadata

This example is similar to the one described in the previous subclause with additionally having a volumetric timed metadata track, as described in Clause 9, available in the container. The following MPD file therefore includes an additional Adaptation Set with one Representation for this track.

```

<?xml version="1.0" encoding="UTF-8"?>
<MPD
    xmlns="urn:mpeg:dash:schema:mpd:2011"
    xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
    type="static"
    mediaPresentationDuration="PT10S"
    minBufferTime="PT1S"
    profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

    <Period>
        <!-- Main V3C AdaptationSet -->
        <AdaptationSet id="1" codecs="v3c1">
            <!-- V3C Descriptor -->
            <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1" />
            <Representation id="1">
                <BaseURL>v3c.mp4</BaseURL>
            </Representation>
        </AdaptationSet>
    
```

```

<!-- Atlas Tile Adaptation Set 1 -->
<AdaptationSet id="2" codecs="v3t1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" tile_ids="1
2 3" />
    <Representation dependencyId="1">
        <BaseURL>v3c_t1.mp4</BaseURL>
    </Representation>
</AdaptationSet>

<!-- Atlas Tile Adaptation Set 2 -->
<AdaptationSet id="3" codecs="v3t1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" tile_ids="4
5 6"/>
    <Representation dependencyId="1">
        <BaseURL>v3c_t2.mp4</BaseURL>
    </Representation>
</AdaptationSet>

<!-- Occupancy -->
<AdaptationSet id="4" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="occp" tile_ids="1 2 3" />
    </EssentialProperty>
    <Representation>
        <BaseURL>v3c_t1_occupancy.mp4</BaseURL>
    </Representation>
</AdaptationSet>

<AdaptationSet id="5" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="occp" tile_ids="4 5 6" />
    </EssentialProperty>
    <Representation>
        <BaseURL>v3c_t2_occupancy.mp4</BaseURL>
    </Representation>
</AdaptationSet>

```

```

<!-- Geometry -->
<AdaptationSet id="6" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
  <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
  <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
    <v3c:videoComponent type="geom" tile_ids="1 2 3" />
  </EssentialProperty>
  <Representation>
    <BaseURL>v3c_t1_geometry.mp4</BaseURL>
  </Representation>
</AdaptationSet>

<AdaptationSet id="7" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
  <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
  <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
    <v3c:videoComponent type="geom" tile_ids="4 5 6" />
  </EssentialProperty>
  <Representation>
    <BaseURL>v3c_t2_geometry.mp4</BaseURL>
  </Representation>
</AdaptationSet>

<!-- Attribute -->
<AdaptationSet id="8" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
  <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
  <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
    <v3c:videoComponent type="attr" attribute_type="1" tile_ids="1 2 3" />
  </EssentialProperty>
  <Representation>
    <BaseURL>v3c_t1_attribute.mp4</BaseURL>
  </Representation>
</AdaptationSet>

<AdaptationSet id="9" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
  <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
  <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
    <v3c:videoComponent type="attr" attribute_type="1" tile_ids="4 5 6" />
  </EssentialProperty>
  <Representation>
    <BaseURL>v3c_t2_attribute.mp4</BaseURL>
  </Representation>
</AdaptationSet>

```

```

    <!-- Volumetric Metadata -->
    <AdaptationSet id="9" mimeType="application/mp4" codecs="dyvm">
        <Representation id="volumetric-metadata" associationId="1"
associationType="cdsc" >
            <BaseURL>volumetric_metadata.mp4</BaseURL>
        </Representation>
    </AdaptationSet>

    <!-- Atlas Tile Preselections -->
    <Preselection id="1" tag="1" preselectionComponents="2 4 6 8" codecs="v3t1">
        <!--V3C Descriptor -->
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1" />
    </Preselection>

    <Preselection id="2" tag="2" preselectionComponents="3 5 7 9" codecs="v3t1">
        <!--V3C Descriptor -->
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1" />
    </Preselection>

</Period>
</MPD>

```

D.6 Alternative V3C content example

In this MPD example, one V3C content has two alternatives. The first alternative does not have atlas tile encoding while the second alternative has atlas frames with 6 atlas tiles. The atlas tiles for the second alternative are carried in two separate atlas tile tracks, where the first atlas tile track carries atlas tiles 1, 2, and 3, and the second atlas tile track carries atlas tiles 4, 5, and 6.

In total, thirteen Adaptation Sets are signalled in the MPD. As the atlas track without atlas tile encoding and the atlas track with atlas tile encoding are alternatives to each other, their corresponding Adaptation Sets have V3C descriptors with the same value assigned to the @vId and @atlas_id attributes, respectively. Since the two Main Adaptation Sets have V3C descriptors with the same values for the @vId and @atlas_id attributes, a DASH client is able to identify that there are two alternative versions of the V3C content. The MPD also includes a V3C Preselection for the first alternative and two Atlas Tile Preselections for the atlas tile tracks for the second alternative.

```

<?xml version="1.0" encoding="UTF-8"?>
<MPD
    xmlns="urn:mpeg:dash:schema:mpd:2011"
    xmlns:v3c="urn:mpeg:mpegI:v3c:2020"
    type="static"
    mediaPresentationDuration="PT10S"

```



```

minBufferTime="PT1S"
profiles="urn:mpeg:dash:profile:isoff-on-demand:2011">

<Period>
  <!-- Main V3C Adaptation Set 1 -->
  <AdaptationSet id="1" codecs="v3c1">
    <!-- V3C Descriptor -->
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1"
atlas_id="1" />
    <Representation id="1">
      ...
    </Representation>
  </AdaptationSet>

  <!-- Main V3C Adaptation Set 2 -->
  <AdaptationSet id="2" codecs="v3c1">
    <!-- V3C Descriptor -->
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1"
atlas_id="1"/>
    <Representation id="2">
      ...
    </Representation>
  </AdaptationSet>

  <!-- Atlas Tile Adaptation Set 1 -->
  <AdaptationSet id="3" codecs="v3t1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" tile_ids="1
2 3" />
    <Representation dependencyId="2">
      ...
    </Representation>
  </AdaptationSet>

  <!-- Atlas Tile Adaptation Set 2 -->
  <AdaptationSet id="4" codecs="v3t1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" tile_ids="4
5 6"/>
    <Representation dependencyId="2">
      ...
    </Representation>

```

```

</AdaptationSet>

<!-- Occupancy -->
<AdaptationSet id="5" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="occp" atlas_id="1"/>
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<AdaptationSet id="6" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="occp" atlas_id="1" tile_ids="1 2 3" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<AdaptationSet id="7" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="occp" atlas_id="1" tile_ids="4 5 6" />
    </EssentialProperty>
    <Representation>
        ...
    </Representation>
</AdaptationSet>

<!-- Geometry -->
<AdaptationSet id="8" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
    <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
    <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
        <v3c:videoComponent type="geom" atlas_id="1"/>
    </EssentialProperty>
    <Representation>
        ...

```

```

        </Representation>
    </AdaptationSet>

    <AdaptationSet id="9" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="geom" atlas_id="1" tile_ids="1 2 3" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <AdaptationSet id="10" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="geom" atlas_id="1" tile_ids="4 5 6" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <!-- Attribute -->
    <AdaptationSet id="11" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="attr" atlas_id="1" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <AdaptationSet id="12" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="attr" atlas_id="1" tile_ids="1 2 3" />
        </EssentialProperty>
        <Representation>
            ...

```

```

        </Representation>
    </AdaptationSet>

    <AdaptationSet id="13" mimeType="video/mp4" codecs="resv.vvvc.hvc1">
        <EssentialProperty schemeIdUri="urn:mpeg:dash:preselection:2016" />
        <EssentialProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:videoComponent">
            <v3c:videoComponent type="attr" atlas_id="1" tile_ids="4 5 6" />
        </EssentialProperty>
        <Representation>
            ...
        </Representation>
    </AdaptationSet>

    <Preselection id="1" tag="1" preselectionComponents="1 5 8 11" codecs="v3c1">
        <!-- V3C Descriptor -->
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1"
atlas_id="1"/>
    </Preselection>
    <!-- Atlas Tile Preselections -->
    <Preselection id="2" tag="2" preselectionComponents="3 6 9 12" codecs="v3t1">
        <!-- V3C Descriptor -->
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1"
atlas_id="1"/>
    </Preselection>
    <Preselection id="3" tag="3" preselectionComponents="4 7 10 13" codecs="v3t1">
        <!-- V3C Descriptor -->
        <SupplementalProperty schemeIdUri="urn:mpeg:mpegI:v3c:2020:v3c" vId="1"
atlas_id="1"/>
    </Preselection>

</Period>
</MPD>

```

Annex E (informative)

Partial access support

E.1 Partial access utilizing V3C volumetric annotation SEI message family

E.1.1 General

There are cases, where storing data in number of atlases or tiles, that are stored in different tracks, are useful for easy access to one or more regions of V3C content especially in streaming scenarios. One such example is view-frustum culling. It is a known 3D graphics technique for culling objects outside of the user's current view of the scene. This annex describes a design which enables view-frustum culling on multiple levels, allowing to cull entire atlases or tiles based on object and view visibility information provided by Volumetric Annotation SEI message family as defined in ISO/IEC 23090-5.

This annex provides information about three use cases of partial access, where tracks with samples entries specified in subclause 7.4 are discussed per case. NOTE This annex is only applicable to V-PCC and MIV applications.

E.1.2 Content stored in a single atlas with a single tile

Atlas data with one tile is a special case. Neither atlases nor tiles can be culled, i.e., all atlas data shall be provided to V3C decoder. This case is presented for completeness and illustrated in Figure E.1.

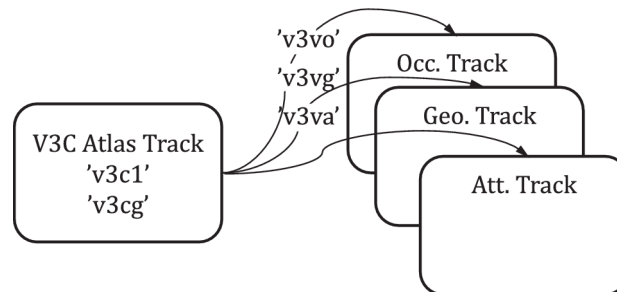


Figure E.1 — Example of multi-track container with single atlas

In this case the partial access can be done only at the renderer. The renderer, using the information provided by volumetric annotation SEI message family as defined in ISO/IEC 23090-5, may perform object-based selective rendering. This document provides several mechanisms for storing SEI messages depending on, how often they are updated. Several options for storing Volumetric Annotation SEI messages in V3C track may therefore be considered:

- SEI messages may be stored in `V3CDecoderConfigurationRecord`
- SEI messages may be stored in `V3CAtlasParamSampleGroupDescriptionEntry`
- SEI messages may be stored in `V3CAtlasSample`

E.1.3 Content stored in a single atlas with multiple tiles

Atlas data with two or more tiles may be stored as multiple tracks as presented in Figure E.2. From a partial access point of view, this design provides ability to selectively cull tiles from an atlas, considering that tiles are stored in different tracks. V3C atlas track may contain only information, which is shared between V3C atlas tile tracks, e.g., atlas parameter sets and SEI messages, whereas V3C atlas tile tracks contain only ACL data of a tile or a group of tiles.

NOTE Splitting atlases into atlas tile tracks that reference video tile tracks depend on the video codec support. However, such video codec support is outside of the scope of this document.

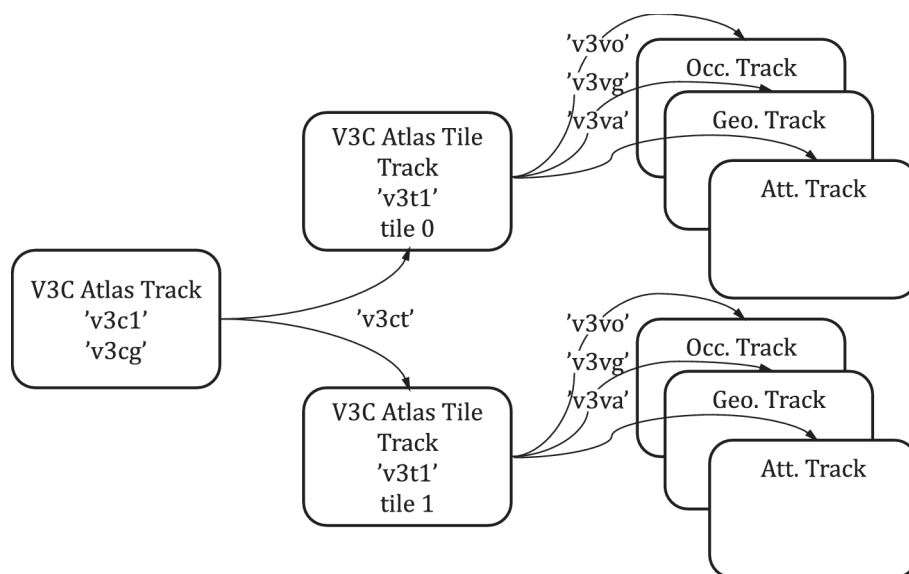


Figure E.2 — Example of multi-track container with single atlas and multiple atlas tiles

In practice this design means that a V3C atlas track may contain atlas parameter sets and SEI messages. With information provided by Volumetric Annotation SEI message family, as defined in ISO/IEC 23090-5, an application can map objects with 3D bounding boxes and visibility information to tiles. With the information on the needed tile, an application may request this tile from a file parser. A file parser can map a tile to an V3C atlas tile track and provides appropriate samples to V3C decoder. V3C atlas track offers three mechanisms to signal messages from the Volumetric Annotation SEI message family:

- SEI messages may be stored in `V3CDecoderConfigurationRecord`; Volumetric Annotation SEI messages in a sample entry would provide bootstrap information for presentation time t_0 .
- SEI messages may be stored in `V3CAtlasParamSampleGroupDescriptionEntry`; SEI messages stored in sample groups of V3C track would offer flexible update mechanism for messages from Volumetric Annotation SEI message family and improve random access functionality for partial access.
- SEI messages may be stored in `V3CAtlasSample`; SEI messages stored in samples of V3C atlas track would provide temporal update and override any information related to SEI messages from Volumetric Annotation SEI message family stored in a sample entry or sample group description entry.

E.1.4 Content stored in multiple atlases

Storage design of more than one atlas with more than one tile is illustrated in Figure E.3. From partial access point of view, this design provides ability to selectively cull atlases, considering that atlases are stored in different tracks, in addition to allowing tile-based culling as presented in Annex E.3.

V3C atlas track with sample entry 'v3cb', as described in subclause 7.5.2, contains common information, which applies to all atlases. As an example of such information is atlas adaptation parameter set, which may contain view parameters shared by different atlases as specified in ISO/IEC 23090-12. Similarly, V3C atlas track with sample entry 'v3cb' may also store Volumetric Annotation SEI messages, which provide information on which objects are contained in which atlases. Moreover, by storing scene object information SEI message in V3C atlas track with sample entry 'v3cb' would indicate that object indices remain unique across multiple atlases, which is helpful if the same object appears in more than one atlas. A scene object information SEI message contains a list of objects in the scene and information related to said objects, e.g., 3D bounding boxes and visibility information. Atlas to object SEI message provide information for mapping objects to atlases.

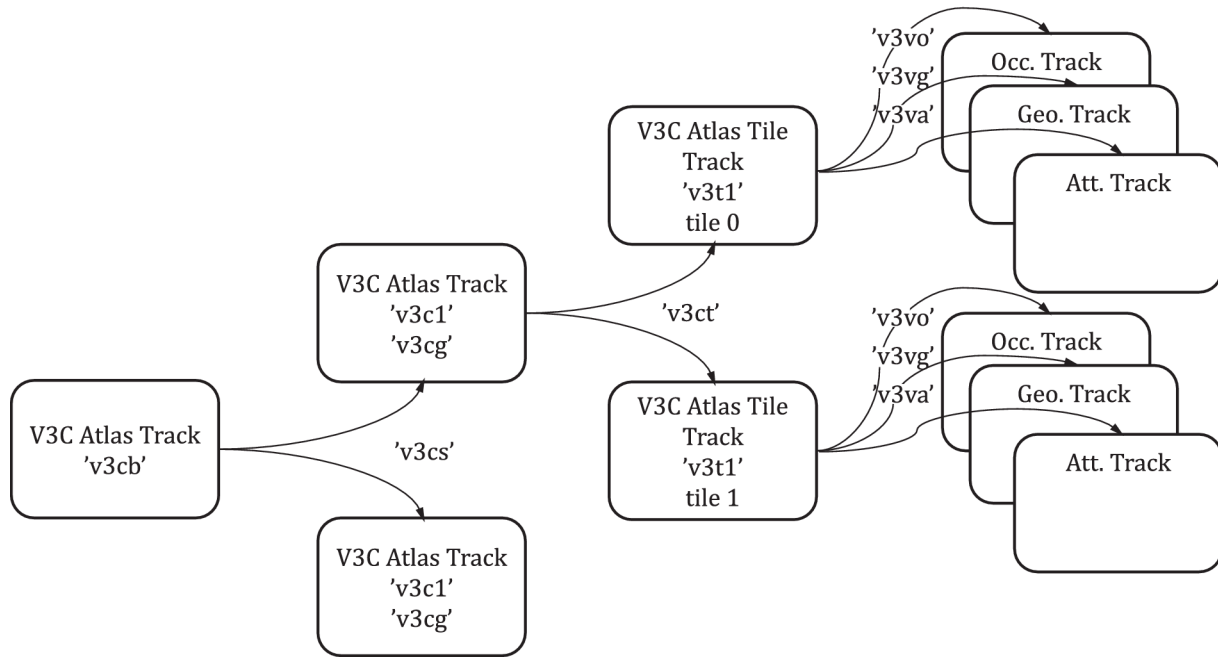


Figure E.3 — Example of multi-track container with more than one atlas

With scene object information SEI message as well as other SEI messages from Volumetric Annotation SEI message family, as defined in ISO/IEC 23090-5, stored in V3C atlas track, an application could identify, which atlases contain information relevant for rendering at a given time. The atlases themselves may be stored in separate V3C atlas tracks with sample entries of 'v3a1' or 'v3ag'. Additionally, tiles could be carried by more than one V3C atlas tile tracks, if more than one tile exists for an atlas as described in Annex E.3. Atlas adaptation parameter set, which contains view parameters shared by different atlases, scene object information SEI message and atlas object association SEI message could be stored in V3C atlas track with sample entry 'v3cb'. The V3C atlas tracks with sample entries of 'v3a1' and 'v3ag' should contain data related to an individual atlas like patch information SEI message and volumetric rectangle information SEI messages to provide mapping information from objects to tiles and patches.

V3C atlas tracks offer several alternative techniques for storing configuration data and SEI messages for all atlases.

- SEI messages may be stored in `V3CDecoderConfigurationRecord`; Volumetric Annotation SEI messages stored in sample entry would provide bootstrap information for presentation time t_0 or in case if partial access information or view information is unlikely to change during presentation
- SEI messages may be stored in `V3CAtlasParamSampleGroupDescriptionEntry`; if Volumetric Annotation SEI messages or view information rarely changes during presentation

- SEI messages may be stored in V3CAtlasSample; Volumetric Annotation SEI messages stored in samples of V3C atlas track with sample entry 'v3cb' would provide temporal update and override any information related to said SEI messages stored in sample entry.

E.2 Partial access using volumetric information timed-metadata tracks

E.2.1 General

This annex describes how the volumetric metadata timed-metadata track specified in subclause 9.6 can be used by players to enable view-frustum culling of atlases and tiles at the container level without having to parse the samples of atlas tracks to search for volumetric annotation SEI messages as described in Annex E. This annex provides information about two partial access use cases.

E.2.2 Content stored in a single atlas with multiple tiles

Atlas data with two or more tiles may be stored as multiple tracks as presented in Figure E.2. When a volumetric metadata timed-metadata track is associated with a V3C atlas track, a player can use the information provided by this track to selectively cull tile from the atlas (e.g., based on the current viewport), even when volumetric annotation SEI messages are not carried in the atlas track. This can be done in both local playback and streaming use cases.

The samples of volumetric metadata timed-metadata track carry information on the different spatial regions defined for the V3C content, including the bounding box for the spatial region and an association with one or more V3C atlas tiles. In addition, the samples of this timed-metadata track may also carry information on different objects within the V3C content and the spatial regions or V3C atlas tile ids associated with those objects.

With information provided by the sample entry and the samples of the volumetric metadata timed-metadata track, a player can determine which spatial regions and/or objects fall within the boundaries of the user's viewport and map those regions/objects to tiles.

With information on the needed tile(s), a player may request the tile(s) from a file parser (e.g., in the case of local playback) or from a media access engine (e.g., in the case of streaming applications). A file parser can map a tile to an V3C atlas tile track and provides appropriate samples to V3C decoder. A media access engine can determine the set of Adaptation Sets in the MPD file whose Representations carry V3C data for those tiles and download media segments from those Representations to be passed to the file parser and eventually deliver appropriate samples to the V3C decoder.

E.2.3 Content stored in multiple atlases

Storage design of more than one atlas with more than one tile is illustrated in Figure E.3. The atlases are stored in separate V3C atlas tracks with sample entries of 'v3a1' or 'v3ag'. Additionally, tiles could be carried by more than one V3C atlas tile tracks, if more than one tile exists for an atlas.

Similar to the case of a single atlas with multiple tiles, when more than one atlas is present in the content, a player is able to identify which set of atlases, and atlas tiles, contain relevant information for the rendering process at any given time during playback using the metadata in the samples of the timed-metadata track(s) associated with the atlas track(s). A player is therefore able to cull any atlases not needed for rendering and only pass relevant atlas information to the file parser or request. Similarly, by first retrieving segments from the Adaptation Set of the timed-metadata track in streaming applications, the player can make an informative decision on which media segments the media access engine should download to render the parts and objects within the content that fall within the view-frustum at any given time.

E.2.4 Content with multiple levels of detail

When the atlas tiles of the content are stored in separate V3C atlas tile tracks and the volumetric metadata timed-metadata track associated with the atlas track defines multiple levels of detail, the player is can use the information in the samples of the timed-metadata track to identify for each region and/or object of interest the set of tiles associated with a target LoD. The player can therefore request increase the level-of-detail of the rendered content by incrementally requesting additional tiles, corresponding to higher LoDs, from the file parser.

In streaming applications, the player identifies the available levels of detail at any given time by first downloading a segment from the Adaptation Set of the volumetric metadata timed-metadata track. Based on the region(s) and/or object(s) of interest, the target LoD for each region/object, and the information in manifest file for the content, the player is able to identify the set of media segments carry information for the corresponding atlas tiles that the media access engine should retrieve.

E.3 Partial access for overlapping spatial subdivisions

E.3.1 General

The volumetric media can be distributed with potentially overlapping spatial subdivisions. For such cases, the V3CSpatialRegion and V3CBoundingBox might not suffice to describe the underlying data, since a point of interest of the media might be present in more than one spatial sub-divisions and/or there can be some subdivisions optimized for rendering from specific viewports. For such cases, it is recommended to use spatial regions with associated recommended viewport(s) for each of the spatial subdivisions. This way, if more than one spatial sub-divisions are identified as relevant, a selection can be made based on the viewing angle and position of the virtual camera of the media client (i.e. the application viewport).

E.3.2 Using viewport spatial regions

To define a viewport spatial region, the spatialRegion@type attribute is set to 1. Then, each recommended viewport is signalled via the v3sr.spatialRegion.viewport@rvIds attribute, for each of the spatial regions of the V3C3DRegions descriptor. Recommended viewports used for this purpose should have the viewportInfo@viewport_type attribute set to “4 - A recommended viewport suggested for a spatial region”, according to

Table 13.

E.3.3 Using cuboid and viewport spatial regions

The same spatial subdivision can have multiple v3sr.spatialRegion elements, therefore enabling spatial access using multiple viewports and/or cuboid regions. The cuboid spatial subdivision should have the spatialRegion@type attribute set to 0 and the viewport spatial subdivision should have the spatialRegion@type attribute set to 1. The client can use both or either types of the v3sr.spatialRegion elements to identify the most relevant subdivision. As an implementation logic example, a client can first compare the cuboid descriptors to make a first selection of relevant spatial regions and then use the viewport descriptors to make the final decision (e.g. by comparing the recommended viewport to the current application viewport).

Annex F (informative)

Examples of using alternate groups

When a V3C video component track has alternatives, the sample entry of each alternative track provides enough information to show the differences between the alternative representations and to select one appropriate representation by the application for play. When the sample entry of tracks in an alternative relationship does not provide enough information to show the differences between the alternative versions, the application may use additional external mechanisms, e.g., DASH MPD, scene descriptions, and so on, to get the information needed to select one appropriate version for play.

Figure H.1 shows an example of a V3C content using track alternatives for the V3C video component tracks. The value of the `scheme_type` of the tracks whose ID is within the range between 2 and 8 is 'vvvc' indicating they are the V3C video components tracks. Among them the tracks whose ID equals to 2, 3, and 4 are the V3C video component tracks related to the V3C atlas track whose ID equals 1 and are directly listed in the track reference list of that track. Even though the tracks whose ID equals 5, 6, 7, and 8 are not directly listed, they are also identified as V3C video component track related to the same V3C atlas track by having the same value of track alternative group information with the tracks whose ID equals 2, 3, and 4. The tracks with the values of `alternate_group` equal to 10, 11, and 12, respectively, are the alternative representations of same a respective component of the V3C content. The value of `original_format` in `RestrictedSchemeInfoBox` shows that the tracks whose ID equals 2, 3, and 4 are AVC encoded representations and the tracks whose ID equals 5, 6, and 7 are HEVC encoded alternative representations of the tracks whose ID equals 2, 3 and 4, respectively. Similarly, the value of `original_format` in `RestrictedSchemeInfoBox` shows that the tracks whose ID equals 8 is a VVC encoded alternative representation of the track whose ID equals 4.

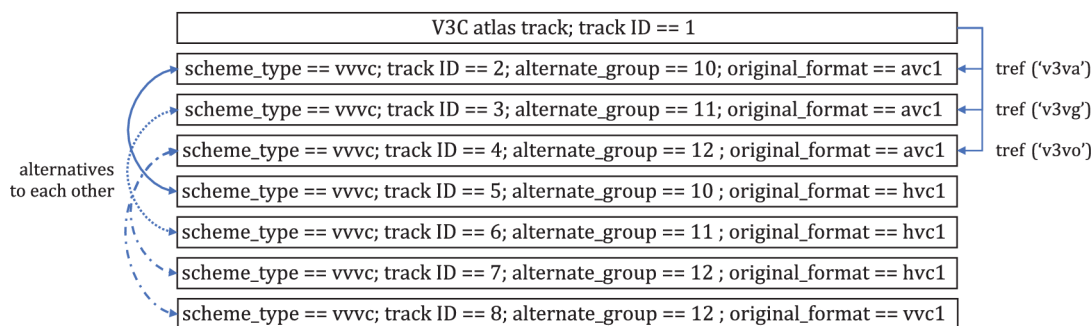


Figure F.1 — An example of track alternatives for an V3C content

Figure H.2 shows an example of a V3C content using track alternatives of V3C video component tracks and V3C atlas tile tracks. The value of the `scheme_type` for the tracks whose ID is within the range between 21 and 28 is set to 'vvvc' indicating they are the V3C video component tracks. The track reference information in the V3C atlas tile tracks show that the tracks whose ID equals 21, 22 and 23 are the V3C video component tracks related to the V3C atlas tile track whose track ID equals 11 and the tracks whose ID equals 24, 25, and 26 are the V3C video components tracks related to the V3C atlas tile track whose track ID equals 12. From the value of the `alternate_group`, the tracks whose ID equals 27 and 28 are identified as the alternative representations of the track whose ID equals 24 and 25, respectively, which means that they are also the V3C video component tracks related to the V3C atlas track whose track ID equals 12. The value of `original_format` in `RestrictedSchemeInfoBox` shows that the tracks whose ID

equals 24 and 25 are AVC encoded representations and the tracks whose ID equals 27 and 28 are HEVC encoded alternative representations of the tracks whose ID equals 24 and 25, respectively.

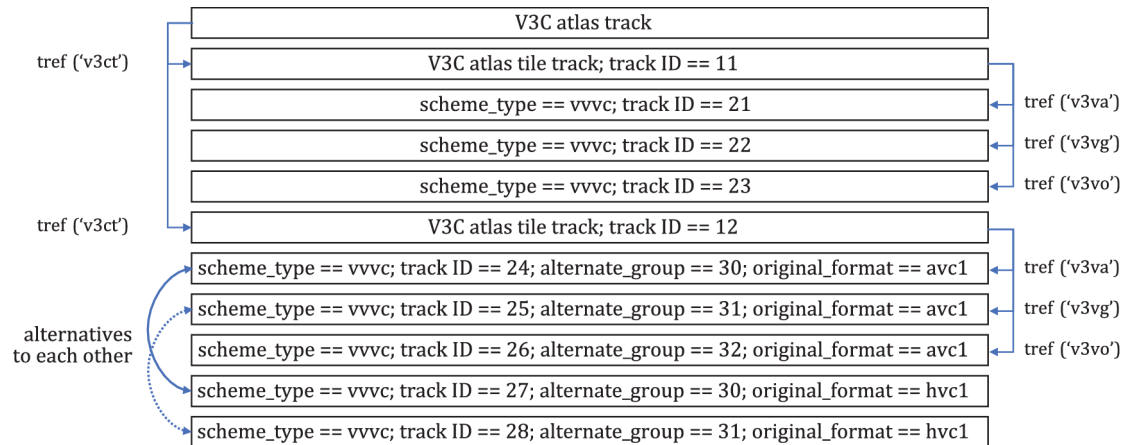


Figure F.2 — An example of track alternatives for a V3C content with V3C atlas tile tracks

Figure H.3 shows an example of two V3C contents in an alternative relationship. The tracks whose ID is in the range between 1 and 4 comprise one V3C content and the tracks whose ID is in the range between 10 and 18 comprise the other V3C content. Two V3C atlas tracks with the same value of `alternate_group` indicate that the two V3C content represented by each V3C atlas track are alternatives to each other. In this example, the V3C content with a V3C atlas track whose track ID equals 1 is a version which does not use V3C atlas tiles but the V3C content with a V3C atlas track whose track ID equals 10 is a version using V3C atlas tiles. For all the V3C video component tracks, the value of the `scheme_type` of the tracks is equal to 'vvvc' and they are directly referenced by either the V3C atlas track or V3C atlas tile tracks. The track reference information in the V3C atlas track whose ID equals 1 show that the tracks whose track ID equals 2, 3 or 4 are its V3C video components tracks. The track reference information in two V3C atlas tile tracks show that the tracks whose ID equals 13, 14 or 15 are the V3C video components tracks related to V3C atlas tile track whose track ID equals 11 and the tracks whose ID is 16, 17 or 18 are the V3C video components tracks related to V3C atlas tile track whose track ID equals 12. All the V3C video component tracks are referenced by the V3C atlas track or V3C atlas tile track only once.

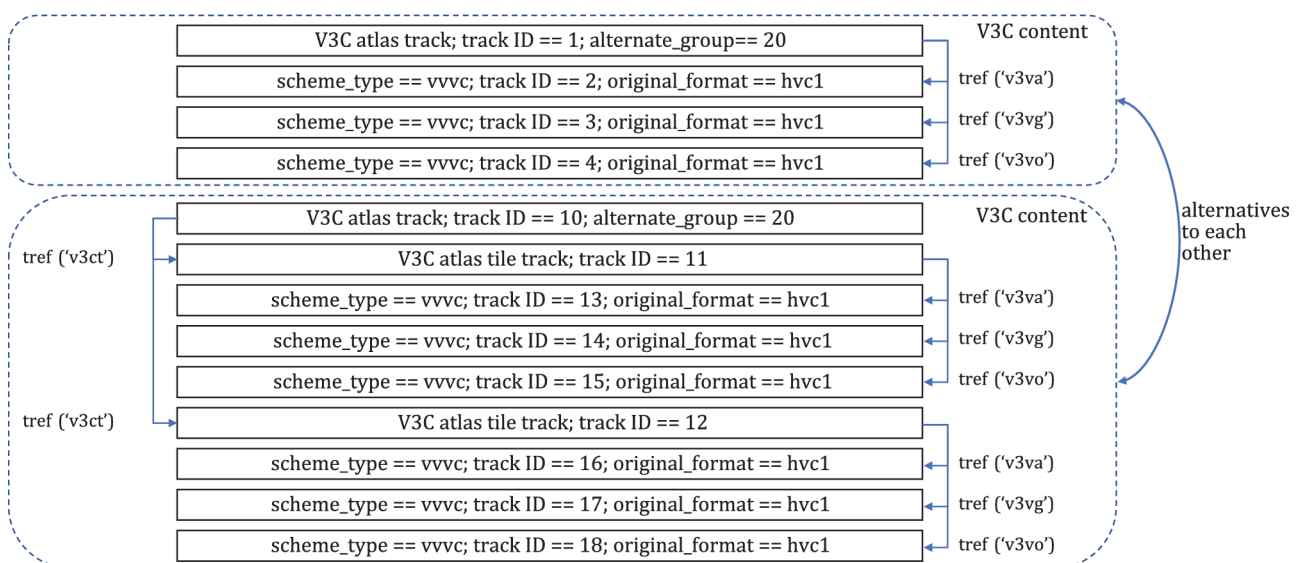


Figure F.3 — An example of V3C content alternatives

Annex G (informative)

Implementation examples of decoding all video components of V3C contents with single decoder instance

G.1 General

For V3C content with multiple video components, each video component will be decoded individually.

For example, Figure I.1 shows the processing when V-PCC file which multi-track encapsulated atlas, occupancy, geometry, and attribute encoded data.

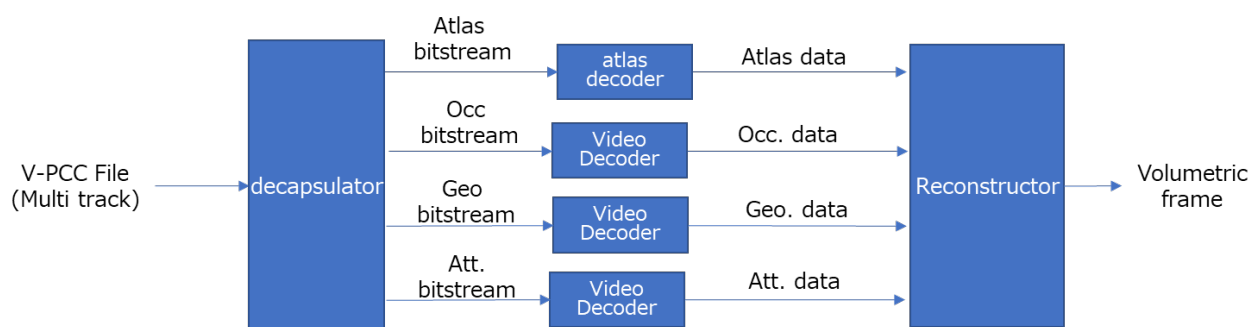


Figure I.1 decoding process of V-PCC content

V-PCC file is decapsulated to atlas, occupancy, geometry, and attribute bitstream in decapsulator, respectively. All bitstream are decoded in decoder, respectively. It is reconstructed into a volumetric frame using all decoded data. This case is typically handled by multiple video decoder instances.

This annex describes implementation examples for decoding all video components of V3C content with single video decoder instance using function defined in other specifications.

NOTE This annex is only applicable to V-PCC and MIV applications.

G.2 Using ISO/IEC 23090-13

ISO/IEC 23090-13 Video decoding interface specifies the interface of Video Decoding Engine (VDE) and operation of elementary stream in VDE.

The “inserting” function is one of the specified operations in ISO/IEC 23090-13 and it can insert an elementary stream into another elementary stream and generate one merged elementary stream.

An implementation example is shown in Figure I.2. A V-PCC file which multi-track encapsulated atlas, occupancy, geometry, and attribute encoded data is decapsulated to atlas, occupancy, geometry, and attribute bitstream in decapsulator, respectively. The occupancy, geometry, and attribute bitstream are input to the VDE. The first “inserting” function of VDI operation insert the geometry bitstream into the occupancy bitstream. Then, the second “inserting” function insert the attribute bitstream into the bitstream which generated by the first “inserting” function. As a result, generated bitstream (as merged bitstream in the figure) contains occupancy, geometry, and attribute. The bitstream contained occupancy, geometry, and attribute is decoded with single decoder instance. The decoded data which is output from

decoder is divided into occupancy, geometry, and attribute data. These data and atlas data are reconstructed to the volumetric frame.

NOTE the “inserting” function needs to regenerate parameter sets to ensure the bitstream conformance.

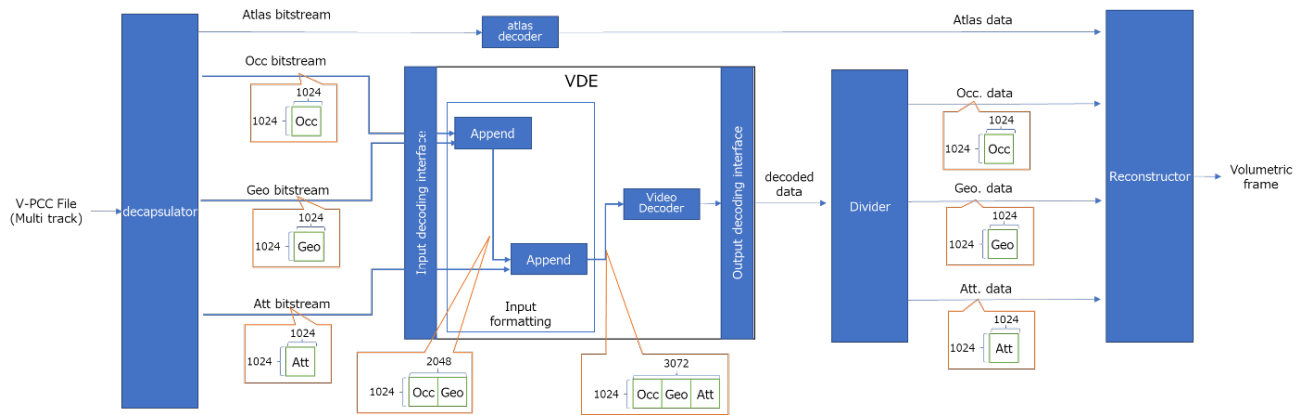


Figure I.2 decoding process using ISO/IEC 23090-13

G.3 Using bitstream reconstruction of HEVC tile

It specifies the bitstream reconstruction using independent HEVC tiles and HEVC tile base tracks in the ISO/IEC 14496-15.

In this example, the V3C video component is encoded as an independent HEVC tile. For example, all V3C video components may be encoded as independent HEVC tile in one picture, and then divide into tracks for each V3C video component (HEVC tile). And the HEVC tile base track is generated to merge HEVC tile tracks which are occupancy, geometry, attribute video component track. Figure I.3 shows the structure of V3C content. The ‘sabt’ track reference in HEVC base track refers to the occupancy, geometry, attribute video component track.

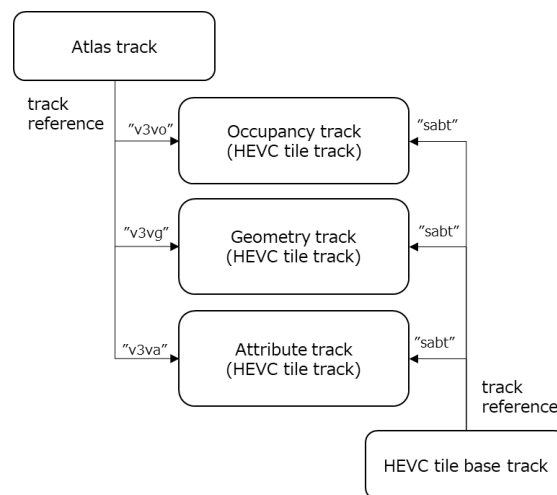


Figure I.3 track structure of V3C content using bitstream reconstruction of HEVC tile

When the client plays the V3C content, it can identify the HEVC tile base track after finding the atlas track. To identify the HEVC tile base track, the client can choose the HEVC tile track that the tracks referenced by

the 'sabt' track reference in the HEVC base track are identical the video component tracks referenced from the track reference in the atlas track.

The decoding process of V-PCC file which multi-track encapsulated atlas, occupancy, geometry, and attribute encoded data is shown in Figure I.4. V-PCC file which multi-track encapsulated atlas, occupancy, geometry, and attribute encoded data is decapsulated to atlas, occupancy, geometry, and attribute bitstream in decapsulator, respectively. The merged bitstream is generated from HEVC tile base track and HEVC tile tracks which are occupancy, geometry, attribute video component track according to bitstream reconstruction defined in ISO/IEC 14496-15.

The bitstream (as merged bitstream in Figure I.4) contained occupancy, geometry, and attribute is decoded with a single decoder instance. The decoded sequence which is output from decoder is divided into occupancy, geometry, and attribute sequences using the information of placement and resolution in 'trif' sample group in the HEVC tile base track. These divided data and atlas data are reconstructed to the volumetric frame.

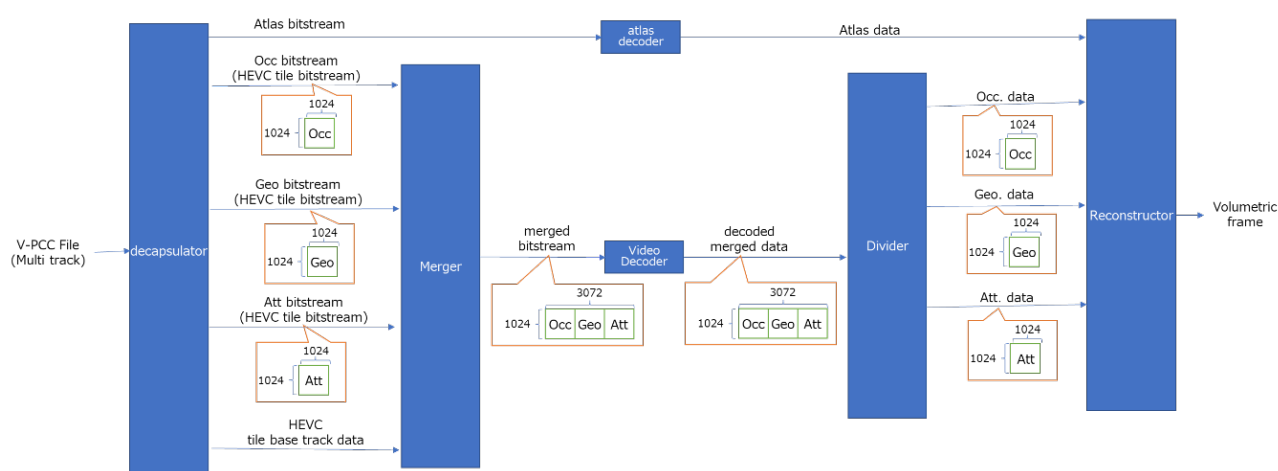


Figure I.4 decoding process using bitstream reconstruction of HEVC tile

Note1 The HEVC picture is rectangular picture. In other words, the merged picture is rectangular picture. To conformant this condition, it need to consider the resolution of each video component and placement of each video component in the merged picture. For example, if all video components are the same resolution, the merged picture can be rectangular picture by arranging from pictures horizontally (or vertically).

Note2 The placement of HEVC tile in the merged bitstream is configured in the HEVC base track. The placement of subpictures may be arranged not only horizontally, but also vertically, or both horizontally and vertically.

G.4 Using bitstream reconstruction of VVC subpicture

It specifies the bitstream reconstruction using independent VVC subpictures and VVC merge base tracks in the ISO/IEC 14496-15.

In this example, the V3C video component is encoded as an independent VVC subpicture. For example, all V3C video components may be encoded as independent subpictures in one picture, and then divide into tracks for each V3C video component(VVC subpicture). And the VVC merge base track is generated to merge VVC subpicture tracks which are occupancy, geometry, attribute video component track. Figure I.5 shows the structure of VVC content. The "subp" track reference in VVC merge base track refers to the occupancy, geometry, attribute video component track.

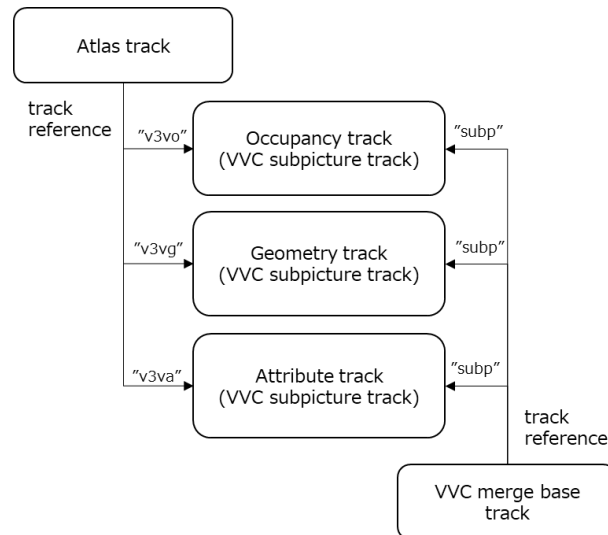


Figure I.5 track structure of V3C content using bitstream reconstruction of VVC subpicture

When the client plays the V3C content, it may identify the VVC merge base track after finding the atlas track. To identify the VVC merge base track, the client may choose the VVC base track that the tracks referenced by the 'subp' track reference in the VVC base track are identical the video component tracks referenced from the track reference in the atlas track.

The decoding process of V-PCC file which multi-track encapsulated atlas, occupancy, geometry, and attribute encoded data is shown in Figure I.6. V-PCC file which multi-track encapsulated atlas, occupancy, geometry, and attribute encoded data is decapsulated to atlas, occupancy, geometry, and attribute bitstream in decapsulator, respectively. The merged bitstream is generated from VVC merge base track and VVC subpicture tracks which are occupancy, geometry, attribute video component track according to bitstream reconstruction defined in ISO/IEC 14496-15.

The bitstream (as merged bitstream in Figure I.6) contained occupancy, geometry, and attribute is decoded with a single decoder instance. The decoded merged data which is output from decoder is divided into occupancy, geometry, and attribute data using the information of placement and resolution in 'trif' sample group in the VVC merge base track. These divided data and atlas data are reconstructed to the volumetric frame.

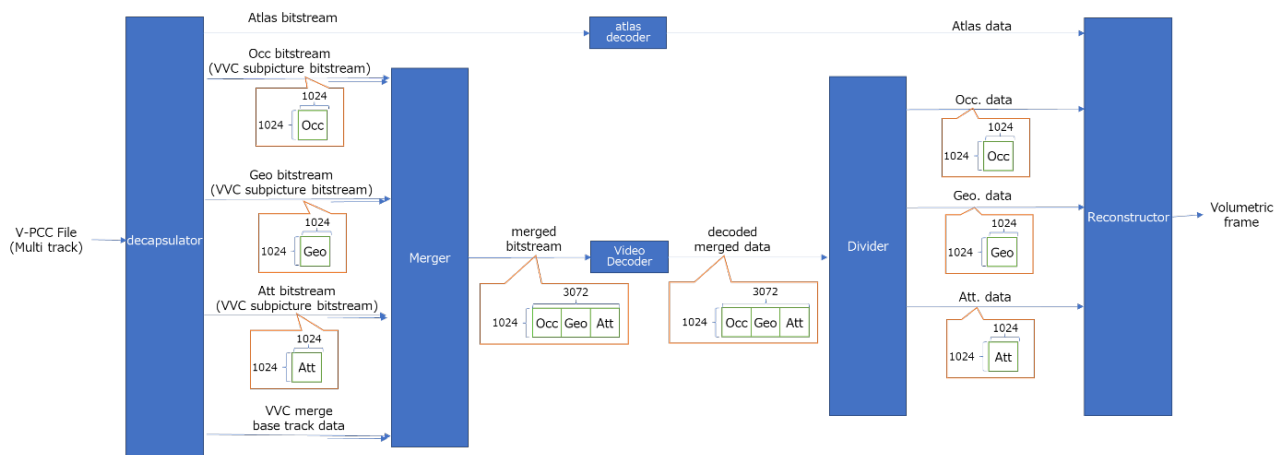


Figure I.6 decoding process using bitstream reconstruction of VVC subpicture

Note1 The VVC picture is rectangular picture. In other words, the merged picture is rectangular picture. To

conformant this condition, it need to consider the resolution of each video component and placement of each video component in the merged picture. In the Figure X.3, all video components are the same resolution, And the merged picture can be rectangular picture by arranging from pictures horizontally.

Note2 The placement of subpictures in the merged bitstream is configured in the VVC merge base track. The placement of subpictures may be arranged not only horizontally, but also vertically, or both horizontally and vertically

Annex H (normative)

Support of 2D snapshot images

H.1 Introduction

As V3C contents are composed of several compressed video bitstreams, it introduces a challenge for quick preview or trick play of the contents. For the applications decoding of several video frames and reconstruction of a volumetric frame is too complex such as quick preview, 2D snapshot image track can be provided. The 2D snapshot image track can contain 2D image of volumetric contents at certain points of time with a camera at a certain position and direction. A client can decode and present 2D snapshot images instead of volumetric contents by decoding compressed bitstreams and compositing/rendering volumetric contents when it is not really needed.

[Ed(SOH) : that support of single track needs to be added.]

H.2 Single directional 2D snapshot image track

H.2.1 Overview

The 2D snapshot image track contains one or more samples of coded bitstream of 2D image of a coded volumetric frame rendered at a certain location and direction. Each sample contains a 2D projected image of a coded volumetric frame whose composition time is same with such sample. There can be more than one 2D snapshot image track for a single CVS and each of them contains different version of snapshot image.

H.2.2 Restriction to the track

The value of `handler_type` of the 2D snapshot image track shall be 'vide.' In other words, the track used as a 2D snapshot image should be the one which could have 'vide' as a value for `handler_type`. All samples in the 2D snapshot image track shall be sync samples. The track shall be represented in the file as restricted video and shall use a generic sample entry 'resv' and `SchemeTypeBox` shall be present in `RestrictedSchemeInfoBox` and `scheme_type` is set to 'sdst'.

H.2.3 Track references

To associate a 2D snapshot image track with the tracks containing V3C data, track reference tool of ISO/IEC 14496-12 shall be used. One or more `TrackReferenceTypeBoxes` shall be added to a `TrackReferenceBox` within the `TrackBox` of the V3C atlas track or V3C atlas tile track, one for each 2D snapshot image tracks. The 2D snapshot image tracks referenced by V3C atlas tile tracks shall have 2D snapshot images specific to atlas tiles carried in such V3C atlas tile tracks. When V3C atlas tile track has reference to 2D snapshot image track, the V3C atlas track shall not include any referenc to 2D snapshot image track. The `TrackReferenceTypeBox` shall contain array of `track_IDs` designating the tracks containing 2D snapshot images which the V3C atlas track or V3C atlas tile track references. The 4CC value of `reference_type` of such `TrackReferenceTypeBox` shall be '2dsi.' When a V3C bitstream is represented by a single-track declaration, one or more such `TrackReferenceTypeBoxes` shall be included in V3C bitstream track.

H.2.4 Indication of camera used for rendering snapshot images

Information about the camera used to render 2D snapshot images is provided as viewport information timed-metadata track. A viewport sample whose composition time is same with a snapshot image provide information about the camera used to render such image. When such viewport timed-metadata track is provided the value of `viewport_type` is set to '0.' The viewport information timed-metadata track shall reference corresponding 2D snapshot image track instead of V3C atlas track and '2dci' shall be used for `reference_type`.

H.3 Multi-directional 2D snapshot image track

H.3.1 Overview

The multi-directional snapshot image track contains one or more samples of coded bitstream of 2D image of a coded volumetric frame. Each sample consist of more than one sub-samples and each sub-sample contains a 2D projected image of a coded volumetric frame where location and direction of projection of each sub-sample are not same each other. `SubSampleInformationBox` shall present in a multi-directional snapshot image track to provide subsample information. The set of location and direction of cameras used for projection shall remain same for a single track. Composition time of the 2D images shall be same with the sample of coded volumetric frame.

H.3.2 Track references

To associate a multi-directional snapshot image track with the tracks containing V3C data, track reference tool of ISO/IEC 14496-12 shall be used. One or more `TrackReferenceTypeBoxes` shall be added to a `TrackReferenceBox` within the `TrackBox` of the V3C atlas track or V3C atlas tile track, one for each 2D snapshot image tracks. The `TrackReferenceTypeBox` shall contain array of `track_IDs` designating the tracks containing 2D snapshot images which the V3C atlas track or V3C atlas tile track references. The 2D snapshot image tracks referenced by V3C atlas tile tracks shall have 2D snapshot images specific to atlas tiles carried in such V3C atlas tile tracks. When V3C atlas tile track has reference to 2D snapshot image track, the V3C atlas track shall not include any referenc to 2D snapthot image track. The 4CC value of `reference_type` of such `TrackReferenceTypeBox` shall be 'mdsi'. When a V3C bitstream is represented by a single-track declaration, one or more such `TrackReferenceTypeBoxes` shall be included in V3C bitstream track.

H.3.3 Restriction to the track

The track referenced from a V3C atlas track or V3C atlas tile track with `reference_type` 'mdsi' shall be represented in the file as restricted video and shall use a generic sample entry 'resv' with following additional requirements:

- `SchemeTypeBox` shall be present in `RestrictedSchemeInfoBox` and `scheme_type` is set to 'mdst'.
- All samples in the track shall be sync samples.
- `SubSampleInformationBox` shall be present and the value of `subsample_count` of `SubSampleInformationBox` shall be same for all entries and greater than one.

H.3.3.1 Multi-dimensional snapshot camera information box

H.3.3.1.1 Definition

Box Type: 'mdst'

Container: SchemeInformationBox

Mandatory: Yes (when the SchemeType is 'mdst')

Quantity: One

The Multi-dimensional snapshot camera information box is used to indicate the information about the camera used to render the snapshots of volumetric frames for each sub-samples indicated by SubSampleInformationBox. *i*-th view port information shall indicate *i*-th sub-sample in bitstream order within a sample.

H.3.3.1.2 Syntax

```
aligned(8) class MultiDimSnapshotCameraInfoBox extends extends FullBox('mdst', version =
0, 0)
{
    unsigned int(8) num_viewports;
    for (int i=1; i <= num_viewports; i++){
        unsigned int(1) camera_extrinsic_flag[i];
        unsigned int(1) camera_intrinsic_flag[i];
        bit(6) reserved = 0;
        ViewportInfo (camera_extrinsic_flag[i], camera_intrinsic_flag[i]);
    }
}
```

H.3.3.1.3 Semantics

`num_viewport` indicates the number of viewport signaled in the sample. The value of this field shall be equal to the value of the value of `subsample_count` of SubSampleInformationBox. *i*-th viewport provides information about the camera for the *i*-th subsample in bitstream order.

`camera_intrinsic_flag[i]` equal to 1 indicates that the intrinsic camera parameters are present in the *i*-th viewport.

`camera_extrinsic_flag[i]` equal to 1 indicates that the extrinsic camera parameters are present in the *i*-th viewport.

`ViewportInfo` provides information about the camera. Then syntax and semantics of this class is specified in subclause 10.2.3 of ISO/IEC 23090-10.

Bibliography

- [1] ISO/IEC 14496-10, *Information technology — Coding of audio-visual objects — Part 10: Advanced Video Coding*.
- [2] ISO/IEC 23008-2, *Information technology — High efficiency coding and media delivery in heterogeneous environments — Part 2: High efficiency video coding*
- [3] ISO/IEC 23090-13:2024 , *Information technology — Coded representation of immersive media — Part 13: Video decoding interface for immersive media*