

IEEE Standard for Immersive Visual Content Coding

IEEE Computer Society

Developed by the
Data Compression Standards Committee

IEEE Std 1857.9™-2021



STANDARDS

IEEE Standard for Immersive Visual Content Coding

Developed by the

Data Compression Standards Committee
of the
IEEE Computer Society

Approved 9 November 2021

IEEE SA Standards Board

Abstract: Efficient coding tool sets for compression, decompression, and reconstructing of the immersive visual content data is provided. The target applications and services include, but are not limited to, virtual reality (VR), such as unmanned aerial vehicle-based VR, augmented reality, panorama video, free-view TV, panoramic stereo video, and other video-/audio-enabled services and applications, such as immersive video streaming, broadcasting, storage, and communication.

Keywords: 3DoF, 6DoF, decoding, encoding, frame, free-view TV, IEEE 1857.9™, image, internet protocol television, IPTV, picture, prediction block, six degrees of freedom, three degrees of freedom, virtual reality, VR

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2022 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 4 March 2022. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-1-5044-8178-6 STD25091
Print: ISBN 978-1-5044-8179-3 STDPD25091

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <https://www.ieee.org/about/corporate/governance/p9-26.html>.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>), appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE SA) Standards Board. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA, and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE does not warrant or represent the accuracy or completeness of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to results and workmanlike effort. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE Standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE is the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter's views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents.**

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and Standards Coordinating Committees are not able to provide an instant response to comments, or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or in revisions to an IEEE standard is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#). An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under US and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include

both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit [IEEE Xplore](#) or [contact IEEE](#). For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#). Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

Patents

IEEE Standards are developed in compliance with the [IEEE SA Patent Policy](#).

IMPORTANT NOTICE

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. IEEE Standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

Participants

At the time this IEEE standard was completed, the Audio Video Coding Working Group had the following membership:

Siwei Ma, *Chair*
Cliff Reader, *Vice Chair*

Yangang Cai
Jie Chen
Weiqiang Chen
Wen Gao
Xuesong Gao
Song Gu
Tiejun Huang
Yawen Lai
Jianliang Lin
Xiangning Liu

Dong Liu
Ke Qiu
Xiaojie Sheng
WenBo Sun
Jun Tian
Ronggang Wang
Yueming Wang
Zhenyu Wang
Rongjie Wang

Qi Wang
Guozhong Wang
Feng Wu
Xiaoyang Wu
Haitao Yang
Lu Yu
Xinggong Zhang
Weimin Zhang
Haiwu Zhao
Xiaozhen Zheng

The following members of the individual Standards Association balloting group voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Oscar Au
Eric Chai
Takuya Fujihashi
Wei Gao
Yun He
Werner Hoelzl
Piotr Karocki
Alexis Kokkonos
Ting Li

Weiping Li
Fan Liang
Dong Liu
Qian Liu
Xiaoxu Luan
Siwei Ma
Luntian Mou
Rajesh Murthy

Joanna Olszewska
R. K. Rannow
Yonghong Tian
Ronggang Wang
Yaowei Wang
Feng Wu
Lu Yu
Yu Yuan
Oren Yuen

When the IEEE SA Standards Board approved this standard on 9 November 2021, it had the following membership:

Gary Hoffman, *Chair*
Jon Walter Rosdahl, *Vice Chair*
John D. Kulick, *Past Chair*
Konstantinos Karachalios, *Secretary*

Edward A. Addy
Doug Edwards
Ramy Ahmed Fathy
J. Travis Griffith
Thomas Koshy
Joseph L. Koepfinger*
David J. Law

Howard Li
Daozhuang Lin
Kevin Lu
Daleep C. Mohla
Chenhui Niu
Damir Novosel
Annette Reilly
Dorothy Stanley

Mehmet Ulema
Lei Wang
F. Keith Waters
Karl Weber
Sha Wei
Howard Wolfman
Daidi Zhong

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1857.9™-2021, IEEE Standard for Immersive Visual Content Coding.

This standard specifies efficient coding tool sets for compression, decompression, and reconstructing of immersive visual content data. The following are the features of this standard: (1) support for interoperability of immersive videos among different devices; (2) support for efficient compression of immersive videos; (3) compatible with traditional 2D video coding standards.

Contents

1. Overview	9
1.1 Scope	9
1.2 Purpose	9
1.3 Word usage	9
1.4 Applications	10
2. Normative references	10
3. Definitions, acronyms, and abbreviations	11
3.1 Definitions	11
3.2 Acronyms and abbreviations	13
4. Operators and mathematical functions	14
4.1 Description	14
4.2 Arithmetic operators	14
4.3 Logical operators	14
4.4 Relation operators	15
4.5 Bitwise operators	15
4.6 Assignment operators	15
4.7 Description methods of bit stream syntax, parsing process, and decoding process	16
5. Panoramic video coding representation and reconstruction	19
5.1 Video source and codec framework	19
5.2 Syntax and semantics	20
5.3 Projection	26
6. Free-view video coding representation and reconstruction	64
6.1 Video source and codec framework	64
6.2 Syntax and semantics	65
6.3 Reconstruction methods	78
7. Bitstream structure of this standard	97
7.1 Description	97
7.2 Bitstream structure inherited from IEEE Std 1857.4™-2018	97
7.3 Bitstream structure inherited from IEEE Std 1857™-2013	98
7.4 Bitstream structure inherited from ISO/IEC 23090-3:2021	99
7.5 Bitstream structure inherited from ISO/IEC 23008-2:2013	101
7.6 Bitstream structure inherited from ISO/IEC 14496-10:2010	102
Annex A (informative) Profile	105
A.1 Description	105
A.2 Profile	105
Annex B (informative) Filtering for near-equiangular pyramid projection	106

IEEE Standard for Immersive Visual Content Coding

1. Overview

1.1 Scope

This standard defines a set of tools for efficient coding of immersive visual content, as well as the corresponding decoding and reconstructing procedure. The immersive visual content includes, but is not limited to, panorama video, free-view video, light field, and three-dimensional (3D) model and other kinds of synthetic visual contents. The coding efficiency is measured by counting the data rates under target visual quality of immersive visual content.

1.2 Purpose

This standard provides efficient coding tool sets for compression, decompression, and reconstructing of the immersive visual content data. The target applications and services include, but are not limited to, virtual reality, such as unmanned aerial vehicle-based virtual reality (VR), augmented reality, panorama video, freeview TV, panoramic stereo video, and other video-/audio-enabled services and applications, such as immersive video streaming, broadcasting, storage, and communication.

1.3 Word usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).^{1, 2}

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*should* equals *is recommended that*).

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

¹ The use of the word *must* is deprecated and cannot be used when stating mandatory requirements, *must* is used only to describe unavoidable situations.

² The use of *will* is deprecated and cannot be used when stating mandatory requirements, *will* is only used in statements of fact.

1.4 Applications

This standard is designed to cover a broad range of applications including, but not limited to, the following:

- Cable TV (CATV) on optical networks, copper, etc.
- Direct broadcast satellite (DBS) video services
- Digital subscriber line (DSL) video services
- Digital terrestrial television broadcasting (DTTB)
- Internet protocol television (IPTV)
- Interactive storage media (ISM), such as optical disks
- Multimedia mailing (MMM)
- Multimedia services on packet networks (MSPN)
- Real-time conversational (RTC) services, such as videoconferencing and videophone
- Remote video surveillance (RVS)

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE Std 1857TM-2013, IEEE Standard for Advanced Audio and Video Coding.^{3, 4}

IEEE Std 1857.4TM-2018, IEEE Standard for Second-Generation IEEE 1857 Video Coding.

ISO/IEC 14496-10:2010/ITU-T H.264, Information technology—Coding of audio-visual objects—Part 10: Advanced Video Coding.⁵

ISO/IEC 23008-2:2013/ITU-T H.265, Information technology—High efficiency coding and media deliver in heterogeneous environments—Part 2: High efficiency video coding.⁶

ISO/IEC 23090-3:2021/ITU-T H.266, Information technology—Coded representation of immersive media—Part 3: Versatile video coding.

³ The IEEE standards or products referred to in Clause 2 are trademarks owned by The Institute of Electrical and Electronics Engineers, Incorporated.

⁴ IEEE publications are available from The Institute of Electrical and Electronics Engineers (<https://standards.ieee.org/>).

⁵ ISO publications are available from the International Organization for Standardization (<https://www.iso.org/>) and the American National Standards Institute (<https://www.ansi.org/>).

⁶ ISO/IEC 23008-2:2013/ITU-T H.265 has been withdrawn.

3. Definitions, acronyms, and abbreviations

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.⁷

angle of view: The range of visible areas on a spherical video frame, which is defined by azimuth and pitch angles.

asymmetric circular projection: A method of representing a spherical video frame with a rectangular frame. This method maps a spherical frame to a main viewpoint region, a vice-main viewpoint region, and a secondary viewpoint region, and then maps these three regions into the rectangular frame.

azimuth: The horizontal coordinate describing the location of a point on the sphere.

binocular stereo equirectangular projection: A method of representing a binocular stereo equirectangular video frame with a rectangular frame, wherein the left and right eye spherical video frames are respectively mapped to two rectangular regions with equirectangular projection, and these two rectangular regions are stitched into the rectangular frame.

bitstream: The binary data stream formed by the encoded image.

camera parameters: Camera parameters are usually composed of intrinsic parameters and extrinsic parameters. The intrinsic parameters describe internal parameters of the camera (such as focal distance and radial lens parameters), and the extrinsic parameters indicate the external position and orientation of the camera in the three-dimensional world.

chroma: A matrix of samples or a single sample of either of two-color difference signals, the symbols are Cr and Cb.

component: One of the three sample matrices (luminance and two chrominance) of a frame or a sample of the three matrices.

cubic projection: A method of representing a spherical video frame with a rectangular frame, wherein the spherical video frame is mapped to an external hexahedron, and the six faces of the hexahedron are stitched into the rectangular frame.

cylinder projection: A method of representing a spherical video frame with a rectangular frame, wherein the spherical frame is mapped to a circumscribed cylinder, and the cylinder is spread into the rectangular frame.

decoder: The entity that completes the decoding process.

decoding process: The process of generating a decoded frame from syntax elements.

depth-image-based rendering (DIBR): A technique that uses depth and texture information from multiple cameras to synthesize texture information of a virtual camera.

⁷ *IEEE Standards Dictionary Online* is available at: <http://dictionary.ieee.org>. An IEEE Account is required for access to the dictionary, and one can be created at no charge on the dictionary sign-in page.

depth map: An image that contains information relating to the distance of the surfaces of scene objects from a viewpoint.

encoder: The entity that completes the encoding process.

encoding process: The process to generate a bitstream conforming to this standard.

equal area bipolar square projection: A method of representing a spherical video frame by using a plane video frame. The method divides the spherical frame into three regions of the south pole, the north pole, and the equator, then maps the three regions into three rectangles, and finally splices the three rectangles into a plane.

equirectangular projection: A method of representing a spherical video frame with a rectangular frame by spreading the spherical video frame along the equator.

frame: The representation of the spatial information of the video signal which consists of a matrix of luminance samples (Y) and two matrices of two chrominance samples (Cb and Cr).

free-view video: Provides user with a six degrees of freedom viewing experience by permission of moving viewpoint in the X, Y, and Z directions, as well as rotating viewpoint around the X, Y, and Z axes in three-dimensional space. *Syn:* six degrees of freedom (**6DoF**) video

icosahedron projection: A method of representing a spherical video frame with a rectangular frame, wherein a spherical video frame is mapped to an external icosahedron, each face of the icosahedron is an equilateral triangle, and all the faces are stitched into the rectangular frame.

luma: A matrix of samples or a single sample representing the luminance signal, the symbol is Y.

main view directed cubic projection: A method of representing a spherical video frame with a rectangular frame. The method maps a spherical video frame to an external hexahedron, defines a certain face of the hexahedron as a primary viewpoint, down-samples the other faces of the hexahedron, and splices all the sampled faces into the rectangular frame.

main viewpoint: A region that covers the user's field of view at a certain time.

minimal deformation bipolar square projection: A method of representing a spherical video frame with a rectangular frame, wherein the spherical frame is divided into three regions including up, middle, and down regions. The up region and down region are mapped into two squares according to an equal area projection method. The middle region is mapped to a rectangular region by equirectangular projection. The two squares and the rectangular region are stitched into the rectangular frame.

near-equiangular cubic projection: A method of representing a spherical video frame with a rectangular frame, wherein the spherical video frame is mapped to a normal hexahedron in near-equiangular manner, and all faces of the normal hexahedron are stitched into the rectangular frame.

near-equiangular pyramid projection: A method of representing a spherical video frame with a rectangular frame. This method maps the spherical frame to an external pyramid, the ground of the pyramid is mapped to a square by pseudo-equiangular projection, and finally the square and other faces of the external pyramid are mapped into the rectangular frame.

panoramic video: A spherical video composed of a set of pixels obtained by uniformly sampling the rays on a spherical surface (or a partially spherical surface) which an observer centered on. *Syn:* three degrees of freedom (**3DoF**) video

parse: The process of obtaining a syntax element from a bitstream.

pitch: The vertical coordinate describing the location of a point on the sphere.

profile: A subset of grammar, semantics, and algorithms specified in this specification.

region of interest: A region that is attractive to user.

representation model: A rectangular video frame that can be used to reconstruct an immersive video frame.

reserved: The values of some specific syntax elements are not defined but saved for extensions to this standard in the future.

ROI-based equirectangular projection: A method of representing a spherical video frame with a rectangular frame, by defining a rectangular region in the equirectangulars as a region of interest (ROI), and the remaining regions outside the region of interest are down-sampled.

secondary viewpoint: The region that is far from the main viewpoint area.

sequence: The highest level of syntax structure for the encoded bitstream, including one or more consecutive encoded images.

slice: A number of consecutive maximum coding unit rows arranged in some order.

spherical video: A video sequence consisting of spherical video frames.

start code: A 32-bit binary code word whose form is unique throughout the bitstream.

syntax element: The result of parsing the data unit in the bitstream.

vice-main viewpoint: The region between the main and secondary viewpoints.

virtual equirectangular projection: A rectangular frame representation of a hypothetical spherical video frame with equirectangular projection.

3.2 Acronyms and abbreviations

3DoF	three degrees of freedom
6DoF	six degrees of freedom
DIBR	depth image based rendering
ERP	equirectangular projection
FoV	field of view
ROI	region of interest
VR	virtual reality

4. Operators and mathematical functions

4.1 Description

The mathematical operators used in this standard are similar to those used in the C programming language. However, integer division and arithmetic shift operations are specifically defined. Numbering and counting conventions begin with 0 unless otherwise specified.

4.2 Arithmetic operators

The arithmetic operators are defined in Table 1.

Table 1—Arithmetic operator definition

Arithmetic operator	Definition
+	Addition
−	Subtraction (binary operator) or negation (unary prefix operator)
×	Multiplication
a^b	A power operation that represents the power of b to a ; it can also indicate superscript
/	The division operation is truncated along the direction of the value of 0; for example, $7/4$ and $-7/-4$ are truncated to 1, $-7/4$ and $7/-4$ to -1
÷	Division, no truncation or rounding
$\frac{a}{b}$	Division, no truncation or rounding
$\sum_{i=a}^b f(i)$	The cumulative sum of functions when the argument i takes all integer values from a to b (including b)
$a \% b$	Modular operation, a divided by the remainder of b , where a and b are positive integers
exp	Exponential operation
sqrt	Square root operation
arctg	Arctangent operation
arcsin	Anti-sinusoidal operation
arccos	Inverse cosine operation
abs	Absolute value
max	Take a larger value
min	Take a smaller value
tan	Tangent operation
sin	Sinusoidal operation
cos	Cosine operation

4.3 Logical operators

The logical operator definitions are shown in Table 2.

Table 2—Logical operator definitions

Logical operators	Definition
$a \ \&\& \ b$	Logical AND operations between a and b
$a \ \ b$	OR logical operation between a and b
!	Logical NON operation

4.4 Relation operators

The relation operator definitions are shown in Table 3.

Table 3—Relational operator definition

Relational operator	Definition
>	More than
>=	More than or equal to
<	Less than
<=	Less than or equal to
==	Equal to
!=	Not equal to

4.5 Bitwise operators

The bitwise operator definitions are shown in Table 4.

Table 4—Bitwise operator definition

Bit operator	Definition
&	And operation.
	Or operation.
~	Inverse operation.
$a \gg b$	Move a to the right b bit in the form of 2 Complement integer. Define this operation only when b takes a positive number.
$a \ll b$	Move a to the left b bit in the form of 2 Complement integer. Define this operation only when b takes a positive number.

4.6 Assignment operators

The assignment operation definitions are shown in Table 5.

Table 5—Assignment calculation definition

Assignment operation	Definition
=	Assignment operator.
++	Increment, $x++$ is equivalent to $x = x + 1$. When used for array subscripting, find the variable value before the self-addition operation.
--	Decrement, $x--$ is equivalent to $x = x - 1$. When used for array subscripting, find the variable value before the decrement operation.
+=	Add a specified value, for example $x += 3$ is equivalent to $x = x + 3$, $x += (-3)$ is equivalent to $x = x + (-3)$.
-=	Decrease the specified value, for example $x -= 3$ is equivalent to $x = x - 3$, and $x -= (-3)$ is equivalent to $x = x - (-3)$.

4.7 Description methods of bit stream syntax, parsing process, and decoding process

4.7.1 Description method of bit stream syntax

The description method of a bit stream syntax is similar to C language. The syntax elements of bit stream are represented in bold type, and each syntax element is described by name (English alphabet separated by underline, all letters are lowercase), syntax, and semantics. The values of syntax elements in the syntax table and body are represented in regular fonts.

In some cases, the values of other variables derived from syntax elements can be applied to the grammar table. Such variables are named with a mixture of lowercase and uppercase letters without underscoring in the grammar table or body. Variables beginning with a capital letter are used to decode the current and related syntactic structures, as well as subsequent syntactic structures. Variables that begin with a lowercase letter are used only within the section in which they are located.

The relationship between mnemonics of syntax element values and variable values and their values is described in the text. In some cases, both are used equally. Mnemonics are represented by one or more letter groups separated by underscores. Each letter group starts with a capital letter and can also include multiple capital letters.

When the length of the bit string is an integer multiple of 4, it can be represented by hexadecimal symbols. The hexadecimal prefix is "0x", for example, "0x1a" means the bit string "0001 1010".

In the conditional statement, 0 means false and non-0 means true.

The syntax table describes all supersets that conform to the bit stream syntax of this part, and the additional syntax restrictions are described in the relevant section.

Table 6 shows examples of pseudo-code describing the syntax. When a syntax element appears, it means that a data unit is read from the bit stream.

Table 6—Syntax description of pseudo-code

Pseudo-code	Description
/* Statement is a descriptor of a syntax element, or the existence, type, and value of a syntax element. Two examples are given below. */	
syntax_element	ue(v)
conditioning statement	
/* The statement group enclosed by curly brackets is a compound statement, which is regarded as a single statement functionally. */	
{	
statement	
...	
}	
/* “while” Statement to test whether the condition is true, and if true, the body of the loop is repeated until the condition is not true. */	
while (condition)	
statement	
/* “do ... while” Statement first executes the loop body once, and then tests whether the condition is true. If true, the loop body is executed repeatedly until the condition is not true. */	
do	
statement	
while (condition)	
/* “if ... else” Statement first tests the condition. If true, the primary statement is executed, otherwise the alternative statement is executed. If the alternative statement does not need to be executed, the "else" part of the structure and the associated alternative statement can be ignored. */	
if (condition)	
primary statement	
else	
alternative statement	
/* “for” Statement first executes the initial statement, and then tests the condition. If the condition is true, repeat the primary statement and sub-sequence statement until the condition is not true. */	
for (initial statement; condition; subsequent statement)	
primary statement	

The parsing process and decoding process are described by words and pseudo-code similar to C language.

4.7.2 Functions

The functions below are used to describe syntax. These functions assume that there is a bitstream pointer in the decoder. The pointer indicates the position of the next bit to be read from the bitstream. A function name and the parameters in the parentheses make up a function; however, a function may also exist without parameters.

next_start_code(): Finding the next start code in bitstream, pointing the bitstream pointer to the first binary bit of the start code prefix.

read_bits(n): Returns the next n bits of the bitstream, MSB is in front, and the bitstream pointer moves forward n bits. If n is equal to 0, then returns 0 and the bit stream pointer does not move forward.

Functions are also used to describe the description of the parsing process and the decoding process.

4.7.3 Descriptor

Descriptor represents the parse of different syntax elements, as shown in Table 7.

Table 7—Descriptor

Descriptor	Illustration
b(8)	A byte that can be arbitrarily valued. The parsing process is specified by the return value of the function read_bits(8).
f(n)	Take consecutive n binary bits of a specific value. The parsing process is specified by the return value of the function read_bits(n).
i(n)	An n -bit integer. In the syntax table, if n is "v", its number of bits is determined by other syntax element values. The parsing process is specified by the return value of the function read_bits(n), which is represented by the 2's complement of the high order first.
r(n)	There are n consecutive '0's. The parsing process is specified by the return value of the function read_bits(n).
u(n)	An n -bit unsigned integer. In the syntax table, if n is "v", its number of bits is determined by other syntax element values. The parsing process is specified by the return value of the function read_bits(n), which is represented by the high-order first binary representation.

4.7.4 Retention, prohibition, and marking bits

In the bitstream syntax defined in this section, the values of some syntax elements are marked as "reserved" or "forbidden".

"Reserved" defines some specific syntax element values for future extensions to this section. These values should not appear in the bitstream that conforms to this section.

"Forbidden" defines some specific syntax element values that should not appear in the bitstream that conforms to this section.

"marker_bit" means that the value of this bit should be '1'.

The "reserved_bits" in the bitstream indicate that some syntax units are reserved for future extensions of this part, and the decoding process should ignore these bits. "reserved_bits" should not appear for more than 21 consecutive '0's starting from any byte aligned position.

5. Panoramic video coding representation and reconstruction

5.1 Video source and codec framework

The panoramic video (3DoF video) source in this standard is a spherical video which is composed of a set of pixels by uniformly sampling on a spherical surface (or a partially spherical surface) which takes the observer as the center. To facilitate storage and process, the video source is mapped to a plane video through panoramic video content representation model specified in this standard. The coordinate of the pixel in the spherical video frame is (θ, ϕ) , where θ is the azimuth angle and ϕ is the pitch angle. The units of θ and ϕ are radians, the range of θ is $-\pi$ to π , and the range of ϕ is $-\pi/2$ to $\pi/2$. As shown in Figure 1, if (θ, ϕ) is to the left of $(0, 0)$, θ is negative; otherwise θ is positive; if (θ, ϕ) is below $(0, 0)$, then ϕ is negative; otherwise ϕ is positive.

The coordinate of the pixel in the plane video frame is (x, y) , where x represents the index of the column in which the pixel is located in the associated plane video frame, the index of the leftmost column is 0; y represents the index of the row in which the pixel is located in the associated plane video frame, the index of the top row is 0.

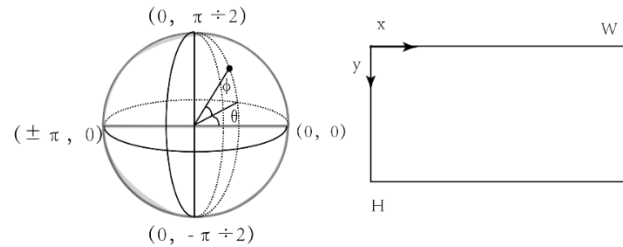


Figure 1—The coordinate relation between spherical and plane video

The overall codec framework is shown in Figure 2. In the encoder, the spherical video source is mapped to a plane video by a panoramic video content representation model, the plane video is encoded by the traditional video coding method, and the representation model information is embedded in the plane video bitstream. In the decoder, according to the representation model information parsed from the bitstream, the plane video is decoded and reconstructed to the spherical video and rendered in the VR terminal device.

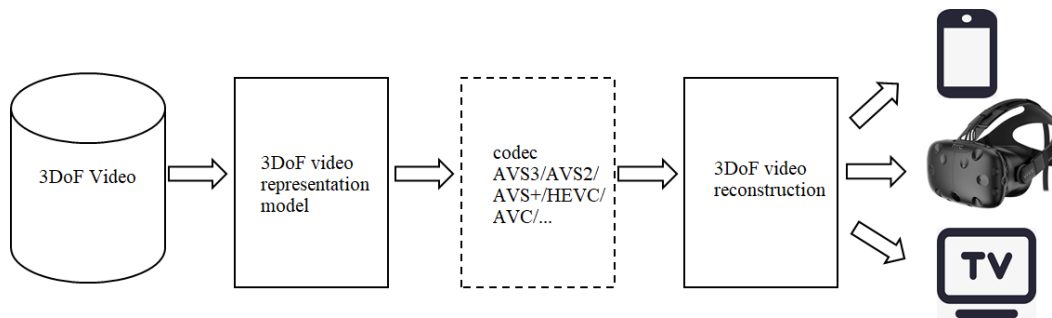


Figure 2—Video codec framework for immersive video (3DoF video)

This standard defines the syntax and semantics of the panoramic video content representation model in bitstream. The syntax is described in 5.2.1, and semantics are described in 5.2.2. This standard specifies the mapping relationship between panoramic video and plane video in 5.3. This standard also specifies the method of embedding panoramic video representation model syntax into traditional video bitstream in Clause 7.

5.2 Syntax and semantics

5.2.1 Syntax

The syntax of panoramic video content representation model definition is shown in Table 8.

Table 8—The syntax of the representation model of panoramic video

The syntax of the representation model of panoramic video	Descriptor
3dof_extension() {	
3dof_profile_id	u(8)
3dof_model	u(8)
marker_bit	f(1)
content_orientation_flag	u(1)
if (content_orientation_flag){	
roll_rotation_degree	i(16)
marker_bit	f(1)
yaw_rotation_degree	i(16)
marker_bit	f(1)
pitch_rotation_degree	i(15)
marker_bit	f(1)
}	
default_fov_flag	u(1)
if (!default_fov_flag){	
horizontal_half_fov	u(8)
vertical_half_fov	u(7)
marker_bit	f(1)
}	
slice_decoding_dependency_flag	u(1)
if (3dof_model == 0 3dof_model == 1 3dof_model == 10)	
if (!default_fov_flag && 3dof_profile_id == 0x40){	
center_offset_h_sign	u(1)
center_offset_h_level	u(8)
center_offset_v_sign	u(1)
center_offset_v_level	u(7)
}	
if (3dof_model == 1) { /* Binocular Stereo Equirectangular Projection */	
stereo_layout	u(1)
}	
}	
else if (3dof_model == 9) { /* Cubic Projection */	
cubic_type	u(4)

default_layout_flag	u(1)
if (!default_layout_flag){	
for (i = 0; i < 6; i++) {	
face_index	u(3)
face_rotation	u(2)
marker_bit	f(1)
}	
}	
}	
else if (3dof_model == 2) { /* ROI-based Equirectangular Projection */	
virtual_source_width	u(14)
marker_bit	f(1)
virtual_source_height	u(14)
marker_bit	f(1)
roi_width	u(14)
marker_bit	f(1)
roi_height	u(14)
}	
else if (3dof_model == 6) { /* Asymmetric Circular Projection */	
main_viewpoint_half_range	u(8)
submain_viewpoint_half_range	u(8)
marker_bit	f(1)
main_viewpoint_radius	u(14)
marker_bit	f(1)
main_viewpoint_center_longitude_sign	u(1)
main_viewpoint_center_longitude_level	u(8)
main_viewpoint_center_latitude_sign	u(1)
main_viewpoint_center_latitude_level	u(7)
}	
else if (3dof_model == 11) { /* Main viewpoint directed cubic projection */	
downsample_factor	u(8)
main_viewpoint_center_longitude_sign	u(1)
main_viewpoint_center_longitude_level	u(8)
marker_bit	f(1)
main_viewpoint_center_latitude_sign	u(1)
main_viewpoint_center_latitude_level	u(7)
}	
else if (3dof_model == 5) { /* Near-equiangular pyramid projection */	
main_viewpoint_center_longitude_sign	u(1)
main_viewpoint_center_longitude_level	u(8)

marker_bit	f(1)
main_viewpoint_center_latitude_sign	u(1)
main_viewpoint_center_latitude_level	u(7)
main_viewpoint_half_range	u(8)
attenuation_factor	u(6)
}	
else if (3dof_model == 4) { /* Minimal deformation bipolar square projection */	
padding_size	u(8)
latitude_offset	u(8)
}	
marker_bit	f(1)
next_start_code()	
}	

5.2.2 Semantics

3dof_profile_id

8-bit unsigned integer. It indicates the profile of panoramic video representation model, detailed information refers to Annex A.

3dof_model

8-bit unsigned integer. It identifies the panoramic video representation model. Table 9 gives the values and meaning of panoramic video. The representation model is used to specify the mapping relationship between panoramic video and plane video, which is described in 5.3.

Table 9—Panoramic video representation model

Value of 3dof_model	Panoramic video representation model
0	Equirectangular projection
1	Binocular stereo equirectangular projection
2	Region of interest (ROI)-based equirectangular projection
3	Near-equiangular cubic projection
4	Minimal deformation bipolar square projection
5	Near-equiangular pyramid projection
6	Asymmetric circular projection
7	Equal-area bipolar square projection
8	Icosahedron projection
9	Cubic projection
10	Cylinder projection
11	Main viewpoint directed cubic projection
Others	Reserved

marker_bit

Binary variable. The value is 1. It is used to avoid the occurrence of pseudo-start codes in the bitstream.

content_orientation_flag

Binary variable. It is used to indicate whether panoramic video content needs to be rotated after it is decoded and reconstructed. A value of '0' indicates that panoramic video content does not need to be rotated, and the value of '1' indicates that panoramic video content needs to be rotated in order of roll, yaw, and pitch respectively. The direction and angle of rotation along each axis are determined by the subsequent syntax elements of `roll_rotation_degree`, `yaw_rotation_degree`, and `pitch_rotation_degree`. Figure 3 shows the relative positions of the three axes of rotation.

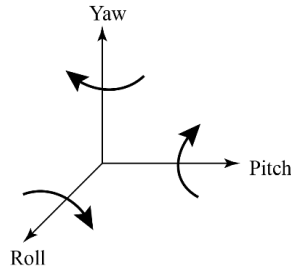


Figure 3—Three axes of rotation of virtual reality

roll_rotation_degree

16-bit unsigned integer. The highest bit indicates the direction of rotation along the roll axis: the value of '0' indicates rotation in the counterclockwise direction, and the value of '1' indicates rotation in the clockwise direction. The lower 15 bits indicate the amplitude of rotation, in unit of 2^{-7} degree, and the range is from 0 to 180 degrees.

yaw_rotation_degree

16-bit unsigned integer. The highest bit indicates the direction of rotation along the yaw axis: the value of '0' indicates rotation in the counterclockwise direction, and the value of '1' indicates rotation in the clockwise direction. The lower 15 bits indicate the amplitude of rotation, in unit of 2^{-7} degree, and the range is from 0 to 180 degrees.

pitch_rotation_degree

15-bit unsigned integer. The highest bit indicates the direction of rotation along the pitch axis: the value of '0' indicates rotation in the counterclockwise direction, and the value of '1' indicates rotation in the clockwise direction. The lower 14 bits indicate the amplitude of rotation, in unit of 2^{-7} degree, and the range is from 0 to 90 degrees.

default_fov_flag

Binary variable. It indicates whether the horizontal view range (hFOV) and vertical view range (vFOV) of the panoramic video are default values. The value of '0' indicates that the values of hFOV and vFOV are determined by the subsequent syntax elements of `horizontal_half_fov` and `vertical_half_fov`, respectively; the value of '1' indicates that hFOV and vFOV take default values, and the values are determined as follows:

```
if (3dof_model == 1) // Binocular stereo equirectangular projection
    hFOV = 180
    vFOV = 180
else if (3dof_model == 10) // Cylinder projection
```

hFOV = 360
vFOV = 90
else // Others
hFOV = 360
vFOV = 180

horizontal_half_fov

8-bit unsigned integer. It indicates half of the horizontal viewing angle range, in unit of degree, ranging from 0 to 180. The horizontal view range hFOV of the panoramic video is derived by:

$$\text{hFOV} = 2 \times \text{horizontal_half_fov}$$

vertical_half_fov

7-bit unsigned integer. It indicates half of the vertical viewing angle range, in unit of degree, ranging from 0 to 90. The vertical view range vFOV of the panoramic video is derived by:

$$\text{vFOV} = 2 \times \text{vertical_half_fov}$$

center_offset_h_sign

Binary variable. It indicates the direction of azimuth offset of the center point of the effective spherical area relative to the center of the sphere (0, 0). A value of '0' indicates an offset in the counterclockwise direction, and a value of '1' indicates an offset in the clockwise direction.

center_offset_h_level

8-bit unsigned integer. It indicates the azimuth offset of the center point of the effective spherical area relative to the center of the sphere (0, 0), in unit of degree, ranging from 0 to 180.

center_offset_v_sign

Binary variable. It indicates the pitch angle deviation direction of the center point of the effective spherical area relative to the center of the sphere (0, 0). A value of '0' indicates an offset in the counterclockwise direction, and a value of '1' indicates an offset in the clockwise direction.

center_offset_v_level

7-bit unsigned integer. It indicates the azimuth offset of the center point of the effective spherical area relative to the center of the sphere (0, 0), in unit of degree, ranging from 0 to 90.

slice_decoding_dependency_flag

Binary variable. When its value is '1', each frame in the video sequence uses the same division method to divide the frame into multiple slices, and the inter-frame prediction reference data used by a block in the current slice shall not exceed the range of its collocated slice in the reference frame. When its value is '0', there's no constraint for the encoding process.

stereo_layout

Binary variable. A value of '0' indicates that the left half of the decoded video frame is used to reconstruct the left eye panoramic video, and the right half is used to reconstruct the right eye panoramic video; A value of '1' indicates that the top half of the decoded video frame is used to reconstruct the left eye panoramic video, and the bottom half is used to reconstruct the right eye panoramic video.

cubic_type

2-bit unsigned integer. It identifies the arrangement type for the six faces of the normal hexahedron, explained in 5.3.11.

default_layout_flag

Binary variable. A value of '1' indicates that the six faces of the normal hexahedron are arranged in a default manner, and a value of '0' indicates that the arrangement of the six faces of the normal hexahedron is determined by the subsequent syntax element of cubic_type.

face_index

3-bit unsigned integer. It identifies the index of the face in the normal hexahedron, the relationship between the face index and the face is illustrated in 5.3.11.

face_rotation

2-bit unsigned integer. It indicates that the pixel coordinates of the corresponding face shall be rotated counterclockwise by some degrees, explained in 5.3.11.

virtual_source_width

14-bit unsigned integer. It indicates the width of the virtual equirectangular projection, explained in 5.3.4.

virtual_source_height

14-bit unsigned integer. It indicates the height of the virtual equirectangular projection, explained in 5.3.4.

roi_width

14-bit unsigned integer. It indicates the width of the region of interest, explained in 5.3.4.

roi_height

14-bit unsigned integer. It indicates the height of the region of interest, explained in 5.3.4.

main_viewpoint_center_longitude_sign

Binary variable. It indicates the azimuth direction of the center of the main viewpoint relative to the center of the sphere (0, 0). A value of '0' indicates counterclockwise direction, and a value of '1' indicates clockwise direction.

main_viewpoint_center_longitude_level

8-bit unsigned integer. It indicates the azimuth offset of the center of the main viewpoint relative to the center of the sphere (0, 0) in unit of degree.

main_viewpoint_center_latitude_sign

Binary variable. It indicates the pitch direction of the center of the main viewpoint relative to the center of the sphere (0, 0). A value of '0' indicates counterclockwise direction, and a value of '1' indicates clockwise direction.

main_viewpoint_center_latitude_level

7-bit unsigned integer. It indicates the pitch offset of the center of the main viewpoint relative to the center of the sphere (0, 0) in unit of degree.

downsample_factor

8-bit unsigned integer. It indicates the sampling density decrease rate of the non-main viewpoint region of the main viewpoint directed cubic projection model.

main_viewpoint_half_range

8-bit unsigned integer. It indicates the angle between the boundary of main viewpoint and the center of the main viewpoint in unit of degree.

submain_viewpoint_half_range

8-bit unsigned integer. It indicates the angle between the boundary of the second primary viewpoint and the center of the primary viewpoint in unit of degree.

main_viewpoint_radius

3-bit unsigned integer. It indicates the radius of the primary view region, which is mapped to a circular area on a plane video in unit of pixel.

attenuation_factor

6-bit unsigned integer. It indicates the sampling density decrease rate of the near-equiangular pyramid projection; its valid value is 1 to 49.

padding_size

8-bit unsigned integer. It indicates the number of pixels to be padded, and its valid value is 1 to 255.

latitude_offset

8-bit unsigned integer. It indicates the latitude offset, and its valid value is 32 to 255.

5.3 Projection

5.3.1 Description

This part specifies the projection relationship between the plane video frame and the spherical video frame after decoding. As shown in Figure 1, the pixel coordinates of the plane video frame are represented by (x, y) , and the spherical video frame coordinates are represented by (θ, ϕ) . The width and height of the plane video frame are represented by variables W and H , and the values of W and H are determined by syntax elements or variables in the conventional video coding standard, explained in Clause 7.

The azimuth and pitch range of the spherical video frame are represented by the variables $hFOV$ and $vFOV$, the values of $hFOV$ and $vFOV$ are determined by the syntax elements of `default_fov_flag`, `horizontal_half_fov`, and `vertical_half_fov` as follows:

```
if (default_fov_flag)
    if (3dof_model == 1) // Binocular stereo equirectangular projection
        hFOV = 180
        vFOV = 180
    else if (3dof_model == 10) // Cylinder projection
        hFOV = 360
        vFOV = 90
    else // Others
```

```

hFOV = 360
vFOV = 180

else

hFOV = horizontal_half_fov × 2
vFOV = vertical_half_fov × 2

```

When 3dof_profile_id is equal to 0x20, the center point C of the spherical video frame shall be at the (0, 0); when 3dof_profile_id is equal to 0x40, the coordinates of the spherical video frame center point C in the azimuth and pitch directions are respectively shifted by hCenterOffset and vCenterOffset, and hCenterOffset and vCenterOffset are derived by the relevant syntax elements as follows:

```

if (center_offset_h_sign == 0)
    hCenterOffset = center_offset_h_level
else
    hCenterOffset = -center_offset_h_level
if (center_offset_v_sign == 0)
    vCenterOffset = center_offset_v_level
else
    vCenterOffset = -center_offset_v_level

```

5.3.2 Equirectangular projection

When the value of 3dof_model is 0, the representation model is equirectangular projection.

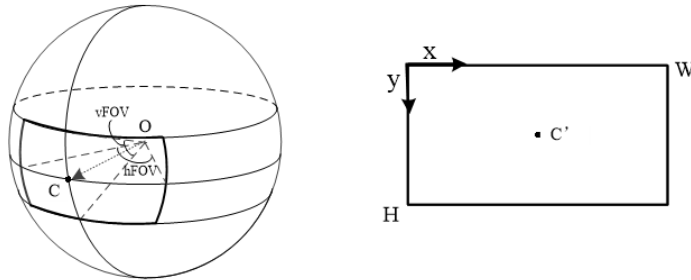


Figure 4—Equirectangular projection (point C is the center of the spherical video frame)

As shown in Figure 4, in general, the center of the spherical video frame is at (hCenterOffset, vCenterOffset), the azimuth range of the effective sphere which corresponding to the plane video frame is (hCenterOffset – hFOV ÷ 2) to (hCenterOffset + hFOV ÷ 2), the pitch range is (vCenterOffset – vFOV ÷ 2) to (vCenterOffset + vFOV ÷ 2). The range of hCenterOffset is –180 to 180, the range of vCenterOffset shall satisfy the following two conditions:

$$(vCenterOffset - vFOV \div 2) \geq -90$$

$$(vCenterOffset + vFOV \div 2) \leq 90$$

The mapping relationship between the pixel coordinates (x, y) of plane video frame and the corresponding pixel coordinates (θ, φ) of spherical video frame is:

$$\begin{aligned}\theta &= 2 \times (\text{hFOV} \div 360) \times \pi \times (x - W \div 2 + 0.5) \div W + (\text{hCenterOffset} \div 180) \times \pi \\ \text{if } (\theta > \pi) \\ \theta &= \theta - 2\pi \\ \text{else if } (\theta < -\pi) \\ \theta &= \theta + 2\pi \\ \varphi &= (\text{vFOV} \div 180) \times \pi \times (H \div 2 - y - 0.5) \div H + (\text{vCenterOffset} \div 180) \times \pi\end{aligned}$$

5.3.3 Binocular stereo equirectangular projection

When the value of `3dof_model` is 1, the model is binocular stereo equirectangular projection. The plane video represented by bitstream is obtained by stitching two rectangular regions, these two regions are mapped from the left and right eye spherical video frames with equirectangular projection respectively. The way of stitching two rectangular regions is determined by the syntax element of `stereo_layout`. Figure 5 gives a diagram of the 180-degree binocular stereo equirectangular projection when `stereo_layout` is equal to 0.

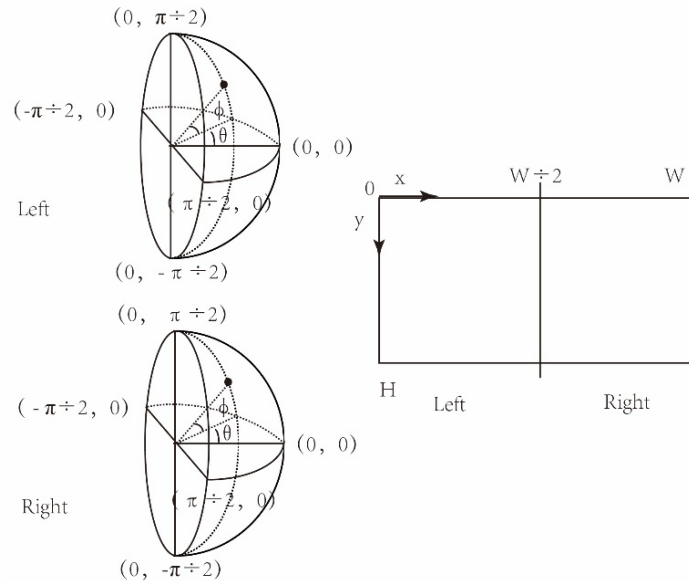


Figure 5—180° binocular stereo equirectangular projection when `stereo_layout` is equal to 0

In general, the center of each eye video frame are $(\text{hCenterOffset}, \text{vCenterOffset})$, the range of azimuth angle is $(\text{hCenterOffset} - \text{hFOV} \div 2)$ to $(\text{hCenterOffset} + \text{hFOV} \div 2)$, the range of pitch angle is $(\text{vCenterOffset} - \text{vFOV} \div 2)$ to $(\text{vCenterOffset} + \text{vFOV} \div 2)$. The range of hCenterOffset is -180 to 180 , the range of vCenterOffset shall satisfy the following two conditions:

$$\begin{aligned}(\text{vCenterOffset} - \text{vFOV} \div 2) &\geq -90 \\ (\text{vCenterOffset} + \text{vFOV} \div 2) &\leq 90\end{aligned}$$

The mapping relationship between the pixel coordinates (x, y) of plane video frame and the corresponding pixel coordinates (θ , φ) of spherical video frame is:

```

if (stereo_layout == 0) // Stitching in left and right order:
    if (x < W/2) // The pixel belongs to the left spherical frame.
         $\theta = (\text{hFOV} \div 360) \times 2 \times \pi \times (x - W \div 4 + 0.5 \div (W/2) + (\text{hCenterOffset} \div 180) \times \pi$ 
        if ( $\theta > \pi$ )
             $\theta = \theta - 2\pi$ 
        else if ( $\theta < -\pi$ )
             $\theta = \theta + 2\pi$ 
    else // The pixel belongs to the right spherical frame.
         $\theta = (\text{hFOV} \div 360) \times 2 \times \pi \times (x - W \times 3 \div 4 + 0.5) \div (W/2) + (\text{hCenterOffset} \div 180) \times \pi$ 
        if ( $\theta > \pi$ )
             $\theta = \theta - 2\pi$ 
        else if ( $\theta < -\pi$ )
             $\theta = \theta + 2\pi$ 
     $\varphi = (\text{vFOV} \div 180) \times \pi \times (H \div 2 - y - 0.5) \div H + (\text{vCenterOffset} \div 180) \times \pi$ 
else // Stitching in up and down order:
    if (y < H/2) // The pixel belongs to the left spherical frame.
         $\varphi = (\text{vFOV} \div 180) \times \pi \times (H \div 4 - y - 0.5 \div (H/2) + (\text{vCenterOffset} \div 180) \times \pi$ 
    else // The pixel belongs to the right spherical frame.
         $\varphi = (\text{vFOV} \div 180) \times \pi \times (H \times 3 \div 4 - y - 0.5) \div (H/2) + (\text{vCenterOffset} \div 180) \times \pi$ 
         $\theta = (\text{hFOV} \div 360) \times 2 \times \pi \times (x - W \div 2 + 0.5) \div W + (\text{hCenterOffset} \div 180) \times \pi$ 
        if ( $\theta > \pi$ )
             $\theta = \theta - 2\pi$ 
        else if ( $\theta < -\pi$ )
             $\theta = \theta + 2\pi$ 

```

5.3.4 ROI-based equirectangular projection

When the value of 3dof_model is 2, the model is the region of interest-based equirectangular projection. As shown in Figure 6, ROI-based ERP is obtained by down-sampling the virtual ERP. The virtual ERP is a plane video frame generated by mapping a spherical video according to the ERP model. The width and height of the ROI-based ERP are calculated by the syntax elements of virtual_source_width and virtual_source_height, respectively. The ROI is the rectangular region defined around the center of ROI-based ERP, and the width and height of the ROI are obtained by the syntax elements of roi_width and roi_height, respectively. In ROI-based ERP, the sampling density of ROI is unchanged, and the regions outside ROI are gradually down-sampled according to their distance to ROI.

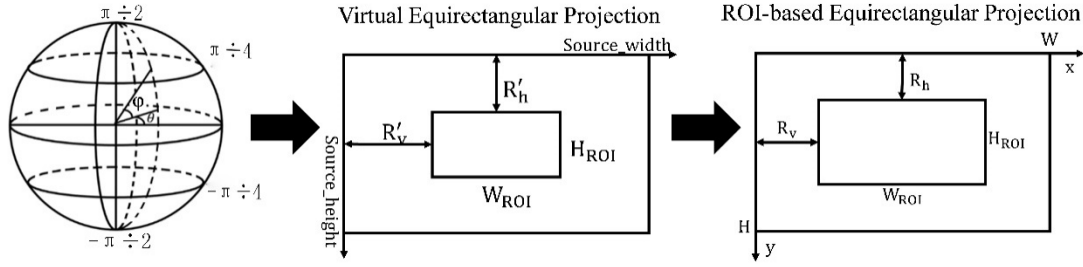


Figure 6—Schematic of ROI-based equirectangular projection

The width, W_{ROI} , and the height, H_{ROI} , of the ROI are equal to the syntax elements of `roi_width` and `roi_height`, respectively.

$$W_{ROI} = \text{roi_width}$$

$$H_{ROI} = \text{roi_height}$$

According to the width, W , and the height, H , obtained from the bitstream, W_{ROI} and H_{ROI} , the distances between the ROI and the plane video boundaries, R_h and R_v , in Figure 6 are calculated:

$$R_h = (W - W_{ROI}) \div 2$$

$$R_v = (H - H_{ROI}) \div 2$$

The width, $w_{Virtual}$, of virtual ERP is equal to the syntax element of `virtual_source_width`, and the height, $h_{Virtual}$, is equal to the syntax element of `virtual_source_height`.

$$w_{Virtual} = \text{virtual_source_width}$$

$$h_{Virtual} = \text{virtual_source_height}$$

The maximum down-sampling ratios rX and rY are defined as follows. The rX is the maximum down-sampling ratios in the horizontal direction when mapping virtual ERP to ROI-based ERP, and the rY is the maximum down-sampling ratios in the vertical direction.

$$rX = (2 \times (W - W_{ROI})) \div ((w_{Virtual} - W_{ROI})) - 1$$

$$rY = (2 \times (H - H_{ROI})) \div ((h_{Virtual} - H_{ROI})) - 1$$

$rH_{Virtual}$ is the distance between virtual ERP and the left boundary of ROI-based ERP, and $rV_{Virtual}$ is the distance between virtual ERP and the top boundary of ROI-based ERP.

$$rH_{Virtual} = (w_{Virtual} - W_{ROI}) \div 2$$

$$rV_{Virtual} = (h_{Virtual} - H_{ROI}) \div 2$$

The correspondence between the planar coordinates (x, y) of the pixels in the plane video frame and the spherical coordinates (θ, ϕ) is as follows.

$$\theta = 2 \times \pi \times (x_{Virtual} - w_{Virtual} \div 2 + 0.5) \div w_{Virtual}$$

$$\phi = \pi \times (h_{Virtual} \div 2 - y_{Virtual} - 0.5) \div h_{Virtual}$$

Where (xVirtual, yVirtual) is the coordinate on virtual ERP corresponding to the planar coordinate (x, y).

The correspondence between xVirtual and x is as follows.

$$\begin{aligned}
 &\text{if } (x < R_h) \\
 &\quad x_{\text{Virtual}} = r_{\text{HVirtual}} - r_{\text{HVirtual}} \div (1 - r_X) + \sqrt{(r_{\text{HVirtual}} \div (1 - r_X))^2 - (2 \times r_{\text{HVirtual}}) \div (1 - r_X) \times (R_h - x))} \\
 &\quad \text{else if } (x > R_h + W_{\text{ROI}}) \\
 &\quad \quad x_{\text{Virtual}} = r_{\text{HVirtual}} + W_{\text{ROI}} + r_{\text{HVirtual}} \div (1 - r_X) - \sqrt{(r_{\text{HVirtual}} \div (1 - r_X))^2 - (2 \times r_{\text{HVirtual}}) \div (1 - r_X) \times (x - r_{\text{HVirtual}} - W_{\text{ROI}}))} \\
 &\quad \text{else} \\
 &\quad \quad x_{\text{Virtual}} = r_{\text{HVirtual}} + x - R_h
 \end{aligned}$$

The correspondence between yVirtual and y is as follows.

$$\begin{aligned}
 &\text{if } (y < R_v) \\
 &\quad y_{\text{Virtual}} = r_{\text{VVirtual}} - r_{\text{VVirtual}} \div (1 - r_Y) + \sqrt{(r_{\text{VVirtual}} \div (1 - r_Y))^2 - (2 \times r_{\text{VVirtual}}) \div (1 - r_Y) \times (R_v - y))} \\
 &\quad \text{else if } (y > R_v + H_{\text{ROI}}) \\
 &\quad \quad y_{\text{Virtual}} = r_{\text{VVirtual}} + H_{\text{ROI}} + r_{\text{VVirtual}} \div (1 - r_Y) - \sqrt{(r_{\text{VVirtual}} \div (1 - r_Y))^2 - (2 \times r_{\text{VVirtual}}) \div (1 - r_Y) \times (y - r_{\text{VVirtual}} - H_{\text{ROI}}))} \\
 &\quad \text{else} \\
 &\quad \quad y_{\text{Virtual}} = r_{\text{VVirtual}} + y - R
 \end{aligned}$$

5.3.5 Near-equiangular cubic projection

When the value of 3dof_model is 3, the model is near-equiangular cubic projection. The plane video frame represented by bitstream is shown in Figure 7, which is obtained by mapping a sphere video frame into a normal hexahedron according to the near equiangular relation and arranging the six faces in the arrangement way of 3×2.

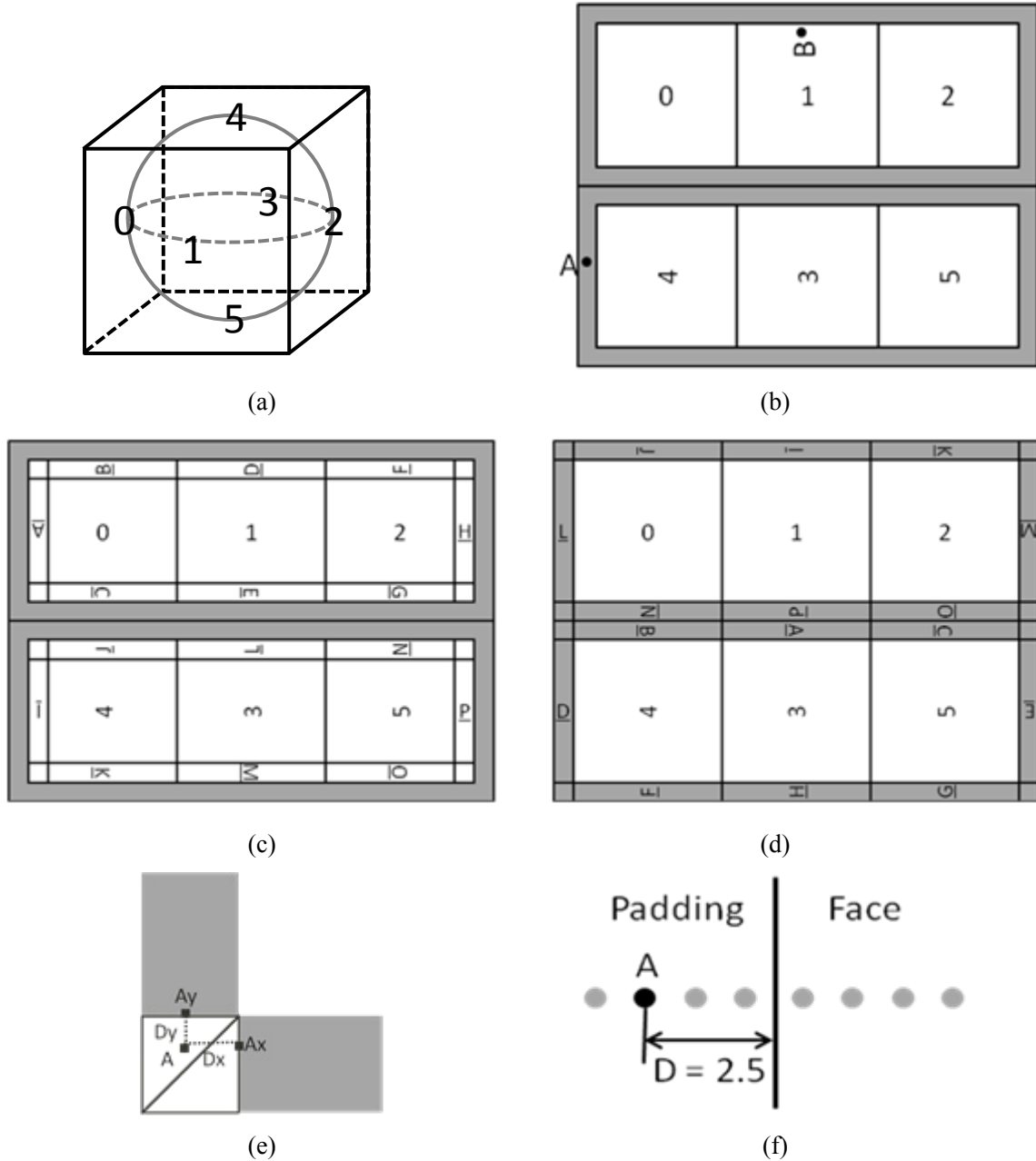


Figure 7—Near-equiangular cubic projection

A normal hexahedron is tangent to the spherical frame. The spherical coordinates of tangent points are $(-\pi \div 2, 0)$, $(0, 0)$, $(\pi \div 2, 0)$, $(\pi, 0)$, $(0, \pi \div 2)$ and $(0, -\pi \div 2)$, respectively, and their corresponding six faces of the normal hexahedron are numbered with 0, 1, 2, 3, 4, and 5. Then, some faces are rotated as follows: face 4 is rotated by 90° counterclockwise, face 3 is rotated by 90° clockwise, face 5 is rotated by 90° counterclockwise. The top 3×1 group is composed by face 0, face 1 and face 2, and the bottom 3×1 group is composed of face 3, face 4, and face 5. Finally, the two 3×1 groups are padded by P row/column pixels around the boundaries to get the near-equiangular hexahedron video frame, which is shown in Figure 7(b). Specifically, when the luminance component is padded, P is equal to 4; when the chrominance component is padded, P is equal to 2. In addition, the P row (column) pixels which are padded outward in Figure 7(d) are obtained by rotating