



DPC: 25/30509948 DC

BSI Group Headquarters

389 Chiswick High Road London W4 4AL

Tel: +44 (0)20 8996 9000

Fax: +44 (0)20 8996 7400

www.bsigroup.com

Date: 06 October 2025

Origin: International

Latest date for receipt of comments: 29 November 2025

Project No. 2025/00439

Responsible committee: IST/37 Coding of picture, audio, multimedia and hypermedia information

Interested committees:

Title: Draft BS ISO/IEC 21794-2 Information technology — Plenoptic image coding system (JPEG Pleno). Part 2: Light field coding.

Please notify admin.start@bsigroup.com if you are aware of any keywords that might assist in clarifying or identifying the standard or of the content of this standard:

- i) has any issues related to 3rd party IPR, patent or copyright
- ii) affects other national standard(s)
- iii) requires additional national guidance or information

WARNING: THIS IS A DRAFT AND MUST NOT BE REGARDED OR USED AS A BRITISH STANDARD. THIS DRAFT IS NOT CURRENT BEYOND 29 November 2025

This draft is issued to allow comments from interested parties; all comments will be given consideration prior to publication. No acknowledgement will normally be sent. **See overleaf for information on the submission of comments.**

No copying is allowed, in any form, without prior written permission from BSI except as permitted under the Copyright, Designs and Patent Act 1988 or for circulation within a nominating organization for briefing purposes. Electronic circulation is limited to dissemination by e-mail within such an organization by committee members.

Further copies of this draft may be purchased from BSI Shop <http://shop.bsigroup.com> or from BSI Customer Services, Tel: +44(0) 20 8996 9001 or email cservices@bsigroup.com. British, International and foreign standards are also available from BSI Customer Services.

Information on the co-operating organizations represented on the committees referenced above may be obtained from <https://standardsdevelopment.bsigroup.com/>.

E-mail: admin.start@bsigroup.com

Introduction

This draft standard is based on national and international discussions. Your comments on this draft are invited and will assist in the preparation of the consequent standard.

For international standards, comments will be reviewed by the relevant UK national committee before sending the consensus UK vote and comments to the international committee, which will then decide appropriate action. If the international standard is approved, it is usual for the text to be published as a British Standard.

For national standards, comments will be reviewed by the relevant UK national committee and the resulting standards published as a British Standard.

UK Vote

Please indicate whether you consider the UK should submit a negative (with supporting technical reasons) or positive vote on this draft. Please indicate if you are aware of any reason why this draft standard should not be published as a British Standard.

Submission of Comments

- **Annotated drafts are not acceptable and will be rejected.**
- All comments should be submitted online at <https://standardsdevelopment.bsigroup.com/>. You will need to register in order to comment.

Template for comments and secretariat observations

Date: xx/xx/20xx	Document: ISO/DIS xxxx
------------------	------------------------

1	2	(3)	4	5	(6)	7
MB	Clause No./ Subclause No./Annex (e.g. 3.1)	Paragraph/ Figure/Table/Note	Type of comment	Comment (justification for change) by the MB	Proposed change by the MB	Secretariat observations on each comment submitted
	3.1	Definition 1	ed	Definition is ambiguous and needs clarifying.	Amend to read '...so that the mains connector to which no connection...'	
	6.4	Paragraph 2	te	The use of the UV photometer as an alternative cannot be supported as serious problems have been encountered in its use in the UK.	Delete reference to UV photometer.	



DRAFT International Standard

ISO/IEC DIS 21794-2

Information technology — Plenoptic image coding system (JPEG Pleno) —

Part 2: Light field coding

*Technologies de l'information — Système de codage d'images
plénoptiques (JPEG Pleno) —*

Partie 2: Codages des champs de lumière

ICS: 35.040.30

ISO/IEC JTC 1/SC 29

Secretariat: JISC

Voting begins on:
2025-10-06

Voting terminates on:
2025-12-29

This document has not been edited by the ISO Central Secretariat.

THIS DOCUMENT IS A DRAFT CIRCULATED FOR COMMENTS AND APPROVAL. IT IS THEREFORE SUBJECT TO CHANGE AND MAY NOT BE REFERRED TO AS AN INTERNATIONAL STANDARD UNTIL PUBLISHED AS SUCH.

IN ADDITION TO THEIR EVALUATION AS BEING ACCEPTABLE FOR INDUSTRIAL, TECHNOLOGICAL, COMMERCIAL AND USER PURPOSES, DRAFT INTERNATIONAL STANDARDS MAY ON OCCASION HAVE TO BE CONSIDERED IN THE LIGHT OF THEIR POTENTIAL TO BECOME STANDARDS TO WHICH REFERENCE MAY BE MADE IN NATIONAL REGULATIONS.

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.

Reference number
ISO/IEC DIS 21794-2:2025(en)

Full document details: 25/30509948 DC

© ISO/IEC 2025  **Documents**
Document Sharing Hub



COPYRIGHT PROTECTED DOCUMENT

© 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
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword	iv
Introduction	v
1 Scope	1
2 Normative references	1
3 Terms and definitions	1
4 Symbols and abbreviated terms	3
4.1 Symbols	3
4.2 Abbreviated terms	6
5 Conventions	7
5.1 Naming conventions for numerical values	7
5.2 Operators	7
5.2.1 Arithmetic operators	7
5.2.2 Logical operators	8
5.2.3 Relational operators	8
5.2.4 Precedence order of operators	8
5.2.5 Mathematical functions	9
6 General	9
6.1 Functional overview on the decoding process	9
6.2 Encoder requirements	11
6.3 Decoder requirements	11
7 Organization of the document	11
Annex A (normative) JPEG Pleno Light Field superbox	12
Annex B (normative) 4D transform mode	29
Annex C (normative) JPEG Pleno light field reference view decoding	73
Annex D (normative) JPEG Pleno light field normalized disparity view decoding	81
Annex E (normative) JPEG Pleno Light Field Intermediate View superbox	88
Annex F (normative) Slanted 4D transform mode	114
Annex G (Normative) Profiles and Levels for JPEG Pleno Light Field Coding System	136
Bibliography	139

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 documents 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).

This 2nd edition of ISO/IEC 21794-2 ("Plenoptic image coding system (JPEG Pleno) Part 2: Light field coding") integrates AMD1 of ISO/IEC 21794-2 ("Profiles and levels for JPEG Pleno Light Field Coding") and includes the specification of an additional coding mode, entitled Slanted 4D transform mode and its associated profile.

ISO/IEC 21794-2 ("Plenoptic image coding system (JPEG Pleno) Part 2: Light field coding") specifies two coding modes: the 4D transform mode and the 4D prediction mode. The 4D prediction mode is efficient for light fields of all baselines but depends on the availability of accurate depth information. The 4D transform mode, although not relying on any sort of depth information, is only efficient for coding narrow baseline light fields. The Slanted 4D transform mode, based on 4D transformations, is efficient for light fields with both narrow and wide baselines and does not rely on the availability of depth information.

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 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*.

A list of all parts in the ISO/IEC 21794 series can be found on the ISO website.

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 is part of a series of standards for a system known as JPEG Pleno. This document defines the JPEG Pleno framework. It facilitates the capture, representation, exchange and visualization of plenoptic imaging modalities. A plenoptic image modality can be a light field, point cloud or hologram, which are sampled representations of the plenoptic function in the form of, respectively, a vector function that represents the radiance of a discretized set of light rays, a collection of points with position and attribute information, or a complex wavefront. The plenoptic function describes the radiance in time and in space obtained by positioning a pinhole camera at every viewpoint in 3D spatial coordinates, every viewing angle and every wavelength, resulting in a 7D function.

JPEG Pleno specifies tools for coding these modalities while providing advanced functionality at system level, such as support for data and metadata manipulation, editing, random access and interaction, protection of privacy and ownership rights.

Information technology — Plenoptic image coding system (JPEG Pleno) —

Part 2: Light field coding

1 Scope

This document specifies a coded codestream format for storage of light field modalities as well as associated metadata descriptors that are light field modality specific. This document also provides information on the encoding tools.

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.

ITU-T Rec. T.800, / ISO/IEC 15444-1, *Information technology — JPEG 2000 image coding system — Part 1: Core coding system*

ITU-T Rec. T.801, / ISO/IEC 15444-2, *Information technology — JPEG 2000 image coding system — Part 2: Extensions*

ISO/IEC 21794-1:2020, *Information technology — Plenoptic image coding system (JPEG Pleno) — Part 1: Framework*

ISO/IEC 60559, *Information technology — Microprocessor Systems — Floating-Point arithmetic*

ITU-T T.81, / ISO/IEC 10918-1, *Information technology – Digital compression and coding of continuous-tone still images – Requirements and guidelines*

ITU-T Rec. T.84, / ISO/IEC 10918-3, *Information technology – Digital compression and coding of continuous-tone still images: Extensions*

3 Terms and definitions

For the purposes of this document the terms and definitions given in ISO/IEC 21794-1 and the following apply.

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

- ISO Online browsing platform: available at <https://www.iso.org/obp>
- IEC Electropedia: available at <http://www.electropedia.org/>

3.1 arithmetic coder

entropy coder that converts variable length strings to variable length codes (encoding) and vice versa (decoding)

3.2

bit-plane

two-dimensional array of bits

3.3

4D bit-plane

four-dimensional array of bits

3.4

coefficient

numerical value that is the result of a transformation or linear regression

3.5

compression

reduction in the number of bits used to represent source image data

3.6

depth

distance of a point in 3D space to the camera plane

3.7

disparity view

image that for each pixel of the subaperture view contains the apparent pixel shift between two subaperture views along either horizontal or vertical axis

3.8

hexadeca-tree

division of a 4D region into 16 (sixteen) 4D subregions

3.9

pixel

collection of sample values in the spatial image domain having all the same sample coordinates

EXAMPLE A pixel may consist of three samples describing its red, green and blue value.

3.10

plenoptic function

amount of radiance in time and in space by positioning a pinhole camera at every viewpoint in 3D spatial coordinates, every viewing angle and every wavelength, resulting in a 7D representation

3.11

reference view

subaperture view that is used as one of the references to generate the intermediate views

3.12

subaperture view

subaperture image

image taken of the 3D scene by a pinhole camera positioned at a particular viewpoint and viewing angle

3.13

texture

pixel attributes

EXAMPLE Colour information, opacity, etc.

3.14

transform

transformation

mathematical mapping from one signal space to another

4 Symbols and abbreviated terms

4.1 Symbols

<code>Codestream_Body()</code>	coded image data in the codestream without <code>Codestream_Header()</code>
<code>Codestream_Header()</code>	codestream header preceding the image data in the codestream
$\tilde{D}^{DEC}(t,s,v,u)$	decoded normalized disparity value at view (t,s) for pixel location (v,u)
$\tilde{D}(t,s,v,u)$	normalized disparity value at view (t,s) for pixel location (v,u)
$DPEC_k$	pointer to contiguous codestream for normalized disparity view k
D_{shift}	scaling parameter to translate quantized normalized disparity maps to positive range
DCODEC	disparity view codec type
f	focal length
FPW_p	fixed-weight merging parameter for view p
$H(t,s)$	view hierarchy value for view (t,s)
$HCC(t,s)$	horizontal camera centre coordinate for view (t,s)
$H_D(t,s)$	binary value defining the availability of a normalized disparity view (t,s)
J_0	Lagrangian encoding cost
J_1	Lagrangian encoding cost of spatial partitioning
J_2	Lagrangian encoding cost of view partitioning
$KR_{p,c}$	sparse filter regressor mask of texture component c for view p
<code>LightField()</code>	JPEG Pleno light field codestream
$LSW_j^{p,c}$	quantized least-squares merging weight of texture component c for view p , $j = 1, 2, \dots, NLS_p$
MIDV	absolute value of the minimum value over all quantized normalized disparity views
$MMODE_p$	view merging mode for intermediate view p
MSP_p	sparse filter order for view p
NLS_p	number of least-squares merging coefficients for intermediate view p
NRT_p	regressor template size parameter for sparse filter for view p
NC	number of components in an image
N_I	number of intermediate views
N_{NDV}	number of reference normalized disparity views
N_p^D	number of normalized disparity reference views for intermediate view p
N_p^T	number of texture reference views for intermediate view p

N_{REF}	number of reference views
N_{RES}	number of prediction residual views
N_{sp}	total available number of regressors for sparse filter
Plev	level a particular codestream complies to
Ppih	profile a particular codestream complies to
Q_p	2D image of dimensions $V \times U$, defines the occlusion state-based segmentation at Intermediate view p
Q	normalized disparity quantization parameter
R	rate or bitrate, expressed in bit per sample
RCODEC	prediction residual view codec type
RDATA	array of bytes containing for a single prediction residual view the RCODEC codestream after header information has been stripped
RENCODING	array of bytes containing for a single prediction residual view the full DCODEC codestream
RGB	colour data for the red, green and blue colour component of a pixel
RHEADER	array of bytes containing for a single prediction residual view the header information from the RCODEC codestream
$RPEC_j$	pointer to contiguous codestream for prediction residual view j
s	coordinate of the addressed subaperture image along the s -axis
S	size of the light field image along the s -axis (COLUMNS)
s_{ii}^{Tr}	subscript of the column index of the reference view, $ii = 1, 2, \dots, N_p^T$ in the light field array in row-wise scanning order
s_{jj}^{Dr}	subscript of the column index of the reference normalized disparity view, $jj = 1, 2, \dots, N_p^D$ in the light field array in row-wise scanning order
SF_p	binary variable, determines if sparse filter is used (true) or not (false)
$SPW_j^{p,0}$	quantized sparse filter coefficients of texture component c for view p , $j = 1, 2, \dots, MSP_p$
$\widehat{SPW}_j^{p,c}$	de-quantized sparse filter coefficients of texture component c for view p , $j = 1, 2, \dots, MSP_p$
t	coordinate of the addressed subaperture image along the t -axis
T	size of the light field image along the t -axis (ROWS)
t_{ii}^{Tr}	subscript of the row index of the reference view, $ii = 1, 2, \dots, N_p^T$ in the light field array in row-wise scanning order
t_{jj}^{Dr}	subscript of the row index of the reference normalized disparity view, $jj = 1, 2, \dots, N_p^D$ in the light field array in row-wise scanning order

(t_k^D, s_k^D)	view coordinate subscripts for normalized disparity view k
(t_l^X, s_l^X)	view coordinate subscripts for reference view l
(t_p^I, s_p^I)	view coordinate subscripts for intermediate view p
$t_k \times s_k \times v_k \times u_k$	4D block dimensions at the 4D block partitioning stage
$t_b \times s_b \times v_b \times u_b$	4D block dimensions at the bit-plane hexadeca-tree decomposition stage
TCODEC	reference view codec type
TDATA	array of bytes, containing for a single reference view, the TCODEC codestream, after header information has been stripped
TENCODING	array of bytes, containing for a single reference view the full TCODEC codestream
THEADER	array of bytes, containing for a single reference view the header information from the TCODEC codestream
$TPEC_l$	pointer to contiguous codestream for reference view l
u	sample coordinate along the u -axis within the addressed subaperture image
U	size of the subaperture image along the u -axis (WIDTH)
v	sample coordinate along the v -axis within the addressed subaperture image
V	size of the subaperture image along the v -axis (HEIGHT)
$VCC(t, s)$	vertical camera centre coordinate for view (t, s)
VPP_p	view prediction parameters for intermediate view p
$X(t, s, v, u, c)$	texture value at view (t, s) for pixel location (v, u) for texture component c
$X^{DEC}(t, s, v, u, c)$	decoded texture value at view (t, s) for pixel location (v, u) for texture component c
$X_W^{(t_1, s_1)}(t_2, s_2)$	result of warping the texture view (t_1, s_1) to view location (t_2, s_2)
Δx	horizontal distance between a pair of camera centres
Δy	vertical distance between a pair of camera centres
YCbCr	colour data for the luminance, the blue chrominance and the red chrominance component of a pixel
$z(t, s, v, u)$	depth value at view (t, s) for pixel location (v, u)
$\hat{\theta}_i^p$	distance based merging weight for reference view $i = 1, \dots, N_p^T$ at intermediate view p
α_i^p	distance based factor, used for defining the merging weight, at intermediate view p for reference view $i = 1, \dots, N_p^T$

Γ_p	binary matrix, defining the locations of the non-zero merging weights in merging weight matrix $\Theta_{p,c}$ at intermediate view p . It is identical between all colour components c
$\theta_j^{p,c}$	de-quantized least-squares merging weight of texture component c for view p , $j = 1, 2, \dots, NLS_p$
$\theta_{p,c}^{sp}$	sparse filter coefficients at intermediate view p for colour component c
$\Theta_{p,c}$	merging weight matrix for intermediate view p for colour component c
$Y_{p,c}$	locations of the non-zero elements of $\Psi_{(v,u)}$
$\Psi_{(v,u)}$	regressor template at pixel location (v,u)
Ω_p^{Dr}	set of reference normalized disparity views for intermediate view p
Ω_p^{occlD}	set of occluded pixels, which remain to be inpainted, during normalized disparity view synthesis at intermediate view p
Ω_p^{occlT}	set of occluded pixels, which remain to be inpainted, during texture view synthesis at intermediate view p
Ω_p^{Tr}	set of reference views for intermediate view p

4.2 Abbreviated terms

2D	two dimensional
3D	three dimensional
4D	four dimensional
DCT	discrete cosine transform
EPI	Epipolar Plane Image
floating point	floating point notation as specified in ISO/IEC 60559
HTTP	hypertext transfer protocol
IDCT	inverse DCT
IPR	intellectual property rights
IV	Intermediate view; subaperture view that is generated from surrounding reference view(s)
JPEG	Joint Photographic Experts Group
JPL	JPEG Pleno file format
LSB	least significant bit
MSB	most significant bit
R-D	rate-distortion
RV	reference view

URL	uniform resource locator
XML	eXtensible Markup Language

5 Conventions

5.1 Naming conventions for numerical values

Integer numbers are expressed as bit patterns, hexadecimal values or decimal numbers. Bit patterns and hexadecimal values have both a numerical value and an associated particular length in bits.

Hexadecimal notation, indicated by prefixing the hexadecimal number by "0x", may be used instead of binary notation to denote a bit pattern having a length that is an integer multiple of 4. For example, 0x41 represents an eight-bit pattern having only its second most significant bit and its least significant bit equal to 1. Numerical values that are specified under a "**Code**" heading in tables that are referred to as "code tables" are bit pattern values (specified as a string of digits equal to 0 or 1 in which the left-most bit is considered the most-significant bit). Other numerical values not prefixed by "0x" are decimal values. When used in expressions, a hexadecimal value is interpreted as having a value equal to the value of the corresponding bit pattern evaluated as a binary representation of an unsigned integer (i.e. as the value of the number formed by prefixing the bit pattern with a sign bit equal to 0 and interpreting the result as a two's complement representation of an integer value). For example, the hexadecimal value 0xF is equivalent to the 4-bit pattern '1111' and is interpreted in expressions as being equal to the decimal number 15.

5.2 Operators

NOTE Many of the operators used in document are similar to those used in the C programming language.

5.2.1 Arithmetic operators

+	addition
–	subtraction (as a binary operator) or negation (as a unary prefix operator)
×	multiplication
/	division without truncation or rounding
<<	left shift; $x \ll s$ is defined as $x \times 2^s$
>>	right shift; $x \gg s$ is defined as $\lfloor x/2^s \rfloor$
++	increment with 1
--	decrement with 1
umod	$x \text{ umod } a$ is the unique value y between 0 and $a-1$ for which $y + Na = x$ with a suitable integer N
&	bitwise AND operator; compares each bit of the first operand to the corresponding bit of the second operand If both bits are 1, the corresponding result bit is set to 1. Otherwise, the corresponding result bit is set to 0.
^	bitwise XOR operator; compares each bit of the first operand to the corresponding bit of the second operand If both bits are equal, the corresponding result bit is set to 0. Otherwise, the corresponding result bit is set to 1.

5.2.2 Logical operators

	logical OR
&&	logical AND
!	logical NOT

5.2.3 Relational operators

>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to
==	equal to
!=	not equal to

5.2.4 Precedence order of operators

Operators are listed in descending order of precedence. If several operators appear in the same line, they have equal precedence. When several operators of equal precedence appear at the same level in an expression, evaluation proceeds according to the associativity of the operator either from right to left or from left to right.

Operators	Type of operation	Associativity
()	expression	left to right
[]	indexing of arrays	left to right
++, --	increment, decrement	left to right
!, -	logical not, unary negation	
×, /	multiplication, division	left to right
umod	modulo (remainder)	left to right
+, -	addition and subtraction	left to right
&	bitwise AND	left to right
^	bitwise XOR	left to right
&&	logical AND	left to right
	logical OR	left to right
<<, >>	left shift and right shift	left to right
<, >, <=, >=	relational	left to right

5.2.5 Mathematical functions

$ x $	absolute value, is $-x$ for $x < 0$, otherwise x
$\text{sign}(x)$	sign of x , zero if x is zero, +1 if x is positive, -1 if x is negative
$\text{clamp}(x, \text{min}, \text{max})$	clamps x to the range $[\text{min}, \text{max}]$: returns min if $x < \text{min}$, max if $x > \text{max}$ or otherwise x
$\lceil x \rceil$	ceiling of x ; returns the smallest integer that is greater than or equal to x
$\lfloor x \rfloor$	floor of x ; returns the largest integer that is less than or equal to x
$\text{round}(x)$	rounding of x to the nearest integer, equivalent to $\text{sign}(x) \lfloor x + 0.5 \rfloor$

6 General

6.1 Functional overview on the decoding process

This document specifies the JPEG Pleno Light Field superbox and the JPEG Pleno light field decoding algorithm. The generic JPEG Pleno Light Field superbox syntax is specified in [Annex A](#).

The specified light field decoding algorithm distinguishes three coding modes:

- **4D transform mode:** this mode is specified in [Annex B](#) and is based on a 4D inverse discrete cosine transform (IDCT), 4D block partitioning, and 4D bit-plane hexadeca-tree decoding.
- **4D prediction mode:** this mode is based the prediction of intermediate views based on reference views and normalized disparity maps. The signalling syntax and decoding of the reference views is addressed in [Annex C](#), the normalized disparity views in [Annex D](#), and the prediction parameters and residual views in [Annex E](#). The intermediate views are reconstructed in a decoding process that involves view warping, view merging and prediction error correction.
- **Slanted 4D transform mode:** this mode is specified in [Annex F](#) and is based on changing the EPI slants/slopes in the 4D blocks by applying a geometric transformation before the conventional 4D-DCT stage of the 4D Transform mode architecture, notably to make the slopes of the lines composing the EPIs as aligned with one of the separable 4D-DCT dimensions as possible.

The overall architecture of the three coding modes ([Figure 1](#)) provides the flexibility to configure the encoding and decoding system depending on the requirements of the addressed use case.

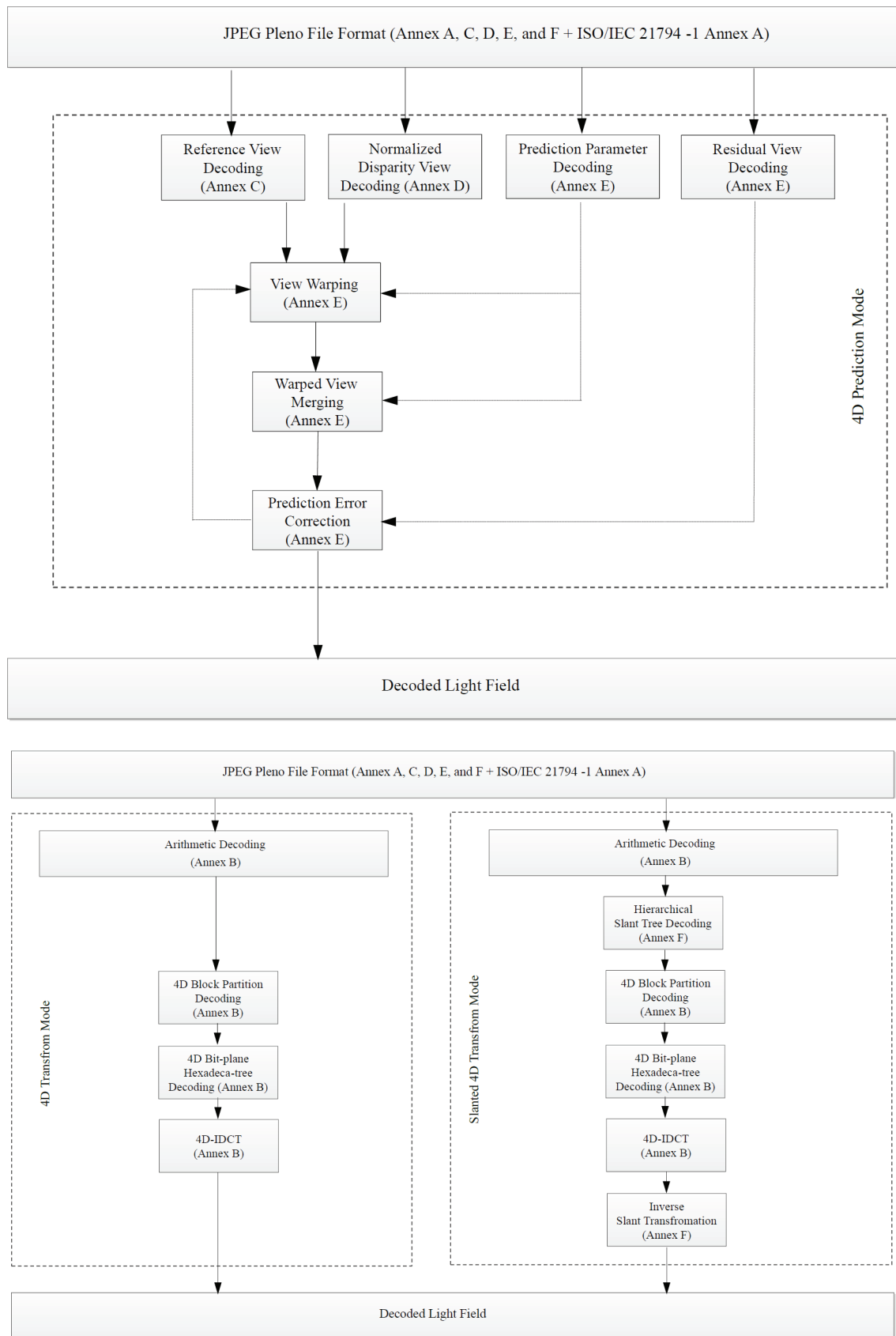


Figure 1 — Generic JPEG Pleno light field decoder architecture.

6.2 Encoder requirements

An encoding process converts source light field data to coded light field data.

In order to conform with this document, an encoder shall conform with the codestream format syntax and file format syntax specified in the annexes for the encoding process(es) embodied by the encoder.

6.3 Decoder requirements

A decoding process converts coded light field data to reconstructed light field data. [Annexes A](#) through [F](#) describe and specify the decoding process.

A decoder is an embodiment of the decoding process. In order to conform to this document, a decoder shall convert all, or specific parts of, any coded light field data that conform to the file format syntax and codestream syntax specified in [Annex A](#) to [F](#) to a reconstructed light field.

7 Organization of the document

[Annex A](#) specifies the description of the JPEG Pleno Light Field superbox.

This document specifies three approaches to represent a compressed representation of light field data: the 4D Transform mode is specified in [Annex B](#), the 4D Prediction mode is specified in [Annex C](#), [Annex D](#) and [Annex E](#). [Annex C](#) details the signalling of the reference view data, [Annex D](#) the signalling of the normalized disparity views and, [Annex E](#) the signalling of the prediction parameters to generate the intermediate views and residual view data to compensate for prediction errors, and [Annex F](#) specifies the Slanted 4D Transform mode. [Annex G](#) defines the profiles and levels of the three coding modes.

Annex A (normative)

JPEG Pleno Light Field superbox

A.1 General

This annex specifies the use of the JPEG Pleno Light Field superbox which is designed to contain compressed light field data and associated metadata. The listed boxes shall comply with their definitions as specified in ISO/IEC 21794-1.

This document may redefine the binary structure of some boxes defined as part of the ISO/IEC 15444-1 or ISO/IEC 15444-2 file formats. For those boxes, the definition found in this document shall be used for all JPL files.

A.2 Organization of the JPEG Pleno Light Field superbox

[Figure A.1](#) shows the hierarchical organization of the JPEG Pleno Light Field superbox contained by a JPL file. This illustration does not specify nor imply a specific order to these boxes. In many cases, the file will contain several boxes of a particular box type. The meaning of each of those boxes is dependent on the placement and order of that particular box within the file.

This superbox is composed out of the following core elements:

- a JPEG Pleno Light Field Header box containing parameterization information about the light field such as size and colour parameters;
- a JPEG Pleno Light Field Reference View box containing the compressed reference views of the light field;
- a JPEG Pleno Light Field Disparity View box signalling disparity information for all or a subset of subaperture views;
- a JPEG Pleno Light Field Intermediate View box containing prediction parameters and eventual compressed residual signals for subaperture views not encoded as reference views.

[Table A.1](#) lists all boxes defined as part of this document. Boxes defined as part of the ISO/IEC 15444-1 or ISO/IEC 15444-2 file formats are not listed. A box that is listed in [Table A.1](#) as “Required” shall exist within all conforming JPL files. For the placement of and restrictions on each box, see the relevant section defining that box.

Note that the IPR, XML, and UUID boxes defined in [Annex A](#) can be signalled, as well at the level of the JPEG Pleno Light Field box, to carry light field specific metadata.

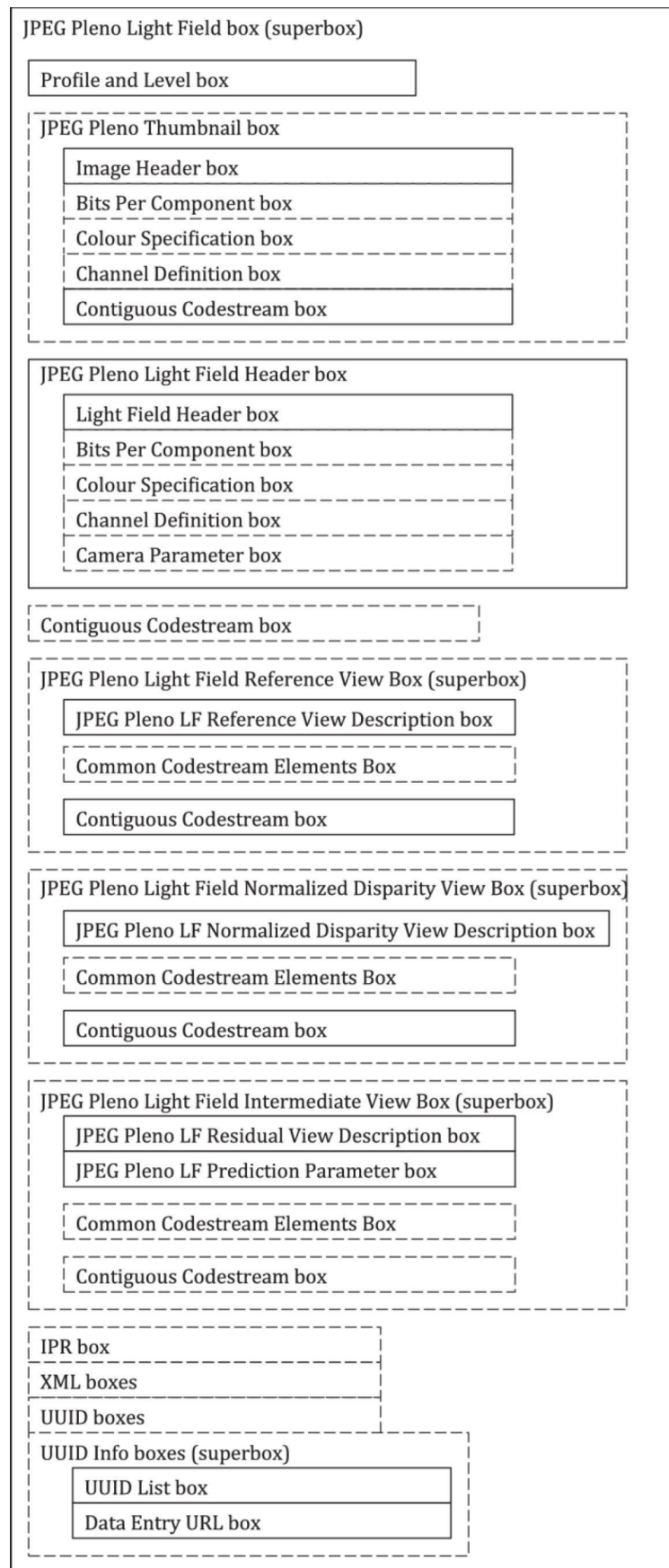


Figure A.1 — Hierarchical organization of a JPEG Pleno Light Field superbox

A.3 Defined boxes

A.3.1 Overview

The following boxes should be interpreted properly by all conforming readers. Each of these boxes conforms to the standard box structure as defined in ISO/IEC 21794-1:2020, Annex A. The following clauses define the value of the DBox field. It is assumed that the LBox, TBox and XLBox fields exist for each box in the file as defined in ISO/IEC 21794-1:2020, Annex A.

Table A.1 — Defined boxes

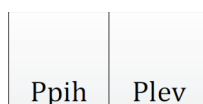
Box name	Type	Superbox	Required?	Comments
JPEG Pleno Light Field box	'jplf' (0x6A70 6C66)	Yes	Yes	This box contains a series of boxes that contain the encoded light field, its parameterization and associated metadata. (Defined in ISO/IEC 21794-1:2020, Annex A)
JPEG Pleno Profile and Level box	'jppl' (0x6A70 706C)	No	Yes	This box indicates to which profile and associated level the file format and codestream complies. (Defined in Annex A.3.2)
JPEG Pleno Light Field Header box	'jplh' (0x6A70 6C68)	Yes	Yes	This box contains generic information about the file, such as the number of components, bits per component and colour space. (Defined in Annex A.3.3)
Light Field Header box	'lhdr' (0x6C68 6472)	No	Yes	This box contains fixed length generic information about the light field, such as light field dimensions, subaperture image size, number of components, codec and bits per component. (Defined in Annex A.3.3.2)
Camera Parameter box	'lfcv' (0x6C66 6370)	No	No	This box signals intrinsic and extrinsic camera parameters for calibration of the light field data. (Defined in Annex A.3.3.3)
Contiguous Codestream box	'jp2c' (0x6A70 3263)	No	No	This box contains a JPEG Pleno codestream (Defined in Annex A.3.4)
JPEG Pleno Light Field Reference View superbox	'lfrv' (0x6C66 7276)	Yes	No	This box contains a series of boxes that contain the encoded reference views and their associated parameters. (Defined in Annex C.2)
JPEG Pleno Light Field Reference View Description box	'lfrd' (0x6C66 7264)	No	No	This box signals which views are encoded as reference views and their encoding configuration. (Defined in Annex C.3.1)
Common Codestream Elements box	'lfcc' (0x6C66 6363)	No	No	This box contains the redundant part of the signalled codestreams. (Defined in Annex C.3.2)
JPEG Pleno Light Field Normalized Disparity View superbox	'lfdv' (0x6C66 6476)	Yes	No	This box contains a series of boxes that contain the encoded normalized disparity views and their associated parameters. (Defined in Annex D.2)
JPEG Pleno Light Field Normalized Disparity View Description box	'lfdd' (0x6C66 6464)	No	No	This box signals for which views normalized disparity information is signalled and their encoding configuration. (Defined in Annex D.3.1)

Table A.1 (continued)

Box name	Type	Superbox	Required?	Comments
JPEG Pleno Light Field Intermediate View superbox	'lfiv' (0x6C66 6976)	Yes	No	This box contains a series of boxes that contain both the prediction parameters for the intermediate views and the encoded residual views. (Defined in Annex E.2)
JPEG Pleno Light Field Prediction Parameter box	'lfpp' (0x6C66 7070)	No	No	This box signals prediction parameter information for the intermediate views. (Defined in Annex E.3.1)
JPEG Pleno Light Field Residual View Description box	'lfre' (0x6C66 7265)	No	No	This box signals the encoding configuration for the residual views containing the prediction errors. (Defined in Annex E.3.2)

A.3.2 JPEG Pleno Profile and Level box

Profile and levels are defined in [Annex G](#). The type of the JPEG Pleno Profile and Level box shall be 'jppl' (0x6A70686F). The contents of the box shall have the organization as in [Figure A.2](#), and its format shall be as in [Table A.2](#).



Key

Ppih profile of the codestream (as defined in [Annex G](#))

Plev level of the codestream (as defined in [Annex G](#))

Figure A.2 — Organization of the contents of a JPEG Pleno Profile and Level box**Table A.2 — Format of the contents of the JPEG Pleno Profile and Level box**

Field name	Size (bits)	Value
Ppih	16	Variable, defined in Annex G
Plev	16	Variable, defined in Annex G

A.3.3 JPEG Pleno Light Field Header box

A.3.3.1 General

The JPEG Pleno Header box contains generic information about the file, such as the number of components, bits per component and colour space. This box is a superbox. Within a JPL file, there shall be one and only one JPEG Pleno Header box. The JPEG Pleno Header box shall be located anywhere after the File Type box and before the Contiguous Codestream box. It also shall be at the same level as the JPEG Pleno Signature and File Type boxes. It shall not be inside any other superbox within the file.

The type of the JPEG Pleno Header box shall be 'jplh' (0x6A706C68).

This box contains several boxes. Other boxes may be defined in other documents and may be ignored by conforming readers. The boxes contained within the JPEG Pleno Header box that are defined within this document are shown in [Figure A.3](#):

- The Light Field Header box specifies information about the reference grid geometry, bit depth and the number of components. This box shall be the first box in the JPEG Pleno Header box and is specified in [A.3.3.2](#).

- The Bits Per Component box specifies the bit depth of the components in the file in cases where the bit depth is not constant across all components. Its structure shall be as specified in ISO/IEC 15444-1.
- The Colour Specification boxes specify the colour space of the decompressed image. Their structures shall be as specified in ISO/IEC 15444-2. There shall be at least one Colour Specification box within the JPEG Pleno Header box. The use of multiple Colour Specification boxes provides the ability for a decoder to be given multiple optimization or compatibility options for colour processing. These boxes shall be positioned anywhere in the JPEG Pleno Header box provided that they come after the Light Field Header box. All Colour Specification boxes shall be contiguous within the JPEG Pleno Header box.
- The Channel Definition box defines the channels in the image. Its structure shall be as specified in ISO/IEC 15444-1. This box shall be positioned anywhere in the JPEG Pleno Header box, provided that it comes after the Light Field Header box.



Key

lhdr	Light Field Header box
bpcc	Bits Per Component box
colr ⁱ	Colour Specification boxes
cdef	Channel Definition box

Figure A.3 — Organization of the contents of a JPEG Pleno Header box

A.3.3.2 Light Field Header box

A.3.3.2.1 General

This box contains fixed length generic information about the light field, such as light field dimensions, subaperture image size, number of components, codec and bits per component. The contents of the JPEG Pleno Header box shall start with a Light Field Header box. Instances of this box in other places in the file shall be ignored. The length of the Light Field Header box shall be 30 bytes, including the box length and type fields. Much of the information within the Light Field Header box is redundant with information stored in the codestream itself.

All references to "the codestream" in the descriptions of fields in this Light Field Header box apply to the codestream found in the first Contiguous Codestream box in the file. Files that contain contradictory information between the Light Field Header box and the first codestream are not conforming files. However, readers may choose to attempt to read these files by using the values found within the codestream.

The type of the Light Field Header box shall be 'lhdr' (0x6C68 6472) and the contents of the box shall have the format as in [Figure A.4](#) and [Table A.3](#):

- **ROWS (*T*):** The value of this parameter indicates the number of rows of the subaperture view array. This field is stored as a 4-byte big-endian unsigned integer.
- **COLUMNS (*S*):** The value of this parameter indicates the number of columns of the subaperture view array. This field is stored as a 4-byte big-endian unsigned integer.
- **HEIGHT (*V*):** The value of this parameter indicates the height of the sample grid. This field is stored as a 4-byte big-endian unsigned integer.
- **WIDTH (*U*):** The value of this parameter indicates the width of the sample grid. This field is stored as a 4-byte big-endian unsigned integer.
- **NC:** This parameter specifies the number of components in the codestream and is stored as a 2-byte big-endian unsigned integer. The value of this field shall be equal to the value of the NC field in the LFC

marker in the codestream (as defined in [B.3.2.6.3](#)). If no Channel Definition Box is available, the order of the components for colour images is R-G-B-Aux or Y-U-V-Aux.

- **BPC:** This parameter specifies the bit depth of the components in the codestream, minus 1, and is stored as a 1-byte field ([Table A.4](#)).

The low 7-bits of the value indicate the bit depth of the components. The MSB indicates whether the components are signed or unsigned. If the MSB is 1, then the components contain signed values. If the MSB is 0, then the components contain unsigned values. If the components vary in bit depth or sign, or both, then the value of this field shall be 255 and the Light Field Header box shall also contain a Bits Per Component box defining the bit depth of each component (as defined in [A.3.3.2.2](#)).

- **C:** This parameter specifies the compression algorithm used to compress the image data. It is encoded as a 1-byte unsigned integer. If the value is 0, the 4D transform mode coding is activated. If the value is 1, the 4D prediction mode is activated. If the value is 2, the Slanted 4D transform mode is activated. All other values are reserved for ISO/IEC use.
- **UnkC:** This field specifies if the actual colour space of the image data in the codestream is known. This field is encoded as a 1-byte unsigned integer. Legal values for this field are 0, if the colour space of the image is known and correctly specified in the Colour Space Specification boxes within the file, or 1 if the colour space of the light field is not known. A value of 1 will be used in cases such as the transcoding of legacy images where the actual colour space of the image data is not known. In these cases, while the colour space interpretation methods specified in the file may not accurately reproduce the image with respect to an original, the image should be treated as if the methods do accurately reproduce the image. Values other than 0 and 1 are reserved for ISO/IEC use.
- **IPR:** This parameter indicates whether this JPL file contains intellectual property rights information. If the value of this field is 0, this file does not contain rights information, and thus the file does not contain an IPR box. If the value is 1, then the file does contain rights information and thus does contain an IPR box as defined in ISO/IEC 15444-1. Other values are reserved for ISO/IEC use.

ROWS	COLUMNS	HEIGHT	WIDTH	NC	BPC	C	UnkC	IPR
------	---------	--------	-------	----	-----	---	------	-----

Key

ROWS (<i>T</i>)	number of rows of the subaperture view array
COLUMNS (<i>S</i>)	number of columns of the subaperture view array
HEIGHT (<i>V</i>)	subaperture view height
WIDTH (<i>U</i>)	subaperture view width
NC	number of components
BPC	bits per component
C	compression type
UnkC	colour space unknown
IPR	intellectual property

Figure A.4 — Organization of the contents of a Light Field Header box

Table A.3 — Format of the contents of the Light Field Header box

Field name	Size (bits)	Value
ROWS	32	1 to $(2^{32} - 1)$
COLUMNS	32	1 to $(2^{32} - 1)$
HEIGHT	32	1 to $(2^{32} - 1)$
WIDTH	32	1 to $(2^{32} - 1)$
NC	16	1 to 16 384
C	8	0 to 2
BPC	8	See Table A.4
UnkC	8	0 to 1
IPR	8	0 to 1

Table A.4 — BPC values

Values (bits) MSB LSB	Component sample precision
x000 0000 to x010 0101	Component bit depth = value + 1. From 1 bit deep to 38 bits deep respectively (counting the sign bit, if appropriate).
0xxx xxxx	Components are unsigned values.
1xxx xxxx	Components are signed values.
1111 1111	Components vary in bit depth.
	All other values reserved for ISO/IEC use.

A.3.3.2.2 Bits Per Component box

The Bits Per Component box specifies the bit depth of each component. If the bit depth of all components in the codestream is the same (in both sign and precision), then this box shall not be present. Otherwise, this box specifies the bit depth of each individual component. The order of bit depth values in this box is the actual order in which those components are enumerated within the codestream. The exact location of this box within the JPEG Pleno Header box may vary provided that it follows the Light Field Header box.

There shall be one and only one Bits Per Component box inside a JPEG Pleno Header box.

The type of the Bits Per Component box shall be 'bpcc' (0x6270 6363). The contents of this box shall be as in [Table A.5](#) and [Figure A.5](#).

**Key**

BPCⁱ bits per component

Figure A.5 — Organization of the contents of a Bits Per Component box**Table A.5 — Format of the contents of the Bits Per Component box**

Field name	Size (bits)	Value
BPC ⁱ	8	See Table A.6

This parameter specifies the bit depth of component *i*, minus 1, encoded as a 1-byte value ([Table A.6](#)). The ordering of the components within the Bits Per Component box shall be the same as the ordering of the components within the codestream. The number of BPCⁱ fields shall be the same as the value of the NC field

from the Light Field Header box. The value of this field shall be equivalent to the respective Ss_{iz}^i field in the LFC marker in the codestream. The low 7-bits of the value indicate the bit depth of this component. The MSB indicates whether the component is signed or unsigned. If the MSB is 1, then the component contains signed values. If the MSB is 0, then the component contains unsigned values.

Table A.6 — BPCⁱ values

Values (bits)		Component sample precision
MSB	LSB	
x000 0000 to x010 0101		Component bit depth = value + 1. From 1 bit deep to 38 bits deep respectively (counting the sign bit, if appropriate).
0xxx xxxx		Components are unsigned values.
1xxx xxxx		Components are signed values.
		All other values reserved for ISO/IEC use.

A.3.3.3 Camera Parameter box

A.3.3.3.1 General

The Camera Parameter box provides information on the positioning of the local reference grid in the global reference grid, its size, and calibration information about the light field. This box is optional.

Camera models can be represented by matrices with particular properties that represent the mapping of the 3D world coordinate system to the image coordinate system. This 3D to 2D transform depends on a number of parameters, known as the intrinsic and extrinsic parameters.

A.3.3.3.2 JPEG Pleno camera parameters

As specified in ISO/IEC 21794-1, JPEG Pleno provides a mechanism to co-register plenoptic data contained by the JPL file on a 3D reference grid system. This reference grid system exists out of a global and a local reference grid. The global reference grid allows the positioning of the individual modalities in the represented 3D scene. In addition, each JPEG Pleno Light Field, Point Cloud and Hologram box shall be assigned a local reference grid to address their sampled plenoptic data. This local reference grid is specified by signalling its translation and rotation with respect to the global reference grid. The rotation angles shall be determined utilizing the right-hand rule for curve orientation.

The parameters related to the local reference grid are signalled per plenoptic object in the associated JPEG Pleno Light Field, Point Cloud or Hologram box.

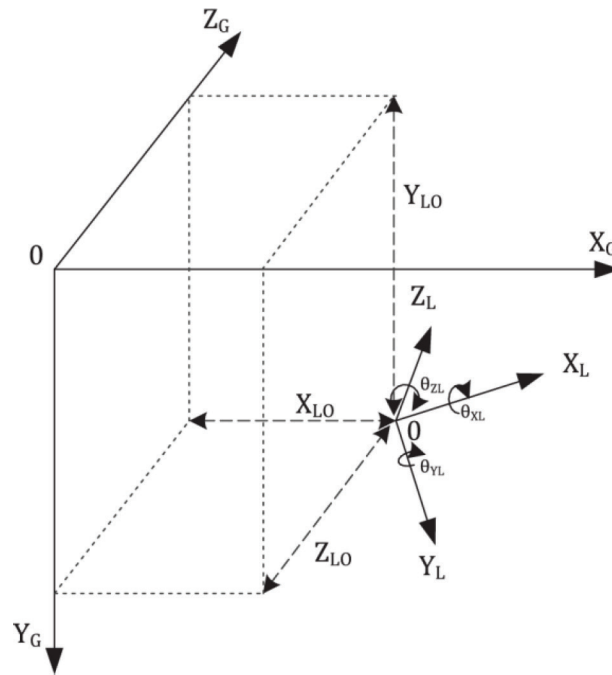


Figure A.6 — The global and local reference grid

In [Figure A.6](#), boundaries and coordinate axes of the global and one local reference grid are shown. In each case, the samples or coefficients coincident with the left and upper boundaries are included in a given bounding box, while samples or coefficients along the right and/or lower boundaries are not included in that bounding box.

A.3.3.3.3 Camera modelling and calibration

A.3.3.3.3.1 General

The camera modelling and calibration is described based on the local reference grid (X_L, Y_L, Z_L) in [Figure A.6](#). Both intrinsic and extrinsic parameters are being signalled to model and calibrate the camera setting and behaviour. The intrinsic parameters are the camera parameters that are internal and fixed to a particular camera/digitization setup, allowing the mapping between camera coordinates and pixel coordinates in the image plane^[1]. The extrinsic parameters are the camera parameters that are external to the camera and may change with respect to the 3D local reference grid, defining the location and orientation of the camera with respect to the 3D local reference grid coordinate system.

A.3.3.3.3.2 Intrinsic camera parameters

Considering a pinhole camera model, the centre of projection is the 'optical centre' (**C** in [Figure A.7](#)). The camera's 'principal axis' (Z_{CAM} in [Figure A.7](#)) is the line perpendicular to the image plane that passes through the pinhole. Its intersection with the image plane is known as the 'principal point' (**p** in [Figure A.7](#)) and it is the geometrical centre of the image. The parameters u_0 and v_0 are the principal points offsets, which are the coordinates of the principal point relative to the coordinate axes u and v ([Figure A.7](#)).

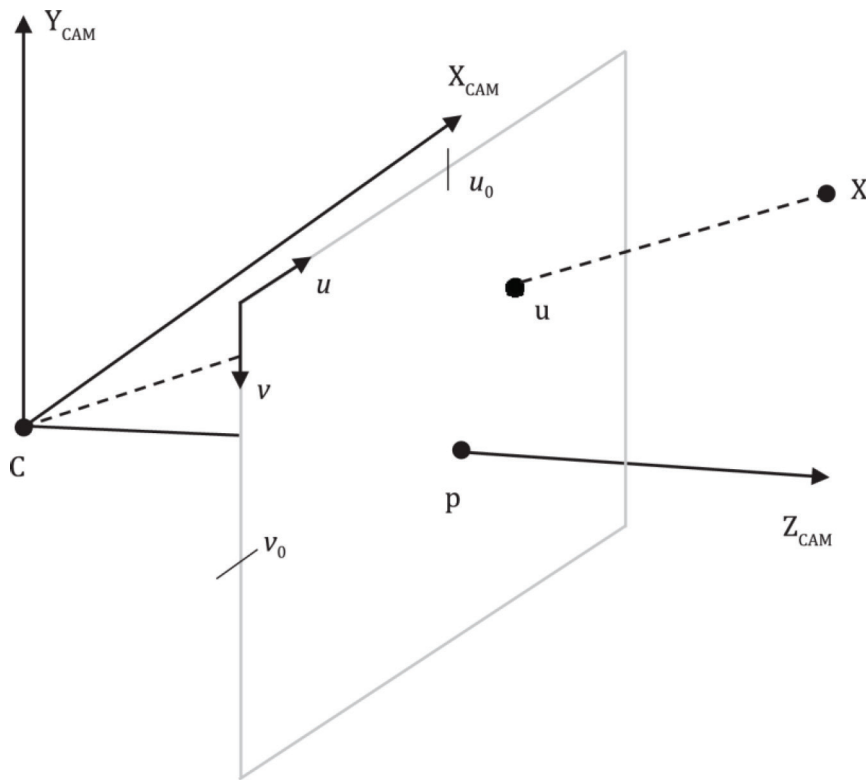


Figure A.7 — Pinhole camera geometry

The focal lengths f_u and f_v correspond to the distance between the optical centres of the cameras and their respective image planes. They are represented in terms of pixel dimensions in the u and v directions. For example, if the camera focal length is given in mm one needs to convert it to pixel dimensions using [Formulae \(A.1\)](#) and [\(A.2\)](#). For square pixels the sensor height is equal to the sensor width, and f_u is equal to f_v .

$$f_u \text{ (pixel)} = (\text{focal length (mm)} / \text{sensor width (mm)}) \times \text{image width (pixel)} \quad (\text{A.1})$$

$$f_v \text{ (pixel)} = (\text{focal length (mm)} / \text{sensor height (mm)}) \times \text{image height (pixel)} \quad (\text{A.2})$$

The axis skew parameter sk causes shear distortion in the projected image, and for most of the cameras its value is equal to zero. The parameters f_u , f_v , sk , u_0 and v_0 completely characterize the mapping of an image point from camera to pixels coordinates. They are known as the intrinsic or internal parameters of a camera system and can be represented by the transformation matrix $\mathbf{K}$ in [Formula \(A.3\)](#):

$$\mathbf{K} = \begin{bmatrix} f_u & sk & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{A.3})$$

The matrix $\mathbf{K}$ is known as the calibration matrix. In general, the mapping from 3D local reference grid to the image is linear. A camera system is said to be calibrated when its intrinsic parameters are known, otherwise it is an uncalibrated camera system.

A.3.3.3.3 Extrinsic camera parameters

The parameters that relate the camera orientation and position to a 3D local reference grid coordinate system are called the extrinsic or external parameters. The geometric quantities (rotational and translational components) describing the relative position and orientation of the cameras are called the extrinsic parameters of the camera system. The rotation and translation can be represented in an extrinsic

matrix taking the form of a rigid transformation matrix: a 3×3 rotation matrix $\mathbf{R}$, and a 3×1 translation column-vector $\mathbf{t}_M = (XCC, YCC, ZCC)^T$ that can be represented in a 3×4 matrix, as in [Formula \(A.4\)](#).

$$\mathbf{R}|\mathbf{t}_M = \begin{bmatrix} r_{11} & r_{12} & r_{13} & XCC \\ r_{21} & r_{22} & r_{23} & YCC \\ r_{31} & r_{32} & r_{33} & ZCC \end{bmatrix} \quad (\text{A.4})$$

Matrix $\mathbf{P}$ ([Formula \(A.5\)](#)), known as the projection matrix, or camera matrix, represents the pose of the 3D local reference grid coordinates relative to the image coordinates. It contains 6 independent parameters (Degrees of Freedom - DoF): 3 for rotation and 3 for translation. These parameters are expressed in the local reference grid ([Figure A.6](#)).

$$\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}_M] \quad (\text{A.5})$$

The 3×4 matrix $\mathbf{P}$ relates the sensor plane 2D image coordinates $\mathbf{u} = (u, v)$ ($\tilde{\mathbf{u}} = (u, v, 1)$ in homogeneous coordinates) to the 3D local reference grid coordinates $\mathbf{X} = (X_L, Y_L, Z_L)$ ($\tilde{\mathbf{X}} = (X_L, Y_L, Z_L, 1)$ in homogeneous coordinates) via [Formula \(A.5\)](#). The mapping between a point in the 3D world into a 2D image is given by $\tilde{\mathbf{u}} = \mathbf{P} \tilde{\mathbf{X}}$, where $\tilde{\mathbf{u}}$ is the image point in homogeneous coordinates, $\mathbf{P}$ is the camera matrix and $\tilde{\mathbf{X}}$ is the 3D local reference grid point in homogeneous coordinates. The extrinsic camera parameter matrix represents the current status of the camera in the 3D scene.

For example, a rotation in 3D local reference grid space involves an axis around which to rotate, and an angle of rotation, according to the right-handed coordinates.

[Formulae \(A.6\)](#), [\(A.7\)](#) and [\(A.8\)](#) show the rotation matrix values for rotations around these 3 axes:

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_{X_{CAM}} & -\sin\theta_{X_{CAM}} \\ 0 & \sin\theta_{X_{CAM}} & \cos\theta_{X_{CAM}} \end{bmatrix}, \text{ rotation around the } X_L \text{ axis, rotates } Y_L, Z_L, \text{ leaving the } X_L \text{ coordinates fixed} \quad (\text{A.6})$$

$$R_y = \begin{bmatrix} \cos\theta_{Y_{CAM}} & 0 & \sin\theta_{Y_{CAM}} \\ 0 & 1 & 0 \\ -\sin\theta_{Y_{CAM}} & 0 & \cos\theta_{Y_{CAM}} \end{bmatrix}, \text{ rotation around the } Y_L \text{ axis, rotates } X_L, Z_L, \text{ leaving the } Y_L \text{ coordinates fixed} \quad (\text{A.7})$$

$$R_z = \begin{bmatrix} \cos\theta_{Z_{CAM}} & \sin\theta_{Z_{CAM}} & 0 \\ -\sin\theta_{Z_{CAM}} & \cos\theta_{Z_{CAM}} & 0 \\ 0 & 0 & 1 \end{bmatrix}, \text{ rotation around the } Z_L \text{ axis, rotates } X_L, Y_L, \text{ leaving the } Z_L \text{ coordinates fixed.} \quad (\text{A.8})$$

Any rotation can be expressed as a combination of the three rotations about the three axes, as per [Formula \(A.9\)](#):

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_{X_{CAM}} & -\sin\theta_{X_{CAM}} \\ 0 & \sin\theta_{X_{CAM}} & \cos\theta_{X_{CAM}} \end{bmatrix} \begin{bmatrix} \cos\theta_{Y_{CAM}} & 0 & \sin\theta_{Y_{CAM}} \\ 0 & 1 & 0 \\ -\sin\theta_{Y_{CAM}} & 0 & \cos\theta_{Y_{CAM}} \end{bmatrix} \begin{bmatrix} \cos\theta_{Z_{CAM}} & \sin\theta_{Z_{CAM}} & 0 \\ -\sin\theta_{Z_{CAM}} & \cos\theta_{Z_{CAM}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (A.9)$$

[Formula \(A.10\)](#) shows the mapping of a 3-D local reference grid point (X_L, Y_L, Z_L) to the image coordinate system (u, v) , for a calibrated system:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & sk & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} XCC \\ YCC \\ ZCC \end{bmatrix} \begin{bmatrix} X_L \\ Y_L \\ Z_L \\ 1 \end{bmatrix} \quad (A.10)$$

A.3.3.3.4 Camera Parameter box definition

The type of the Camera Parameter box shall be 'lfcP' (0x6C66 6370) and the contents of the box shall have the format as in [Figure A.8](#).

If the Camera Parameter box is not signalled, all parameters specified in [Figure A.8](#) and [Table A.7](#) shall be initialized to zero. The scaling values S_{GLX} , S_{GLY} and S_{GLZ} will be initialized to 1.

PP	X_{LO}	Y_{LO}	Z_{LO}	θ_{XL}	θ_{YL}	θ_{ZL}	S_{GLX}	S_{GLY}	S_{GLZ}	ExtInt	Baseline _x	Baseline _y
	XCC(0,0)	YCC(0,0)	ZCC(0,0)	$\theta_{XCAM}(0,0)$	$\theta_{YCAM}(0,0)$	$\theta_{ZCAM}(0,0)$						
	f(0,0)	sW(0,0)	sH(0,0)	sk(0,0)	u ₀ (0,0)	v ₀ (0,0)						
	XCC(0,1)	YCC(0,1)	ZCC(0,1)	$\theta_{XCAM}(0,1)$	$\theta_{YCAM}(0,1)$	$\theta_{ZCAM}(0,1)$						
	f(0,1)	sW(0,1)	sH(0,1)	sk(0,1)	u ₀ (0,1)	v ₀ (0,1)						
	XCC(T-1,S-1)	YCC(T-1,S-1)	ZCC(T-1,S-1)	$\theta_{XCAM}(T-1,S-1)$	$\theta_{YCAM}(T-1,S-1)$	$\theta_{ZCAM}(T-1,S-1)$						
	f(T-1,S-1)	sW(T-1,S-1)	sH(T-1,S-1)	sk(T-1,S-1)	u ₀ (T-1,S-1)	v ₀ (T-1,S-1)						

Key**Light field position in global reference grid**

PP	precision of coordinates (Precision Prec = 16×2^{PP})
X_{LO}	position of the origin of the local reference grid in the global reference system along the X_G coordinate axis
Y_{LO}	position of the origin of the local reference grid in the global reference system along the Y_G coordinate axis
Z_{LO}	position of the origin of the local reference grid in the global reference system along the Z_G coordinate axis
θ_{XL}	rotation offset around the X_G axis (in rad)
θ_{YL}	rotation offset around the Y_G axis (in rad)
θ_{ZL}	rotation offset around the Z_G axis (in rad)
S_{GLX}	scaling of local reference grid system with respect to global reference grid system for the X-axes before rotation
S_{GLY}	scaling of local reference grid system with respect to global reference grid system for the Y-axes before rotation
S_{GLZ}	scaling of local reference grid system with respect to global reference grid system for the Z-axes before rotation

Extrinsic parameters for pinhole camera corresponding to subaperture view (t, s)

ExtInt	signals which extrinsic and intrinsic camera parameters are signalled
Baseline _x	horizontal camera baseline, used when $XCC(t,s) = s \times \text{Baseline}_x + XCC(0,0)$, and hence, $XCC(0,1)$, $XCC(0,2)$, ..., $XCC(T-1,S-1)$ do not need to be signalled
Baseline _y	vertical camera baseline, used when $YCC(t,s) = t \times \text{Baseline}_y + YCC(0,0)$, and hence, $YCC(0,1)$, $YCC(0,2)$, ..., $YCC(T-1,S-1)$ do not need to be signalled
XCC (t, s)	camera centre of subaperture view (t, s) in local reference grid along X_L coordinate axis
YCC(t, s)	camera centre of subaperture view (t, s) in local reference grid along Y_L coordinate axis
ZCC (t, s)	camera centre of subaperture view (t, s) in local reference grid along Z_L coordinate axis

$\theta_{X_{CAM}}(t, s)$	camera rotation offset around the X_L axis (in rad)
$\theta_{Y_{CAM}}(t, s)$	camera rotation offset around the Y_L axis (in rad)
$\theta_{Z_{CAM}}(t, s)$	camera rotation offset around the Z_L axis (in rad)

Intrinsic parameters for pinhole camera corresponding to subaperture view (t, s)

$f(t, s)$	focal length (in mm)
$sW(t, s)$	sensor width (in mm)
$sH(t, s)$	sensor height (in mm)
$sk(t, s)$	sensor skew
$u_0(t, s)$	horizontal principle point offset
$v_0(t, s)$	vertical principle point offset

NOTE 1 **PP** indicates the floating-point precision issued for the coordinates.

NOTE 2 $X_{LO}, Y_{LO}, Z_{LO}, \theta_{XL}, \theta_{YL}, \theta_{ZL}, S_{GLX}, S_{GLY}, S_{GLZ}, \theta_{X_{CAM}}(t, s), \theta_{Y_{CAM}}(t, s)$ and $\theta_{Z_{CAM}}(t, s)$ utilize the chosen floating-point precision.

Figure A.8 — Organization of the contents of the Camera Parameter box

The geometrical coordinates of the centre of the camera when acquiring the view (t, s) are denoted as

$XCC(t, s)$, camera centre on X_L

$YCC(t, s)$, camera centre on Y_L

$ZCC(t, s)$, camera centre on Z_L .

An example of camera centre coordinates for a planar camera array is given in [Figure A.9](#), where both the horizontal and vertical coordinates are illustrated for five views in the camera array. The camera centres XCC and YCC are used together with the normalized disparity maps $\tilde{D}$ to obtain horizontal and vertical disparity maps between a pair of views in the light field. For usage examples see Annex [D.4](#) and Annex [E.4](#).

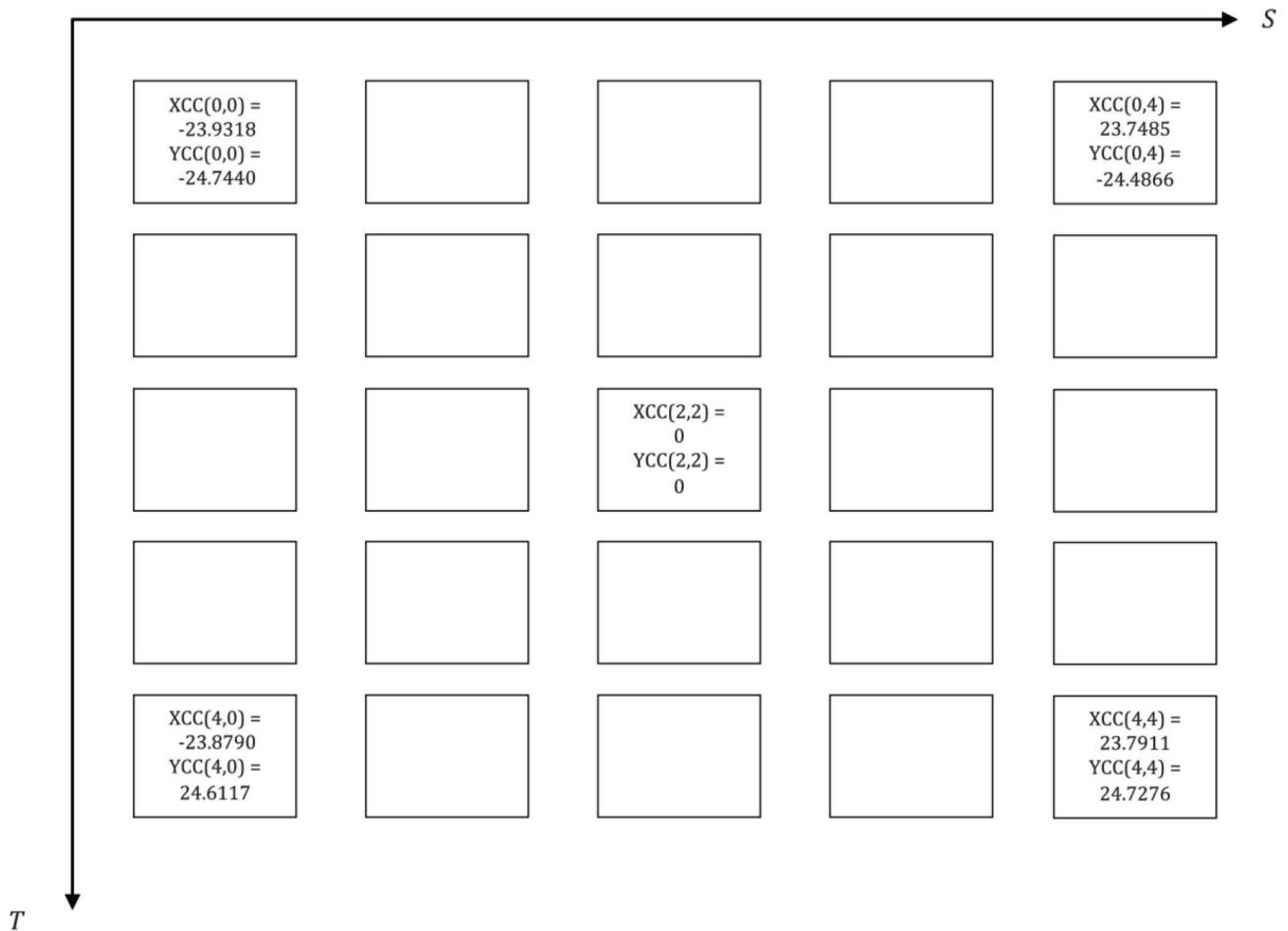


Figure A.9 — Organization of subaperture views and associated planar camera calibration information

Table A.7 — Format of the contents of the Camera Parameter box

Field name	Size (bits)	Value
PP	8	0 to (2^8-1)
X_{LO}	variable	big endian floating-point
Y_{LO}	variable	big endian floating-point
Z_{LO}	variable	big endian floating-point
θ_{XL}	variable	big endian floating-point
θ_{YL}	variable	big endian floating-point
θ_{ZL}	variable	big endian floating-point
S_{GLX}	variable	big endian floating-point
S_{GLY}	variable	big endian floating-point
S_{GLZ}	variable	big endian floating-point
ExtInt	16	See Table A.8
Baseline _x	32	single precision, big endian floating-point
Baseline _y	32	single precision, big endian floating-point
XCC(0,0)	32	single precision, big endian floating-point
YCC(0,0)	32	single precision, big endian floating-point
ZCC(0,0)	32	single precision, big endian floating-point

Table A.7 (continued)

Field name	Size (bits)	Value
$\theta_{\text{Xcam}}(0,0)$	32	single precision, big endian floating-point
$\theta_{\text{Ycam}}(0,0)$	32	single precision, big endian floating-point
$\theta_{\text{Zcam}}(0,0)$	32	single precision, big endian floating-point
$f(0,0)$	32	single precision, big endian floating-point
$sW(0,0)$	32	single precision, big endian floating-point
$sH(0,0)$	32	single precision, big endian floating-point
$sk(0,0)$	32	single precision, big endian floating-point
$u_0(0,0)$	32	single precision, big endian floating-point
$v_0(0,0)$	32	single precision, big endian floating-point
XCC(0,1)	32	single precision, big endian floating-point
YCC(0,1)	32	single precision, big endian floating-point
...	...	...
$u_0(T-1,S-1)$	32	single precision, big endian floating-point
$v_0(T-1,S-1)$	32	single precision, big endian floating-point

Table A.8 — Meaning of ExtInt bits

Bit position	Value	Meaning
0 (LSB)	0	$XCC(t,s) = s \times \text{BaselineX} + XCC(0,0)$ Signal BaselineX and XCC(0,0), the remaining $(T \times S)-1$ XCC entries in Table A.7 are not signalled
	1	The $T \times S$ XCC(t,s) entries in Table A.7 are signalled
1	0	$YCC(t,s) = t \times \text{BaselineY} + YCC(0,0)$ Signal BaselineY and YCC(0,0), the remaining $(T \times S)-1$ YCC entries in Table A.7 are not signalled
	1	The $T \times S$ YCC(t,s) entries in Table A.7 are signalled
2	0	$ZCC(t,s) = ZCC(0,0)$ and the remaining $(T \times S)-1$ ZCC entries in Table A.7 are not signalled
	1	The $T \times S$ ZCC(t,s) entries in Table A.7 are signalled
3	0	$\theta_{\text{Xcam}}(t,s) = \theta_{\text{Xcam}}(0,0)$ and the remaining $(T \times S)-1$ entries in Table A.7 are not signalled
	1	The $T \times S$ $\theta_{\text{Xcam}}(t,s)$ entries in Table A.7 are signalled
4	0	$\theta_{\text{Ycam}}(t,s) = \theta_{\text{Ycam}}(0,0)$ and the remaining $(T \times S)-1$ entries in Table A.7 are not signalled
	1	The $T \times S$ $\theta_{\text{Ycam}}(t,s)$ entries in Table A.7 are signalled
5	0	$\theta_{\text{Zcam}}(t,s) = \theta_{\text{Zcam}}(0,0)$ and the remaining $(T \times S)-1$ entries in Table A.7 are not signalled
	1	The $T \times S$ $\theta_{\text{Zcam}}(t,s)$ entries in Table A.7 are signalled
6	0	$f(t,s) = f(0,0)$ and the remaining $(T \times S)-1$ entries in Table A.7 are not signalled
	1	The $T \times S$ $f(t,s)$ entries in Table A.7 are signalled
7	0	$sW(t,s) = sW(0,0)$ and the remaining $(T \times S)-1$ entries in Table A.7 are not signalled
	1	The $T \times S$ $sW(t,s)$ entries in Table A.7 are signalled
8	0	$sH(t,s) = sH(0,0)$ and the remaining $(T \times S)-1$ entries in Table A.7 are not signalled
	1	The $T \times S$ $sH(t,s)$ entries in Table A.7 are signalled
9	0	$sk(t,s) = sk(0,0)$ and the remaining $(T \times S)-1$ entries in Table A.7 are not signalled
	1	The $T \times S$ $sk(t,s)$ entries in Table A.7 are signalled
10	0	$u_0(t,s) = \lfloor U/2 \rfloor$ and the $(T \times S)$ entries in Table A.7 are not signalled
	1	The $T \times S$ $u_0(t,s)$ entries in Table A.7 are signalled.

Table A.8 (continued)

Bit position	Value	Meaning
11	0	$0(t,s) = \lfloor V/2 \rfloor$ and the $(T \times S)$ entries in Table A.7 are not signalled
	1	The $T \times S \setminus 0(t,s)$ entries in Table A.7 are signalled
12-15	0	Reserved for future ISO/IEC use

A.3.4 Contiguous Codestream box

The Contiguous Codestream box contains a JPEG Pleno codestream.

The type of a Contiguous Codestream box shall be 'jp2c' (0x6A70 3263). The contents of the box shall be as in [Figure A.10](#) and [Table A.9](#):

**Figure A.10** — Organization of the contents of the Contiguous Codestream box**Table A.9** — Format of the contents of the Contiguous Codestream box

Field name	Size (bits)	Value
Code	Variable	Variable

Code

This field contains valid and complete JPEG Pleno codestream components as specified in [Annexes B, C, D, E](#) and [F](#).

Annex B (normative)

4D transform mode

B.1 General

This annex describes an instantiation of 4D transform mode encoder. Next, the codestream syntax is specified and subsequently the light field decoding process is detailed.

B.2 4D transform mode encoding

B.2.1 High-level coding architecture

In the 4D transform mode,^[2] the light field is encoded with a four-step process ([Figure B.1](#)). First, the 4D light field data is divided into fixed-sized 4D blocks that are independently encoded according to a predefined and fixed scanning order. However, if any light field dimensions are not multiple of such fixed-sized 4D blocks, the sizes of the 4D blocks at the light field boundaries shall be truncated to fit in the light field dimensions. The initial blocks can be further partitioned into a set of non-overlapping 4D sub-blocks, where the optimal partitioning parameters are derived based on a rate-distortion (R-D) criterion. Each sub-block is independently transformed by a variable block-size 4D DCT. Subsequently, the transformed blocks are quantized and entropy coded using hexadeca-tree bit plane decomposition and adaptive arithmetic encoding, producing a compressed representation of the light field. This coding procedure is applied to each colour component independently.

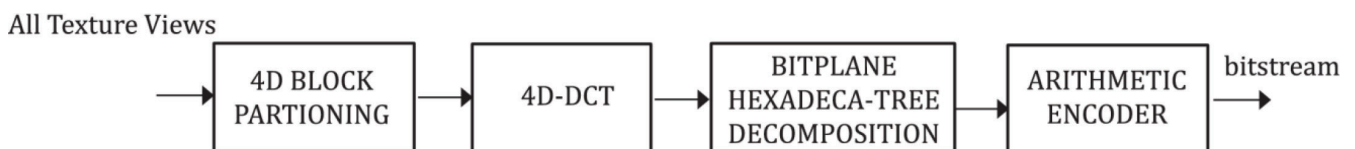


Figure B.1 — 4D transform mode encoding architecture

A sample (pixel) of the light field is referenced in a 4D coordinate system along the t , s , v and u -axes, where t and s are representing the coordinates of the addressed subaperture view, and v and u the sample coordinates within the subaperture images as illustrated in [Figure B.2](#). The blocks are scanned in the directions t , s , v and u , with direction u corresponding to the inner loop of the scan. In [Table B.1](#) a pseudo-code describes the scan of the 4D blocks. Each 4D block will generate a separate codestream embedded in the codestream, which can be independently decoded in support of random access.

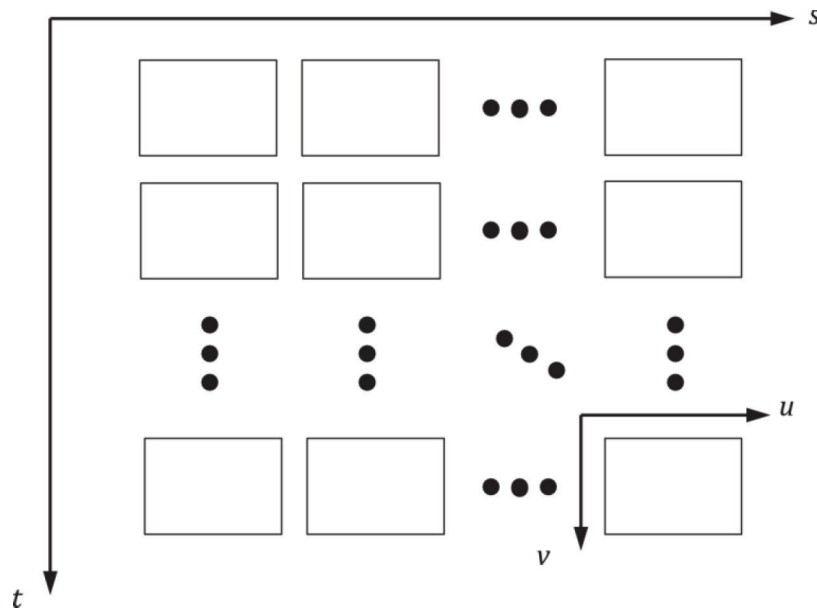
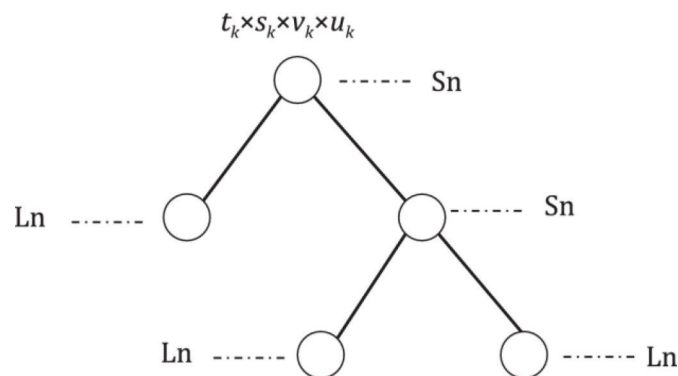


Figure B.2 — 4D structure

The 4D block partitioning, as well as the clustering of the bit-planes of transform coefficient – for efficient encoding – are signalled using tree structures (Figure B.3). The partitioning of the 4D blocks in sub-blocks is signalled with a binary tree using ternary flags. These flags signal whether:

- a block is transformed as is;
- is split into 4 blocks in the s, t (view) dimensions;
- is split into 4 blocks in the u, v (spatial) dimensions.

**Key**

- Sn split node
Ln leaf node

Figure B.3 — Binary tree representing 4D-block partitioning of a $t_k \times s_k \times v_k \times u_k$ 4D block

Before subsequently applying the 4D-DCT on the sub-blocks, a level-shift operation is performed to reduce the dynamic range requirements of the DCT (Annex B.2.3.1). The deployed 4D DCT (Annex B.2.3.2) is separable, i.e. with 1D transforms computed separately in each of the 4 directions. An example of the computation flow of the 4D separable transform is depicted in Figure B.4. The order of the 1D transforms is arbitrary.

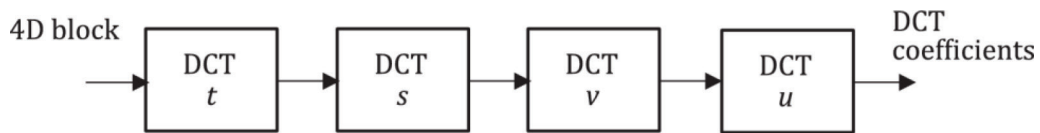
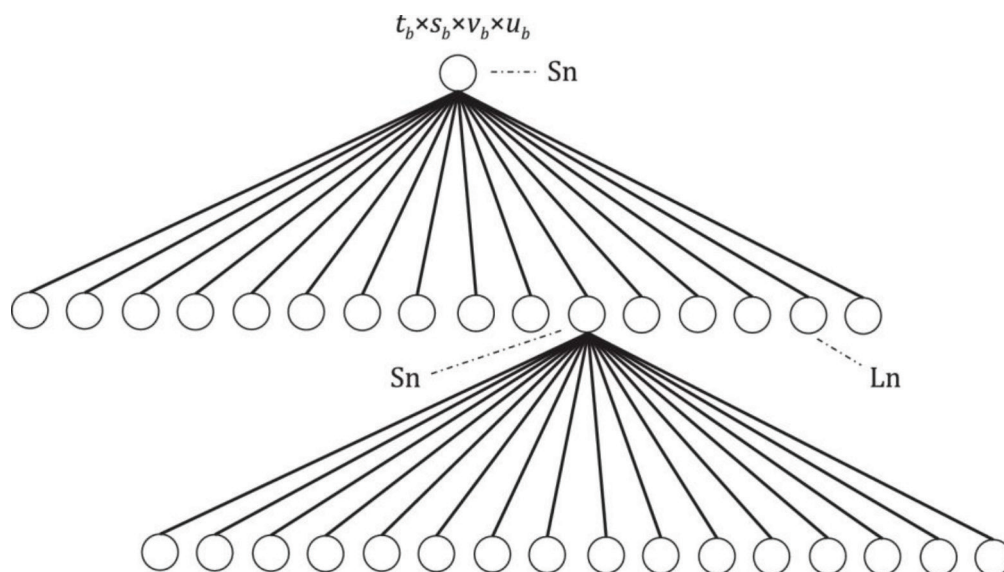


Figure B.4 — Separable forward 4D-DCT

After the 4D-DCT is performed, the set of transform coefficients is sliced into 4D bit-planes. A transform coefficient is considered non-significant on a 4D bit-plane if its bits belonging to higher 4D bit-planes are all zero. Otherwise, the transform coefficient is considered to be significant. A hexadeca-tree with ternary flags is used to group the non-significant transform coefficients and thus localize the significant transform coefficients (Figure B.5). The ternary flags signal that:

- either a block of transform coefficients containing a significant coefficient at the current bit-plane is split into 16 blocks in the four t , s , v , u dimensions,
- or a block of transform coefficients not containing any significant coefficient at the current bit-plane is not split,
- or a block of transform coefficients containing a significant coefficient at the current bit-plane is discarded.

The 4D bit-planes are scanned from the most significant bit-plane to the least significant one, where the least significant 4D bit-plane being determined by the desired quantization level. Both the hexadeca-tree bits and the bits from the transform coefficients are encoded using an adaptive arithmetic coder.

**Key**

- Sn split node
Ln leaf node

Figure B.5 — Hexadeca-tree representing the clustering of bit-planes of 4D transform coefficients of a $t_b \times s_b \times v_b \times u_b$ 4D block

The structure that clusters the non-significant 4D transform coefficients and thus localizes the significant ones is a hexadeca-tree, that is encoded using ternary flags. They signal that a block of transform coefficients containing a significant coefficient at the current bit-plane is split into 16 blocks in the four t , s , v , u dimensions, or that a block of transform coefficients not containing any significant coefficient at the current bit-plane is not split, or that a block of transform coefficients containing a significant coefficient at the current bit-plane is discarded. The 4D bit-planes are scanned from the most significant to the least significant one, where the least significant 4D bit-plane being determined by the desired quantization level. Both the hexadeca-tree

bits and the bits from the coefficients are encoded using an adaptive arithmetic coder. The hexadeca-tree structure mentioned above is depicted in [Figure B.5](#).

The following two sections provide more insight on the partitioning strategy and the entropy and quantization steps.

B.2.2 Partitioning optimization

The partitioning optimization for each block in [Figure B.6](#) is obtained as follows: initially each block is transformed by a full-size DCT ([B.2.3.2](#)), and the Lagrangian encoding cost, J_0 is evaluated ([Table B.5](#)). This cost is defined as $J_0 = D + \lambda R$, where D is the distortion incurred when representing the original block by the quantized version and R is the rate needed to encode it. The procedure of [Table B.5](#) is used to evaluate this cost.

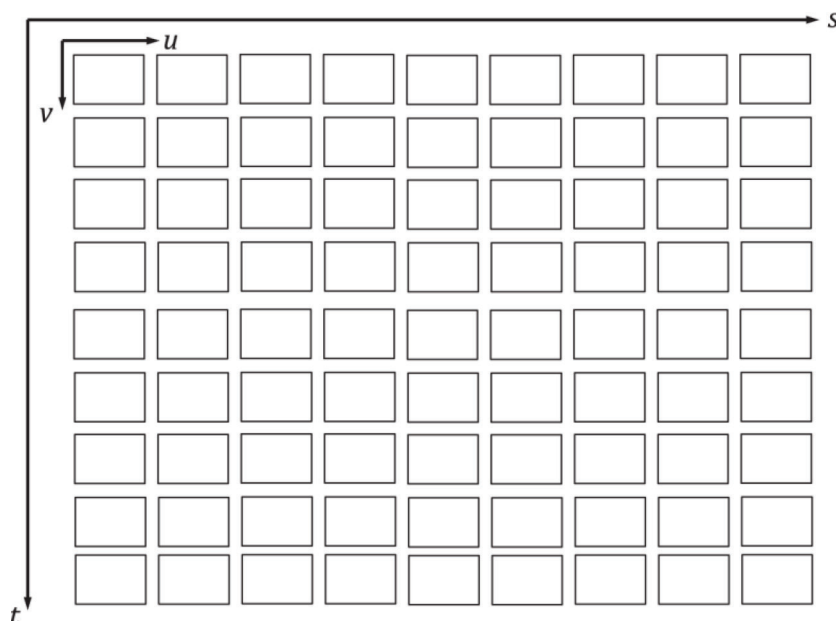


Figure B.6 — 4D Block of a light field

Next, the block can be partitioned in four sub-blocks each one with approximately a quarter of the pixels in the spatial dimensions. For example, let us consider a block B of dimensions $t_k \times s_k \times v_k \times u_k$. This block (pictured in [Figure B.7](#) and [Figure B.8](#)) will be subdivided in four sub-blocks of sizes $t_k \times s_k \times \lfloor v_k/2 \rfloor \times \lfloor u_k/2 \rfloor$, $t_k \times s_k \times \lfloor v_k/2 \rfloor \times (u_k - \lfloor u_k/2 \rfloor)$, $t_k \times s_k \times (v_k - \lfloor v_k/2 \rfloor) \times \lfloor u_k/2 \rfloor$ and $t_k \times s_k \times (v_k - \lfloor v_k/2 \rfloor) \times (u_k - \lfloor u_k/2 \rfloor)$, respectively.

[Figure B.8](#) shows a 4D block with dimensions $t_k \times s_k \times v_k \times u_k$ in the root node. When applying the spatial split (signalled with the *spatialSplit* flag) to the root node, the tree in [Figure B.8](#) is obtained. [Figure B.7](#) illustrates the four ways that a single view is partitioned using the *spatialSplit* flag, corresponding to four nodes of [Figure B.8](#). The optimal partition for each sub-block is computed by means of the recursive procedure described in [Table B.5](#) and the Lagrangian costs of the four sub-blocks are added to compute the Lagrangian cost J_S (Spatial R-D cost).

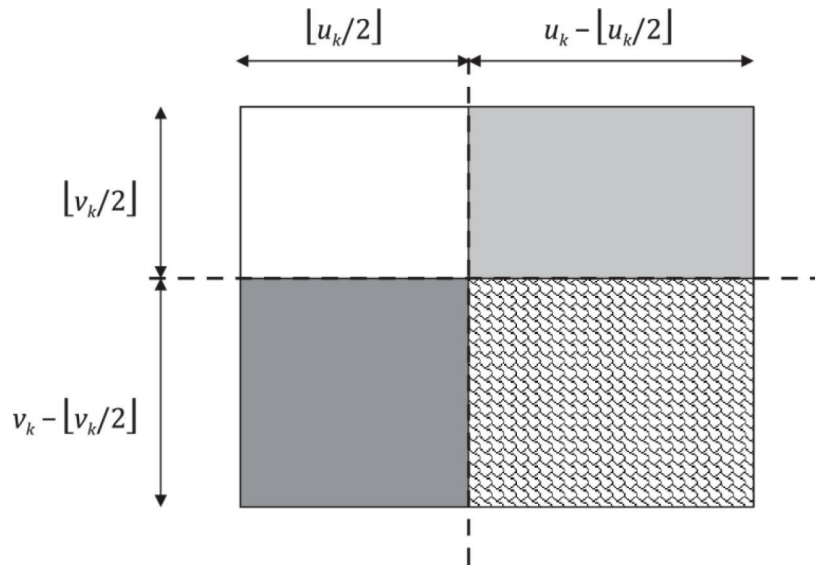


Figure B.7 — Spatial partitioning of block with spatial dimensions $v_k \times u_k$

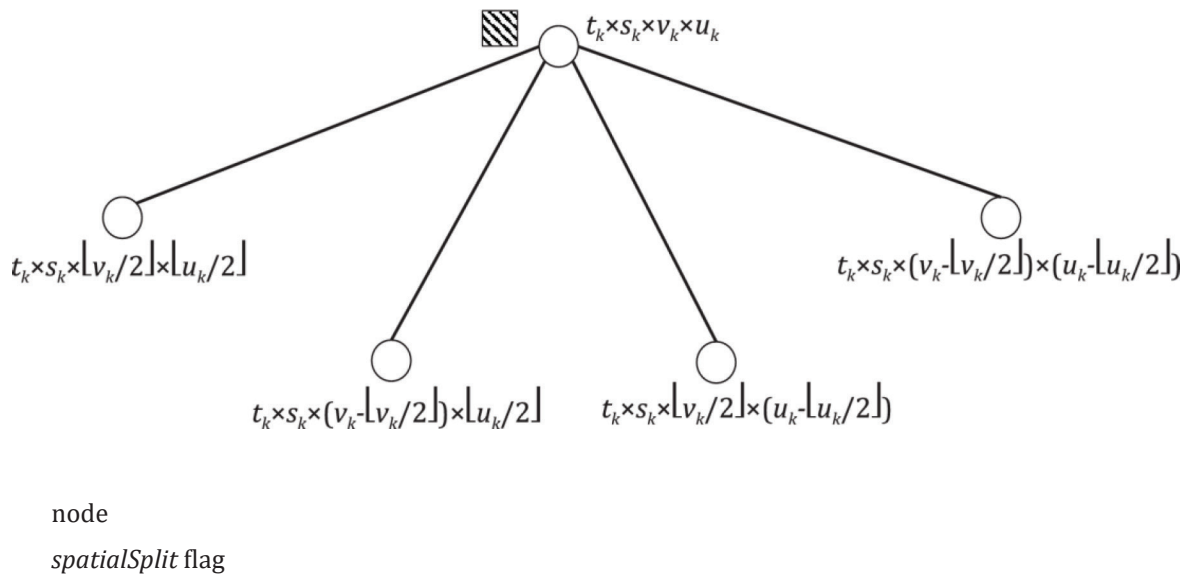


Figure B.8 — Hierarchical 4D partitioning of a 4D block with dimensions $t_k \times s_k \times v_k \times u_k$ using the *spatialSplit* flag

The block can be partitioned in four sub-blocks each one with approximately a quarter of the pixels in the view dimensions. For example, let us consider again a block B of dimensions $t_k \times s_k \times v_k \times u_k$. This block (pictured in [Figure B.9](#) and [Figure B.10](#)) will be subdivided in four sub-blocks of sizes $\lfloor t_k/2 \rfloor \times \lfloor s_k/2 \rfloor \times v_k \times u_k$, $\lfloor t_k/2 \rfloor \times (s_k - \lfloor s_k/2 \rfloor) \times v_k \times u_k$, $(t_k - \lfloor t_k/2 \rfloor) \times \lfloor s_k/2 \rfloor \times v_k \times u_k$, $(t_k - \lfloor t_k/2 \rfloor) \times (s_k - \lfloor s_k/2 \rfloor) \times v_k \times u_k$, respectively [Figure B.10](#)). When applying the view split (signalled with the *viewSplit* flag) to the root node, the tree in [Figure B.10](#) is obtained. [Figure B.9](#) illustrates the four ways that a 4D block is partitioned using the *viewSplit* flag, corresponding to four the nodes of [Figure B.10](#). The optimal partition for each sub-block is computed by means of the recursive procedure described in [Table B.5](#) and the Lagrangian costs of the four sub-blocks are added to compute the Lagrangian cost J_V (View R-D cost).

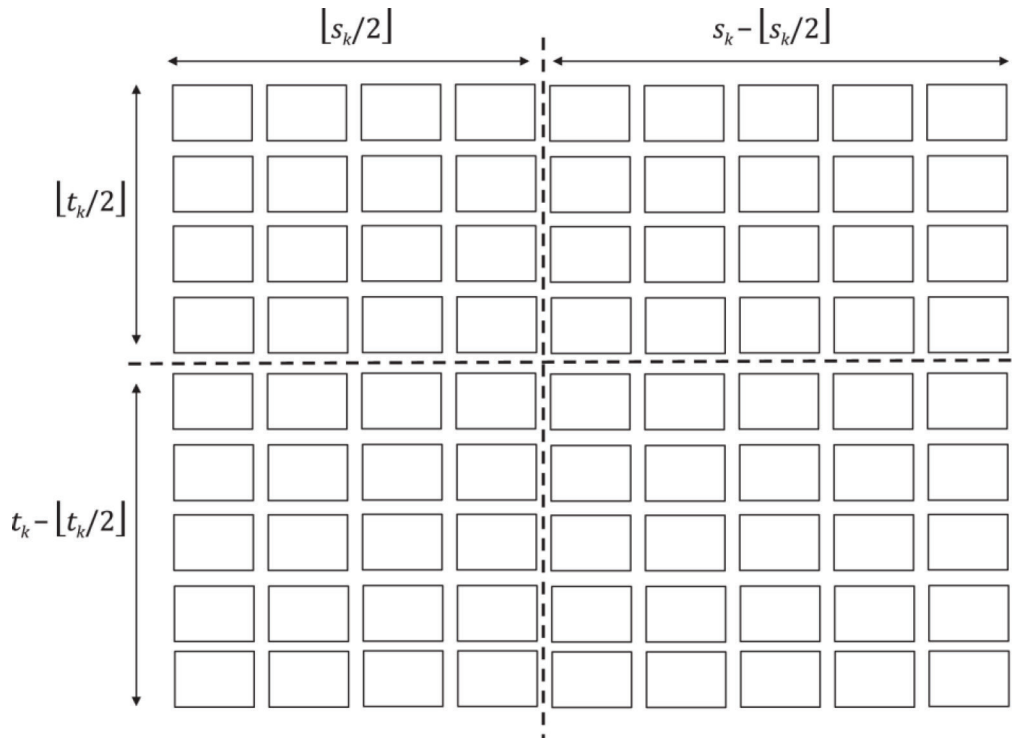
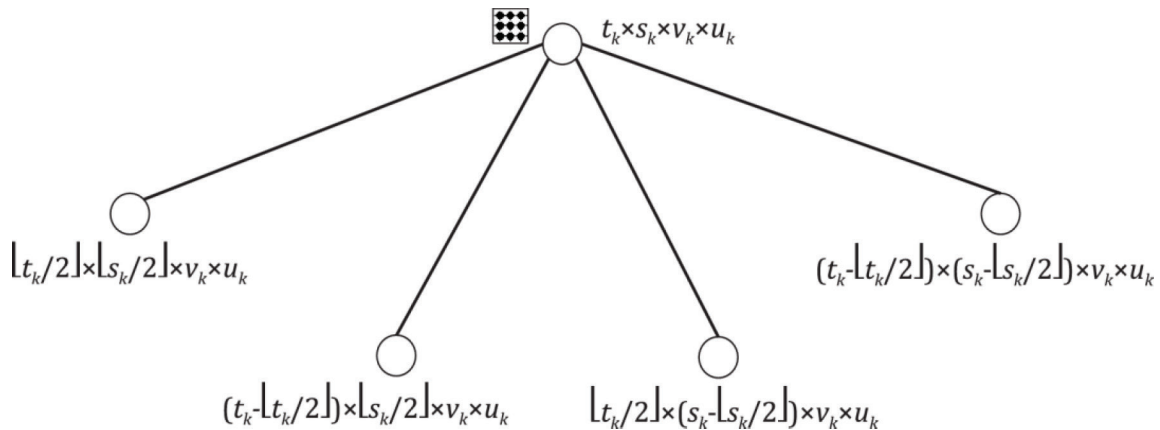


Figure B.9 — View partitioning of a 4D block of dimensions $t_k \times s_k \times v_k \times u_k$



Key



node



viewSplit flag

Figure B.10 — Hierarchical 4D partitioning of a 4D block with dimensions $t_k \times s_k \times v_k \times u_k$, using the *viewSplit* flag

Finally, the three Lagrangian costs (J_0 , J_S and J_V) are compared (Table B.5) and the one presenting the lowest value is chosen.

The recursive partition procedure (Table B.5) keeps track of this tree (Table B.11) and returns a *partitionString* (Table B.5) that represents the optimal tree. The string is obtained as follows: once the lowest cost is chosen, the current value of *partitionString* is augmented by appending to it the flag corresponding to the lowest cost chosen. Then, the string returned by the recursive call that leads to the minimum cost is also appended to the end of the *partitionString* and the procedure returns both the minimum cost J_0 , J_S or J_V and the updated *partitionString*.

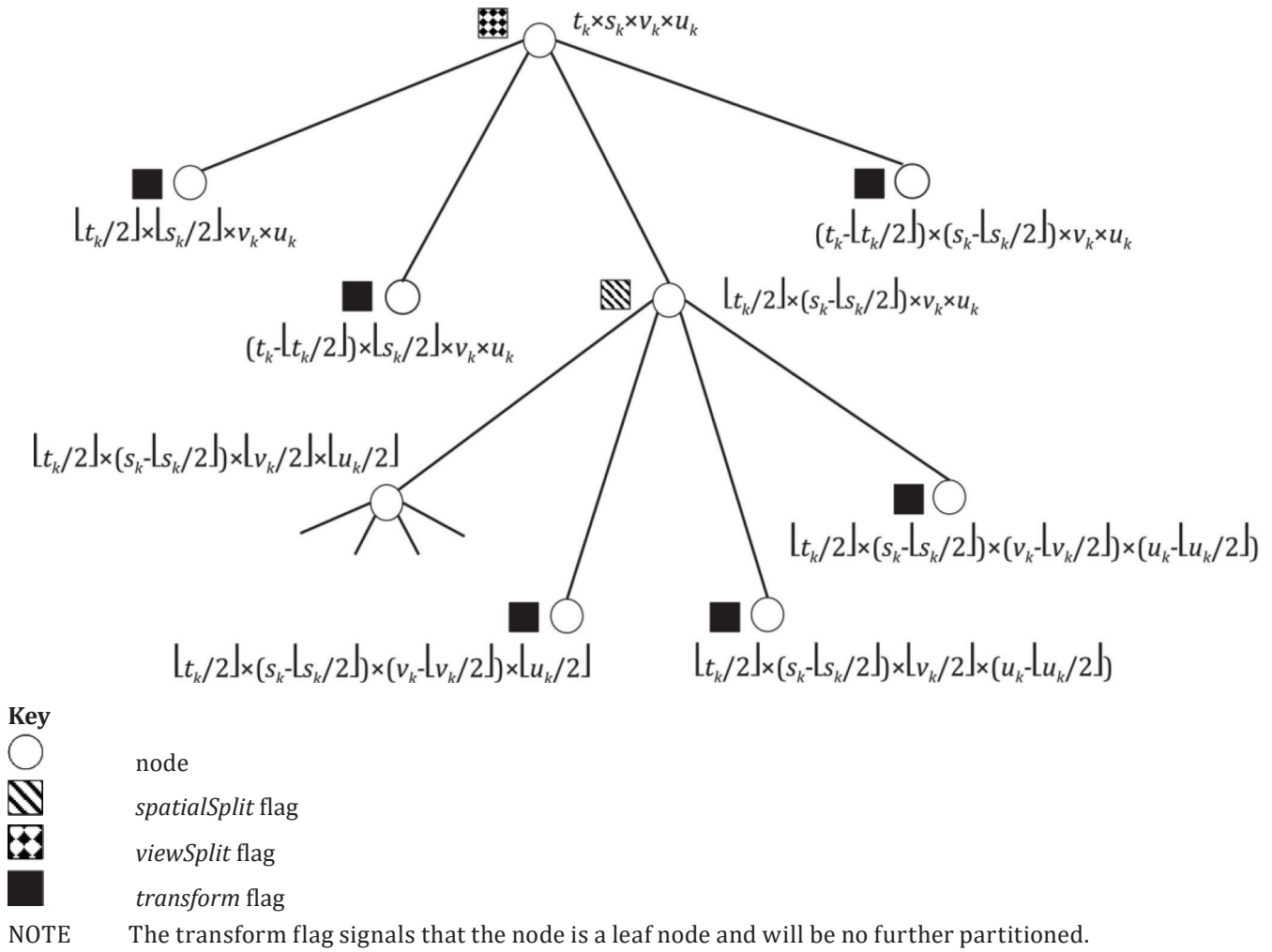


Figure B.11 — Hierarchical 4D partitioning using the *viewSplit* flag and the *spatialSplit* flag

After the optimal partition tree is found, the Encode Partition procedure ([Table B.6](#)) is called to encode it.

B.2.3 Forward 4D-DCT

B.2.3.1 Level shift

Subsequently, the subblocks are subject to a DCT. However, before processing the forward DCT for a block of source light field samples, if the samples of the component are unsigned, those samples shall be level shifted to a signed representation. if the MSB of $Ssiz^i$ from the LFC marker segment (see Annex [B.3.2.6.3](#)) is zero, all samples $x(u, v, s, t)$ of the i th component are level shifted by subtracting the same quantity from each sample as follows:

$$x(u, v, s, t) \leftarrow x(u, v, s, t) - 2^{Ssiz^i}$$

B.2.3.2 Forward 4D-DCT function

First all components have to be converted to the same precision (bit-depth). Each colour component c is by multiplied by $2 \times (BPC - Ssiz^c)$ before the forward 4D-DCT.

For a given light field $x(u,v,s,t)$ the corresponding 4D-DCT representation $X(i,j,p,q)$ is computed by transforming the block along each dimension as follows.

$$\begin{aligned}
 X^{(u)}(i,v,s,t) &= \sqrt{u_k} \alpha(i) \sum_{u=0}^{u_k-1} X(u,v,s,t) \cos\left[\frac{\pi(2u+1)i}{2u_k}\right]; \quad i=0,1,\dots, u_k-1; \quad v=0,1,\dots, v_k-1 \\
 &\quad s=0,1,\dots, s_k-1; \quad t=0,1,\dots, t_k-1 \\
 X^{(uv)}(i,j,s,t) &= \sqrt{v_k} \alpha(j) \sum_{v=0}^{v_k-1} X^{(u)}(i,v,s,t) \cos\left[\frac{\pi(2v+1)j}{2v_k}\right]; \quad i=0,1,\dots, u_k-1; \quad j=0,1,\dots, v_k-1 \\
 &\quad s=0,1,\dots, s_k-1; \quad t=0,1,\dots, t_k-1 \\
 X^{(uvs)}(i,j,p,t) &= \sqrt{s_k} \alpha(p) \sum_{s=0}^{s_k-1} X^{(uv)}(i,j,s,t) \cos\left[\frac{\pi(2s+1)p}{2s_k}\right]; \quad i=0,1,\dots, u_k-1; \quad j=0,1,\dots, v_k-1 \\
 &\quad p=0,1,\dots, s_k-1; \quad t=0,1,\dots, t_k-1 \\
 X(i,j,p,q) &= \sqrt{t_k} \alpha(q) \sum_{t=0}^{t_k-1} X^{(uvs)}(i,j,p,t) \cos\left[\frac{\pi(2t+1)q}{2t_k}\right]; \quad i=0,1,\dots, u_k-1; \quad j=0,1,\dots, v_k-1 \\
 &\quad p=0,1,\dots, s_k-1; \quad q=0,1,\dots, t_k-1
 \end{aligned}$$

where $\alpha(0) = \sqrt{\frac{1}{N}}$ and $\alpha(n) = \sqrt{\frac{2}{N}}$; $n=1,2,\dots, N-1$, where N is the size of the transform. Output coefficients are represented as 32-bit integers. As indicated earlier, the transform order is arbitrary.

B.2.4 Quantization and entropy encoding

The quantization and entropy encoding rely on the R-D optimized hexadeca-tree structure, which is constructed based on the procedure listed in [Table B.7](#) and which is further discussed in this paragraph. This tree is uniquely represented by a series of ternary flags: *lowerBitplane*, *splitBlock* and *zeroBlock* ([Table B.44](#)). The hexadeca-tree is built by recursively subdividing a 4D block until all sub-blocks reach a $1 \times 1 \times 1 \times 1$ 4D block-size. The hexadeca-tree is built by recursively subdividing a 4D block. Starting from a 4D block of size $t_b \times s_b \times v_b \times u_b$, and a bit-plane initially set to *maxBitplane* ([Table B.11](#)), 3 operations are performed:

- i) **Lower the bit-plane:** in this case, the descendant of the node is another block with the same dimensions as the original one but represented with precision *bitplane*-1. This is used to indicate for all pixels of the block that the binary representation of their magnitudes at the current bitplane and above are zero. This situation is encoded by the ternary flag value *lowerBitPlane*.
- ii) **Split the block:** in this case, the node will have up to 16 children, each one associated to a sub-block with approximately half the length of the original block in all four dimensions. For example, a block B of size $t_b \times s_b \times v_b \times u_b$ can be split in the following sub-blocks:

$$\begin{aligned}
 &B_{0000} \text{ of size } (\lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor), \\
 &B_{0001} \text{ of size } (\lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor), \\
 &B_{0010} \text{ of size } (\lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor), \\
 &B_{0011} \text{ of size } (\lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor), \\
 &B_{0100} \text{ of size } (\lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor), \\
 &B_{0101} \text{ of size } (\lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor), \\
 &B_{0110} \text{ of size } (\lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor), \\
 &B_{0111} \text{ of size } (\lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor), \\
 &B_{1000} \text{ of size } (t_b - \lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor), \\
 &B_{1001} \text{ of size } (t_b - \lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor), \\
 &B_{1010} \text{ of size } (t_b - \lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor), \\
 &B_{1011} \text{ of size } (t_b - \lfloor t_b/2 \rfloor \times \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor), \\
 &B_{1100} \text{ of size } (t_b - \lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor),
 \end{aligned}$$

B_{1101} of size $(t_b - \lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor)$,
 B_{1110} of size $(t_b - \lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times \lfloor u_b/2 \rfloor)$,
 B_{1111} of size $(t_b - \lfloor t_b/2 \rfloor \times s_b - \lfloor s_b/2 \rfloor \times v_b - \lfloor v_b/2 \rfloor \times u_b - \lfloor u_b/2 \rfloor)$.

There are 16 possible sub-blocks, but depending on the size of the parent block, some of these descendant sub-blocks will have one or more of their lengths equal to zero and shall be skipped. All descendants have the same bit-depth as the parent. This situation is indicated by the flag value *splitBlock* ([Table B.44](#)).

- iii) **Discard the block:** the node has no descendants and is represented by an all-zeros block. This situation is indicated by the flag value *zeroBlock* ([Table B.44](#)).

The procedure described in [Table B.7](#) recursively subdivides the input block, as determined by the ternary flags in the segmentation string until a $1 \times 1 \times 1 \times 1$ 4D block-size is reached.

Given a particular hexadeca-tree, specified by a unique *segmentationString* of ternary segmentation flags together with a particular block, the data can be encoded by means of the recursive procedure described in [Table B.8](#). The inputs to this procedure are the transformed block to be encoded and the optimal partition string. It recursively subdivides the input block, as determined by the ternary flags in the segmentation string. Then the magnitude of the single coefficient of this block is encoded one bit at a time, ranging from the current bit plane to the *minimumBitPlane* ([Table B.8](#)), using an arithmetic encoder with a different context information for each bit. If the coefficient is not zero valued, its signal is encoded as well.

The rate and the distortion achieved depend heavily on the choice of the segmentation tree as well as the data itself, and those should be matched. The procedure described in [Table B.7](#), recursively chooses the optimal segmentation tree to use in the encoding of a given block in a rate-distortion sense. The optimization works as follows: it starts with *bitplane*=*maximumBitplane*, *segmentationString*=*null*, variables J_0 and J_1 both set to infinity and the full transformed input block. The transformed block is scanned and all its coefficients are compared to a threshold given by 2^{bitplane} . If the magnitudes of all of them are less than the threshold, the optimization procedure is recursively called with the same block as input, but with a *bitplane* value decreased by one (*bitplane*-1). The values returned by this recursive call are the new Lagrangian cost J_0 , and a rate-distortion optimized segmentation string *lowerSegmentationString*. However, if any coefficient is above the threshold, the transformed block is segmented into up to 16 sub-blocks as previously described. The optimization procedure is called recursively for each sub-block and the returned Lagrangian costs are added to obtain the new Lagrangian cost J_1 . The segmentation strings returned from these calls are concatenated to form the *splitSegmentationString*.

Another Lagrangian cost J_2 is evaluated considering the resulting cost if the block was replaced by a block entirely composed of zeros. The lowest cost is chosen, and the segmentation string is updated as follows:

- If the minimum cost is J_0 , the input segmentation string is augmented by appending a flag *lowerBitplane* followed by the *lowerSegmentationString*.
- If the minimum cost is J_1 , the input segmentation string is augmented by appending a flag *splitBlock* followed by the *splitSegmentationString*.
- If the minimum cost is J_2 , the input segmentation string is augmented by appending a flag *zeroBlock*.

The procedure returns the lowest cost and the resulting associated segmentation string.

The 4D coefficients, flags, and probability context information generated during the encoding process, are input to the arithmetic encoder, that generates the compressed representation of the light field ([Table B.4](#)). The adaptive statistical binary arithmetic coding is detailed in [Table B.12](#), [Table B.13](#) and [Table B.14](#). The arithmetic coding requires transmitting only the information needed to allow a decoder to determine the particular fractional interval between 0 and 1 to which the sequence is mapped, adapting to changing statistics in the codestream.