INTERNATIONAL STANDARD

ISO/IEC 14496-16

Fourth edition
2011-11-01

AMENDMENT 4
2017-10

Information technology — Coding of audio-visual objects —

Part 16: Animation Framework eXtension (AFX)

AMENDMENT 4: Pattern-based 3D mesh coding (PB3DMC)

Technologies de l'information — Codage des objets audiovisuels —

Partie 16: Extension du cadre d'animation (AFX)

AMENDEMENT 4: Codage de maille 3D fondé sur un modèle

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

Reference number
ISO/IEC 14496-16:2011/Amd.4:2017(E)

© ISO/IEC 2017

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2017, Published in Switzerland

All rights reserved. Unless otherwise specified, 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
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

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. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

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

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).

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

For an explanation on 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 the following URL: www.iso.org/iso/foreword.html.

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 14496 series can be found on the ISO website.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/Amd4:2017

© ISO/IEC 2017 – All rights reserved

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

Information technology — Coding of audio-visual objects —

Part 16: Animation Framework eXtension (AFX)

AMENDMENT 4: Pattern-based 3D mesh coding (PB3DMC)

5.2 Geometry tools

Add the following subclauses:

5.2.7 Pattern-based 3D mesh coding (PB3DMC)

5.2.7.1 Overview

In practical applications, many 3D models consist of a large number of connected components. And these multi-connected 3D models usually contain lots of repetitive structures in various transformations, as shown in Figure Amd4.1. In order to increase their efficiency, compression methods for this kind of 3D models should be able to extract the redundancy existing in the repetitive structures.

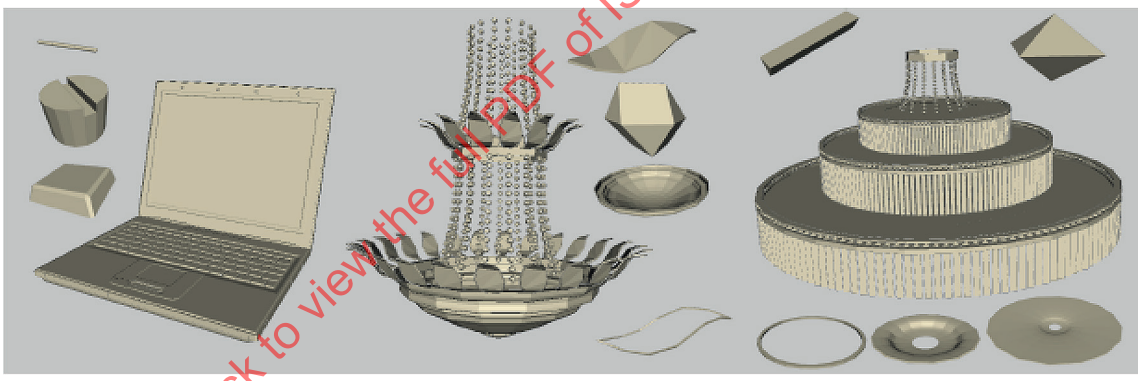


Figure Amd4.1 — 3D models with a large number of connected components and repetitive structures

This document presents an efficient compression algorithm for multi-connected 3D models by taking advantage of discovering repetitive structures in the input models. It allows discovering of structures repeating in various positions, orientations and scaling factors, where the 3D model is then organized into "pattern-instance" representation. A pattern is the representative geometry of the corresponding repetitive structure. The connected components belonging to a repetitive structure are called instances of the corresponding pattern and represented by the pattern ID and their transformation, i.e. the combination of reflection, translation, rotation and possible uniform scaling, with regards to the pattern. The instance transformation consists of four parts: reflection part, translation part, rotation part and possible scaling part.

The repetitive structure discovery-based compression algorithm proposed in this subclause, brings significant compression gain compared to the static 3D model compression algorithms provided by SC3DMC when 3D models present repetitive features. This document defines the compressed bitstream syntax and semantics for this repetitive structure discovery-based compression approach.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.2 General compression approach

5.2.7.2.1 Identification of repetitive and symmetric structures in a 3D model

This subclause defines several terms that are used all across this document and presents the different steps that are used in order to identify repetitive structures.

There are two types of repetitive structures in 3D models:

— ***Unconnected repetitive structure***: One unconnected repetitive structure consists of all the connected components which are invariant in various positions, orientations and scaling factors. One unconnected repetitive structure includes one pattern, which corresponds to one connected component in this structure, and all the other connected components are its instances.

— ***Connected repetitive structure***: One connected repetitive structure consists of all the surface patches which are invariant in various positions, orientations and scaling factors. In other words, it is a repetitive structure that may be found within one connected component. Connected repetitive structure is sometimes also called ***symmetric structure***. This document uses both naming. Similar to unconnected repetitive structure, one symmetric structure includes one pattern, which corresponds to one surface patch in this structure, and all the other surface patches are instances of the pattern.

PB3DMC discovery of the two types of repetitive structures is done in four steps as described below, considering that the input 3D model is as shown in Figure Amd4.2.

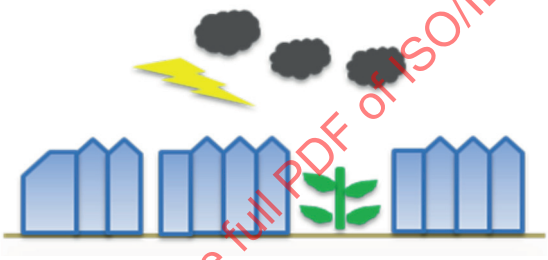


Figure Amd4.2 — Input 3D model

STEP 1: Identification of unconnected repetitive structures and unique part.

The input 3D model is divided into two parts, the unconnected repetitive structures and the unique part which includes all those components that are not included in any unconnected repetitive structures. In Figure Amd4.2, clouds, leaves and houses on the right are all unconnected repetitive structures.

STEP 2: Choose patterns and instances within unconnected repetitive structures.

Following STEP 1, for each unconnected repetitive structures, one pattern shall be chosen among all repetitions. Other repetitions then become instances of their related pattern. The input 3D model is then divided into three parts as follows:

— A: Patterns of all unconnected repetitive structures (see Figure Amd4.3);

— B: Instances of all unconnected repetitive structures (see Figure Amd4.4);

— C: Unique part which is not included in any unconnected repetitive structures (see Figure Amd4.5).

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

 © ISO/IEC 2017 – All rights reserved

Figure Amd4.3 — A: Patterns of all unconnected repetitive structures

Figure Amd4.4 — B: Instances of all unconnected repetitive structures

Figure Amd4.5 — C: Unique part which is not included in any unconnected repetitive structures

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

As shown in Figure Amd4.6, part A can be further divided into two types:

— D: Patterns of unconnected repetitive structures which do not include any symmetric structures;

— E: Patterns of unconnected repetitive structures which include symmetric structures.

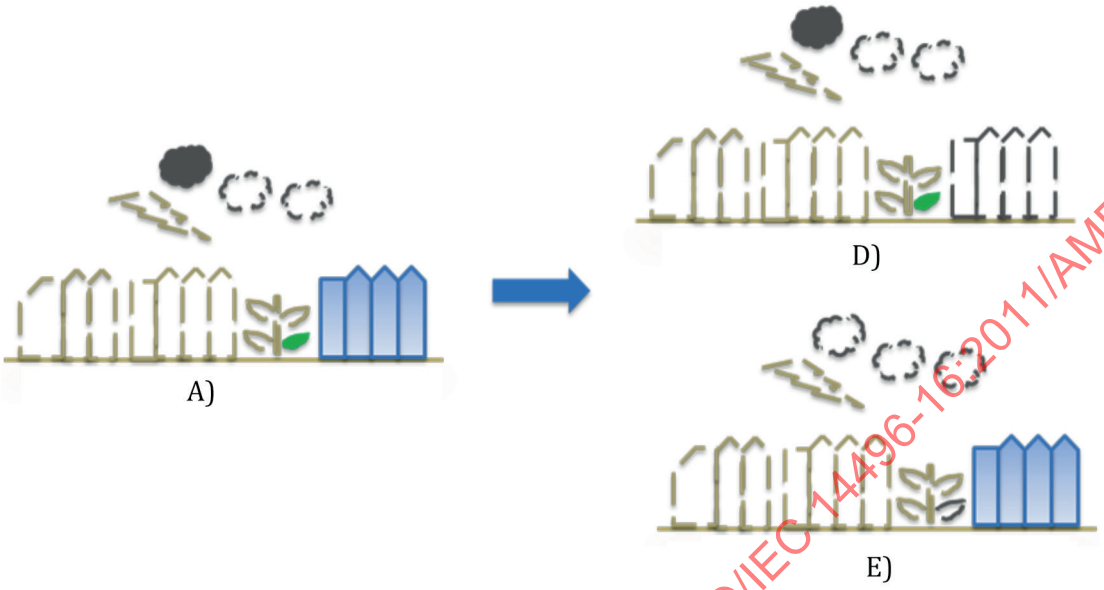


Figure Amd4.6 — Part A further divided into two types

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

As shown in Figure Amd4.7, part C can be further divided into two types:

— F: Unique part which is not included in any unconnected repetitive structures and which does not include unconnected repetitive structures and symmetric structures;

— G: Unique part which is not included in any unconnected repetitive structures and which includes symmetric structures.

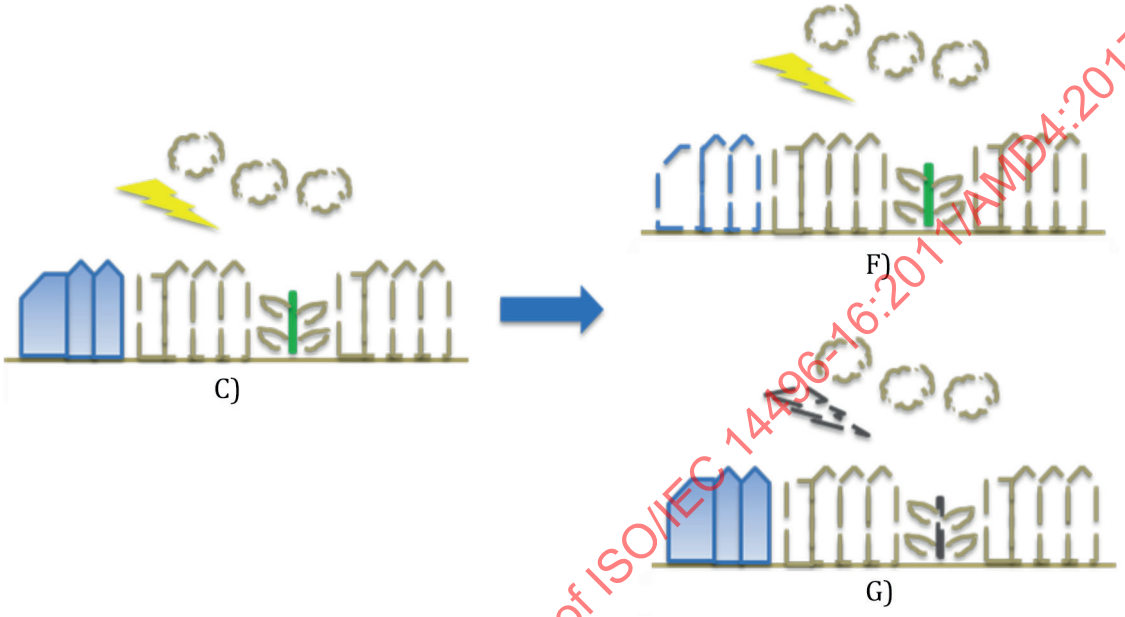


Figure Amd4.7 — Part C can be further divided into two types

STEP 3: Identification of symmetric structures.

Since instances will be coded with reference to their related patterns, symmetric structures are identified either within pattern or unique parts in the 3D model. Consequently, as shown in Figure Amd4.8, the input in STEP 3 is the input model, except instances of all unconnected repetitive structures discovered in STEP 2.

NOTE The input in STEP 3 is the input model, except instances of unconnected repetitive structure.

Figure Amd4.8 — Input of STEP 3

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

STEP 4: Choose patterns and instances within symmetric structures.

Similar to STEP 2 on repetitive structures, STEP 4 consists choosing patterns and instances within symmetric structures. As shown in Figure Amd4.9, the following four types of data are defined:

— H: Patterns of all symmetric structures;

— I: Instances of all symmetric structures;

— J: Unique parts of those unique components which do not belong to any unconnected repetitive structures but have symmetric structures;

— K: Unique parts on those unconnected-repetitive-structure patterns including symmetric structures.

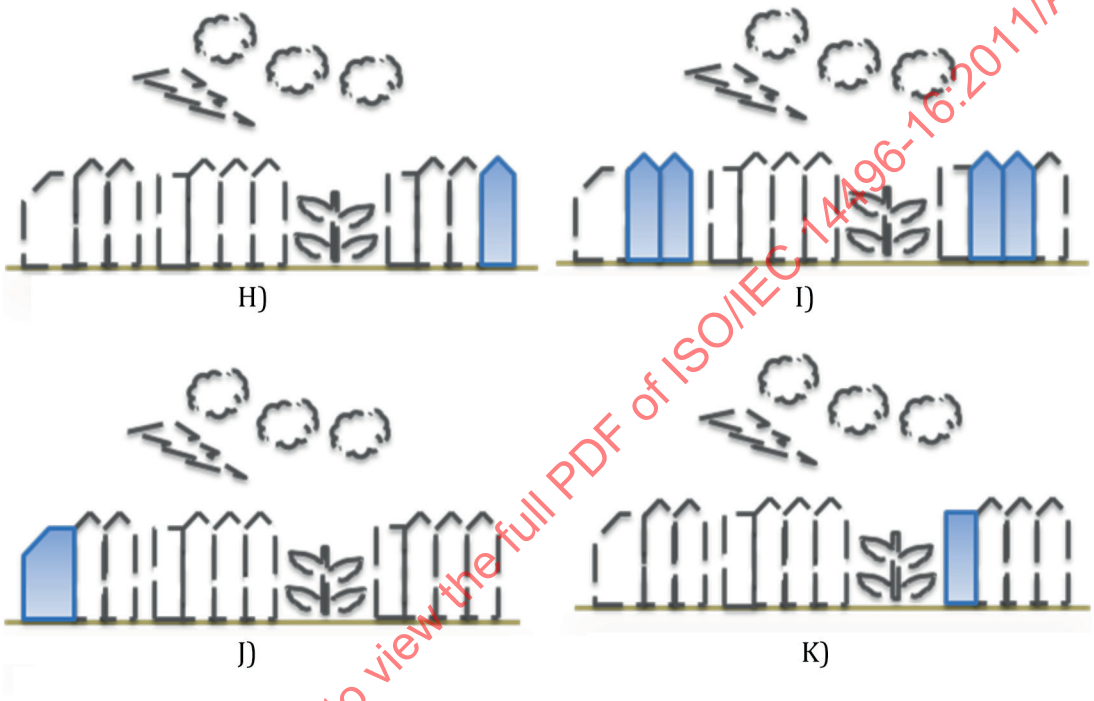


Figure Amd4.9 — Result of STEP 4

5.2.7.2.2 Two-instance reconstruction modes

While the bitstream needs to embed all the instance data, it should also be efficient and should address several applications where sometimes either bitstream size or decoding efficiency or error resilience matters the most.

Therefore, two options are proposed in reconstructing the data of one instance, i.e. its pattern ID (ID being the actual position of the patterns in the compressed bitstream of patterns: 1 for first pattern, 2 for second pattern, etc.), its reflection transformation part (F), its translation transformation part (T), its rotation transformation part (R) and its scaling transformation part (S), from the bitstream. Both of them have their own pros and cons.

Option (A) elementary instance data mode (ID, F, T, R, S, ID, F, T, R, S...): Using this mode, the pattern ID, reflection transformation part, translation transformation part, rotation transformation part and scaling transformation part of one instance are packed together in the bitstream.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

 © ISO/IEC 2017 – All rights reserved

Pros:

— It is error resilient. The decoder can recover from losing the transformation of some instances.

— Online decoding, which means that the instances can be decoded one by one during actual reading of the compressed bitstream. There is no need to wait for the reading of the whole compressed bitstream to be finished.

— Higher codec speed.

— The codec needs no buffer.

Cons:

— Relatively larger compressed 3D model size.

Option (B) grouped instance data mode (ID, ID, F, F, T, T, R, R, S, S): Using this mode, the pattern ID, reflection transformation part, translation transformation part, rotation transformation part and scaling transformation part of one instance are packed together in the bitstream.

Pros:

— Relatively smaller compressed 3D model size.

Cons:

— The decoder is no longer error resilient.

— Off-line decoding, which means the decoder can only start decoding after reading the whole compressed bitstream.

— Lower codec speed.

— Buffer is necessary.

The bitstream definition includes both of the above two options. The encoder can choose the one which fits its application better.

Since instances may have larger reconstruction error than its related patterns (error being defined as the distance between the original component and the component restored from the pattern and instance transformation), some data fields of the bitstream are defined to denote the compressed instance reconstruction error to guarantee the decoded 3D model quality. Whether or not to record the decoding error of an instance is based on the quality requirement.

5.2.7.2.3 Reconstruction of instance transformation

The instance transformation is reconstructed from four parts: reflection part, rotation part, translation part and possible scaling part.

— The reflection part is represented by a 1-bit flag.

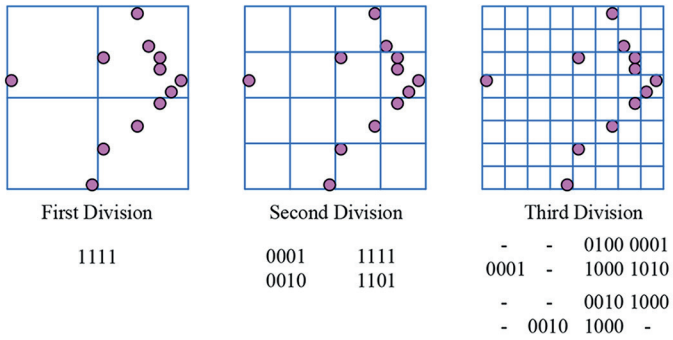
— The rotation part is reconstructed from the three Euler angles (alpha, beta, gamma).

— The translation part is represented by a vector (x, y, z) (translation vector).

— The scaling part is represented by the uniform scaling factor S of the instance.

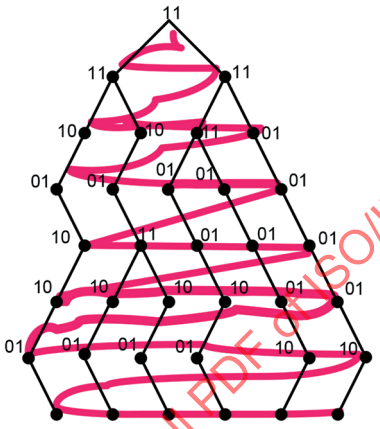
IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved



NOTE The number under each sub-figure is the corresponding occupancy codes.

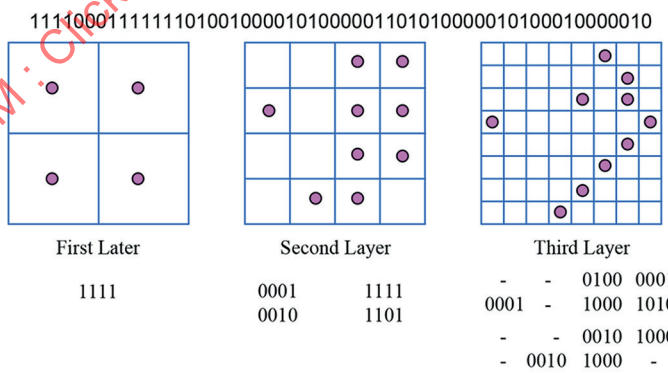
a) 2D example of space subdivision for quad-tree construction



NOTE 1 The red line illustrates the breadth first traversal of the binary tree.

NOTE 2 The occupancy codes of the tree nodes, sorted according to the breadth first traversal, construct the occupancy code sequence that describes the tree structure.

b) Example of breadth first traversal of the binary tree



NOTE For a predefined traversal order “top left–top right–bottom right–bottom left”, the input bitstream is decoded.

c) 2D example of quad-tree reconstruction process

Figure Amd4.10 — Reconstruction of instance transformation

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

 © ISO/IEC 2017 – All rights reserved

While using grouped instance transformation mode, all instance translation vectors are compressed by octree (OT) decomposition-based compression algorithm, which recursively subdivides the bounding box of all instance translation vectors in an octree data structure, as illustrated by the 2D example in Figure Amd4.10 a). Each octree node subdivision is represented by the 8-bit long occupancy code, which uses a 1-bit flag to signify whether a child node is nonempty. An occupancy code sequence describing the octree is generated by breadth first traversing the octree, as illustrated by the 2D example in Figure Amd4.10 b). To decode these instance translation vectors, the octree is reconstructed by breadth first traversing the octree from top to bottom, as shown in Figure Amd4.10 c). First, the top layer is obtained by decoding the first long occupancy code. If any nodes at Layer i have more than one "1" in its occupancy code, e.g. "01100000", the codec decodes necessary number of symbols to append as the children of such nodes. This process continues until all the leaf nodes have only one "1" or are the terminal code. The occupancy code sequence is decoded by dividing it into several intervals and decoded them with different probability models. Since instances may have extremely close translation vectors, which are called as duplicate translation vectors, some data fields of the bitstream are defined to denote these duplicate translation vectors.

5.2.7.2.4 Reconstruction of instance attributes

In practical applications, besides geometry, 3D models usually have various attributes such as normal, colour and texture coordinates. Requiring instances to have the same attributes of patterns will limit the number of repetitive structures that can be discovered and decrease the compression ratio of PB3DMC. Thus, only the geometry is checked during repetitive structure discovery and the instances may have attributes different from the corresponding pattern's attributes. There are two reconstruction modes for instance attributes.

— Share attribute mode: The instance shares the pattern attribute data and does not need data fields to represent its attributes.

— Specific attribute mode: The instance has its own attributes and need separate data fields to represent its attributes in the bitstream.

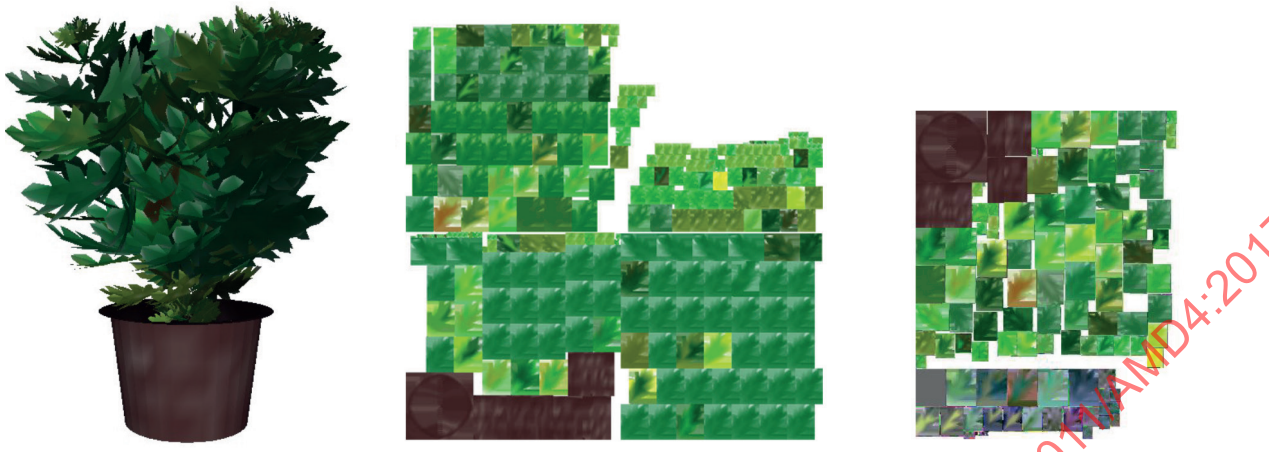
When the elementary instance data mode is used, one data field is defined to denote how to reconstruct the attributes of an instance from the bitstream. The attribute data of one instance (A) follows the other data of the instance, i.e. (ID, F, T, R, S, A, ID, F, T, R, S, A…). When the grouped instance data mode is used, all instances should either share the pattern attribute data or have their own attribute data. The instance data part of the bitstream is like (ID, ID, F, F, T, T, R, R, S, S, A, A).

5.2.7.2.5 Reconstruction of textured image(s)

It is expected that the repetitive structures in 3D models share textured image(s)/portion(s), as well as geometry, as shown in Figure Amd4.11. Given the geometric matching relationship between patterns and instances, the texture redundancy can be removed. Thus, the compression ratio can be further improved by removing the textured image redundancy with the help of geometric repetition information.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved



a) Input 3D model with texture | **b) Original textured image containing many repeated parts** | **c) Re-organized textured image after removing redundancy**

Figure Amd4.11 — Reconstruction of a textured image

Let I and P denote the instance and the corresponding pattern, respectively. Let I_d and P_d denote the reconstructed I and P, respectively. Two options for reconstructing the textured image(s)/portion(s) of I_d are proposed.

a) **Regular textured image reconstruction mode**

I_d uses the reconstructed textured image (portion) indicated by the reconstructed texture coordinates of itself or P_d. There are three possible cases that could invoke this mode in the encoder side.

— I and P use the same texture coordinates.

— The textured image contents of I and P are extremely similar, although they use different texture coordinates. In this case, the redundant portion of the textured image will be removed before compression and I_d will reuse the texture coordinates of P_d.

— I and P use the different texture coordinates and their textured image contents are different. In this case, I_d will have its own reconstructed texture coordinates.

b) **Compensated textured image reconstruction mode**

The textured image(s)/portion(s) of I_d is reconstructed from the textured image(s)/portion(s) indicated by the reconstructed texture coordinates of I_d and P_d. This mode will be invoked at the encoder side if the texture content of I and P are similar but have non-ignorable difference. In this case, the corresponding textured image(s)/portion(s) indicated by the texture coordinates of I is updated to the difference from the texture content of P. The textured image(s)/portion(s) of I_d is reconstructed by the following formula:

$$I_I(x,y) = I_I(x,y) - I_P(x,y) + 128 \quad DeTEX_{I_d}(x,y) = TEX_{I_d}(x,y) + TEX_{P_d}(x,y) - 128$$

where

$DeTEX_{I_d}(x,y)$	is the reconstructed textured image of I_d;
$TEX_{I_d}(x,y)$ and $TEX_{P_d}(x,y)$	are the textured images indicated by the reconstructed texture coordinates of I_d and P_d.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

 © ISO/IEC 2017 – All rights reserved

Using the method described above, both the texture coordinates and textured images need to be modified. The details of bitstream definition related to texture coordinate decoding is described in 5.2.7.4.11. The details of textured image decoding are out of the scope of this document.

5.2.7.2.6 Reconstruction of the unique part

Unique part refers to those components of the original model which are not repetitive. The components belonging to unique part are called as unique components. The unique components are reconstructed by translating the corresponding decoded components to the reconstructed positions of the original components.

Although the centre of those components need to be compressed and outputted to the bitstream later, the rate-distortion performance will be optimized as the bounding box of those components could be much smaller (in most cases) and less quantization bits are required for the same coding error.

Especially, if there is no or not enough repetitive structures to guarantee the bitrates saving using the pattern-instance representation, all the components of the input 3D model will be regarded as unique part. Compared with using the raw representation, there will still be bitrates saving.

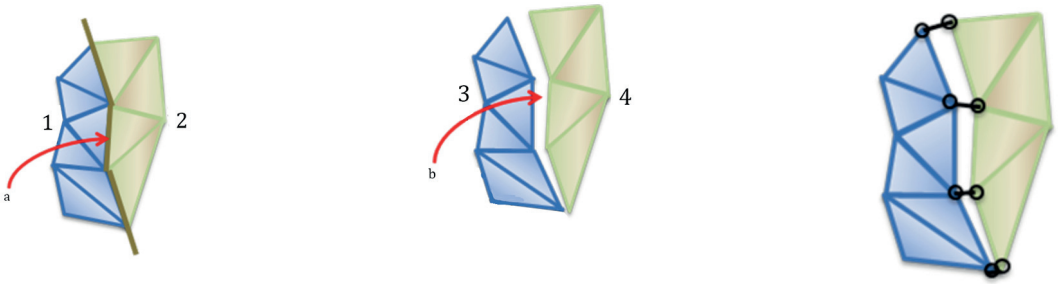
5.2.7.2.7 Reconstruction of symmetric structures

As described in 5.2.7.2.1, there might be repetitive structures among various components and/or within one component. The former is called *unconnected repetitive structures* because any instance of this kind of repetitive structures does not share boundaries with the other parts of the 3D model. The latter is called *connected repetitive structures* or *symmetric structures* because any instance of the symmetric structures shares boundaries with the other parts of the 3D model.

The boundary of an instance could be defined as consisting of vertices, triangles or other elements of the 3D model. Because of the unavoidable vertex position coding error introduced by most 3D model compression algorithms, using the same method to represent and compress symmetric structures might cause cracks on the boundaries of the decoded symmetric instances, as shown in Figures Amd4.12 a) and b). In order to avoid these cracks, stitching information should be recorded in the compressed bitstream for stitching the decoded symmetric instances and their adjacent parts in the 3D model. One example of the stitching information, as shown in Figure Amd4.12 c), is the difference between the vertex positions recovered from different instances which correspond to the same vertex on the original 3D model.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved



a) Common boundary in the input model

b) Cracks between adjacent instances in the decoded model

c) Stitching information could be the difference between vertices (black lines)

NOTE The crack on the common boundary of adjacent symmetric instances caused reconstruction during decoding and one example of the stitching information for removing the crack.

Key

1 symmetric instance, I_1

2 symmetric instance, I_2

3 decoded symmetric instance, I_1

4 decoded symmetric instance, I_2

a Common boundaries of I_1 and I_2.

b The crack caused by recovering I_1 and I_2 from different decoded patterns and transformations.

Figure Amd4.12 — Reconstruction of symmetric structures

For one unconnected repetitive structure whose pattern consists of symmetric structures, there are two options to represent its instances. Let I and P denote the instance and the corresponding pattern, respectively. Let P_S denote the pattern of the symmetric structure belonging to P. For the sake of simplicity, suppose P has no unique part when represented by the instances of P_S, then I could be represented as:

— Option A: one instance of P, which is represented by instances of P_S;

— Option B: instances of P_S.

As instances of symmetric structures might decrease the quality of the decoded 3D model and need extra stitching information, option A is chosen to limit the number of symmetric instances.

Thus, for the purpose of benefiting compression, discovering repetitive structures on the surface of a 3D model can be divided into two steps:

— first, discover the unconnected repetitive structures;

— then, discover the symmetric structures on the surfaces of the patterns of the unconnected repetitive structures and the unique part.

As a result of the two-step repetitive structure discovery, the entire original 3D model is represented by the following:

a) patterns, which consist of the patterns of unconnected repetitive structures not including any symmetric structures (D), the patterns of symmetric structures (H), the unique parts on those unconnected-repetitive-structure patterns including symmetric structures (K), and the unique parts of those unique components which do not belong to any unconnected repetitive structures but have symmetric structures (J). All the objects here are indexed according to their actual position in the bitstream;

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

 © ISO/IEC 2017 – All rights reserved

b) instances of symmetric structure patterns (I) (only one pattern per symmetric structure), which are represented by the pattern IDs (actual position in the bitstream) and transformations from the symmetric patterns to instances;

c) stitching information that is used to stitch the decoded symmetric pattern instances and their adjacent parts on the decoded 3D model, which could be other decoded symmetric pattern instances or the two types of unique parts defined in a);

d) instances of unconnected-repetitive-structure patterns (B);

e) unique components (G), which do not belong to any unconnected repetitive structures and do not include any symmetric structures;

The decoder reconstructs the original 3D model as follows.

— Reconstruct the unique components, which do not belong to any unconnected repetitive structures and do not include any symmetric structures.

  Reconstruct all patterns defined in a). All the patterns defined in a) are indexed according to their positions in the bitstream.

— Reconstruct all symmetric instances.

— Reconstruct all unconnected-repetitive-structure patterns and unique components which include symmetric structures, using the recovered patterns, symmetric instances and stitching information. Each unconnected-repetitive-structure pattern or unique component is indexed using the minimum index of its patterns defined in a).

— Reconstruct all unconnected-repetitive-structure instances.

5.2.7.2.8 Error compensation

When reconstructing an instance by applying transformation matrix on the corresponding pattern, it may result in a larger reconstruction error than when directly decoding the corresponding component. Thus, an error compensation mode may be used for compensating the reconstruction error of the vertices of those instances which suffer from large vertex reconstruction error. The reconstruction error of a vertex is defined as the distance between its original and reconstructed positions. As the amount of vertex reconstruction error to be encoded may vary drastically, different levels of quantization are used for different vertex reconstruction error according to its scale.

The information related with error compensation is recorded in three layers, i.e. the bitstream, instance and vertex layer, as shown in Figure Amd4.13.

At the bitstream header layer, an error compensation flag informs whether or not error compensation data is present in the bitstream. When this flag is set to 1 (true), a lookup table containing all three numbers of quantization bits that can be used for error compensation follows.

At the instance layer, each instance has one flag to indicate whether or not error compensation data is present for at least one of its vertices. If no error compensation data is present in the whole bitstream (as indicated by the error compensation flag in the header layer), all these instance error compensation flags are omitted.

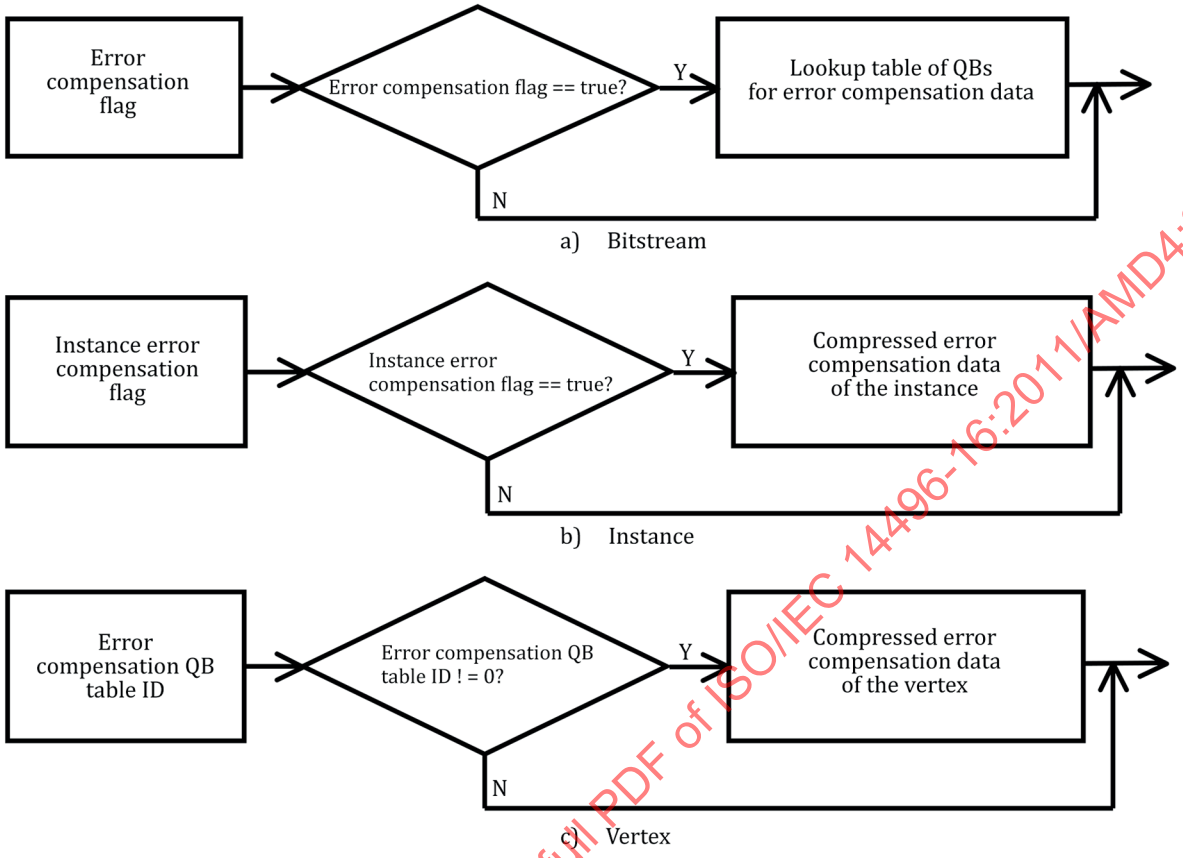
At the vertex layer, if the error compensation mode of its parent instance is activated, a quantization bits (QB) lookup table ID is present in order to indicate to the decoder the number of quantization bits used when encoding.

NOTE The choice of a 3-value QB lookup table rather than directly recording the number of quantization bits for each vertex allows saving 3 bits per vertex (2 bits for QB lookup table entry vs 5 bits for number of quantization bits). Since there are many vertex information entries in the bitstream, the introduction of such a lookup table helps saving significant bitrate.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

All vertices of an instance in which at least one vertex has error compensation data contain QB lookup table ID. The QB lookup table allows using three different levels of quantization since the first ID (0) of the lookup table is reserved for vertices that do not have compensated reconstruction error.

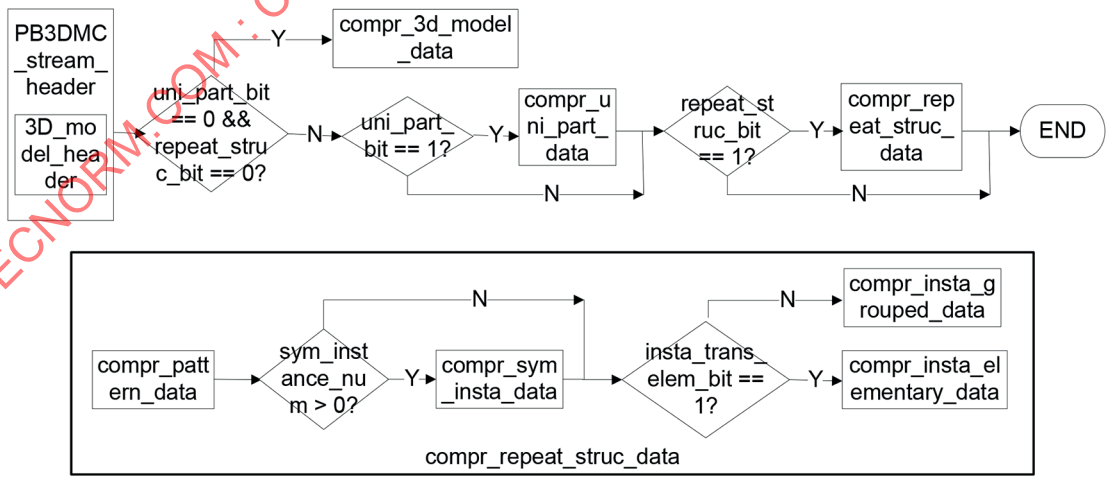


NOTE The information related with error compensation is at the a) bitstream, b) instance and c) vertex levels.

Figure Amd4.13 — Information related with error compensation

5.2.7.3 General structure of the compressed bitstream

The general structure of the compressed bitstream of PB3DMC is as shown in Figure Amd4.14.



NOTE The block diagram in the bottom part is the detail explanation of the **compr_repeat_struc_data** block in the upper part.

Figure Amd4.14 — PB3DMC compressed bitstream

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

The bitstream starts with the header buffer (**PB3DMC_stream_header**), which contains all the necessary information for decoding the compressed stream: information of whether or not there are unique part in the original model, information of whether or not there is at least one repetitive structure in the original model, information of whether or not the "*grouped instance transformation mode*" or "*elementary instance transformation mode*" is used in this bitstream, information of the original 3D model, information of the type of attributes instances may have, the 3D model compression method used for compressing geometry, connectivity and attributes of all 3D objects (patterns and other parts if necessary), etc.

If there is no unique part and repetitive structure in the original model (**uni_part_bit** == 0 && **repeat_struc_bit** == 0), the left part of the bitstream is the compressed input 3D model using the 3D model compression method indicated in **PB3DMC_stream_header**. Otherwise, the next part in the bitstream is the compressed result of all unique components if there are some. If there is at least one repetitive structure, the next data field is the compressed result of all patterns, which consist of the patterns of unconnected repetitive structures not including any symmetric structures, the patterns of symmetric structures, the unique parts on those unconnected-repetitive-structure patterns including symmetric structures, and the unique parts of those unique components which do not belong to any unconnected repetitive structures but have symmetric structures. If there is at least one symmetric structure, the next data field is the compressed transformation of all instances of symmetric structure patterns. Depending on which instance transformation packing mode is chosen in this bitstream, either **compr_insta_grouped_data** or **compr_insta_elementary_data** is the next part in the bitstream, which contains the compressed transformation of all instances of un-connected repetitive structures.

5.2.7.4 Bitstream syntax and semantics

5.2.7.4.1 Conventions

The mathematical operators used to describe this specification for repeated structure discovery-based compression algorithm are similar to those used in the C programming language. However, integer divisions with truncation and rounding are specifically defined. Numbering and counting loops generally begin from zero.

In addition to the syntax functions, categories and descriptors already used in SC3DMC specification, the following two functions are used:

f(n): fixed-length coded bit string using n bits (written from left to right) for each symbol. n depends on the code length for each symbol.

ec(v): entropy-coded (e.g. arithmetic coded) syntax element, including possibly configuration symbols.

5.2.7.4.2 PB3DMC_stream class

5.2.7.4.2.1 Syntax

class **PB3DMC_stream**{		
PB3DMC_stream_header		
PB3DMC_stream_data		
}		

5.2.7.4.2.2 Semantics

PB3DMC_stream_header: This data field contains the header buffer.

PB3DMC_stream_data: This data field contains the data buffer.

In the following definition, "symmetric instance" refers to the instance of symmetric structure patterns and "instance" refers to the instance of unconnected repetitive structures. Patterns consist of the patterns of unconnected repetitive structures not including any symmetric structures, the patterns of symmetric structures, the unique parts on those unconnected-repetitive-structure patterns including

IECNORM.COM: Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

symmetric structures, and the unique parts of those unique components which do not belong to any unconnected repetitive structures but have symmetric structures.

5.2.7.4.3 PB3DMC_stream_header class

5.2.7.4.3.1 Syntax

	Num. of bits	Descriptor
class **PB3DMC_stream_header**{		
version_no	4	
uni_part_bit	1	
repeat_struc_bit	1	
if(repeat_struc_bit == 1){		
pattern_num	8	
if(pattern_num == 255){		
pattern_num_2	16	
}		
sym_instance_num	16	
instance_num	16	
if(instance_num == 65535){		
instance_num_2	32	
}		
insta_trans_elem_bit	1	
use_scaling_bit	1	
}		
3d_model_compr_mode	2	
3d_model_header		
if(repeat_struc_bit == 1){		
QP_translation	5	
QP_rotation	5	
error_compen_enable_bit	1	
if(error_compen_enable_bit == 1){		
error_compen_QB_table[0...2]	3 × 5	
}		
}		
reserved_bits		For byte alignment
}		

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.3.2 Semantics

version_no: This 4-bit unsigned integer indicates the version number of bitstream specification for repetition discovery-based 3D model codec (PB3DMC). Bitstream version 0 includes the structures repeating in various positions, orientations and scaling factors, as defined in [w13178].

version_no	Meaning
0	The structures repeating in various positions, orientations and scaling factors are included in the bitstream [w13178].
1~15	ISO reserved

uni_part_bit: This 1-bit unsigned integer indicates whether there is unique part, which does not belong to any unconnected repetitive structures and does not include any symmetric structures, in the 3D model. 0 means there is no unique part and 1 means there is unique part. Note that for those multi-connected 3D model without any repetitive structure, the entire input 3D model is regarded as unique part.

repeat_struc_bit: This 1-bit unsigned integer indicates whether or not there is at least one repetitive structure in the 3D model. 0 for no repetitive structure and 1 for repetitive structure.

uni_part_bit = 0 && repeat_struc_bit = 0 means the 3D model contains only one connected component and the original 3D model can be directly reconstructed from the left part of the bitstream.

pattern_num: This 8-bit unsigned integer indicates the number of all patterns if it is less than 255. The minimum value of pattern_num is 1.

pattern_num_2: This 16-bit unsigned integer indicates the number of all patterns if it is not less than 255. In this case, the total pattern number is (pattern_num_2 + 255)

sym_instance_num: This 16-bit unsigned integer indicates the number of all symmetric instances

instance_num: This 16-bit unsigned integer indicates the number of all instances if it is less than 65535. The minimum value of instance_num is 1.

instance_num_2: This 32-bit unsigned integer indicates the number of all instances if it is not less than 65535. In this case, the total instance number is (instance_num_2 + 65535)

insta_trans_elem_bit: This 1-bit unsigned integer indicates whether "grouped instance transformation mode" or "elementary instance transformation mode" is used in this bitstream. 0 for "grouped instance transformation mode" and 1 for "elementary instance transformation mode".

use_scaling_bit: This 1-bit unsigned integer indicates whether instance transformation includes scaling factors, 1 for scaling factors being included in instance transformation and 0 for not. When the scaling factors of most instances equal 1, the instance transformation does not include the scaling factor. That means, all instances shall have the same size with the corresponding patterns.

3d_model_compr_mode: This 2-bit unsigned integer indicates the 3D model decoding method used to reconstruct patterns, unique part and the original 3D model itself if it includes no repetitive structures.

3d_model_compr_mode	Meaning
0	SC3DMC
1	3DMC Extension
2 and 3	ISO reserved

3d_model_header: This data field contains the 3D model header buffer.

QP_translation: This 5-bit unsigned integer indicates the quality parameter of instance translation. The minimum value of QP_Translation is 3 and maximum is 31. This is the global quality parameter for instance translation. It can be over written by the quality parameter for the translation of each individual instance. See 4.8.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

QP_rotation: This 5-bit unsigned integer indicates the quality parameter of instance rotation. The minimum value of QP_Rotation is 3 and maximum is 31. This is the global quality parameter for instance rotation. It can be over written by the quality parameter for the rotation of each individual instance. See 4.8.

error_compen_enable_bit: This 1-bit unsigned integer indicates whether or not there are data fields of compressed coding error compensation data for some instances in the bitstream. 0 means there is no data field of compressed coding error compensation data of instances in the bitstream and 1 means there is at least one data field of compressed coding error compensation data for at least one instance in the bitstream.

error_compen_QB_table: This data field contains three predefined numbers of quantization bits for compensated reconstruction error, each of which is represented by one 5-bit unsigned integer.

5.2.7.4.4 3d_model_header class

5.2.7.4.4.1 Syntax

	Num. of bits	Descriptor
class **3d_model_header**{		
ver_num	32	
tri_num	32	
default_coord_bbox	1	
if(default_coord_bbox == '0'){		
coord_bbox	6 × 32	
}		
QP_coord	5	
normal_binding		
if(normal_binding != 'not_found'){		
default_normal_bbox	1	
QP_normal	5	
}		
color_binding		
if(color_binding != 'not_found'){		
default_color_bbox	1	
if(defalut_color_bbox == '0'){		
color_bbox	6 × 32	
}		
QP_color	5	
}		
multi_texCoord_num	5	
if(multi_texCoord_num != 0){		
for (i = 0; i < multi_texCoord_num; i++){		
texCoord_binding		
default_texCoord_bbox	1	
if(default_texCoord_bbox == '0'){		
texCoord_bbox	4 × 32	
}		
QP_texCoord	5	
}		

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

	Num. of bits	Descriptor
}		
multi_attribute_num	5	
if(multi_attribut_num != 0){		
for (i = 0; i < multi_attribute_num; i++){		
attribute_dim_num		
attribute_binding		
default_attribute_bbox	1	
if(default_attribute_bbox == '0'){		
attribute_bbox	2 × attribute_dim_num × 32	
}		
QP_attribute	5	
}		
}		
}		

5.2.7.4.4.2 Semantics

ver_num: This 32-bit unsigned integer contains the number of vertices of the entire 3D model. This value can be used to verify the decoded 3D model.

tri_num: This 32-bit unsigned integer contains the number of triangles of the entire 3D model. This value can be used to verify the decoded 3D model.

default_coord_bbox: This 1-bit unsigned integer indicates whether a default bounding box is used for the entire 3D model's geometry. 0 means using another bounding box and 1 means using the default bounding box. The default bounding box is defined as $x_{min} = 0$, $y_{min} = 0$, $z_{min} = 0$, $x_{max} = 1$, $y_{max} = 1$ and $z_{max} = 1$.

coord_bbox: This data field contains the bounding box of the entire 3D model's geometry. The geometry bounding box is defined by (x_{min}, y_{min}, z_{min}, x_{max}, y_{max}, z_{max}).

QP_coord: This 5-bit unsigned integer indicates the quality parameter of the 3D model geometry. The minimum value of QP_coord is 3 and maximum is 31. If there is at least one repetitive structure, QP_coord is used as the number of quantization bits for the geometry of patterns.

normal_binding: This 2-bit unsigned integer indicates the binding of normals to the 3D model. The admissible values are described below.

normal_binding	Binding
0	not_bound
1	bound_per_vertex
2	bound_per_face
3	bound_per_corner

default_normal_bbox: This 1-bit unsigned integer should always be '0', which indicates that a default bounding box is used for the normal of the entire 3D model. The default bounding box of normal is defined as $nx_{min} = 0$, $ny_{min} = 0$, $nz_{min} = 0$, $nx_{max} = 1$, $ny_{max} = 1$ and $nz_{max} = 1$.

QP_normal: This 5-bit unsigned integer indicates the quality parameter of the 3D model normal. The minimum value of QP_normal is 3 and maximum is 31. If there is at least one repetitive structure, QP_normal is used as the number of quantization bits for the normal of patterns.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

color_binding: This 2-bit unsigned integer indicates the binding of colours to the 3D model. The following table shows the admissible values.

color_binding	Binding
0	not_bound
1	bound_per_vertex
2	bound_per_face
3	bound_per_corner

default_color_bbox: This 1-bit unsigned integer indicates whether a default bounding box is used for the colour of the entire 3D model. 0 means using another bounding box and 1 means using the default bounding box. The default bounding box is defined as $r_{min} = 0$, $g_{min} = 0$, $b_{min} = 0$, $r_{max} = 1$, $g_{max} = 1$ and $b_{max} = 1$.

color_bbox: This data field contains the bounding box of the colour of the entire 3D model. The colour bounding box is defined by (r_{min}, g_{min}, b_{min}, r_{max}, g_{max}, b_{max}).

QP_color: This 5-bit unsigned integer indicates the quality parameter of the colour. The minimum value of QP_color is 3 and maximum is 31. If there is at least one repetitive structure, QP_color is used as the number of quantization bits for the colour of patterns.

multi_texCoord_num: This 5-bit unsigned integer gives the number of texture coordinates per vertex/corner.

texCoord_binding: This 2-bit unsigned integer indicates the binding of texture coordinates to the 3D model. The following table shows the admissible values.

texCoord_binding	Binding
0	forbidden
1	bound_per_vertex
2	forbidden
3	bound_per_corner

default_texCoord_bbox: This 1-bit unsigned integer indicates whether a default bounding box is used for the texture coordinates. 0 means using another bounding box and 1 means using the default bounding box. The default bounding box is defined as $u_{min} = 0$, $v_{min} = 0$, $u_{max} = 1$ and $v_{max} = 1$.

texCoord_bbox: This data field contains the bounding box of the texture coordinate of the entire 3D model. The texture coordinate bounding box is defined by (u_{min}, v_{min}, u_{max}, v_{max}).

QP_texCoord: This 5-bit unsigned integer indicates the quality parameter of texture coordinates. The minimum value of QP_texCoord is 3 and maximum is 31. If there is at least one repetitive structure, QP_texCoord is used as the number of quantization bits for the texture coordinates of patterns.

multi_attribute_num: This 5-bit unsigned integer indicates the number of attributes per vertex/face/corner.

attribute_binding: This 2-bit unsigned integer indicates the binding of attributes to the 3D model. The following table shows the admissible values.

attribute_binding	Binding
0	forbidden
1	bound_per_vertex
2	bound_per_face
3	bound_per_corner

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

default_attribute_bbox: This 1-bit unsigned integer indicates whether a default bounding box is used for the attributes. 0 means using another bounding box and 1 means using the default bounding box. The default bounding box is defined as attribute_min[1..attribute_dim] = 0, attribute_max[1..attribute_dim] = 1.

attribute_bbox: This data field contains the bounding box of the attribute. The attribute bounding box is defined by (attribute_min[1..attribute_dim], attribute_max[1..attribute_dim]).

QP_attribute: This 5-bit unsigned integer indicates the quality parameter of the attribute. The minimum value of QP_attribute is 3 and maximum is 31. If there is at least one repetitive structure, QP_attribute is used as the number of quantization bits for the attribute of patterns.

5.2.7.4.5 PB3DMC_stream_data class

5.2.7.4.5.1 Syntax

class **PB3DMC_stream_data**{		
if(uni_part_bit == 0 && repeat_struc_bit == 0){		
compr_3d_model_data		
}		
else{		
if(uni_part_bit == 1){		
compr_uni_part_data		
}		
if(repeat_struc_bit == 1){		
compr_repeat_struc_data		
}		
}		
}		

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.5.2 Semantics

compr_3d_model_data: This data field contains the compressed 3D model, which has only one connected component and could be reconstructed by the decoding method indicated by 3d_model_compr_mode.

compr_uni_part_data: This data field contains the compressed unique part data, which is defined as those components not belonging to any connected repetitive structures and not including any symmetric structures.

compr_repeat_struc_data: This data field contains the compressed repetitive structure data.

5.2.7.4.6 compr_uni_part_data class

5.2.7.4.6.1 Syntax

	Num. of bits	Descriptor
class **compr_uni_part_data**{		
compr_uni_comp_data		
compr_uni_comp_transl	bit_num_uni_comp_transl()	f(bit_num_uni_comp_transl())
}		

5.2.7.4.6.2 Semantics

compr_uni_comp_data: This data field contains the compressed geometry, connectivity and properties of all unique components, which are used to reconstruct all unique components by the decoding method indicated by 3d_model_compr_mode. All reconstructed unique components are translated to the positions decoded from compr_uni_comp_transl.

compr_uni_comp_transl: This data field contains the compressed translation vectors for all unique components. The unique component translation vectors are decoded by first de-quantization and then entropy decoding. This data field uses the same order of unique component with compr_uni_comp_data.

bit_num_uni_comp_transl(): This function computes the number of bits used for quantizing each unique component translation vector based on QP_coord.

5.2.7.4.7 compr_repeat_struc_data class

5.2.7.4.7.1 Syntax

	Num. of bits	Descriptor
class **compr_repeat_struc_data**{		
compr_pattern_data		
if(sym_instance_num > 0){		
compr_sym_insta_data		
compr_stitch_data		
}		
compr_pattern_transl	bit_num_pattern_transl()	f(bit_num_pattern_transl())
if(insta_trans_elem_bit == 1){		
compr_insta_elementary_data		
}		
else{		
compr_insta_grouped_data		
}		
}		

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.7.2 Semantics

compr_pattern_data: This data field contains the compressed geometry, connectivity and attributes of all patterns, which is used to reconstruct all patterns by the decoding method indicated by 3d_model_compr_mode. Here patterns consist of the patterns of unconnected repetitive structures not including any symmetric structures, the patterns of symmetric structures, the unique parts on those unconnected-repetitive-structure patterns including symmetric structures, and the unique parts of those unique components which do not belong to any unconnected repetitive structures but have symmetric structures. All the reconstructed patterns will be translated to the positions reconstructed from compr_pattern_transl.

compr_sym_insta_data: This data field contains the compressed information of the symmetric instances.

compr_stitch_data: This data field contains the compressed data for stitching symmetric instances and their adjacent parts on the decoded 3D model.

compr_pattern_transl: This data field contains the compressed translation vectors for those components corresponding to the patterns of un-connected repetitive structures. The pattern translation vectors are reconstructed by first de-quantization and then entropy decoding. This data field uses the same order of patterns as compr_pattern_data.

compr_insta_elementary_data: This data field contains the compressed transformation data for all instances using the "elementary instance transformation mode". It is compressed in a manner that is byte aligned.

compr_insta_grouped_data: This data field contains the compressed transformation data for all instances using the "grouped instance transformation mode". It is compressed in a manner that is byte aligned.

bit_num_pattern_transl(): This function computes the number of bits used for quantizing translation vector of each pattern component based on QP_coord.

5.2.7.4.8 compr_sym_insta_data class

5.2.7.4.8.1 Syntax

class **compr_sym_insta_data**{		
if(insta_trans_elem_bit != 1){		
compr_sym_insta_elementary_data		
}		
else{		
compr_sym_insta_grouped_data		
}		
}		

5.2.7.4.8.2 Semantics

compr_sym_insta_elementary_data: This data field contains the compressed transformation data for all symmetric instances using the "elementary instance transformation mode". It is compressed in a manner that is byte aligned. The detail definition of compr_sym_insta_elementary_data is the same with compr_insta_elementary_data.

compr_sym_insta_grouped_data: This data field contains the compressed transformation data for all symmetric instances using the "grouped instance transformation mode". It is compressed in a manner that is byte aligned. The detail definition of compr_sym_insta_grouped_data is the same with compr_insta_grouped_data.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.9 compr_insta_elementary_data class

5.2.7.4.9.1 Syntax

	Num. of bits	Descriptor
class **compr_insta_elementary_data**{		
insta_transl_bbox	6 × 32	
for (i = 0; i < numofInstance; i ++){		
elem_insta_QP_translation_flag	1	
elem_insta_QP_rotation_flag	1	
if(elem_insta_QP_translation_flag = 1){		
elem_insta_QP_translation	5	
}		
if(elem_insta_QP_rotation_floag == 1){		
elem_insta_QP_rotation	5	
}		
compr_elem_insta_patternID		ec(v)
elem_insta_flip_flag	1	
elem_insta_reflection_flag	1	
elem_insta_attribute_header		
compr_elem_insta_transl	bit_num_insta_transl()	f(bit_num_insta_transl())
compr_elem_insta_rotat_spherical		
if(use_scaling_bit){		
compr_elem_insta_scaling		ec(v)
}		
if(error_compen_enable_bit == 1){		
elem_insta_error_compen_flag	1	
if(elem_insta_error_compen_flag == 1){		
compr_elem_insta_error_compen_data		
}		
}		
compr_elem_insta_attribute_data		
}		
}		

5.2.7.4.9.2 Semantics

insta_transl_bbox: This data field contains the bounding box of all translation vectors. The bounding box is defined by insta_transl_x_{min}, insta_transl_y_{min}, insta_transl_z_{min}, insta_transl_x_{max}, insta_transl_y_{max}, insta_transl_z_{max}.

elem_insta_QP_translation_flag: This 1-bit unsigned integer indicates whether or not the i^{th} instance has its own quality parameter for translation.

elem_insta_QP_rotation_flag: This 1-bit unsigned integer indicates whether or not the i^{th} instance has its own quality parameter for rotation.

elem_insta_QP_translation: This 5-bit unsigned integer indicates the quality parameter of the translation vector of i^{th} instance. The minimum value of elem_insta_QP_translation is 3 and maximum is 31.

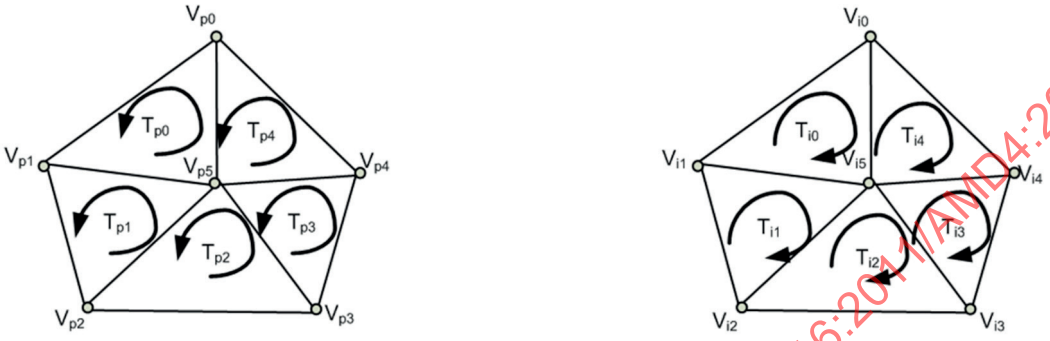
elem_insta_QP_rotation: This 5-bit unsigned integer indicates the quality parameter of the rotation angles of i^{th} instance. The minimum value of elem_insta_QP_rotation is 3 and maximum is 31.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

 © ISO/IEC 2017 – All rights reserved

compr_elem_insta_patternID: This data field contains the compressed pattern ID of i[th] instance.

elem_insta_filp_flag: This 1-bit unsigned integer indicates whether or not the i[th] instance is flipped compared with the corresponding pattern. A flipped instance means the instance triangle normals are in the opposite direction of the corresponding pattern triangles. 0 means the i[th] instance is not flipped and 1 means the i[th] instance is flipped.



a) Pattern **b) Instance**

Figure Amd4.15 — Example of flipped instance with regards to original pattern

elem_insta_reflection_flag: This 1-bit unsigned integer indicates whether the transformation of i[th] instance includes reflection transformation along the coordinate axes. 0 means the transformation of i[th] instance does not include reflection and 1 means the transformation of i[th] instance includes reflection.

elem_insta_attribute_header: This data field contains the attribute header of i[th] instance.

compr_elem_insta_transl: This data field contains the compressed translation vector of i[th] instance.

compr_elem_insta_rotat_spherical: This data field contains the compressed rotation transformation of i[th] instance in spherical mode.

compr_elem_insta_scaling: This data field contains the compressed scaling factor of i[th] instance.

elem_insta_error_compensate_flag: This 1-bit unsigned integer indicates whether or not the next part of the bitstream is the compressed coding error compensation data of i[th] instance. 0 means the next part of the bitstream is not the compressed coding error compensation data of i[th] instance and 1 means the next part of the bitstream is the compressed coding error compensation data of i[th] instance

compr_elem_insta_error_compen_data: This data field contains the compressed coding error compensation data of i[th] instance.

compr_elem_insta_attribute_data: This data field contains the compressed attribute data of i[th] instance.

bit_num_insta_transl(): This function computes the number of bits used for quantizing instance translation vectors based on QP_coord.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.10 compr_elem_insta_error_compen_data class

5.2.7.4.10.1 Syntax

	Num. of bits	Descriptor
class **compr_elem_insta_error_compen_data**{		
for(i = 0; i < numOfVertex; i ++){		
elem_compen_err_QB_id	2	
if(elem_compen_err_QB_id != 0){		
compr_ver_compen_err_data		
}		
}		
}		

5.2.7.4.10.2 Semantics

elem_compen_err_QB_id: This 2-bit unsigned integer indicates the index of the number of quantization bits for the j^{th} vertex of the instance in error_compen_QB_table. A 0 value means there is no error compensation for this vertex. A value of 1 refers to first entry in the error_compen_QB_table (2 for second and 3 for third).

compr_ver_compen_err_data: This data field contains the compressed compensated error of the j^{th} vertex of the instance, which is the quantized x, y and z value of its reconstruction error in this order.

5.2.7.4.11 elem_insta_attribute_header class

5.2.7.4.11.1 Syntax

	Num. of bits	Descriptor
class **elem_insta_attribute_header**{		
if(has_valid_attribute()){		
elem_insta_share_pattern_attribute_bit	1	
}		
if(has_available_attribute()) && elem_insta_share_pattern_attribute_bit == 0){		
if(normal_binding != 'not_found'){		
elem_insta_normal_compr_mode	2	
}		
if(color_binding != 'not_found'){		
elem_insta_color_compr_mode	2	
}		
if(multi_texCoord_num != 0){		
for (i = 0; i < multi_texCoord_num; i++){		
elem_insta_texCoord_compr_mode	2	
}		
}		
if(multi_attribute_num != 0){		
for (i = 0; i < multi_attribute_num; i++){		
elem_insta_attribute_compr_mode	2	
}		
}		

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.11.2 Semantics

elem_insta_share_pattern_attribute_bit: This 1-bit unsigned integer indicates whether or not the instance share all attributes with the corresponding pattern. 0 means the instance does not share all attributes with the corresponding pattern and all or parts of its attributes needs to be compressed. One means the instance shares all attributes with the corresponding pattern.

elem_insta_normal_compr_mode: This 2-bit unsigned integer indicates the reconstruction mode of normal data of the instance. The following table shows its admissible values.

elem_insta_normal_compr_mode	Mode	Meaning
0	share	The instance shares normal data with the corresponding pattern.
1	no_pred	The instance has its specific normal data which is reconstructed by de-quantization and entropy decoding, without prediction.
2	ISO reserved	
3	ISO reserved	

elem_insta_color_compr_mode: This 2-bit unsigned integer indicates the reconstruction mode of colour data of the instance. The following table shows its admissible values.

elem_insta_color_compr_mode	Mode	Meaning
0	share	The instance shares colour data with the corresponding pattern.
1	no_pred	The instance has its specific colour data which is reconstructed by de-quantization and entropy decoding, without prediction.
2	ISO reserved	
3	ISO reserved	

elem_insta_texCoord_compr_mode: This 2-bit unsigned integer indicates the reconstruction mode of textured image and texture coordinates of the instance. The lower bit indicates the reconstruction mode of the textured image of the instance. 0 means regular mode and 1 means compensated mode. The higher bit indicates the reconstruction mode of the texture coordinates of the instance. 0 means share mode and 1 means specific mode. The following table shows its admissible values.

elem_insta_texCoord_compr_mode	Mode	Meaning
0	share	The instance shares texture with the corresponding pattern.
1	ISO reserved	
2	no_pred	The instance has its specific texture coordinate data which is reconstructed by de-quantization and entropy decoding, without prediction. The texture of the reconstructed instance is to be obtained by decoding the texture coordinates from the bitstream and mapping the textured image portion indicated by the decoded texture coordinates.
3	texture_residual	The instance has its specific texture coordinate data which is reconstructed by de-quantization and entropy decoding, without prediction. The texture of the reconstructed instance is to be reconstructed by compensated mode using the texture of the corresponding pattern and the texture indicated by the reconstructed texture coordinates.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

elem_insta_attribute_compr_mode: This 2-bit unsigned integer indicates the reconstruction mode of attribute data of the instance. The following table shows its admissible values.

elem_insta_attribute_compr_mode	Mode	Meaning
0	share	The instance shares attribute data with the corresponding pattern.
1	no_pred	The instance has its specific attribute data which is reconstructed by de-quantization and entropy decoding, without prediction.
2	ISO reserved	
3	ISO reserved	

5.2.7.4.12 has_available_attribute() function

5.2.7.4.12.1 Syntax

```
bool has_available_attribute(){
  if(normal_binding != 'not_found' || color_binding != 'not_found || multi_texCoord_num != 0 || multi_attrib-
ute_num != 0
    return true;
  else
    return false;
}
```

5.2.7.4.12.2 Semantics

This function decides whether or not there is some attribute data needs to be reconstructed.

5.2.7.4.13 compr_elem_insta_rotat_spherical class

5.2.7.4.13.1 Syntax

	Num. of bits	Descriptor
class **compr_elem_insta_rotat_spherical**{		
compr_elem_insta_rotat_alpha	bit_num_rotat_alpha()	f(bit_num_rotat_alpha())
compr_elem_insta_rotat_beta	bit_num_rotat_beta()	f(bit_num_rotat_beta())
compr_elem_insta_rotat_gamma	bit_num_rotat_gamma()	f(bit_num_rotat_gamma()
}		

5.2.7.4.13.2 Semantics

The rotation of i^{th} instance in spherical mode is represented by 3 angles, alpha, beta and gamma.

compr_elem_insta_rotat_alpha: This data field contains the compressed alpha of i^{th} instance's rotation.

compr_elem_insta_rotat_beta: This data field contains the compressed beta of i^{th} instance's rotation.

compr_elem_insta_rotat_gamma: This data field contains the compressed gamma of i^{th} instance's rotation.

bit_num_rotat_alpha(): This function adaptively computes the number of bits for the alpha value of i^{th} instance's rotation based on QP_coord and the scale of the corresponding pattern.

bit_num_rotat_beta(): This function computes the number of bits for the beta value of i^{th} instance's rotation based on QP_coord and the scale of the corresponding pattern.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

bit_num_rotat_gamma(): This function computes the number of bits for the gamma value of i^{th} instance's rotation based on QP_coord and the scale of the corresponding pattern.

5.2.7.4.14 compr_elem_insta_attribute_data class

5.2.7.4.14.1 Syntax

class **compr_elem_insta_attribute_data**{		
if(has_valid_attribute()) && elem_insta_share_pattern_attribute_ bit == 0){		
if(normal_binding != 'not_found' && elem_insta_normal_compr_ mode != 'share'){		
compr_elem_insta_normal_data		
}		
if(color_binding != 'not_found' && elem_insta_color_compr_ mode != 'share'){		
compr_elem_insta_color_data		
}		
if(multi_texCoord_num != 0){		
for (i = 0; i < multi_texCoord_num; i++){		
if(elem_insta_texCoord_compr_mode[i]!= 'share'{		
compr_elem_insta_texCoord_data		
}		
}		
}		
if(multi_attribute_num != 0){		
for (i = 0; i < multi_attribute_num; i++){		
if(elem_insta_attribute_compr_mode[i]!= 'share'{		
compr_elem_insta_attribute_data		
}		
}		
}		
}		
}		

5.2.7.4.14.2 Semantics

compr_elem_insta_normal_data: This data field contains the compressed normal of i^{th} instance.

compr_elem_insta_color_data: This data field contains the compressed colour of i^{th} instance.

compr_elem_insta_texCoord_data: This data field contains the compressed texture coordinates of i^{th} instance.

compr_elem_insta_attribute_data: This data field contains the compressed attribute data of i^{th} instance.

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.15 compr_insta_grouped_data class

5.2.7.4.15.1 Syntax

	Num. of bits	Descriptor
class **compr_insta_grouped_data**{		
elem_insta_QP_translation_flag[1..numofInstance]	Number of instances	
elem_insta_QP_rotation_flag[1..numofInstance]	Number of instances	
for (i = 0; i < numofInstance; i ++){		
if(elem_insta_QP_translation_flag[i] == 1){		
elem_insta_QP_translation[i]	5	
}		
}		
for (i = 0; i < numofInstance; i ++){		
if(elem_insta_QP_rotation_flag[i] == 1){		
elem_insta_QP_rotation[i]	5	
}		
}		
reserved_bits		For byte alignment
compr_insta_patternID_header	16	
compr_insta_patternID_data		ec(v)
insta_flip_flag_data	Number of instances	
insta_reflection_flag_data	1 × Number of instances	
compr_insta_transl_header	16	
compr_insta_transl_data		
compr_insta_rotat_header	16	
compr_insta_rotat_data		
if(use_scaling_bit){		
compr_insta_scaling_header	16	
compr_insta_scaling_data		ec(v)
}		
if(error_compen_enable_bit == 1){		
elem_insta_error_compen_flag[1..numofInstance]	Number of instances	
for (i = 0; i < numofInstance; i ++){		
if(elem_insta_error_compen_flag[i] == 1){		
compr_elem_insta_error_compen_data		
}		
}		
}		
if(has_valid_attribute()){		
compr_insta_attribute_header		ec(v)
compr_insta_attribute_data		
}		
}		

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved

5.2.7.4.15.2 Semantics

All the following data fields use the same instance order.

compr_insta_patternID_header: A 16-bit header for the compressed pattern IDs of all instances. This data field is unused when using fixed-length codec or entropy codec which can determine compressed bitstream length automatically for coding patternID_data.

compr_insta_patternID_data: This data field contains the compressed pattern IDs of all instances.

insta_flip_flag_data: This data field contains the flip flags of all instances. It is compressed in a manner that is byte aligned.

insta_reflection_flag_data: This data field contains the reflection flags of all instances. It is compressed in a manner that is byte aligned.

compr_insta_transl_header: A 16-bit header for the compressed translation vectors of all instances. This data field is unused when using fixed-length codec or entropy codec which can determine compressed bitstream length automatically for coding transl_data.

compr_insta_transl_data: This data field contains the compressed translation vectors of all instances.

compr_insta_rotat_header: A 16-bit header for the compressed rotation transformation parts of all instances. This data field is unused when using fixed-length codec or entropy codec which can determine compressed bitstream length automatically for coding rotat_data.

compr_insta_rotat_data: This data field contains the compressed rotation transformation parts of all instances. It is compressed in a manner that is byte aligned. See full description in 5.2.7.4.17.

compr_insta_scaling_header: A 16-bit header for the compressed scaling factors of all instances. This data field is unused when using entropy codec which can determine compressed bitstream length automatically for coding scaling_data.

compr_insta_scaling_data: This data field contains the compressed scaling factors of all instances.

compr_insta_attribute_header: This data field contains the compressed elem_insta_attribute_header data of all instances.

compr_insta_attribute_data: This data field contains the compressed attribute data of all instances.

5.2.7.4.16 compr_insta_transl_data class

5.2.7.4.16.1 Syntax

	Num. of bits	Descriptor
class **compr_insta_transl_data**{		
grouped_insta_transl_bbox	4 × 32	
num_node	24	
num_dupli_leaf	[$\log_2$ Number of instances × 0,05]	
for (i = 0; i < num_dupli_leaf; i ++){		
dupli_leaf_id	[$\log_2$ Number of instances]	
dupli_insta_transl_num_flag	1	
if(**dupli_insta_transl_num_flag** == 1){		
num_dupli_insta_transl	4	
}		
}		
num_interval_bound	4	
for (i = 0; i < num_interval_bound; i ++){		

IECNORM.COM : Click to view the full PDF of ISO/IEC 14496-16:2011/AMD4:2017

© ISO/IEC 2017 – All rights reserved