INTERNATIONAL ORGANIZATION FOR STANDARDIZATION

**ORGANISATION INTERNATIONALE NORMALISATION**

**ISO/IEC JTC 1/SC 29/WG 11**

**CODING OF MOVING PICTURES AND AUDIO**

**ISO/IEC JTC1/SC29/WG11**

**MPEG2015//N15268**

**February 2015, Geneva, Switzerland**

|  |  |
| --- | --- |
| **Title:** | **Encoder Input Format for MPEG-H 3D Audio** |
| **Source:** | **Audio Subgroup** |
| **Status:** | **Approved** |

# Introduction

This document fully describes the input formats used for the MPEG-H 3D Audio, i.e.

* Channel-based input
* Object-based input
* HOA-based input

This may form the basis of a more comprehensive input format to be used for the subsequent collaborative technology development process.

# Channel-based Input

Channel-based input is delivered as a set of monophonic channel signals, where each channel signal is represented as a monophonic .WAV file.

## WAV File Naming & Calibration

The monophonic .WAV files adhere to the following naming convention:

<item\_name>\_A<azimuth\_angle>\_E<elevation\_angle>.wav

Azimuth angles are given in degrees ±180 (positive values go to the left w.r.t the frontal direction, i.e. mathematically positive). They are represented as a three-digit number always including a sign (padded from the left with zeros if necessary).

Elevation angles are given in degrees ±90 (positive values go upward w.r.t the ear/listener height). They are represented as a two-digit number always including a sign (padded from the left with zeros if necessary).

Examples: TestItemName\_A+030\_E+00.wav

TestItemName\_A-030\_E+00.wav

TestItemName\_A+000\_E+90.wav

In the case of LFE channels, the naming is

<item\_name>\_LFE<lfe\_number>.wav

where lfe\_number is either 1 or 2.

For the .WAV files to be accepted as valid input material, the azimuth and elevation angles of all files have to be within the angular tolerance ranges defined in the loudspeaker table in Annex 3 of the CfP. The .WAV files of the input channels are mapped to the corresponding loudspeakers without any modification of the .WAV file content.

For the CfP, the .WAV files shall contain content at 48kHz and 24bits.

The content of the monophonic .WAV files are level-calibrated and delay-aligned such that no correction on the playback side is necessary for a proper reproduction on a loudspeaker setup which is calibrated according to N13196, Calibration of 22.2 Multichannel Sound Reproduction, i.e. equivalent to a spherical loudspeaker arrangement.

# Object-based Input (with optional Channel-based Layer)

Input is delivered as a metadata file and a set of monophonic audio signals, where each audio signal is represented as a monophonic .WAV file. The audio signal might either contain a channel audio signal or an object audio signal.

## WAV File Naming & Calibration

For the CfP, the .WAV files are mono files sampled at 48kHz and 24bit.

### WAV File Naming of Object Audio Signal

.WAV files containing object audio signals are named:

<item\_name>\_<object\_id\_number>.wav

The <object\_id\_number> is a three-digit number counted from zero (padded from the left with zeros if necessary). Example: TestItemName\_005.wav

### WAV File Naming of Channel Audio Signal

.WAV files containing channel audio signals are named and mapped to loudspeakers according to Section 2.1.

### Calibration

The object audio signals files are level-calibrated and delay-aligned. For example, a listener in the sweet spot listening position perceives two events that occur in two objects at the same sample index as happening at the same time independent of the object rendering positions. The perceived level and delay of an object will not change if the position of the object is changed. The calibration of the audio signals assumes that the loudspeakers are calibrated according to N13196, Calibration of 22.2 Multichannel Sound Reproduction.

For details on channel audio signals see Section 2.1

## Object Metadata File Definition

The object metadata file (<item\_name>.OAM) is used to describe metadata for a combined scene consisting of channels and objects. It contains the number of objects participating in a scene, plus the number and names of all channel signal files also belonging to this same scene.

The file begins with a header providing overall information on the scene description. A series of channel and object description data fields is following after the header. The number format is defined to be Little-Endian.

For channel audio signals, only the filename of each .WAV file is given.

To allow time-variant object properties, each object description field contains a timestamp (audio sample index) and an audio file index. A series of object descriptions thus can describe sampled movements of objects.

The timestamp (sample index) of all but the first object description field has always to be greater than the timestamp of the preceding object description of the same object index in the file. For one timestamp value, the object descriptions of all objects of the scene shall be inserted in ascending order of their object indices.

The object descriptions of all objects of a scene are repeated with a period of 1024 samples, even if no change of object properties has occurred. Repeating the position simplifies the implementation and also gives clear indication about from which point in time interpolation of positions have to start (see interpretation).

The file header must be followed at least by <**number\_of\_channel\_signals**> channel description fields, each referring to the filename of a WAV files, and at least by <**number\_of\_object\_signals**> object description fields, each referring to the first sample of the WAV files.

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bytes** | **Data format** |
| description\_file () {  scene\_description\_header()  while (end\_of\_file == 0) {  for (i=0; i<**number\_of\_object\_signals**; i++) {  object\_metadata(i)  }  }  } |  |  |

scene\_description\_header() – a header providing overall information on the scene description.

object\_metadata(i) – object description data for i-th object.

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bytes** | **Data format** |
| scene\_description\_header() {  **format\_id\_string**  **format\_version**  if (format\_version > 2) {  **hasDynamicObjectPriority**  }  **number\_of\_channel\_signals**  **number\_of\_object\_signals**  **description\_string**  for (i=0; i<number\_of\_channel\_signals; i++) {  **channel\_file\_name**  }  for (i=0; i<number\_of\_object\_signals; i++) {  **object\_description**  }  } | 4  2  2  2  2  32  64  64 | char  unsigned int  unsigned int  unsigned int  unsigned int  char  char  char |

**format\_id\_string** – unique character identifier “OAM ”

**format\_version** – version number of the file format.

**number\_of\_channel\_signals** – number of channels compiling the scene. Note: This number might be zero if the item is only object based.

**number\_of\_object\_signals** – number of simultaneous objects compiling the scene. Note: This number might be zero if the item is only channel based.

**description\_string** – description string containing a human readable content description. If shorter than 32 bytes, it is followed by padding null characters. If the string is 32 bytes long, the string is terminated without a null character.

**channel\_file\_name** – description string containing the file name of the according audio channel file. If shorter than 64 bytes, it is followed by padding null characters. If the string is 64 bytes long, the string is terminated without a null character. The file name follows the naming scheme as defined in Section 2.1.

**object\_description** – description string containing human readable text describing the object. If shorter than 64 bytes, it is followed by padding null characters. If the string is 64 bytes long, the string is terminated without a null character.

**hasDynamicObjectPriority** – **flag indicating whether there is dynamic object priority data present in** object\_metadata() **or not.**

|  |  |  |
| --- | --- | --- |
| **Syntax** | **No. of bytes** | **Data format** |
| object\_metadata() {  **sample\_index**  **object\_index**  **position\_azimuth**  **position\_elevation**  **position\_radius**  **gain\_factor**  if (format\_version > 1) {  **spread**  }  if (format\_version > 2) {  if (hasDynamicObjectPriority) {  **dynamic\_object\_priority**  }  }  } | 8  2  4  4  4  4  4  4 | unsigned int  unsigned int  32-bit float  32-bit float  32-bit float  32-bit float  32-bit float  32-bit float |

**sample\_index** – sample based timestamp, representing the time position within the audio signal in samples, to which this object description is assigned. The first sample of the content is referenced by sample\_index = 0.

**object\_index** – object number, referring to the assigned audio signal (and wave file) of the object. The first object is referenced by object\_index = 0.

**position\_azimuth** – position of the object: azimuth (°), has to be in the range -180…180.

**position\_elevation** – position of the object: elevation (°), has to be in the range -90…90.

**position\_radius** – position of the object: radius (m), has to be non-negative.

**gain\_factor** – (linear) factor to modify gain of the object, e.g. 1.0.

**spread** – parameter that determines the angular extent (°) of the region, to which the

energy of an audio element is distributed. The value shall be in the range of 0...180.

**dynamic\_object\_priority** – priority of the object. This field can take values between 0 and 7. The object may be discarded from rendering and decoding if the priority is lower than 7. If objects are discarded, the objects with lowest priority should be discarded first.

Thus, every object has given positions (azimuth, elevation, and radius) at defined timestamps. For every given position the renderer calculates panning gains. The panning gains between any given pair of adjacent timestamps will be linearly interpolated. The renderer’s task is to calculate the loudspeaker signals in such way that the perceived directions are in accordance with the object positions for a listener located in the sweet spot position (that is, the origin of the setup coordinate system). The interpolation is to be implemented in a way that the given object position is reached exactly at the corresponding sample\_index.

## Object Metadata File & Object Signal File Interpretation/Rendering

For the purpose of clarity of the interpretation of the format, a simple rendering algorithm based on the “Vector Base Amplitude Panning” (VBAP) algorithm by Ville Pulkki, as published in [1,2], is available as a binary. The reference renderer transforms the scene, described by the object metadata file and its object descriptions to WAV files containing loudspeaker signals for 22.2. For each loudspeaker signal, channel based content (if present) is added “as is” by the renderer. This results in output signals that enable reproduction on spherical loudspeaker setups without further correction.

The VBAP algorithm reproduces the content as intended by the mixer in the sweet-spot position. The algorithm is not designed to reproduce correct object positions for off-sweet-spot listening positions. However, this is the case for most panning algorithms that would be used for channel-based content creation as well.

The VBAP renderer applies a triangle mesh for panning gain calculations as described in [1]. The three vertices of each triangle are defined by loudspeaker positions.

The VBAP renderer uses the mesh of triangles (labels defined in International Standard IEC 62574 TC100) as shown in Table 1:

| **Triangle #** | **Vertex 1** | **Vertex 2** | **Vertex 3** |
| --- | --- | --- | --- |
| 1 | TpFL | TpFC | TpC |
| 2 | TpFC | TpFR | TpC |
| 3 | TpSiL | BL | SiL |
| 4 | BL | TpSiL | TpBL |
| 5 | TpSiL | TpFL | TpC |
| 6 | TpBL | TpSiL | TpC |
| 7 | BR | TpSiR | SiR |
| 8 | TpSiR | BR | TpBR |
| 9 | TpFR | TpSiR | TpC |
| 10 | TpSiR | TpBR | TpC |
| 11 | BL | TpBC | BC |
| 12 | TpBC | BL | TpBL |
| 13 | TpBC | BR | BC |
| 14 | BR | TpBC | TpBR |
| 15 | TpBC | TpBL | TpC |
| 16 | TpBR | TpBC | TpC |
| 17 | TpSiR | FR | SiR |
| 18 | FR | TpSiR | TpFR |
| 19 | FL | TpSiL | SiL |
| 20 | TpSiL | FL | TpFL |
| 21 | BtFL | FL | SiL |
| 22 | FR | BtFR | SiR |
| 23 | BtFL | FLc | FL |
| 24 | TpFC | FLc | FC |
| 25 | FLc | BtFC | FC |
| 26 | FLc | BtFL | BtFC |
| 27 | FLc | TpFC | TpFL |
| 28 | FL | FLc | TpFL |
| 29 | FRc | BtFR | FR |
| 30 | FRc | TpFC | FC |
| 31 | BtFC | FRc | FC |
| 32 | BtFR | FRc | BtFC |
| 33 | TpFC | FRc | TpFR |
| 34 | FRc | FR | TpFR |

Table 1 – Vertices (loud speaker names) of triangle mesh for VBAP panning.

The 22.2 setup does not support sources below the listener position (elevation < 0°) except for the front where three bottom speakers allow for lower object position reproduction and the side-front (between front loudspeakers FL/FR and side loudspeakers SiL/SiR). A useful calculation of audio sources below the limits given by the loudspeaker setup is not possible. Therefore, the reference renderer limits the object’s minimum elevation according to the given azimuth of the used audio object. [Note: The object format, as described in Section 3.2 does not have this restriction.]

The minimum elevation is determined by the lowest loudspeaker positions available in the reference 22.2 setup. For example, an object at azimuth 45° (equals BtFL) can have a minimum elevation of -15°. If the elevation of the object is lower, its elevation will be automatically adjusted to the minimum value prior to calculating the VBAP panning gains.

The minimum elevation is determined as follows depending on the azimuth angle of the audio object:

* Object in the front with azimuth between BtFL (45°) and BtFR (-45°): minimum elevation = -15°
* Object in the back with azimuth between SiL (90°) and SiR (-90°): minimum elevation = 0°
* Object azimuth between SiL (90°) and BtFL (45°): minimum elevation is determined by the direct connecting line between SiL and BtFL
* Object azimuth between SiR (-90°) and BtFR (-45°): minimum elevation is determined by the direct connecting line between SiR and BtFR

# HOA-based Input

HOA-based input is delivered as a set of monophonic channel signals, where each channel signal is represented as a monophonic .WAV file, 32-bits IEEE float with a sampling rate of 48 kHz.

The content of each .WAV file is a time-domain HOA real coefficients signal, called HOA component, i.e. .

The sound field description (SFD) is given by

|  |  |  |
| --- | --- | --- |
|  |  | (SFD) |

The time-domain HOA real coefficients are given by .

denotes the inverse time-domain Fourier Transformation where corresponds to .

The HOA renderer provides output signals dedicated to driving a spherical loudspeaker arrangement (from a time and level adjustment point of view). If the reproduction system of a test site is not a spherical arrangement, then this test site will have to perform its own proper time and level compensation before feeding loudspeakers.

For other information concerning the HOA format (including NFC), please refer to [3] ], [4] Annex F.

## HOA Component File Naming

The monophonic .WAV files adhere to the following naming convention:

<item\_name>\_.wav,

where denotes the item HOA order and denotes the HOA component with the order index (single digit), , , the azimuthal frequency index (single digit). Details are given in Table 2:

|  |  |
| --- | --- |
|  | <item\_name>\_<>\_00+.wav |
|  | <item\_name>\_<>\_11+.wav |
|  | <item\_name>\_<>\_11-.wav |
|  | <item\_name>\_<>\_10+.wav |
|  | <item\_name>\_<>\_22+.wav |
|  | <item\_name>\_<>\_22-.wav |
|  | <item\_name>\_<>\_21+.wav |
|  | <item\_name>\_<>\_21-.wav |
|  | <item\_name>\_<>\_20+.wav |
|  | <item\_name>\_<>\_33+.wav |
| … | … |

Table 2 – Relation of HOA component waveform to .WAV file name;

## HOA Metadata File Definition & Interpretation

The associated metadata files adhere to the following naming convention:

<item\_name>.ham

This text file includes the following information:

* A flag indicating the presence/absence of Near Field pre-Compensation (NFCon/NFCoff)
  + NFCflag=[1|0];
    - 1: NFC information present
    - 0: no NFC information
* NFC Reference Distance [in m] if applicable
  + NFCrefDist = xxx;
    - distance in meters
* Other information can be ignored

The following conventions apply for the content:

* Content value range satisfies [4], Annex C.5.1.
* Azimuth of 0º means front, (positive values go to the left w.r.t the frontal direction, i.e. mathematically positive).
* Use of real Spherical Harmonic functions
* The normalization convention used is N3D
* The Condon-Shortley Phase is not used (no (-1)m term in associated Legendre functions of the Spherical Harmonics)
* The sound speed is set to 343 m/s.

Internally the MPEG 3D Audio HOA renderer uses a rendering matrix MLxC to transfer the C=(N+1)2 HOA coefficients to L speaker channels. The ordering of HOA coefficients with regard to the columns index c of this matrix is given by ACN ordering: . For details refer to [4], subclause 12.4.3.

# References

[1] V. Pulkki, "Virtual Sound Source Positioning Using Vector Base Amplitude Panning", J. Audio Eng. Soc., vol. 45, pp. 456-466 (1997 June).

[2] <http://www.acoustics.hut.fi/software/vbap/VBAP_demo/vbap_v0.1.tar.gz>

[3] ISO/IEC 14496-11/2005.

[4] ISO/IEC DIS 23008-3