**INTERNATIONAL ORGANISATION FOR STANDARDISATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

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

**CODING OF MOVING PICTURES AND AUDIO**

**ISO/IEC JTC1/SC29/WG11 MPEG/M52350**

**January 2019, Brussels, BE**

|  |  |
| --- | --- |
| **Source** | **Philips** |
| **Status** | **Draft** |
| **Title** | **Philips CE-1.1-related response: V-PCC alignment** |
| **Author** | Bart Kroon |

# Introduction

This contribution is on the alignment of ISO/IEC 23090-12 MIV with ISO/IEC 23090-5 V-PCC.

Choices in this proposal are informed by the agreed alignment conditions [M51454 §2.1.3]:

1. Copy the syntax from Part 5 to Part 12 (analogous to HEVC annexes).
2. Only syntax structures that change are copied. Others are by normative reference.
3. Change syntax element names and semantics, but only when necessary and keep syntax descriptors.
4. Add additional syntax elements based on the V-PCC extension flags.
5. Align each meeting while both standards progress.
6. Align our test model (TMIV 4) to this syntax.
7. The name of Part 5 should be inclusive.

# Proposed MIV syntax and semantics

General considerations:

* When this proposal refers to a syntax structure from MIV WD3 without defining it, the proposal is to keep the structure unchanged.
* When this proposal refers to a syntax structure from V-PCC DIS, the proposal is for MIV to reference that structure from V-PCC.
* The MIV specification uses the same syntax functions and descriptors as V-PCC. (Both copied from older standards.) The proposal is to include these by reference instead of redefining them when appropriate.
* In general, the proposal is to take out all duplicate parts from the MIV specification when appropriate for increased clarity and reduced editing time.

## MIV parameter set

### MIV parameter set syntax (MPS)

|  |  |
| --- | --- |
| miv\_parameter\_set( ) { | **Descriptor** |
| ... /\* same as vpcc\_parameter\_set( ) \*/ |  |
| **vps\_extension\_present\_flag** | u(1) |
| if( vps\_extension\_present\_flag ) { |  |
| **vps\_extension\_length** | ue(v) |
| **mps\_msp\_present\_flag** | u(1) |
| if(miv\_sequence\_params\_present\_flag ) |  |
| miv\_sequence\_params( ) |  |
| while( more\_data\_in\_mps\_extension( ) ) |  |
| **mps\_extension\_data\_flag** | u(1) |
| } |  |
| byte\_alignment( ) |  |
| } |  |

[Ed.(BK): Align extension mechanism with V-PCC.]

### MIV parameter set syntax functions

more\_data\_in\_mps\_extension( ) which is used only in the miv\_parameter\_set( ) is specified to return TRUE if and only if the number of bits read after vps\_extension\_length, not including vps\_extension\_length, is less than 8\*vps\_extension\_length. Otherwise the function returns FALSE.

### MIV parameter set semantics

**mps\_msp\_present\_flag** shall be equal to 1 in bitstreams conforming to this version of this Specification.

**mps\_extension\_data\_flag** may have any value.

### MIV parameter set discussion

The MIV parameter set (MPS) extends the V-PCC parameter set using the available extension mechanism. This enables the MIV parameter set to reuse the VPS V-PCC unit type (vuh\_unit\_type == VPCC\_VPS).

## MIV sequence parameters (MSP)

### MIV sequence parameter syntax

|  |  |
| --- | --- |
| miv\_sequence\_params( ) { | **Descriptor** |
| **msp\_profile\_idc** | u(8) |
| **msp\_depth\_params\_num\_bits\_minus8** | u(4) |
| view\_params\_list( ) |  |
| **msp\_depth\_low\_quality\_flag** | u(1) |
| **msp\_num\_groups\_minus1** | ue(v) |
| **msp\_max\_entities\_minus1** | ue(v) |
| **msp\_viewing\_space\_present\_flag** | u(1) |
| if( msp\_viewing\_space\_present\_flag ) |  |
| viewing\_space( ) |  |
| } |  |

### MIV sequence parameter semantics

**msp\_profile\_idc** indicates the MIV profile component to which the IVS conforms as specified in Annex X. Bitstreams shall not contain values of msp\_profile\_idc other than those specified in Annex X. Other values of msp\_profile\_idc are reserved for future use by ISO/IEC.

The other syntax elements correspond to the MIV WD3 syntax elements without the msp\_ prefix.

### Discussion on MIV sequence parameters

The MIV sequence parameters replace the IV sequence parameters of MIV WD3. The main differences are that the MIV profile\_tier\_level( ) is replaced by msp\_profile\_idc and the extension mechanism is moved up to miv\_parameter\_set( ).

## Depth quantization

### Depth quantization syntax

|  |  |
| --- | --- |
| depth\_quantization( v ) { | **Descriptor** |
| **quantization\_law**[ v ] | u(8) |
| if( quantization\_law[ v ] == 0 ) { |  |
| **norm\_disp\_low**[ v ] | fl(32) |
| **norm\_disp\_high**[ v ] | fl(32) |
| } |  |
| **depth\_occ\_map\_threshold\_default**[ v ] | u(v) |
| **~~depth\_start\_default\_present\_flag~~**~~[ v ]~~ | ~~u(1)~~ |
| ~~if( depth\_start\_default\_present\_flag[ v ] )~~ |  |
| **~~depth\_start\_default~~**~~[ v ]~~ | ~~u(v)~~ |
| } |  |

### Depth quantization discussion

The mpd\_depth\_start[ p ] field is always present.

## MIV atlas sequence parameter set RBSP (MSPS)

### MIV atlas sequence parameter set RBSP syntax

|  |  |
| --- | --- |
| miv\_atlas\_sequence\_parameter\_set\_rbsp( ) { | **Descriptor** |
| ... /\* same as atlas\_sequence\_parameter\_set\_rbsp( ) \*/ |  |
| **asps\_extension\_present\_flag** | u(1) |
| if( asps\_extension\_present\_flag ) { |  |
| **msps\_masp\_present\_flag** | u(1) |
| if( msps\_masp\_present\_flag ) |  |
| miv\_atlas\_sequence\_params( ) |  |
| while( more\_rbsp\_data( ) ) |  |
| **msps\_extension\_data\_flag** | u(1) |
| } |  |
| rbsp\_trailing\_bits( ) |  |
| } |  |

### MIV atlas sequence parameter set RBSP semantics

**msps\_masp\_present\_flag** when present shall be equal to 1 in bitstreams conforming to this version of this Specification. When not present, the value of msps\_masp\_present\_flag is inferred to be 0.

**msps\_extension\_data\_flag** may have any value.

### MIV atlas sequence parameter set RBSP discussion

The MIV atlas sequence parameter set (MSPS) extends the V-PCC atlas sequence parameter set RBSP using the available extension mechanism. This enables the MIV atlas sequence parameter set to reuse the ASPS NAL unit type (nal\_unit\_type == NAL\_ASPS).

## MIV atlas sequence parameters (MASP)

### MIV atlas sequence parameters syntax

|  |  |
| --- | --- |
| miv\_atlas\_sequence\_params( ) { | **Descriptor** |
| if ( vuh\_atlas\_id == 0 ) |  |
| **masp\_omaf\_v1\_compatible\_flag** | u(1) |
| **masp\_group\_id** | u(v) |
| } |  |

### MIV atlas sequence parameter semantics

**masp\_omaf\_v1\_compatible\_flag** specifies that there is attribute video data of type Texture that is compatible for carriage within ISO/IEC 23009-2 1st edition (2019). When omaf\_v1\_compatible\_flag is equal to 1, it is a requirement of bitstream conformance that at least one sub-set of patches of this atlas conforms to a projection format specified in ISO/IEC 23009-2 1st edition (2019). When msps\_masp\_present\_flag = 0, the value of masp\_omaf\_v1\_compatible\_flag is inferred to be 0.

**masp\_group\_id** specifies the group index of the i-th atlas. The number of bits used for the representation of masp\_group\_id is Ceil(Log2( num\_groups\_minus1+1) ). The value of group\_id shall be in the range of 0 .. num\_groups\_minus1.When msps\_masp\_present\_flag = 0, the value of masp\_group\_id is inferred to be 0.

NOTE: With num\_groups\_minus1 == 0 all masp\_group\_id values are equal to 0 and thus there is no grouping.

### MIV atlas sequence parameter discussion

The masp\_omaf\_v1\_compatible\_flag semantics are modelled after the omaf\_v1\_compatible\_flag. In MIV WD3 the flag precedes the loop over the atlases. This has been translated into the vuh\_atlas\_id == 0 condition. We wrote the semantics carefully not to restrict to the V-PCC sample stream format. With the V-PCC unit stream format it is in principle possible that, the attribute video data is a separate bitstream.

## MIV patch data unit (MPU)

### MIV patch data unit syntax

|  |  |
| --- | --- |
| miv\_patch\_data\_unit( p ) { | **Descriptor** |
| **mpu\_view\_id**[ p ] /\* new semantics for pdu\_projection\_id \*/ | u(v) |
| **pdu\_2d\_pos\_x**[ p ] | u(v) |
| **pdu\_2d\_pos\_y**[ p ] | u(v) |
| **pdu\_2d\_delta\_size\_x**[ p ] | se(v) |
| **pdu\_2d\_delta\_size\_y**[ p ] | se(v) |
| **mpu\_view\_pos\_x**[ p ] /\* new semantics for pdu\_3d\_pos\_x \*/ | u(v) |
| **mpu\_view\_pos\_y**[ p ] /\* new semantics for pdu\_3d\_pos\_y \*/ | u(v) |
| **mpu\_depth\_start**[ p ] /\* new semantics for pdu\_3d\_pos\_min\_z \*/ | u(v) |
| if( asps\_normal\_axis\_max\_delta\_value\_enabled\_flag ) |  |
| **mpu\_depth\_end**[ p ] /\* new semantics for pdu\_3d\_pos\_delta\_max\_z \*/ | u(v) |
| **pdu\_orientation\_index**[ p ] | u(v) |
| if( afps\_lod\_bit\_count > 0 ) |  |
| **pdu\_lod**[ p ] | u(v) |
| if( asps\_point\_local\_reconstruction\_enabled\_flag ) |  |
| point\_local\_reconstruction\_data( p ) |  |
| } |  |

[Ed.(BK): There is a bug in V-PCC DIS. In this proposal the mpu\_view\_id aka pdu\_projection\_id field has been moved up to fix the bug. To be aligned with V-PCC.]

### MIV patch data unit semantics

**mpu\_view\_pos\_x**[ p ] specifies the horizontal coordinate in luma samples, respectively, of the top-left corner of the i-th patch in the mpu\_view\_id[ p ]-th view. The value of mpu\_view\_pos\_x[ p ] shall be in the range of 0 to projection\_plane\_width\_minus1[ mpu\_view\_id[ p ] ], inclusive. The number of bits used to represent mpu\_view\_pos\_x[ p ] is afps\_3d\_pos\_x\_bit\_count\_minus1 + 1.

NOTE: The number of bits is identical to V-PCC.

**mpu\_view\_pos\_y**[ p ] specifies the vertical coordinate in luma samples, respectively, of the top-left corner of the i-th patch in the mpu\_view\_id[ p ]-th view. The value of mpu\_view\_pos\_y[ p ] shall be in the range of 0 to projection\_plane\_height\_minus1[ mpu\_view\_id[ p ] ], inclusive. The number of bits used to represent mpu\_view\_pos\_y[ p ] is afps\_3d\_pos\_y\_bit\_count\_minus1 + 1.

NOTE: The number of bits is identical to V-PCC.

**mpu\_depth\_start**[ p ] specifies the start of the valid range of depth values for the p-th patch MpuDepthStart[ p ], as follows:

MpuDepthStart[ p ] = mpu\_depth\_start[ p ] << atgh\_pos\_min\_z\_quantizer

The number of bits used to represent mpu\_depth\_start[ p ] is equal to (gi\_geometry\_3d\_coordinates\_bitdepth\_minus1 + 1 – atgh\_pos\_min\_z\_quantizer + ( mpu\_view\_id[ p ] > 5 ) ? 2 : 1 ).

NOTE: The number of bits is identical to V-PCC.

**mpu\_depth\_end**[ p ] when present specifies the (inclusive) end of the valid range of depth values for the p-th patch MpuDepthEnd[ p ], as follows:

MpuDepthEnd[ p ] = mpu\_depth\_end[ p ] << atgh\_pos\_min\_z\_quantizer

The number of bits used to represent mpu\_depth\_end[ p ] is equal to (gi\_geometry\_3d\_coordinates\_bitdepth\_minus1 + 1 – atgh\_pos\_min\_z\_quantizer quantizer + ( mpu\_view\_id[ p ] > 5 ) ? 2 : 1 ).

When mpu\_depth\_end[ p ] is not present, MpuDepthEnd[ p ] is inferred to be the maximum possible depth value.

NOTE: The number of bits is identical to V-PCC.

**mpu\_view\_id**[ p ] indicates the view representation pair index associated with the p-th patch. The number of bits used to represent mpu\_view\_id[ p ] is equal to Ceil(Log2(num\_views\_minus1 + 1)).

NOTE: The number of bits is different from V-PCC.

### MIV patch data unit discussion

The proposal is to use the V-PCC patch\_data\_unit with modified semantics. The number of bits of all u(v) fields is kept the same, except for mpu\_view\_id[ p ]. This increases the overlap between implementations and also allows MIV to consider inter coding patch data units without much specification or additional implementation work.

Surprisingly, there is no extension mechanism in V-PCC DIS for patch\_data\_unit, patch\_information\_data, atlas\_tile\_group\_data\_unit or atlas\_tile\_group\_layer\_rbsp. In favor of compatibility, we accept the following differences with MIV WD3:

* **depth\_occ\_map\_threshold\_present\_flag**[ a ][ p ]is inferred to be 0.
* **depth\_occ\_map\_threshold**[ a ][ p ] is inferred as when not present in MIV WD3.
* **depth\_start\_present\_flag**[ a ][ p ] is inferred to be 1.
* **entity\_id**[ a ][ p ]...

This proposal does not provide a fleshed out solution for entity\_id[ a ][ p ] but the pdu\_lod[ p ] field could be repurposed unless there is a better solution. The ideal solution would be when one of the mentioned V-PCC structures would have an extension mechanism.

# MIV profiles

The msp\_profile\_idc adds to the complex of V-PCC profile/tier/level fields. A typical MIV usage could be:

* ptl\_tier\_flag = 0 (Main tier)
* ptl\_profile\_codec\_group\_idc = 1 (HEVC Main10)
* ptl\_profile\_pcc\_toolset\_idc = 2 (a new value defined by MIV to allow multiple atlases)
* ptl\_profile\_reconstruction\_idc = 0 (Rec0, no reconstruction)
* msp\_profile\_idc
* ptl\_level\_idc = 3.0 (a higher level that matches with the MIV CTC pixel rate constraints)

We propose the following values for msp\_profile\_idc to get started:

* Basic: WD3 level of functionality
* Extended: Include compatible V-PCC functionality
* Unconstrained: The only restrictions are from V-PCC Annex A

Below tables are a first stab at defining these profiles.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **msp\_profile\_idc** | | |
| **Syntax element** | **0 (Basic)** | **1 (Extended)** | **2 (Unconstrained)** |
| vuh\_unit\_type | != VPCC\_OVD | ? | ‒ |
| vps\_raw\_patch\_enabled\_flag | 0 | ? | ‒ |
| vps\_extension\_present\_flag | 1 | 1 | ‒ |
| ptl\_profile\_pcc\_toolset\_idc | 2 | 2 | ‒ |
| ptl\_profile\_reconstruction\_idc | Rec0 | Rec0 | ‒ |
| occupancy\_information( ) | All zeros | ? | ‒ |
| gi\_geometry\_MSB\_align\_flag | 0 (In alignment) |  |  |
| ai\_attribute\_MSB\_align\_flag | 0 (In alignment) | ? | ‒ |
| asps\_long\_term\_ref\_atlas\_frames\_flag | 0 | ? | ‒ |
| asps\_num\_ref\_atlas\_frame\_lists\_in\_asps | 0 | 0 | ‒ |
| asps\_use\_eight\_orientations\_flag | 0 | ‒ | ‒ |
| asps\_remove\_duplicate\_point\_enabled\_flag | 0 | 0 |  |
| asps\_pixel\_deinterleaving\_flag | 0 | 0 |  |
| asps\_patch\_precedence\_order\_flag | 0 | 0 |  |
| afps\_lod\_bit\_count | 0 | 0 |  |
| afps\_override\_eom\_for\_depth\_flag | 0 | 0 |  |
| afps\_raw\_3d\_pos\_bit\_count\_explicit\_mode\_flag | 0 | 0 |  |
| afti\_single\_tile\_in\_atlas\_frame\_flag | 0 | ‒ |  |
| atgh\_type | I\_TILE\_GRP | ? |  |
| atgdu\_patch\_mode | I\_INTRA | ? |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ptl\_profile\_pcc\_toolset\_idc** | | |
| **Syntax element** | **0** | **1** | **2** |
| asps\_enhanced\_occupancy\_map\_for\_depth\_flag | 0 | ‒ | 0 |
| vps\_map\_count\_minus1 | min( 1, LevelMapCount – 1 ) | LevelMapCount – 1 | min( 0, LevelMapCount – 1 ) |
| vps\_multiple\_map\_streams\_present\_flag  (when vps\_map\_count\_minus1 > 0) | when present, 1 | ‒ | ‒ |
| vps\_atlas\_count\_minus1 | 0 | 0 | ‒ |
| asps\_point\_local\_reconstruction\_enabled\_flag | 0 | ‒ | 0 |
| ai\_attribute\_dimension\_minus1 | 2 | ‒ | 2 |
| ai\_attribute\_dimension\_partitions\_minus1 | 0 | ‒ | 0 |
| ai\_attribute\_partition\_channels\_minus1 | ‒ | 2 | ‒ |
| asps\_use\_eight\_orientations\_flag | 0 | ‒ | ‒ |
| asps\_45degree\_projection\_patch\_present\_flag | 0 | ‒ | 0 |

# How to combine MIV and V-PCC data

This section describes how to combine immersive video and point cloud data in a single MIV bitstream.

## Coordinate systems

MIV has aligned itself with OMAF when picking a coordinate system. V-PCC does not specify the coordinate system, although the software reportedly uses OpenGL coordinate system.

* We recommend that V-PCC adopts the OMAF coordinate system. The reason is that OMAF 1st edition is ready and it is confusing to have multiple coordinate systems within MPEG-I. The required change to TMC2 would be insignificant.
* We also recommend that if V-PCC decides on a coordinate system, MIV will align to that coordinate system, even when the coordinate system is not OMAF. The motivation is that alignment with V-PCC is more important to MIV.

## Use multiple parameter sets

It is permitted to have multiple vpcc\_parameter\_set's with different vps\_vpcc\_parameter\_set\_id's. In this way a bitstream may contain both a vpcc\_parameter\_set and miv\_parameter\_set, and all atlas data can be consistent with that. This will result in having separate atlases for immersive video and point cloud data.

## Orthographic camera model

The point cloud camera cube can be described by six orthographic cameras. By adding an orthograhic projection type to MIV, point cloud data can be represented in a MIV bitstream. With this small addition, it is possible to mix point cloud and immersive video patches even within the same atlas.

# Implementation effort

This contribution comes with a software branch of TMIV that adds the following:

* A partial implementation of a V-PCC HLS codec.
* A partial implementation of the proposed MIV HLS codec.
* A decoding experiment to print out a V-PCC bitstream.
* An encoding experiment to transcode MIV WD3 to the proposed MIV bitstream.

Although incomplete, we consider this software branch to be useful for TMIV 4.0.

The codecs are implemented using a class per syntax structure. All classes are unit tested. The V-PCC HLS codec is largely independent of MIV. The only tweak is that the bit depth of the pdu\_projection\_id field can be overriden. The MIV HLS codec provides a partial implementation of the syntax structures in this proposal. It builds on top of the V-PCC HLS codec.

Danillo has supplied a V-PCC bitstream of the longdress sequence using a branch of TMC2. The software is able to decode the V-PCC units and demux the video (OVD, AVD and GVD). From the AD the NAL units are extracted. The effort to decode the atlas data is ongoing.

The motivation to make a second implementation (as discussed during the 128th MPEG meeting) is to "debug" the V-PCC specification. This turned out to be a good decision. We have reported multiple issues regarding the DIS and TMC2 to the V-PCC editorial team and/or TMC2 software coordinators. They have already addressed some of the comments.

We strive to make DIS comments available as NB comments and we are confident that the more important issues will be addressed in the V-PCC FDIS. We recommend that the TMIV and TMC2 software coordinators maintain a joint testing effort to increase conformance of both projects to the V-PCC specification, and to spot errors in the specification.

# Acknowledgements

I would like to thank the V-PCC editing team and software coordinators, especially Danillo, Julien and Alexis, for supporting me in this work and responding to my many comments and questions. Danillo has provided me with a private crash course, answers to obscure technical questions, test bitstreams and log-files to work with.