**INTERNATIONAL ORGANISATION FOR STANDARDISATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

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

**CODING OF MOVING PICTURES AND AUDIO**

**ISO/IEC JTC1/SC29/WG11 MPEG2019/m51037**

**October 2019, Geneva, CH**

|  |  |
| --- | --- |
| **Source** | **Apple Inc.** |
| **Status** | **Input document** |
| **Title** | **Simplification for EOM patch data unit** |
| **Author** | **Jungsun Kim, Alexis Michael Tourapis, Khaled Mammou** |

# Introduction

This contribution suggests syntax changes to simplify the usage of the EOM data unit in the V-PCC specification. Currently, the EOM data unit includes a syntax element that specifies the total number of intermediate points associated with the current EOM data unit. The process requires that the number of intermediate points in each patch is stored in and the associated patches derived for each EOM data unit. This process results in a limitation in that the associated patches within an EOM data unit are specified as consecutive patches in patch order. In this contribution, we suggest to signal the associated patch index for each patch that belongs to an EOM data unit instead of its intermediate point number. In addition, this contribution proposes to remove redundant information related to the EOM data unit. Signalling patch indices instead of the number of points can result in no difference in the performance for lossless coding when one EOM patch is used and a 0.3%~0.7% loss for lossless coding when more than one EOM patches are used caused from the attribute video coding(32 frames). However, at the same time, parsing and extraction of EOM information is considerably simplified.

# Identified Issues and Solutions

## An EOM patch and its related signaling in the current V-PCC specification

The current V-PCC specification [3] signals the enhanced occupancy map (EOM) related parameters in multiple structures, including the atlas sequence and frame parameter sets as follows:

#### Atlas sequence parameter set syntax (d39)

|  |  |
| --- | --- |
| atlas\_sequence\_parameter\_set( ) { | **Descriptor** |
| **…** |  |
| **asps\_enhanced\_occupancy\_map\_for\_depth\_flag** | u(1) |
| if( asps\_enhanced\_occupancy\_map\_for\_depth\_flag ) { |  |
| if( vpcc\_map\_count\_minus1[ asps\_atlas\_sequence\_parameter\_set\_id ] = = 0 ) |  |
| **asps\_enhanced\_occupancy\_map\_fix\_bit\_count\_minus1** | u(4) |
| **…** |  |
| } |  |

#### Atlas frame parameter set RBSP syntax

|  |  |
| --- | --- |
| atlas\_frame\_parameter\_set\_rbsp( ) { | **Descriptor** |
| **…** |  |
| if( asps\_enhanced\_occupancy\_map\_for\_depth\_flag &&  asps\_enhanced\_occupancy\_map\_attribute\_patch\_present\_flag ) { |  |
| **afps\_eom\_number\_of\_patch\_bit\_count\_minus1** | u(8) |
| **afps\_eom\_max\_bit\_count\_minus1** | u(5) |
| } |  |
| … |  |
| } |  |

#### EOM patch data unit syntax

|  |  |
| --- | --- |
| eom\_patch\_data\_unit( patchIdx ) { | **Descriptor** |
| **epdu\_2d\_shift\_x**[ patchIdx  ] | u(v) |
| **epdu\_2d\_shift\_y**[ patchIdx  ] | u(v) |
| **epdu\_2d\_delta\_size\_x**[ patchIdx   ] | se(v) |
| **epdu\_2d\_delta\_size\_y**[ patchIdx   ] | se(v) |
| **epdu\_patch\_count\_minus1**[ patchIdx ] | u(v) |
| for( p = 0; p < epdu\_patch\_count\_minus1[ patchIdx ] + 1; p++ ) |  |
| **epdu\_points**[patchIdx **]**[ p ] | u(v) |
| } |  |

## Proposed changes

### Signalling epdu\_associated\_patch\_index instead of epdu\_points[ patchIdx ][ p ]

### The current design of the EOM data unit requires signaling its associated intermediate points based on the order of the patches that include them. For example, the intermediate points from patch N come before the intermediate points from patch N+1. Instead of this approach, we suggest explicitly signaling for each EOM patch the index of each regular patch that is associated with it and contains intermediate points. This can make the extraction process of intermediate points more flexible as well as more straight forward. For example, it permits us to place similar colours that belong at distinct patches together instead of keeping them separately.

### In particular, a syntax element, pdu\_associated\_patch\_list, is signaled for this purpose that is of a fixed bit size, i.e. 8 bits, or is ue(v) coded. Alternatively, se(v) coding could also be used by first coding the first patch index in the list, and then coding differentially the remaining epdu\_patch\_count\_minus1[ patchIdx ] indices based on the previous in order index. It should be noted that in this method we also allow the same patch to be repeated in the same or even multiple EOM patches. This could be done for error resiliency purposes or to improve correlation between neighboring samples for coding efficiency purposes.

### The suggested syntax change using u(8) signaling is as follows:

|  |  |
| --- | --- |
| eom\_patch\_data\_unit( patchIdx ) { | **Descriptor** |
| **epdu\_2d\_shift\_x**[ patchIdx  ] | u(v) |
| **epdu\_2d\_shift\_y**[ patchIdx  ] | u(v) |
| **epdu\_2d\_delta\_size\_x**[ patchIdx   ] | se(v) |
| **epdu\_2d\_delta\_size\_y**[ patchIdx   ] | se(v) |
| **epdu\_patch\_count\_minus1**[ patchIdx ] | u(v) |
| for( p = 0; p < epdu\_patch\_count\_minus1[ patchIdx ] + 1; p++ ) |  |
| **~~epdu\_points~~**~~[~~~~patchIdx~~ **~~]~~**~~[ p ]~~ | ~~u(v)~~ |
| **epdu\_associated\_patch\_list**[patchIdx][ p ] | u(8) |
| } |  |

This new syntax element can be defined as:

**epdu\_associated\_patch\_list**[patchIdx][p] indicates the coding order index of a patch in the current tile group that is associated with the intermediate attribute points of the p-th patch of the patch data unit of index patchIdx. It is a requirement of bitstream conformance that epdu\_member\_patch\_list[patchIdx][p] shall not point to a patch of type I\_RAW, P\_RAW, I\_EOM, or P\_EOM.

Based on this modification we believe that the entire section 8.4.4.7 is not necessary. Furthermore, section 9.4.4.21 can be updated as follows:

|  |
| --- |
| Inputs to this process are:  …  - The decoded enhanced occupancy map related syntax elements, asps\_enhanced\_occupancy\_map\_fix\_bit\_count\_minus1.   * the list of EOM patches, eomPatches. (this should be defined ahead)   …  This clause is invoked when asps\_enhanced\_occupancy\_map\_for\_depth\_flag is equal to 1.  ~~The variable eomPointCnt is initialized to 0.~~  The array eomPointCnt of size M. for i from 0 to M-1, eomPointCnt [ i ] = 0.  For u = 0..Patch2dSizeX[ p ] − 1, v = 0..Patch2dSizeY[ p ] – 1, the following applies:   * The conversion of patch coordinates to atlas coordinates as specified in clause 9.4.5 is invoked with patch coordinates (u, v), and patch index p, as inputs and atlas coordinates (x, y) as the output.   – If OFrame[ y ][ x ] is not equal to 0 and ( D1 – D0 ) is greater than 1, the following ordered steps apply:   1. A variable D0 is set equal to GFrame[ 0 ][ y ][ x ]. 2. A variable D1 is derived as follows:  * If vpcc\_map\_count\_minus1[ atlasIdx ] is equal to 0, a variable D1 is set equal to D0 + asps\_enhanced\_occupancy\_map\_fix\_bit\_count\_minus1 + 1. * Otherwise ( vpcc\_map\_count\_minus1[atlasIdx] is greater than 0 ) D1 is set equal to GFrame[ 1 ][ y ][ x ]  1. If ( D1 – D0 ) is greater than 1, the following applies:  * A variable eomCode is derived as follows:   eomCode = (1 << (D1 – D0 – 1 ) ) − OFrame[ y ][ x ]   * For b = 0 .. (D1 – D0 – 2 ), if ( ( ( eomCode >> b ) & 0x01 ) = = 1 ), the following applies:   + Clause 9.4.6 is invoked with the patch coordinates ( u, v ), the atlas coordinates ( x, y ), the patch index p, and depth ( D0 + b + 1 ) and the output is array pos of dimension 3.   + The vector, pos, is copied over to a new entry in RecPcGeo.   for( n=0; n < 3; n++ )  RecPcGeo[ PointCnt ][ n ] = pos[ n ]  If p is epdu\_member\_patch\_list[eomPtchIdx][ i ], for aIdx = 0 .. ai\_attribute\_count[ atlasIdx ]− 1, attributes are assigned to the newly added point as follows:  ~~eomPos = EomPointOffset[p] + eomPointCnt~~  inBlockPos = eomPointCnt[eomPtchIdx]% ( PatchPackingBlockSize \* PatchPackingBlockSize )  nbBlock = eomPointCnt[eomPtchIdx]/ (PatchPackingBlockSize \* PatchPackingBlockSize )  xBlock = nbBlock % ( EomPatch2dSizeX [ p ]/ PatchPackingBlockSize )  yBlock = nbBlock / ( EomPatch2dSizeY [ p ]/ PatchPackingBlockSize )  xx = xBlock \* PatchPackingBlockSize + inBlockPos % PatchPackingBlockSize  + EomPatch2dShiftX[ p ]  yy = yBlock \* PatchPackingBlockSize + inBlockPos / PatchPackingBlockSize  + EomPatch2dShiftY[ p ]  attrDim = ai\_attribute\_dimension\_minus1[ atlasIdx ][ aIdx ] + 1  for( cIdx = 0; cIdx  <  attrDim; cIdx+ + )  RecPcAttr[ PointCnt ][ aIdx ][ cIdx ] = AFrame[ aIdx ][ 0 ][ cIdx ][ yy ][ xx ]  AttrPresent[ PointCnt ] = 1   * + eomPointCnt[eomPtchIdx ] += 1   + PointCnt += 1 |

### afps\_eom\_number\_of\_patch\_bit\_count\_minus1 and afps\_eom\_max\_bit\_count\_minus1

The syntax element afps\_eom\_number\_of\_patch\_bit\_count\_minus1 is used to signal epdu\_patch\_count\_minus1[ patchIdx ]. We suggest to signal the syntax element epdu\_patch\_count\_minus1[ patchIdx ] with a fixed bit size such as 8 instead of signaling its bit size or using ue(v) coding[[1]](#footnote-1). The syntax element afps\_eom\_max\_bit\_count\_minus1 then is not needed based on the previous suggestion. The syntax change is as follows:

|  |  |
| --- | --- |
| atlas\_frame\_parameter\_set\_rbsp( ) { | **Descriptor** |
| **…** |  |
| ~~if( asps\_enhanced\_occupancy\_map\_for\_depth\_flag &&  asps\_enhanced\_occupancy\_map\_attribute\_patch\_present\_flag ) {~~ |  |
| **~~afps\_eom\_number\_of\_patch\_bit\_count\_minus1~~** | ~~u(8)~~ |
| **~~afps\_eom\_max\_bit\_count\_minus1~~** | ~~u(5)~~ |
| } |  |
| … |  |
| } |  |

### Removing asps\_enhanced\_occupancy\_map\_fix\_bit\_count\_minus1

The syntax element asps\_enhanced\_occupancy\_map\_fix\_bit\_count\_minus1 indicates how many bits will be used to represent the locations of intermediate points in the case of the single layer/map system. It works as the depth value of the second layer in the case of the two layer/map system. Since intermediate point location information is signaled in the occupancy map, the maximum bitdepth is limited by the bitdepth of the occupancy map video. Therefore, asps\_enhanced\_occupancy\_map\_fix\_bit\_count\_minus1 can be derived from the values of the gi\_geometry\_nominal\_2d\_bitdepth\_minus1and oi\_occupancy\_nominal\_2d\_bitdepth\_minus1 syntax elements. At the encoder side, asps\_enhanced\_occupancy\_map\_fix\_bit\_count\_minus1can be given to define the maximum allowed depth for intermediate points

# Experimental results

## Signaling of a single EOM Patch

When the number of EOM patches and attributes related to the EOM patches are exactly the same as the anchor, the proposed EOM patch shows no difference in the performance.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **bpip ratio [%]** | | | | **Shift in G:T bits ratio [pp]** | **Shift in G:T bits ratio [pp]** |
|  | All Intra | **Total** | **Tot.Geom** | **Geometry** | **Colour** |
| cat2-A | 8ivfbv2\_loot\_vox10 | 100.0% | 99.97% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | 8ivfbv2\_redandblack\_vox10 | 100.0% | 99.93% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | 8ivfbv2\_soldier\_vox10 | 100.0% | 99.94% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | queen | 100.0% | 99.93% | 100.0% | 100.0% | 0.0% | 0.0% |
| cat2-B | 8ivfbv2\_longdress\_vox10 | 100.0% | 99.94% | 100.0% | 100.0% | 0.0% | 0.0% |
| cat2-C | basketball\_player\_vox11 | 100.0% | 99.98% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | dancer\_player\_vox11 | 100.0% | 99.97% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Cat2-A average** | 100.0% | 99.94% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Cat2-B average** | 100.0% | 99.94% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Cat2-C average** | 100.0% | 99.98% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Overall average** | 100.0% | 99.96% | 100.0% | 100.0% | 0.0% | 0.0% |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **bpip ratio [%]** | | | | **Shift in G:T bits ratio [pp]** | **Shift in G:T bits ratio [pp]** |
|  | Random Access | **Total** | **Tot.Geom** | **Geometry** | **Colour** |
| cat2-A | 8ivfbv2\_loot\_vox10 | 100.0% | 99.97% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | 8ivfbv2\_redandblack\_vox10 | 100.0% | 99.93% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | 8ivfbv2\_soldier\_vox10 | 100.0% | 99.93% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | queen | 100.0% | 99.90% | 100.0% | 100.0% | 0.0% | 0.0% |
| cat2-B | 8ivfbv2\_longdress\_vox10 | 100.0% | 99.94% | 100.0% | 100.0% | 0.0% | 0.0% |
| cat2-C | basketball\_player\_vox11 | 100.0% | 99.98% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | dancer\_player\_vox11 | 100.0% | 99.97% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Cat2-A average** | 100.0% | 99.94% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Cat2-B average** | 100.0% | 99.94% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Cat2-C average** | 100.0% | 99.98% | 100.0% | 100.0% | 0.0% | 0.0% |
|  | **Overall average** | 100.0% | 99.95% | 100.0% | 100.0% | 0.0% | 0.0% |

## Signaling more than one EOM Patches

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **bpip ratio [%]** | | | | **Shift in G:T bits ratio [pp]** | **Shift in G:T bits ratio [pp]** |
|  | All Intra | **Total** | **Tot.Geom** | **Geometry** | **Colour** |
| cat2-A | 8ivfbv2\_loot\_vox10 | 100.4% | 99.99% | 100.00% | 100.4% | -0.1% | -0.1% |
|  | 8ivfbv2\_redandblack\_vox10 | 100.6% | 99.96% | 100.00% | 100.7% | -0.1% | -0.1% |
|  | 8ivfbv2\_soldier\_vox10 | 100.7% | 99.95% | 99.99% | 100.9% | -0.1% | -0.1% |
|  | queen | 101.2% | 99.95% | 100.00% | 101.4% | -0.2% | -0.2% |
| cat2-B | 8ivfbv2\_longdress\_vox10 | 100.5% | 99.97% | 100.00% | 100.6% | -0.1% | -0.1% |
| cat2-C | basketball\_player\_vox11 | 100.3% | 99.97% | 99.99% | 100.3% | 0.0% | 0.0% |
|  | dancer\_player\_vox11 | 100.3% | 99.98% | 100.00% | 100.3% | 0.0% | 0.0% |
|  | **Cat2-A average** | 100.7% | 99.96% | 100.00% | 100.8% | 0.0% | -0.1% |
|  | **Cat2-B average** | 100.5% | 99.97% | 100.00% | 100.6% | 0.0% | -0.1% |
|  | **Cat2-C average** | 100.3% | 99.98% | 99.99% | 100.3% | 0.0% | 0.0% |
|  | **Overall average** | 100.5% | 99.97% | 100.00% | 100.6% | 0.0% | -0.1% |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **bpip ratio [%]** | | | | **Shift in G:T bits ratio [pp]** | **Shift in G:T bits ratio [pp]** |
|  | Random Access | **Total** | **Tot.Geom** | **Geometry** | **Colour** |
| cat2-A | 8ivfbv2\_loot\_vox10 | 100.4% | 99.99% | 100.00% | 100.4% | -0.1% | -0.1% |
|  | 8ivfbv2\_redandblack\_vox10 | 100.6% | 99.96% | 100.00% | 100.7% | -0.1% | -0.1% |
|  | 8ivfbv2\_soldier\_vox10 | 100.8% | 99.96% | 100.00% | 100.9% | -0.1% | -0.1% |
|  | queen | 101.3% | 99.93% | 100.00% | 101.5% | -0.2% | -0.2% |
| cat2-B | 8ivfbv2\_longdress\_vox10 | 100.5% | 99.97% | 100.00% | 100.6% | -0.1% | -0.1% |
| cat2-C | basketball\_player\_vox11 | 100.3% | 99.99% | 100.00% | 100.3% | 0.0% | 0.0% |
|  | dancer\_player\_vox11 | 100.3% | 100.00% | 100.01% | 100.3% | 0.0% | 0.0% |
|  | **Cat2-A average** | 100.7% | 99.96% | 100.00% | 100.9% | 0.0% | -0.1% |
|  | **Cat2-B average** | 100.5% | 99.97% | 100.00% | 100.6% | 0.0% | -0.1% |
|  | **Cat2-C average** | 100.3% | 99.99% | 100.01% | 100.3% | 0.0% | 0.0% |
|  | **Overall average** | 100.5% | 99.98% | 100.00% | 100.6% | 0.0% | -0.1% |

# References

1. “V-PCC Test Model v7”, ISO/IEC JTC1/SC29 WG11 (MPEG) output document N18666, Gothenburg, SE, July 2019
2. Common Test Conditions for PCC**,** ISO/IEC JTC1/SC29/WG11 N18665, Gothenburg, SE, July 2019
3. “([V-PCC] Current Draft of ISO/IEC 23090-5 (Video-based Point Cloud Compression )”, ISO/IEC JTC1/SC29 WG11 m51435 Geneva, CH, Oct. 2010

1. In any case, 8 bits for afps\_eom\_number\_of\_patch\_bit\_count\_minus1 is excessive. [↑](#footnote-ref-1)