**TINTERNATIONAL ORGANISATION FOR STANDARDISATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

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

**CODING OF MOVING PICTURES AND AUDIO**

**ISO/IEC JTC1/SC29/WG11 MPEG2020/m53541**

**April 2020, Alpbach, AT**

|  |  |
| --- | --- |
| **Source:** | **Panasonic corporation** |
| **Status :** | **Input document** |
| **Title:** | **[G-PCC] High Level Syntax and Specification Modification** |
| **Author:** | **Noritaka Iguchi** |

**Abstract**

In this contribution, we propose 4 additional syntax in G-PCC DIS (w19088)[1]

Propose 1: Add the tile index in tile inventory

Propose 2: Input format restriction of attribute component

Propose 3: Add the scale and offset of Attribute in SPS

Propose 4: Add the attribute label and instance id to TLV data header

# Tile Inventory

## Add the tile\_id in tile inventory [Propose 1]

Currently, in geometry data unit header, gsh\_tile\_id is signalled that indicates the index of tile correspond to the tile inventory. On the other hands in the tile inventory, tile\_id is not explicit and the order of the tile loop is tile\_id.

For example, if an external application sets the tile inventory, it is expected to give the arbitrary tile\_id, but the current syntax does not support this.

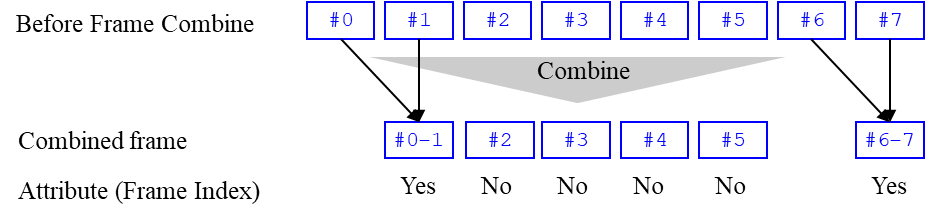
So, we propose to change the tile inventory so that the arbitrary tile\_id can be set.

|  |  |
| --- | --- |
| tile\_inventory( ) { | **Descriptor** |
| **tile\_frame\_idx** | ? |
| **num\_tiles** | u(16) |
| **tile\_id\_flag** | u(1) |
| **tile\_bounding\_box\_bits** | u(8) |
| for( i = 0; i < num\_tiles; i++ ) { |  |
| if(tile\_id\_flag) **tile\_id** | u(v) |
| **tile\_bounding\_box();** |  |
| } |  |
| } |  |

# Attribute Component

## Input format restriction of attribute component [Propose 2]

In previous MPEG128 meeting, we mentioned that when the attribute is a frame index there may be no attribute value depending on the frame in the input point cloud.



The current GPCC is based on the premise that all attribute values are present in the input point cloud and does not support the coding of missing attribute values.

To solve this problem, it is not practical to support the missing attribute value coding in G-PCC.

Therefore, we propose to add a sentence that constrains the input of attribute values to DIS spec as follows.

In the input point cloud sequence, the attribute component shown in the SPS must have an attribute value that correspond to the position. If attribute value is missing, it must be padded before input to the encoder.

## Add the scale and offset of Attribute in SPS [Propose 3]

In the specification (G-PCC DIS), some new attribute type (ex. Normal vector) are added.

However, current G-PCC support only unsigned integer type for attribute encoding/decoding and don’t support signed value.

For example, each normal vector has the value in range [-1, 1]. To encode a normal vector with an 8-bit integer value, the value must be converted to [0, 256] before encoding by scaling and offset.

We proposed to signal the scale and offset value in the attribute loop in the SPS so that the decoder can re-scale and offset to original values.

|  |  |
| --- | --- |
| seq\_parameter\_set( ) { | Descriptor |
| **......** |  |
| **sps\_source\_scale\_factor\_numerator\_minus1** | ue(v) |
| **sps\_source\_scale\_factor\_denominator\_minus1** | ue(v) |
| **sps\_num\_attribute\_sets** | ue(v) |
| for( i = 0; i< sps\_num\_attribute\_sets; i++ ) { |  |
| **attribute\_instance\_id**[ i ] | ue(v) |
| **attribute\_dimension\_minus1**[ i ] | ue(v) |
| **attribute\_bitdepth\_minus1**[ i ] | ue(v) |
| **source\_transfer\_flag** | u(1) |
| **if( source\_transfer\_flag){** |  |
| **source\_attribute\_scale\_log2** | ue(v) |
| **source\_attribute\_offset\_log2** | ue(v) |
| **}** |  |
| if(attribute\_dimension\_minus1[ i ] > 0 ) |  |
| **attribute\_secondary\_bitdepth\_minus1**[ i ] | ue(v) |
| **…** |  |
| } |  |
| … |  |
| } |  |

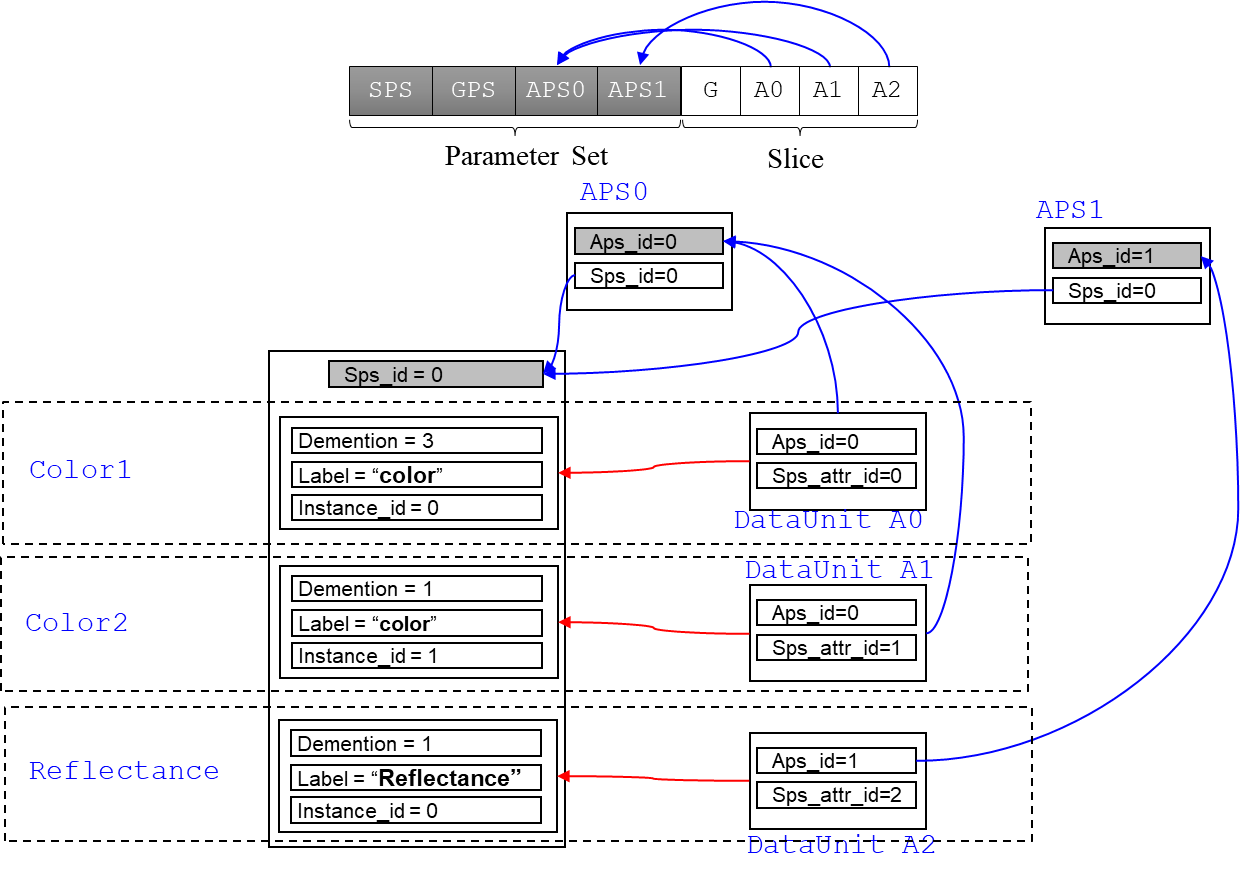
## Add the attribute label and instance id to TLV data header

The below figure show example of the syntax relationship on metadata of Attribute.

・Each attribute data unit is refer APS and SPS.

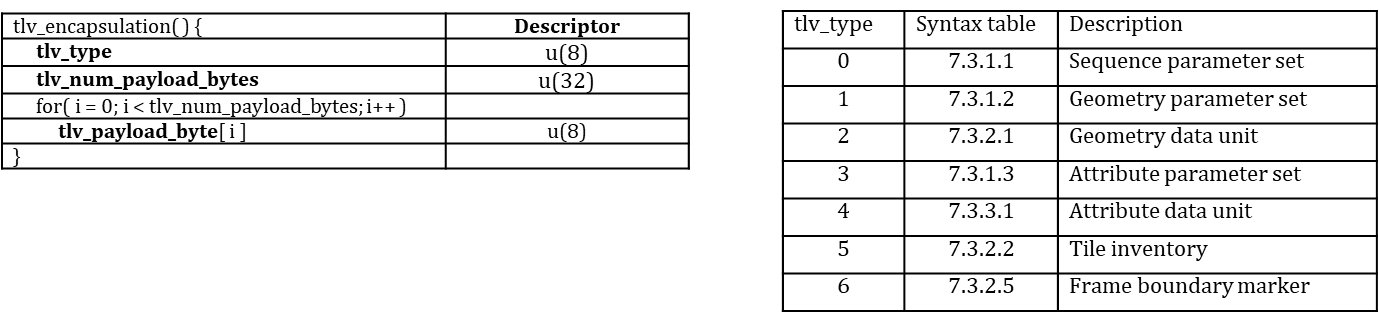
・Multiple attribute data units with different attribute types can refer to one APS.

・Each attribute data unit has sps\_attr\_id which can be used to get the attribute label (attribute type) and the instance id of the attribute in the attribute loop in sps.

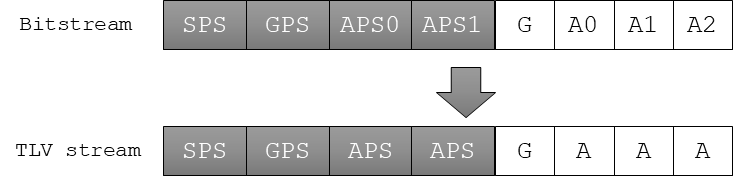


The below figure show the syntax of tlv data unit and table of tlv\_type.

The type of TLV data unit can be determined by analysing the tlv\_type.



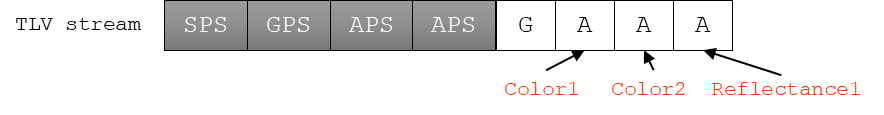
When the bitstream with the above structure is encapsulated in TLV, the same TLV type (4: Attribute data unit) are given to the three types of data units (A0: color1, A1: color2, A2: reflectance). This means that each component of an attribute cannot be identified by analyzing tlv data header (tlv\_type) in TLV layer.



In the system layer, each component may be stored in multiple tracks. In that case it is useful to be able to identify the component by parsing only the TLV data unit header.

We propose to add the attribute\_instance\_id and attribute\_label in SPS to TLV data unit header syntax as below.

With this modification, the component of the attribute information can be identified without analyzing the SPS and attribute data unit header.



|  |  |
| --- | --- |
| tlv\_encapsulation( ) { | **Descriptor** |
| **tlv\_type** | u(8) |
| if( tlv\_type == 4 ){ //Attribute data unit |  |
| **attribute\_component\_flag** | u(1) |
| if( attribute\_component\_flag ){ |  |
| **attribute\_instance\_id** | ue(v) |
| **known\_attribute\_label\_flag** | u(1) |
| if( known\_attribute\_label\_flag ) |  |
| **known\_attribute\_label** | ue(v) |
| else |  |
| **attribute\_label\_four\_bytes** | u(32) |
| } |  |
| } |  |
| **tlv\_num\_payload\_bytes** | u(32) |
| for( i = 0; i < tlv\_num\_payload\_bytes; i++ ) |  |
| **tlv\_payload\_byte**[ i ] | u(8) |
| } |  |

# Conclusion

In this contribution, we propose 4 new syntax of DIS text that need for G-PCC spec.

We recommend adopting these propose modification.

# References

[1] w19088, G-PCC DIS text of ISO/IEC 23090-9, ISO/IEC JTC 1/SC 29/WG 11, Oct., 2019, Geneva CH