INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JTC1/SC29/WG11 MPEG2012/M29231
April 2013, Incheon (KR)

Source **Telecom ParisTech, Canon Research Centre France**
Status **For consideration at the 104th MPEG meeting**
Title **Support for efficient tile access in the HEVC File Format**
Authors Jean Le Feuvre, Cyril Concolato, Franck Denoual, Frédéric Mazé, Eric Nassor, Nael Ouedraogo, Hervé Le Floch

1 Introduction

HEVC provides new coding tools compared to AVC that enable a decoder to partially decode only a region of interest in a large video. The tool in HEVC for that purpose is called “tiles”. This contribution shows that the efficient access to these tiles is not possible using the current HEVC file format. This contribution first describes the HEVC concepts related to tiles and then investigates how to describe these tiles using existing File Format tools and finally proposes new tools, based on extractors, to efficiently access them for the HEVC file format. Additional use cases in particular in the context of DASH are given in contribution m29232.

2 HEVC Tiling

HEVC (JCTVC-L1003) defines different spatial subdivision of pictures: *tiles*, *slices* and *slice segments*. These have been introduced for different purposes: the *slices* are related to streaming issues while the *tiles* and the *slice segments* have been defined for parallel processing.

A *tile* defines a rectangular region of a picture that contains an integer number of *Coding Tree Units* (*CTU*). The tiling is only defined by row and column boundaries as depicted on Figure 1.

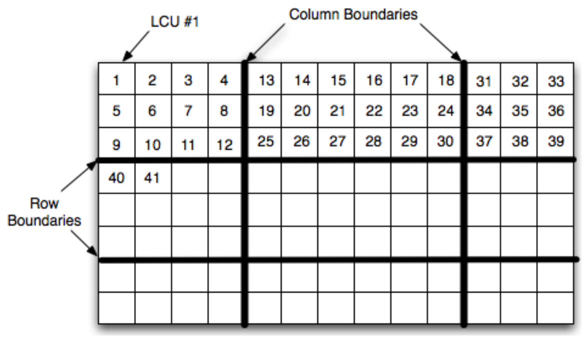

Figure 1: Picture partitioning into tiles

By looking at this figure, we clearly see that tiles are good candidates to represent regions of interests. However, the bitstream organization in terms of syntax and its encapsulation into Network Abstract Layer (NAL) units is rather based on *slices* (as in AVC).

A *slice* in HEVC is a set of *slice segments*, with at least the first *slice segment* being an *independent slice segment*, the others, if any, being *dependent slice segments*. A *slice segment* contains an integer number of consecutive (in raster scan order) *CTU*s. It has not necessarily a rectangular shape (thus less appropriate than tiles for ROI representation). A *slice segment* is encoded in the HEVC bitstream as a *slice_segment_header* followed by *slice_segment_data*. *Independent slice segments* and *dependent slice segments* differ by their header: the *dependent slice segment* has a shorter header because reusing information from the *independent slice segment*'s header. Both independent and dependent *slice segments* contain a list of entry points in the bitstream: either to tiles or to entropy decoding synchronization points.

To better understand the relationships between *slice, slice segments* and *tiles*, we illustrate different configurations below. These configurations differ from the previous figure that corresponds to a case where 1 *tile* has 1 *slice* (containing only 1 *independent slice segment*). On the left side of Figure 2, the picture is partitioned in 2 *tiles* and 1 *slice* (with 5 *slice segments*). On the right the picture is split in 2 *tiles,* the left *tile* having 2 *slices* (each with 2 *slice segments*), the right *tile* having 1 *slice* (with 2 *slice segments*). The HEVC standard defines organization rules between tiles and slice segments that can be summarized as follows (one or both conditions have to be met):

- All *CTU*s in a *slice segment* belong to the same *tile*.
- All *CTU*s in a *tile* belong to the same *slice segment*

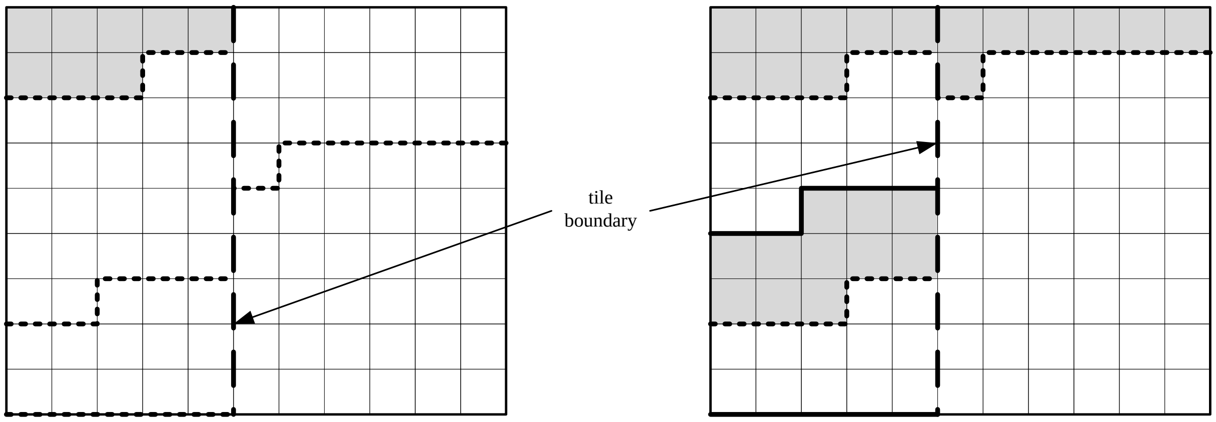

Figure 2: Different tile/slice configurations

While the tile is the appropriate support for regions of interest, the *slice segment* is the entity that will be actually put in NAL units for transport on the network and aggregated to form an access unit (coded picture or sample at file format level). In HEVC, the type of NAL unit is specified in the 2 bytes NAL unit header. For coded slice segment NAL unit, the *slice_segment_header* indicates via the *slice_segment_address* syntax element the address of the 1st coding tree block in the slice segment. The tiling information is provided in a PPS (Picture Parameter Set) NAL unit. The relation between a *slice segment* and a *tile* can then be deduced from these parameters.

While by definition, on tiles borders, the spatial predictions are reset, nothing prevents a *tile* to use temporal predictors from a different *tile* in the reference frame(s). In order to build independent *tiles*, we then constrain at encoding time the motion vectors for the prediction units inside a *tile* to remain in the co-located *tile* in the reference frame(s). In addition, the in-loop filters (deblocking and SAO) have to be deactivated on the *tiles* borders so that no error drift is introduced when decoding only one *tile*. This control of the in-loop filters is already available in HEVC and is set in slice segment header with the **loop_filter_across_tiles_enabled_flag** flag. By explicitly setting this flag to 0, the pixels at the tiles borders won't depend on the pixels on the border of the neighbor tiles. When the 2 conditions on motion vectors and on in-loop filters are met, we then talk about "independently decodable tiles" or "independent tiles".

When a video sequence is encoded as a set of independent tiles, it then enables a tile-based decoding from one frame to another without any risk for missing reference data or propagation of reconstruction errors. This configuration then enables to reconstruct only a spatial part of the original video, for example corresponding to a region of interest. It can be useful to indicate as supplemental information of a video bitstream that this configuration has been used so that tile-based decoding is reliable. This is one object of the contribution JCTVC-L0049 from InterDigital.

3 Review of existing tools

The current ISOBMFF and AVC/SVC FF offer several tools that could potentially be used for describing and accessing HEVC tiles. These tools are:

- Subsamples
- Subsegments
- Extractors

3.1 *Subsample Information*

Subsample information, as illustrated in Figure 3, allows describing the structure of a sample, typically as a contiguous set of NALUs with the same properties. It provides sufficient information for on-demand subsample retrieval/decoding, for example to access NALU corresponding to the slice segments of a given tile in a sample.

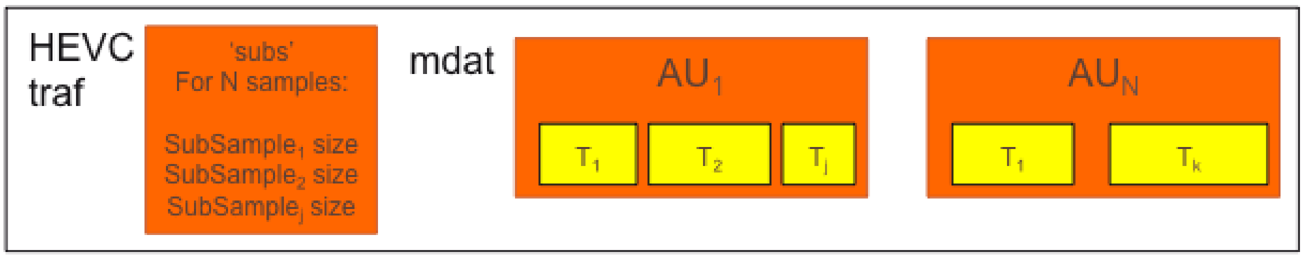

Figure 3 - Subsample information in movie fragments

In a client/server scenario, once subsample information is known at the client side, subsample byte ranges can be aggregated in one or several HTTP requests to retrieve only the needed part. This complicates the download, as only the *"moof"* box should be downloaded first to figure out the interesting parts of the samples in the associated *"mdat"* box; for each movie fragment in the DASH session, a reader has to issue a byte-range request large enough to include the *"moof"* box (or stops downloading once the *"moof"* box is received), and then issue *NxK* byte ranges in HTTP partial requests, where N is the number of samples in the *"moof"* box and K is the number of requested subsamples (e.g. NALUs corresponding to slice segments carrying tiles of interests). The situation gets even more complex and heavy when several movie fragments are present in one media segment. This creates a strong dependency between the media downloader and the media parser.

Furthermore, subsample information provides data partitioning of each sample but does not associate any meta-data with the subsample. If a predefined pattern is used for each sample, i.e. that the n-th NALU of each sample belongs to the same tile, it would be possible to extract only the NALU for a given tile. However, predefined patterns are not convenient practically, as encoders will typically produce a variable number of NALU per tile per sample and possibly in a different order for independently coded tiles. A possible solution is then to use MapGroups as defined in SVC file format, and defining how groups and tiers can be used for HEVC. There is currently no way to associate a given sample/sub-sample to a tile and this is proposed below.

3.2 Data Partitioning for DASH

Another tool for data partitioning introduced during the development of the DASH standard is the level assignment and subsegment-indexing tool. Samples within a file can be tagged to belong to different groups or different sub-tracks; these tags can then be assigned to different levels, which in turn can be used to describe contiguous byte ranges within one or several movie fragments ("moof" and associated "mdat" boxes) using the "ssix" box. Figure 4 illustrates how temporal IDs of an AVC stream may be mapped to different levels, thereby allowing all frames of the lower level to be downloaded in one byte-range request while skipping other levels.

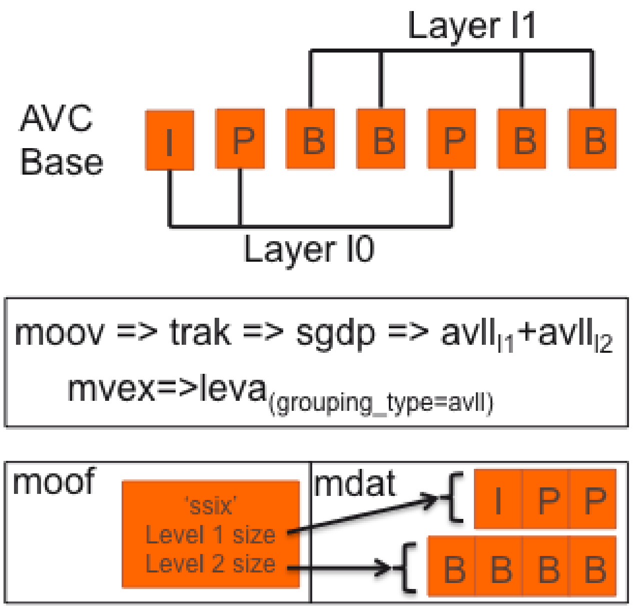

Figure 4 - Possible Level assignment for temporal layers of AVC

However, the level assignment does not allow for sub-sample data partitioning: it works at the track level and would only work for separation of HEVC temporal sub-layers from base layer, not for tiles.

3.3 SVC Data Partitioning

The SVC File format defines a very efficient tool for partitioning of SVC samples, by means of Extractors which replaces a NALU whose syntax and semantics are specific to the file format by another NAL unit (or set of) from another sample in another track. This allows for example storing SVC regions in independent tracks, as illustrated in Figure 5.

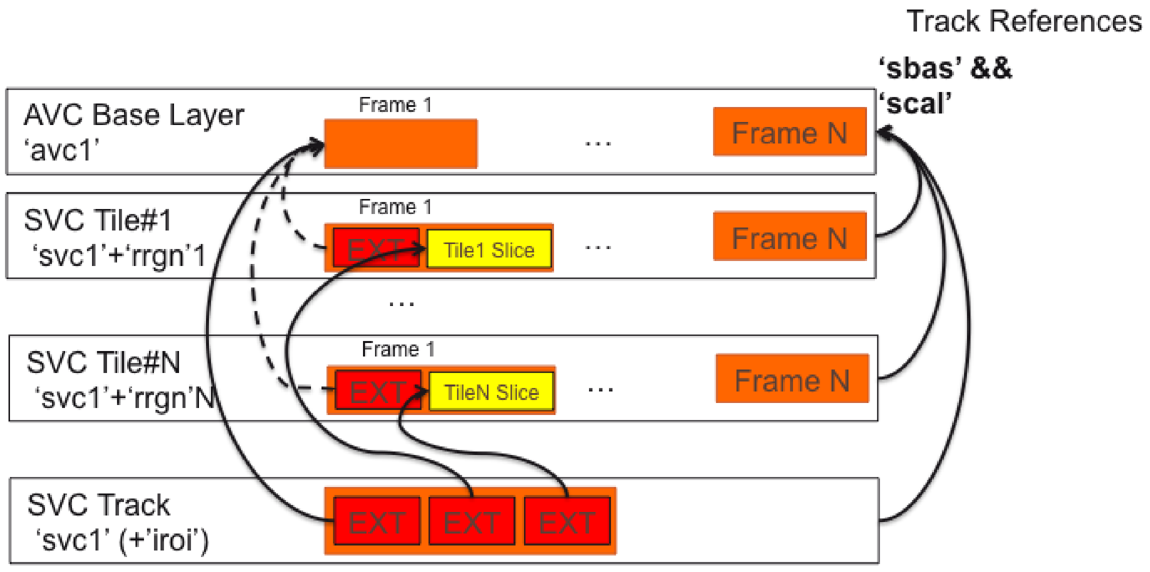

Figure 5 - Storage of SVC files in independently decodable tracks

Such a design allows very efficient HTTP fetching of SVC regions without requiring a complex DASH access engine, as each region can be stored in a single track or file and use existing DASH mechanisms, such as sub-representations or dependent representation, to be retrieved. However, this tool is very specific to SVC and needs modifications for HEVC.

4 Proposal

Based on the previous analysis of the existing file format tools, this contribution proposes new tools to satisfy the above use cases:

- The first tool (in section 4.1) consists in new sample groups used for identifying NALU belonging to a given tile.
- The second tool (in section 4.2) is the usage of extractors in the HEVC file format, using the same approach as the extractor already defined in part 15. This extractor is designed to be forward compatible with possible extensions needed for multi-view or scalable works deriving from HEVC. The notion of extractor is also refined with the notion of sub-layer tracks in HEVC file format, capable of containing temporal or spatial data in a dedicated track.

Finally, we propose (in section 4.3) a simplification for constant sample (or sub-sample) to group association in a file, which has a broader scope than this proposal.

4.1 TileRegion Sample Grouping

4.1.1 Overview

A high-level view of the tile grouping mechanism for an HEVC track is provided in Figure 6**Erreur ! Source du renvoi introuvable.**. It shows a possible usage of the new proposed sample group descriptions to define mapping between tiles and NAL units.

Each tile in the grid of tiles is described by a group description of type ‘trsg’ (for TileRegionSampleGroupEntry). A single SampleGroupDescriptionBox is used to gather all the tile descriptions. An HEVC sample can be described as a set of NALUs, through a grouping description of type ‘tlnm’ (for TileNALUMapEntry) giving the tile to NALU mapping. Each tile is assigned a unique identifier in the sample group description box, called tileID, which is used in the TileNALUMapEntry to associate a NAL unit to a tile.
Note: the ‘sgpd’ box of type ‘trsg’ may not be referenced from an ‘sbgp’ box directly. This may happen if all samples are associated with description by default grouping or if the association is given indirectly by the entries that have tileID values also used in the ‘sgpd’ box of type ‘tlnm’.

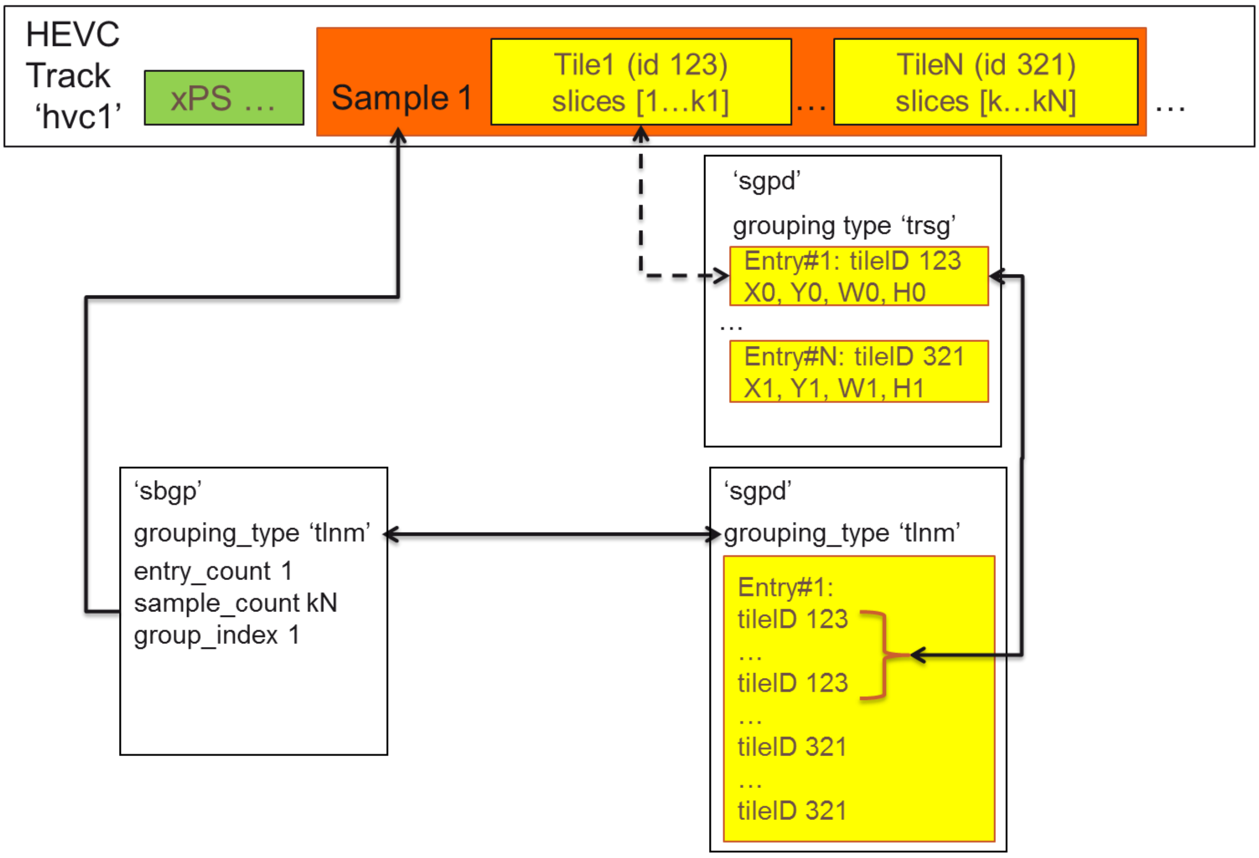

Figure 6 - Usage of tile group description in an HEVC track

4.1.2 Tile Region Sample Group Entry

Group Types:	‘trsg’
Container:	Sample Group Description Box (‘sgpd’)
Mandatory:	No
Quantity:	Zero or more, depending on the number of tiles in the HEVC bitstream

Syntax

```
class TileRegionSampleGroupEntry() extends VisualSampleGroupEntry ('trsg')
{
   unsigned int(16) tileID;
   unsigned int(16) horizontal_offset;
   unsigned int(16) vertical_offset;
   unsigned int(16) region_width;
   unsigned int(16) region_height;
   unsigned int(2) independent;
   unsigned int(6) reserved=0;
}
```

Semantics

`tileID` is a unique identifier for the tile described by this group. Value 0 is reserved for special use in the 'tlnm' box.
`horizontal_offset` and `vertical_offset` give respectively the horizontal and vertical offsets of the top-left pixel of the rectangular region represented by the tile, relative to the top-left pixel of the HEVC frame, in luma samples of the base region.
`region_width` and `region_height` give respectively the width and height of the rectangular region represented by the tile, in luma samples of the HEVC frame.

`independent` specifies that this tile has decoding dependencies only to the same tile in other samples as explained in section 2 (definition of independent tiles), as follows:

- If independent equals 0, the coding dependencies between this tile and other tiles in the same frame or previous frames is unknown,
- If independent equals 1, there are no spatial coding dependencies between this tile and other tiles in the same frame, no temporal dependencies between this tile and the other tiles with different tileID in any reference frames but there can be coding dependencies between this tile and the tile with the same tileID in the reference frames,
- If independent equals 2, there are no coding dependencies between this tile and other tiles with the same tileID in the same frame or in previous frames,
- Value 3 is reserved.

4.1.3 TileNALUMap Entry

Group Type: 'tlnm'
Container: Sample Group Description Box ('sgpd')
Mandatory: No
Quantity: Zero or More

Each sample is associated with a group_description_index in the SampleToGroupBox with grouping_type '`tlnm`'. A SampleGroupDescriptionBox with grouping_type '`tlnm`' contains a TileNALUMapEntry for each group_description_index.

```
class TileNALUMapEntry() extends VisualSampleGroupEntry ('tlnm') {
 unsigned int(6) reserved = 0;
 unsigned int(1) large_size;
 unsigned int(1) rle;
 if (large_size) {
   unsigned int(16) entry_count;
 } else {
   unsigned int(8) entry_count;
 }
 for (i=1; i<= entry_count; i++)
  if (rle) {
   if (large_size) {
    unsigned int(16) NALU_start_number;
   } else {
     unsigned int(8) NALU_start_number;
   }
  }
  unsigned int(16) tileID;
 }
```

```
}
```
Semantics

`large_size` indicates whether the number of NAL units entries in the track samples is represented on 8 or 16 bits.
`rle` indicates whether run-length encoding is used (1) to assign tile identifiers to NAL units or not (0).
`entry_count` specifies the number of entries in the map. Note that when `rle` is equal to 1, the `entry_count` corresponds to the number of runs where consecutives NAL units are associated with the same tile. When `rle` is equal to 0, `entry_count` represents the total number of NAL units.
`NALU_start_number` is the 1-based NALU index in the sample of the first NALU in the current run associated with tileID.
`tileID` is a unique identifier for the tile described by this group. If 0, no tile is associated to these identified NALUs.

4.2 *HEVC Extractors and sub-layer tracks*

4.2.1 Overview

In the previous example shown in Figure 6, all NALUs of all tiles are stored in the same track, which makes accessing an individual tile difficult. The extractors concept and the derived sub-layer concept can be used to improve that.

Figure 7**Erreur ! Source du renvoi introuvable.** illustrates the sub-layer track proposal for the use case of tiling. In this figure, a file contains N+1 tracks: a main track corresponding to the whole video and N tracks each corresponding to a different independently coded tile in the video. Each tile track contains the NAL units carrying the tile information. The track corresponding to the entire video contains general information (SPS, PPS, VPS …) and extractor NALU pointing to the tile tracks. A track reference of type 'subl' is added to the main track indicating that it references sub-tracks.
Additionally, each tile track uses the 'trsg' box (as defined in 4.1) to indicate the spatial part of the main track to which it corresponds.

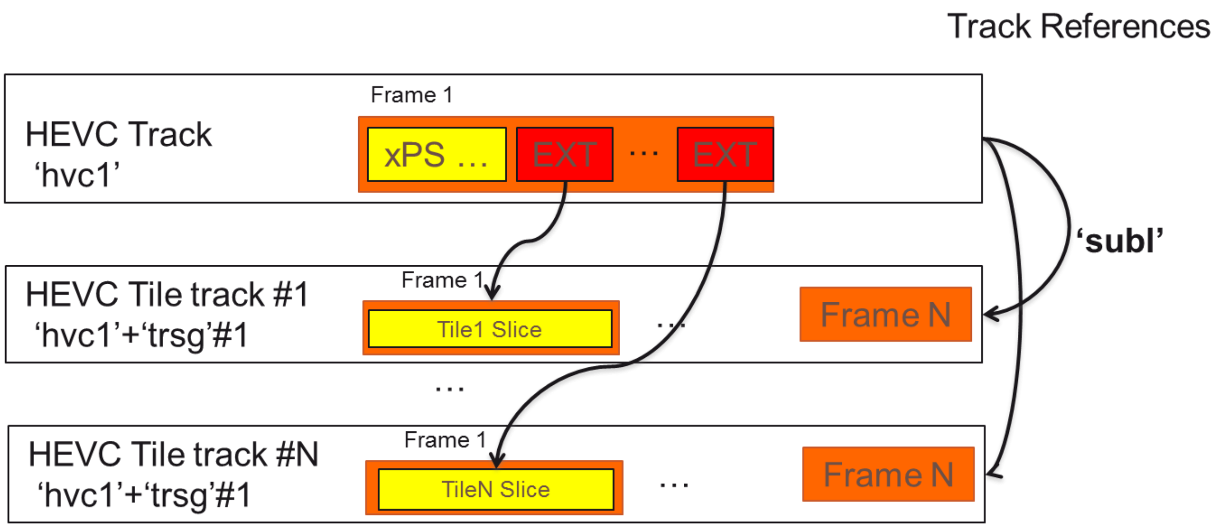

Figure 7 - Sub-layer tracks for HEVC

In Figure 7**Erreur ! Source du renvoi introuvable.**, the tile tracks cannot be processed individually since some needed information is contained in the main track (e.g. SPS, PPS …). The proposal also includes provision for being able to process a tile track individually, as illustrated in Figure 8. In this case, the tile track uses extractors to retrieve the missing information from the main track and an ‘hbas’ track reference is introduced. In case multiple independently decodable tracks are present, selection of the track to display is up to the implementation.

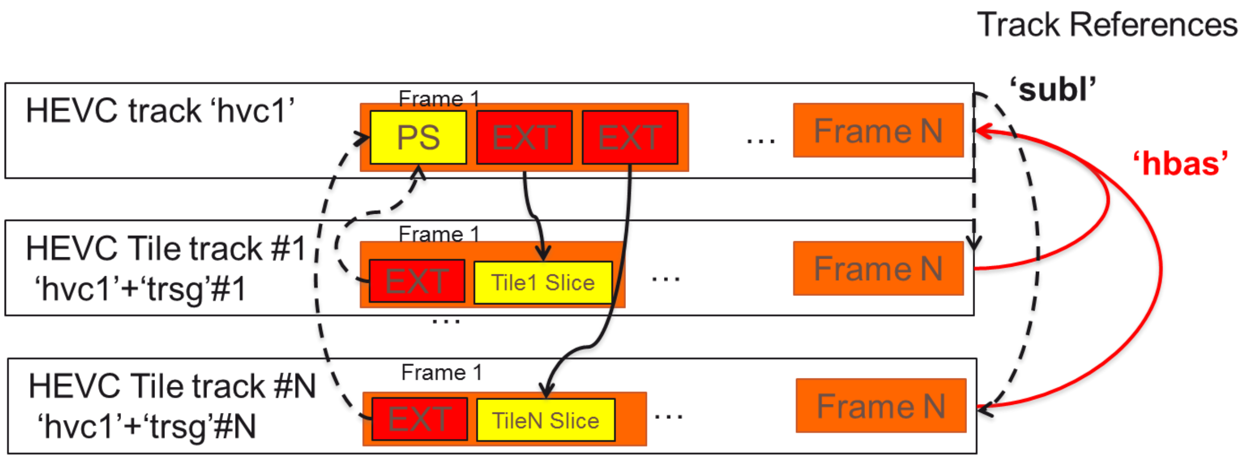

Figure 8 - Subsequence extraction through sub-layer tracks

4.2.2 HEVC Extractor specification text

We propose to extend the semantics of extractors as follows:

Replace

« `NALUnitHeader()`: the first four bytes of SVC and MVC VCL NAL units. nal_unit_type shall be set to the extractor NAL unit type (type 31). »

By

« `NALUnitHeader()`: For an SVC or MVC extractor, the first four bytes of SVC and MVC VCL NAL units. nal_unit_type shall be set to the extractor NAL unit type (type 31). For an HEVC extractor, the first two bytes of HEVC VCL NAL. `nal_unit_type` shall be set to the HEVC extractor NAL unit type (type 47). »

Add the following in the semantics:

« For an extractor referencing HEVC NAL units, the following shall apply:

`forbidden_zero_bit` shall be set as specified in ISO/IEC 23008-2.

`nal_unit_type` shall be set to 47.

nuh_layer_id and **nuh_temporal_id_plus1** shall be copied from the first NALU referenced by this extractor. An extractor in an HEVC track referencing HEVC nal units shall not reference several NAL units with different **nuh_layer_id** and **nuh_temporal_id_plus1 values**.

sample_offset shall be set to 0. »

In the definition of extractor, change

« An extractor contains an instruction to extract data from another track, which is linked to the track in which the extractor resides, by means of a track reference of type '`scal`'. »

with

« An extractor contains an instruction to extract data from another track, which is linked to the track in which the extractor resides, by means of a track reference of type '`scal`' for SVC and MVC tracks, and of type ‘`subl`’ for HEVC.»

Change

« The bytes are copied only from the single identified sample in the track referenced through the indicated ‘`scal`’ track reference. »

with

« The bytes are copied only from the single identified sample in the track referenced through the indicated '`scal`' (for SVC or MVC) or '`subl`' (for HEVC) track reference. »

4.2.3 HEVC Sub-layer specification text

We propose the following modifications to the HEVC file format specification:

In 8.1 add the following in the list of tools for supporting HEVC contents :
« - sub-layer tracks and extractors allowing to group layered data, whether temporal or spatial, in dedicated tracks »

In 8.4.1.1, add the following:
« In order to form the intended HEVC stream, HEVC Extractors as defined in Annex A, if present, shall be replaced with the data they are referencing. Decodable sub-set(s) of the HEVC bitstream may be obtained by ignoring some Extractors pointing to independent tiles or temporal sub-layers.»

Add a new subsection to the HEVC file format specification:
"8.X Sub-layer tracks

Sub-layer tracks are tracks containing parts of an HEVC coded sequence that can be discarded without harming the decoding process of other HEVC NAL units, i.e. discardable sub-samples as defined in 8.4.8 or sub-samples belonging to an independent *TileRegion* sample group entry. Such sub-samples are referred from the HEVC non-sub-layer track using extractors. Other types of sub-samples shall not be stored in sub-layer tracks. Meta-data associated with samples and sub-samples, such as sync flags or sample to group maps, may be set in the sub-layer track, but are overridden by corresponding meta-data associated with the corresponding extractor(s) in the base HEVC track.
HEVC tracks using references to sub-layers shall have a track reference of type 'subl' to the referred sub-layer track(s).
A sub-layer track may be authored with extractors referring to the base track, in order to extract a valid HEVC sequence from a file with multiple sub-layer tracks; in this case and only in this case, the sub layer track shall have a track dependency of type 'hbas' to the base HEVC track.
"

In HEVC configuration record, replace the first reserved(6) bits by :

bit(1) sublayer_representation;
bit(5) reserved = '11111'b;

And add in the semantics:

"sublayer_representation : if set, indicates this HEVC track is a sub-layer track and does not contain the complete HEVC bitstream. Tracks with *sublayer_representation* set to 1 shall have all fields in the associated hvcC copied from the base HEVC track, except from:

- *sublayer_representation* which is set to 1
- numArrays which is set to 0 (i.e. sub-layer tracks shall not contain any NAL array in their configuration)."

4.3 Default Sample to Group Box

4.3.1 Overview

Sample groups provide an efficient and extensible tool to add meta-data to individual samples. It is however possible that the given meta-data may be valid for all the samples of a given track; specifically for the case of sub-layer tracks containing a single tile, a single SampleGroupDescription of type 'trsg' could be specified in the sub-layer track, and used by every sample in the track. To indicate that, each sample has to be flagged for such group(s), which is quite inefficient. We therefore suggest allowing some sample groups to be marked as “default”, i.e. valid for all samples. We suggest to use flags of the SampleGroupDescriptionBox (full box) to describe this.

4.3.2 Specification text

In 8.9.3.2, replace

```
extends FullBox('sgpd', version, 0){
```

with

```
extends  FullBox('sgpd',  version,  flags){  if  ((version==1)  ||  (version==2))  {
unsigned int(32) default_length; }
```

In 8.9.3.3, add the following:

“The following flags are defined for sgpd :
`0x000001 sample-group-is-default`: indicates that all samples in this track or in the current fragment are assigned to this group. If several entries are defined in a default SampleGroupDescriptionBox, all entries apply to all samples in the track or traf.”

5 Conclusion

We recommend adoption of the proposed tools in the HEVC file format.