

JVET-K0260 Flexible Tiles



Rickard Sjöberg, Mitra Damghanian, Martin Pettersson, Jack Enhorn
Ericsson

Tiles in HEVC



- The HEVC tile structure is signaled in the PPS
- Tile row and tile column sizes are specified in CTU units
- Tile partitioning boundaries span across the entire picture
- Tiles provide parallel processing functionality
- The bit cost of using tiles consist of
 1. Prediction restrictions between tiles of the same picture. This includes e.g. intra prediction, context derivation and motion vector prediction
 2. Prediction restrictions between tiles in different pictures apply if MCTS are used
 3. The tile syntax shown to the right and bit-stream offset indicators for each tile in the slice header
 4. CABAC is initialized for each tile
 5. CABAC engine is flushed after each tile, and the bitstream is byte-aligned
- All of these except item 3 also apply to slices
 - But slices comes with a slice header bit cost

pic_parameter_set_rbsp() {	Descriptor
...	
tiles_enabled_flag	u(1)
...	
if(tiles_enabled_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
if(!uniform_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
row_height_minus1[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
...	

Proposal



- We propose to add support for Tiles in VTM
- The following HEVC tile properties are included in the proposal:
 1. That CABAC is initialized for each tile, flushed after each tile and that the tiles are byte aligned
 2. That tiles break predictions as in HEVC
 3. That the tile structure is specified in the PPS
 4. That tile pointers are mandatory for all tiles except the first in a slice
 5. That either all CTUs in a tile belongs to the same slice or all CTUs in a slice belong to the same tile
 6. That the initial QP of a tile is set as in HEVC
 7. That tiles are CTU aligned
 8. That tiles can only have rectangular shape
- The following tile properties different to HEVC tiles are also proposed:
 1. That tile boundaries are not required to span across the entire picture
 2. That tiles are specified individually in raster scan order by their individual tile size either by copying the tile size from the previous tile size in decoding order or by one tile width and one tile height UVLC code word each
 3. That the number of tiles in a picture is signaled in the PPS
 4. That the tile sizes are specified in subtile units to reduce the signaling bit cost

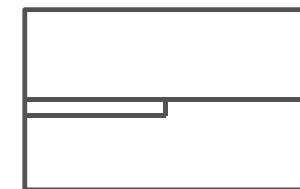
Benefits of flexible tiles



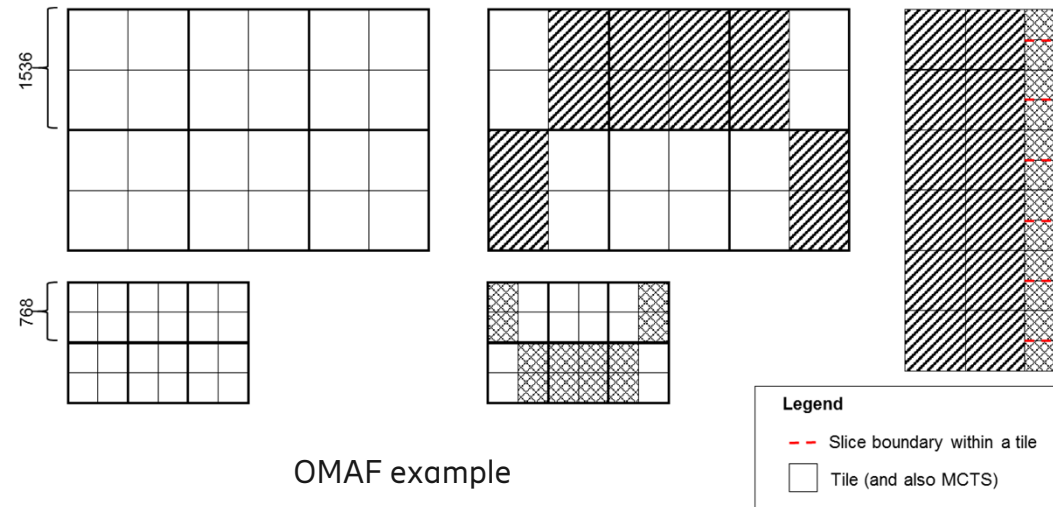
- Provide flexibility to partition pictures into rectangular-shaped CTU-based independent segments
- In HEVC, not all rectangular partitions can be expressed using combination of tiles and slices, such as the example to the right
 - Flexible tiles have less overhead and less boundary artifacts than HEVC partitioning since only necessary boundaries are needed
- HEVC 360 video streaming with mixed resolution tiles require a mix of tiles and slices
 - The example from OMAF and VR Industry forum (VRIF) to the right consist of 24 segments realized with a mix of tiles and slices
 - Slices are more expensive than tiles in terms of bit cost
 - It is cleaner to use tiles as the only partition tool instead of tiles and slices in combination
 - Re-arranging tiles only does not require re-writing of any additional slice headers



Can not be expressed with HEVC

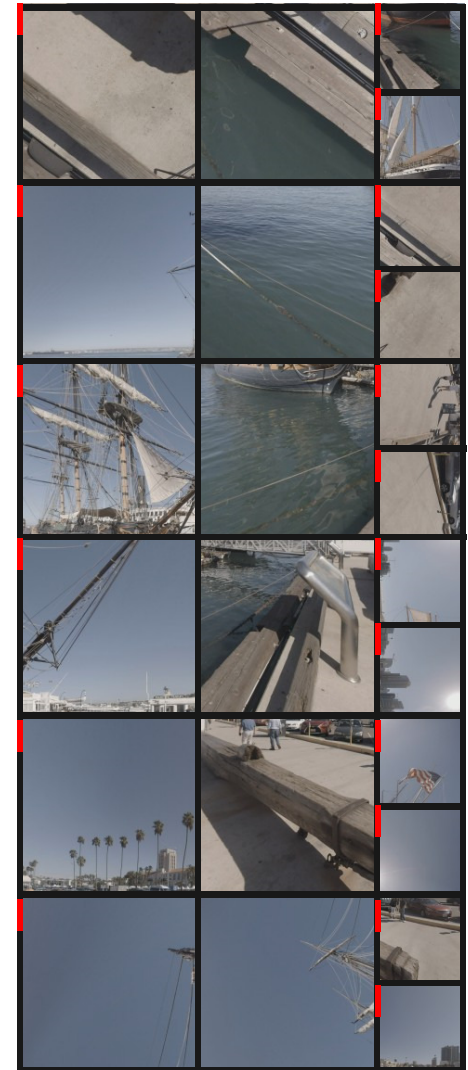


Can be expressed with HEVC



Experiment

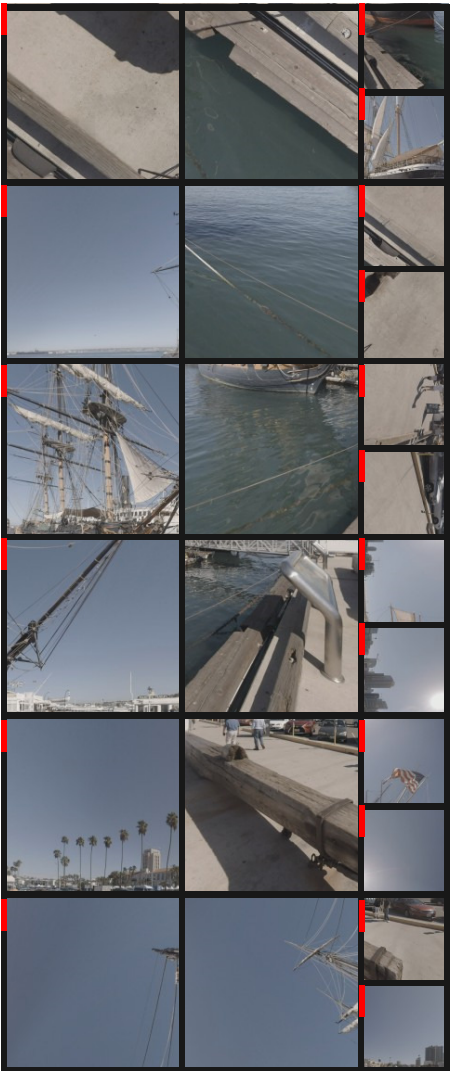
- The partition example from OMAF and VRIF was used
 - 24 tiles per picture as shown to the right
- The coded format is 1920x4608 (level 5)
- The 8192x4096 8-bit sequences from the VCC CfP were coded using RA configuration
- HM 16.18 was used as anchor
 - 24 segments as shown to the right
 - 18 tiles by three tile columns and six tile rows
 - HEVC tiles are restricted such that all CTUs in a slice shall belong to the same tile or all CTUs in a tile shall belong to the same slice
 - 18 slices per picture were used as shown in red in the picture
- HM 16.18 with flexible tiles was used as test
 - 24 tiles with one slice header per picture



Harbor sequence in OMF example format

Results - Overall

Anchor	HM 16.18		
Test	HM 16.18 flexible tiles		
RA	Y	U	V
Trolley360	-0.9%	-0.8%	-0.8%
KiteFlite	-0.7%	-0.6%	-0.6%
Harbor	-1.6%	-1.6%	-1.7%
Overall	-1.1%	-1.0%	-1.1%



Harbor sequence in OMA example format

Results - Harbor



Harbor	HM 16.18				HM 16.18 flexible tiles			
RA	Rate	PSNR			Rate	PSNR		
QP	kbps	Y	U	V	kbps	Y	U	V
37	1120.074	37.0114	45.88007	46.07331	1074.619	37.01161	45.87958	46.07306
32	2272.538	39.7072	47.75722	47.82981	2228.15	39.70656	47.75708	47.83188
27	4666.663	42.54406	49.98094	49.70781	4624.85	42.54374	49.98112	49.70805
22	10027.96	45.31871	51.76117	51.08042	9992.258	45.31864	51.76190	51.08134

Results - KiteFlite



KiteFlite	HM 16.18				HM 16.18 flexible tiles			
RA	Rate	PSNR			Rate	PSNR		
QP	kbps	Y	U	V	kbps	Y	U	V
37	2776.758	34.22481	43.60507	44.74521	2732.590	34.22464	43.60376	44.74446
32	5404.517	37.27455	45.44203	46.34727	5360.646	37.27396	45.44124	46.34620
27	10122.06	40.43554	47.74506	48.28711	10079.38	40.43563	47.74534	48.28699
22	19510.26	43.55319	49.91541	50.10256	19472.14	43.5533	49.91585	50.10268

Results - Trolley



Trolley	HM 16.18				HM 16.18 flexible tiles			
RA	Rate	PSNR			Rate	PSNR		
QP	kbps	Y	U	V	kbps	Y	U	V
37	2300.299	34.45433	44.32295	45.4591	2256.402	34.45415	44.32258	45.45919
32	4362.938	37.70431	46.29957	47.23547	4318.414	37.70388	46.29870	47.23303
27	7791.327	40.94506	48.75346	49.37865	7748.588	40.94521	48.75387	49.37947
22	13939.88	43.84974	50.85303	51.1286	13900.26	43.85007	50.85318	51.12964

Syntax



Proposed syntax

pic_parameter_set_rbsp() {	Descriptor
...	
tiles_enabled_flag	u(1)
...	
if(tiles_enabled_flag) {	
number_of_tiles_in_picture_minus2	ue(v)
subtile_width_minus1	ue(v)
subtile_height_minus1	ue(v)
for (i=0; i < number_of_tiles_in_picture_minus2+2 ; i++) {	
if (i > 0)	
use_previous_tile_size_flag	u(1)
if (use_previous_tile_size_flag == 0) {	
tile_width_minus1[i]	ue(v)
tile_height_minus1[i]	ue(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
...	

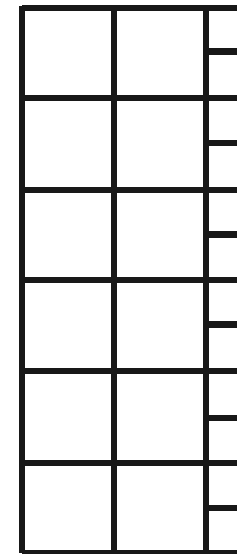
HEVC syntax

pic_parameter_set_rbsp() {	Descriptor
...	
tiles_enabled_flag	u(1)
...	
if(tiles_enabled_flag) {	
num_tile_columns_minus1	ue(v)
num_tile_rows_minus1	ue(v)
uniform_spacing_flag	u(1)
if(!uniform_spacing_flag) {	
for(i = 0; i < num_tile_columns_minus1; i++)	
column_width_minus1[i]	ue(v)
for(i = 0; i < num_tile_rows_minus1; i++)	
row_height_minus1[i]	ue(v)
}	
loop_filter_across_tiles_enabled_flag	u(1)
}	
...	

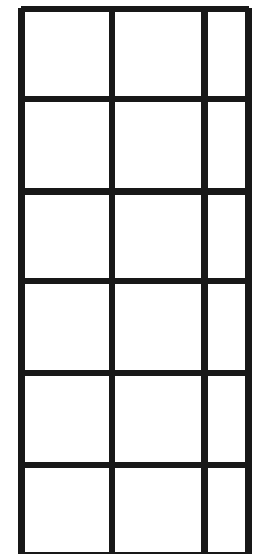
PPS bit cost for specifying the example partition



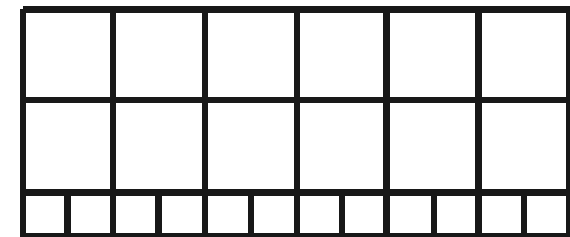
- Using HEVC tiles: 58 bits
 - 5 and 3 bits for specifying the number of rows and columns
 - 1 bit for uniform spacing (off)
 - $2*7+5*7$ bits for specifying the sizes of rows and columns
- Using flexible tiles with proposed syntax (rearranged structure): 50 bits
 - 9 bits for specifying the number of tiles minus2 (UVLC val=22)
 - 5+5 bits for specifying the subtile dimension minus1 (UVLC val=5)
 - 23 copy flag bits
 - 3+3 bits for specifying the first large tiles minus1 (UVLC val=1)
 - 1+1 bits for specifying the first small tile minus1 (UVLC val=0)
- Using flexible tiles without subtiles and copy flags: 297 bits
- Using flexible tiles without copy flags: 115 bits



Partition structure



HEVC tile structure



Re-arranged partition structure

Proposed semantics



tiles_enabled_flag equal to 1 specifies that there is more than one tile in each picture referring to the PPS. **tiles_enabled_flag** equal to 0 specifies that there is only one tile in each picture referring to the PPS. It is a requirement of bitstream conformance that the value of **tiles_enabled_flag** shall be the same for all PPSs that are activated within a CVS.

If **tiles_enabled_flag** is equal to 1, the value of the variable **tile_id** and the value of the two-dimensional array **TileId** are specified as follows:

```
tile_id = 0
for( y= 0; y < PicHeightInCtbsY; y++ )
    for( x = 0; x < PicWidthInCtbsY; x++ )
        TileId[ x ][ y ] = “no tile”
```

number_of_tiles_in_picture_minus2 plus 2 specifies the number of tiles in a picture. **number_of_tiles_in_picture_minus2** shall be in the range of 0 to **PicSizeInCtbsY** – 2, inclusive.

subtile_width_minus1 plus 1 specifies the width of the subtile unit in coding tree units. **subtile_width_minus1** shall be in the range of 0 to **PicWidthInCtbsY** – 1, inclusive.

subtile_height_minus1 plus 1 specifies the height of the subtile unit in coding tree units. **subtile_height_minus1** shall be in the range of 0 to **PicHeightInCtbsY** – 1, inclusive.

use_previous_tile_size_flag equal to 1 specifies that the size of the current tile is equal to the size of the previous tile. **use_previous_tile_size_flag** equal to 0 specifies that the size of the current tile is not equal to the size of the previous tile. When not present, the value of **use_previous_tile_size_flag** is inferred to be equal to 0.

When **use_previous_tile_size_flag** is equal to 1, the value of the *i*'th element of the arrays **tile_height_minus1** and **tile_width_minus1** are derived as follows:

```
tile_height_minus1[ i ] = tile_height_minus1[ i - 1 ]
tile_width_minus1[ i ] = tile_width_minus1[ i - 1 ]
```

Proposed semantics



tile_width_minus1[i] plus 1 specifies the width of the i-th tile in tile units.

tile_height_minus1[i] plus 1 specifies the height of the i-th tile in tile units.

The value of the variable **tile_id** and the value of the two-dimensional array **CTU_tile_assignment** are derived as follows:

```
for( tile_coordinate_y = 0; tile_coordinate_y < PicHeightInCtbsY; tile_coordinate_y++ )
    for( tile_coordinate_x = 0; tile_coordinate_x < PicWidthInCtbsY; tile_coordinate_x++ )
        if( TileId[ tile_coordinate_x ][ tile_coordinate_y ] == “no tile” )
            break
for( ctu_y = 0; ctu_y < ( tile_height_minus1[ i ] + 1 ) * ( subtile_height_minus1 + 1 ) && ( tile_coordinate_y + ctu_y ) < PicHeightInCtbsY ; ctu_y++ )
    for( ctu_x = 0 ; ctu_x < ( tile_width_minus1[ i ] + 1 ) * ( subtile_width_minus1 + 1 ) && ( tile_coordinate_x + ctu_x ) < PicWidthInCtbsY ; ctu_x++ )
        TileId[ tile_coordinate_x + ctu_x ][ tile_coordinate_y + ctu_y ] = tile_id
tile_id++
```

It is a requirement of bitstream conformance that the break statement is reached for each value of i.

It is a requirement of bitstream conformance that the value of **TileId[tile_coordinate_x + ctu_x][tile_coordinate_y + ctu_y]** shall be equal to “no tile” prior to the assignment to the value **tile_id**.

It is a requirement of bitstream conformance that after all assignments of **TileId**, **TileId[x][y]** shall not be equal to “no tile” for x ranging from 0 to **PicWidthInCtbsY-1** inclusive and y ranging from 0 to **PicHeightInCtbsY-1** inclusive.

NOTE X – The derivation of the variable **tile_id** and the value of the two-dimensional array **CTU_tile_assignment** as specified in the semantics of **tile_height_minus1[i]** are executed for each value of i. This means that the derivation is done after each loop of the variable i, including when **use_previous_tile_size_flag** is not present, when **use_previous_tile_size_flag** is equal to 0 and when **use_previous_tile_size_flag** is equal to 1. Likewise, the bitstream conformance condition that the value of **TileId[tile_coordinate_x + ctu_x][tile_coordinate_y + ctu_y]** shall be equal to “no tile” prior to the assignment to the value **tile_id** shall be true for any assignment done in any loop.

Proposed level change vs HEVC



Level	Max luma picture size MaxLumaPs (samples)	Max CPB size MaxCPB (CpbVclFactor or CpbNalFactor bits)		Max slice segments per picture MaxSliceSegments PerPicture	Max # of tile rows MaxTileRows	Max # of tile columns MaxTileCols	Max # of tiles per picture MaxTiles
		Main tier	High tier				
1	36 864	350	-	16	1	1	1
2	122 880	1 500	-	16	1	1	1
2.1	245 760	3 000	-	20	1	1	1
3	552 960	6 000	-	30	2	2	4
3.1	983 040	10 000	-	40	3	3	6
4	2 228 224	12 000	30 000	75	5	5	25
4.1	2 228 224	20 000	50 000	75	5	5	25
5	8 912 896	25 000	100 000	200	11	10	110
5.1	8 912 896	40 000	160 000	200	11	10	110
5.2	8 912 896	60 000	240 000	200	11	10	110
6	35 651 584	60 000	240 000	600	22	20	440
6.1	35 651 584	120 000	480 000	600	22	20	440
6.2	35 651 584	240 000	800 000	600	22	20	440

Proposal summary



The following HEVC tile properties are included in the proposal:

1. That CABAC is initialized for each tile, flushed after each tile and that the tiles are byte aligned
2. That tiles break predictions as in HEVC
3. That the tile structure is specified in the PPS
4. That tile pointers are mandatory for all tiles except the first in a slice
5. That either all CTUs in a tile belongs to the same slice or all CTUs in a slice belong to the same tile
6. That the initial QP of a tile is set as in HEVC
7. That tiles are CTU aligned
8. That tiles can only have rectangular shape

The following tile properties different to HEVC tiles are also proposed:

1. That tile boundaries are not required to span across the entire picture
2. That the number of tiles in a picture is signaled in the PPS
3. That the tile sizes are specified in subtile units to reduce the signaling bit cost
4. That tiles are specified individually in subtile raster scan order by their individual tile size either by copying the tile size from the previous tile size in decoding order or by one tile width and one tile height UVLC code word each



The HEVC rule



One or both of the following conditions shall be fulfilled for each slice and tile:

- All CTUs in a slice belong to the same tile.
- All CTUs in a tile belong to the same slice.