MPEG-5 Essential Video Coding (EVC)

Presented by Ken McCann
Chairman of MPEG Ad hoc Group on EVC
Background
EVC Baseline profile
EVC Main profile
Testing results
Background
Coding efficiency improves with every new generation of video coding standard, but this is not the only factor that determines the industry choice of codec.

- MPEG-5 Essential Video Coding (EVC) aims to address use cases that are currently not well served by other MPEG and ITU-T standards.

EVC uses a new way to create video coding standards:

- Designed to meet both business and technical requirements.
- While remaining consistent with ITU-T/ITU-R/ISO/IEC common patent policy.

Overall goals for EVC:

- Encourage the timely publication of licensing terms to allow reliable business plans to be created.
- Coding efficiency at least as good as HEVC.
- Complexity suitable for practical real-time encoding.

Motivated by observing that the complex licensing structure for HEVC has resulted in only limited adoption in some market segments.
HEVC licensing structure is complex
EVC uses a novel profile structure

Baseline profile

• Includes only technologies that are more than 20 years old or that were submitted with a royalty-free declaration

Main profile

• Adds a small number of additional tools that each provide a significant improvement in terms of compression performance
• Each additional Main profile tool is isolated so that it can be switched off independently of other tools with limited loss of performance
• Contributors were encouraged to submit voluntary declarations on the timely publication of licensing terms

XXX company may have current or pending patent rights relating to the technology described in this contribution and, conditioned on reciprocity, is prepared to grant licenses under reasonable and non-discriminatory terms as necessary for implementation of the resulting ITU-T Recommendation | ISO/IEC International Standard. **Furthermore, XXX company is prepared to make the timely publication of applicable licensing terms within 2 years of FDIS stage either individually or as part of a patent pool.**
- **Initial Working Draft (WD1)** in January 2019
- **Committee Draft (CD)** text in July 2019
- **Draft International Standard (DIS)** text in October 2019
- **Final Draft International Standard (FDIS)** text in April 2020

### Timeline

<table>
<thead>
<tr>
<th>EVC</th>
<th>2019</th>
<th>2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>WD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DIS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FDIS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tool set of Baseline profile
Coding structure

• The coding structure in this section employed a **quad-tree based coding structure which can use blocks up to 64x64**. The quad-tree based coding algorithm was introduced in the early 1990s and an efficient splitting method was disclosed in 1994.


Intra prediction

• **5 directional prediction modes are employed; DC, horizontal (H), vertical (V), diagonal left (DL), diagonal right (DR).** A codeword for prediction mode of the current block is adaptively assigned by using a mapping table between symbol and prediction mode which is selected based on the prediction modes of neighbouring upper and left blocks.

Normal inter prediction

- Inter prediction using uni-directional block-based motion compensation and Bi-directional prediction in the MPEG-2 standard.
- **Temporal direct mode** using the motion vector of the temporally co-located block in H.263.
- **Three neighbouring motion vectors** and a motion vector of temporally co-located blocks

Modes for Inter prediction

- Using motion vectors of neighbouring blocks as candidates for the motion vector for the current block was introduced in the 1980s
- **The competition based motion vector coding** method was proposed in 1999
Signaling MVP index

- 5 MVPs (spatial neighbors, temporal, no prediction)
  - 3 Spatial MVPs: Left, Up, Up-right
  - 1 Temporal MVP
  - No prediction: (0, 0)

- List order
  - MVP0(Left) \(\rightarrow\) MVP1(Up) \(\rightarrow\) MVP2 (Up-right) \(\rightarrow\) MVP3 (Temporal) \(\rightarrow\) MPV4 (0, 0)
  - No list adaptation

- Prior art: Jong Won Kim, Sang Uk, Lee, Hierarchical variable block size motion estimation technique for motion sequence coding, OPTICAL ENGINEERING, VOL.33 NO.8, 1994.08
Baseline profile

-half & quarter-pel interpolation
  • 2D separable filter based on wiener based 8-tap filter for luma, 4-tap filter for chroma component
  • Fractional sample interpolation (Quarter-pel interpolation) was introduced at 1996.

Using multiple reference pictures
  • The usage of multiple reference frames was introduced at 1996.
  • Prior art: Multiframe Block MotionCompensated Video Coding for Wireless Channels][M. Budagavi]
Transform and quantization

• **A discrete cosine transform (DCT)** is applied to a residual block between an original block and the corresponding prediction block.

• The transform size is equal to the coding block size, i.e. from 2x2 to 64x64.

• **The range of the quantization parameter (QP) is from 0 to 51 and a scaling factor (SF) corresponding to each QP is defined by a look-up table.**


Loop filter

- **Deblocking filtering in H.263 Annex J** is used to remove blocking artifacts between coding blocks
- In Annex J of H.263, replace `UpDownRamp(d, STRENGTH)` with `Clip3(-STRENGTH, STRENGTH, d)`
- Prior art: Enhancement of the Telenor proposal for H.26L (Section 5)
Arithmetic coding

- The binary arithmetic coding scheme in JPEG Annex D is applied as the entropy coding engine of the proposed codec.

Coefficient Coding

- To code the transformed and quantized coefficient values, run/level symbols are generated after scanning with a zig-zag pattern.
Tool set of Main profile
**Binary and triple tree (BTT)**

- Binary and triple three splitting from the maximum CTU size
- Split shapes of BTT
  - BTT uses Bi-split and Tri-split with vertical or horizontal direction

- Configure of BTT
  - According to a configuration, allowed split modes and bits assignment are defined

Main profile: Block Partitioning

**Example of allowed split modes**

- **Current CU**
- **BI_VER_SPLIT**
- **BI_HOR_SPLIT**
- **TRI_VER_SPLIT**
- **TRI_HOR_SPLIT**

<table>
<thead>
<tr>
<th>Current CU</th>
<th>BI_VER_SPLIT</th>
<th>BI_HOR_SPLIT</th>
<th>TRI_VER_SPLIT</th>
<th>TRI_HOR_SPLIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>64x64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32x64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16x64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Split configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>log2_cu_12_max_size = 4 (real size is 64)</td>
</tr>
<tr>
<td>log2_cu_14_max_size = 6 (real size is 32)</td>
</tr>
<tr>
<td>log2_tri_split_max_size = 4 (real size is 64)</td>
</tr>
</tbody>
</table>

**Example of bits assignment**

- **Number of split modes 4**
- **Number of split modes 2**
- **Number of split modes 3**
- **Number of split modes 1**

<table>
<thead>
<tr>
<th>Current CU</th>
<th>BI_VER_SPLIT</th>
<th>BI_HOR_SPLIT</th>
<th>TRI_VER_SPLIT</th>
<th>TRI_HOR_SPLIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>64x64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32x64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16x64</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>b0</td>
</tr>
<tr>
<td>b1</td>
</tr>
<tr>
<td>b2</td>
</tr>
</tbody>
</table>

Example of allowed split modes
**Split Unit Coding Order (SUCO)**

- Signaling of child tree coding order for each vertical split
  - 1 flag for each vertical split (SPLIT_BI_VER, SPLIT_TR_VER, SPLIT_QT)
  - Same direction for above two and bottom two for SPLIT_QT

- Applied depth
  - MIN_SUCO_DEPTH: \( \min(\log_2\text{width}, \log_2\text{height}) \)
  - MAX_SUCO_DEPTH: \( \max(\log_2\text{width}, \log_2\text{height}) \)

- Intra/Inter prediction modification according to Left/Right availability
  - LR_10, LR_01, LR_11, LR_00

---

**Main profile: Block coding order**

- **Split Unit Coding Order (SUCO):**
  - Signaling of child tree coding order for each vertical split
  - 1 flag for each vertical split (SPLIT_BI_VER, SPLIT_TR_VER, SPLIT_QT)
  - Same direction for above two and bottom two for SPLIT_QT
  - Applied depth
    - MIN_SUCO_DEPTH: \( \min(\log_2\text{width}, \log_2\text{height}) \)
    - MAX_SUCO_DEPTH: \( \max(\log_2\text{width}, \log_2\text{height}) \)
  - Intra/Inter prediction modification according to Left/Right availability
    - LR_10, LR_01, LR_11, LR_00
**Merge/Skip modes**

- For the skip mode and the merge mode, a motion vector competition scheme is used to select a motion candidate from a given candidate set that includes spatial and temporal motion candidates.
- Only merge indices are transmitted for the block coded in merge or skip mode, and the current block inherits the inter prediction indicator, reference indices, and motion vectors from a neighboring block referred by the coded merge index.
- The merge candidate list for the merge/skip mode is constructed by inserting spatial candidates, temporal candidates, affine candidates, History MVP candidate and the conventional combined candidates and Zero candidates according to a pre-defined insertion order:
  - Spatial candidates 1-4.
  - TMVP candidates 5-7
  - History MVP candidate.
  - Combined candidates
  - Zero candidates

![Diagram of merge/skip modes](image-url)
**AMVR (Adaptive Motion Vector Resolution)**

- MVR (Motion Vector Resolution) competition including super MVR
  - $\frac{1}{4}$, $\frac{1}{2}$, 1, 2, 4
- MVR is applied on MVP and MVD.
- Coupling of MVR and MVP index
- Different MVP for the different MVR
  - More accurate MVP in fine MVR and less accurate in coarse MVR
  - Fixed one MVP for each MVR, based on the hit-ratio of each MVP
MMVD (Merge with Motion Vector Difference)

- Simplified MV expression method based on statistical MVD modeling
  - 3 components (starting point, motion magnitude, motion direction)
- High probability points around MV of skip-candidate list are considered.
  - Starting point: [0~N] (N: MAX number of skip candidate)
  - Motion magnitude: [1/4, 1/2, 1, 2, 4, 8, 16, 32]
  - Motion direction: 4 directions (+0), (-0), (0,+), (0,-)
- Prediction directions (Bi, Uni0, Uni1) of base candidate are all considered.

Search points based on MVD's statistical information

UMVE search process
**Affine**

- **AF_INTER**: model is adaptively selected in block level
  - 4-parameters model, represented by two motion vectors (top-left and top-right corner) of a CU
  - 6-parameters model, represented by three motion vectors (top-left, top-right and bottom-left corner) of a CU
  - Extrapolated affine MVPs similar to affine merge are inserted into beginning of MVP list
- **AF_MERGE**: up to 5 candidates
  - Extrapolated affine candidates from neighbor coded affine block, with the order: F, D, E, G, A
  - Construct several new affine models from the motion vectors of spatial and temporal neighbor blocks
Decoder Side Motion Vector Refinement (DMVR)

- Process invoke
  - Bi-predicted in skip or direct merge mode
- Integer MV offset search, with fractional pel MV offset derived by error surface
- Search range set as 2 samples to reduce memory access.
- Mean-removal SAD is used as matching cost
- Sub-CU 16x16 for large CU size
History based MVP (HMVP)

- History-based Motion Vector Prediction (HMVP) method is an inter coding tool which can be applied to both merge candidate list and Advanced motion vector prediction (AMVP) candidate list construction process.
- In HMVP, a table of HMVP candidates is maintained and updated on-the-fly. Whenever a non-affine inter coded block is decoded, the decoded motion information is used to update the HMVP table in the last position following a First-In-First-Out (FIFO) rule to remove and add entry.
- Subsampling-based HMVP fetching, ratio 1:4
- HMVP candidate inserted following the TMVP for merge and following SMVP for AMVP

Diagram:

1. Load a table with HMVP candidates
2. Decode a block with HMVP candidates
3. Update the table with decoded motion information
Adaptive Loop Filter architecture:

- Design similar to that in VVC
- Filter support:
  - Diamond shape
  - Up-to 7x7 filter support
- Classification:
  - 25 classes
  - 4x4 block bases classifier
- CTB-level flexibility
- Set of fixed filters
  - 64 filters
  - Size of 7x7
- Adaptation Parameter Set (APS)
  - Reusage of the ALF coefficients signaled in APS NAL units
  - An identification index syntax element signaled in the slice header
  - APS buffer size up to 32 entries
Hadamard transform domain filter (HTDF)

- Hadamard transform domain filter (HTDF) is applied to luma reconstructed blocks if quantization parameter is larger than 17. The filter parameters are explicitly derived from the coded information. The filtering is performed on decoded samples right after block reconstruction in domain of Hadamard transform of size 2x2.

![Diagram of Hadamard transform domain filter (HTDF)]

### Diagram:

- **Scan**
- **A** – current pixel
- **BCD** – neighboring pixels
- **A′** B′
- **C′** D′
- **Hadamard transform**
- **Filtering with look-up table**
- **Inverse Hadamard transform**
- **To accumulation buffer**

### Equations:

- **A** B C D
- **r(0) r(1) r(2) r(3)**
- **R(0) R(1) R(2) R(3)**
- **F(0) F(1) F(2) F(3)**
- **f(0) f(1) f(2) f(3)**
- **A′ B′ C′ D′**
Advanced coefficient coding

Key elements:
• Zig-zag scan in inverse order + Last X/Y signaling;
• Chunk transform coefs processing (size of 16);
• Bit-planes context coding;
• Escape to bypass (8 flagA or 1 flagB).

Signaling:
• Context coded (number of contexts):
  • Last X/Y 56
  • Significant flag: 47
  • LevelA & B flag: 18
  • Dependency on causal neighbors (up-to 5).
• TR+Golomb: RemainLevel.
• FLC: Sign Flag
Adaptive transform selection (ATS)

**Intra block**
- CU level on/off decision
  - Applied block size: Up to 32x32
  - DCT2 vs ATS
  - 1 flag for the use of ATS based on RDO evaluation
- Kernel selection
  - 2 flags for used kernels of vertical and horizontal directions

**Inter block**
- TU may be smaller than the CU, and the TU shape & position info is signaled
- Two TU shapes & Two TU positions
- Position dependent core transform: DCT-8 for pos 0, DST-7 for pos 1
**Intra block copy in main profile**

- 128x128 reconstructed samples from the neighboring left CTU can be used as IBC reference purpose.
- Once any of the 64x64 unit reference sample memory in a 128x128 CTU begins to update with the reconstructed samples from the current CU, the previous stored reference samples in the whole 64x64 unit become unavailable for IBC reference purpose.
- IBC mode (IBC flag) is signaled in a CU level. The IBC mode is considered as a prediction mode other than intra and inter prediction modes. The is no need to include current picture as one of the reference pictures in the reference picture list 0. The motion vector of IBC is derived in integer pixel.
- The coding of block vector (BV) is straightforward without using prediction. The coding engine reuses the one used in mvd coding. The BV considered is in integer resolution.
- The maximum allowed block size for IBC mode is signaled in the SPS level.
Testing results
## Coding performance of EVC Baseline profile

### Testing condition
- EVC Baseline profile, ETM3.0
- Anchor: H.264/AVC (JM19.0)

### Coding performance

<table>
<thead>
<tr>
<th></th>
<th>Over JM19.0 (Random Access)</th>
<th></th>
<th>Over JM19.0 (Low Delay)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Y</td>
<td>U</td>
<td>V</td>
</tr>
<tr>
<td>Class A</td>
<td>-38.0%</td>
<td>-33.7%</td>
<td>-38.4%</td>
</tr>
<tr>
<td>Class B</td>
<td>-24.8%</td>
<td>-27.8%</td>
<td>-26.9%</td>
</tr>
<tr>
<td>Class E</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td>-31.4%</td>
<td>-30.8%</td>
<td>-32.7%</td>
</tr>
</tbody>
</table>
## Coding performance of EVC Main profile

### Testing condition
- EVC Main profile, ETM3.0
- Anchor: HEVC (HM16.6)

### Coding performance

<table>
<thead>
<tr>
<th></th>
<th>Over HM16.6 (Random Access)</th>
<th>Over HM16.6 (Low Delay)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Y</td>
<td>U</td>
</tr>
<tr>
<td><strong>Class A</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-30.0%</td>
<td>-27.4%</td>
</tr>
<tr>
<td><strong>Class B</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-23.1%</td>
<td>-23.8%</td>
</tr>
<tr>
<td><strong>Class E</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Overall</strong></td>
<td>-26.5%</td>
<td>-25.6%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EVC Baseline profile

H.264/AVC
11.0Mbps

EVC
6.5Mbps
Thank you!

With special thanks to all AhG members who contributed to this work, especially Kiho Choi who created many of these slides.