Selected coding methods in H.265/HEVC

Andreas Unterweger

Salzburg University of Applied Sciences

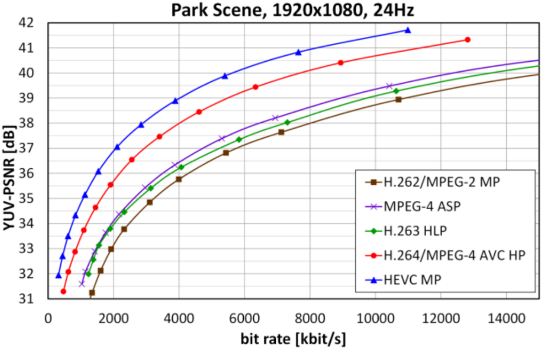
May 29, 2017

What is HEVC?

- HEVC = **H**igh **E**fficiency **V**ideo **C**oding
- Newest video coding standard (ratified in 2013)
- Developed by the Joint Collaborative Team on Video Coding (JCT-VC)
- Also published as part 2 of MPEG-H (similar to MPEG-4 suite)
- Updated three times so far (latest version ratified in late 2016)
- Predecessor: H.264 (ratified 2003 with 21 updates until 2014), used on Blu-rays, in DVB-T(2), DVB-S(2), DVB-C(2) and many more

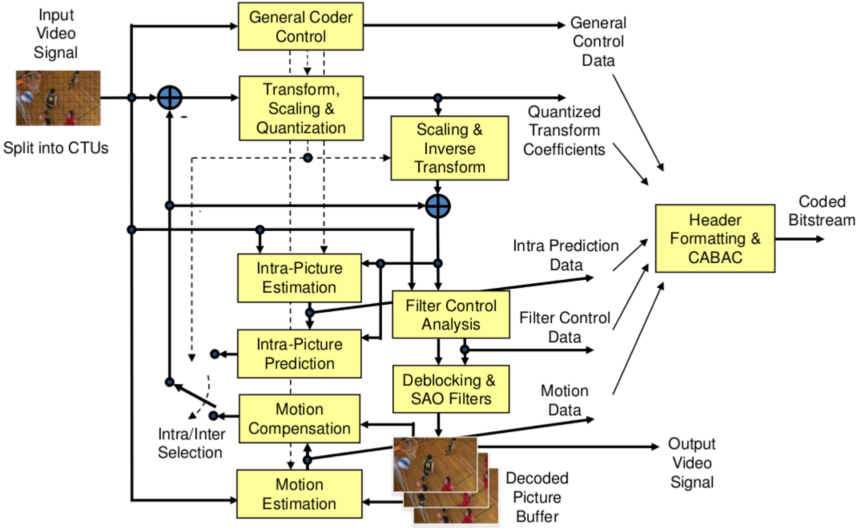
What is expected of HEVC?

- Higher coding efficiency than H.264 (approx. 50%)
- Higher complexity than H.264 to achieve this
- Support for higher resolutions for future video formats
- Encoding and decoding parallelizable in many ways



Source: Ohm et al. (2012)

HEVC architecture



Source: Sullivan et al. (2012)

About this talk

- HEVC is strongly based on H.264 $\rightarrow$ used for comparison
- Overview of new and modified coding tools
- Focus on most significant changes
- Main sources:
  - Overview paper of Sullivan et al.[1]
  - The latest H.265 standard[2]
  - The latest reference software[3] (HM)

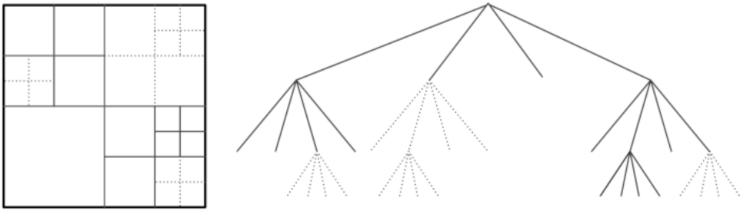
[1] http://iphome.hhi.de/wiegand/assets/pdfs/2012_12_IEEE-HEVC-Overview.pdf

[2] https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-H.265-201612-I!!PDF-E&type=items

[3] https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/trunk/

New block structure I

- H.264: 16x16 macroblocks
  - Intra blocks: 16 4x4 sub-blocks or 1 16x16 block
  - Inter blocks: 16x16, 16x8, 8x16, 8x8 (and subpartitions)
- HEVC: Coding tree blocks (CTBs)
  - 16x16, 32x32 or 64x64 blocks
  - Quadtree-like subpartitioning into coding blocks (CBs)
  - Minimum CB size: 8x8 (or larger if specified)
  - Chroma partitioned accordingly



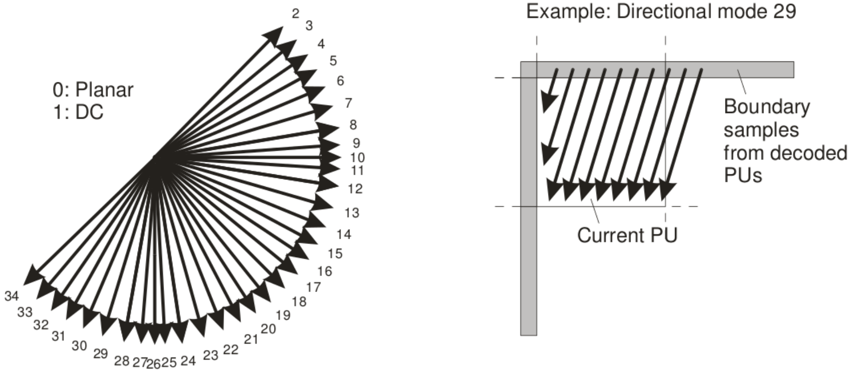
Source: Sullivan et al. (2012)

New block structure II

- H.264: Prediction and transform "static"
  - Prediction is coupled with block partition size
  - Transform size is always 4x4 (or adaptively 8x8 in High profile)
  - Intra/inter decision on higher (16x16 block) level
- HEVC: Prediction and transform flexible
  - CBs split into (min. 4x4) prediction blocks (PBs)
  - CBs split into (min. 4x4) transform blocks (TBs)
  - TB structure may be further partitioned than the PB structure
  - Intra/inter decision on (min. 8x8) CB level

Intra prediction I

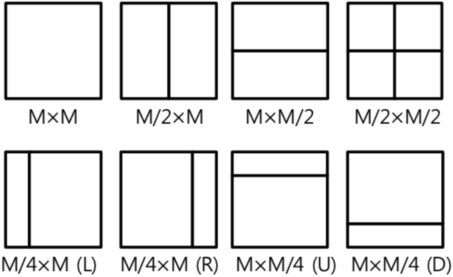
- H.264: Number of total modes depends on block size
  - 16 4x4 sub-blocks or 1 16x16 block
  - 4x4: DC or directional prediction (8 directions)
  - 16x16: DC, plane, horizontal or vertical prediction
  - 8x8 blocks in High profile use 16x16 modes
  - Different modes for chroma blocks (but not for 8x8 luma)
  - Explicit interpolation formulae (containing max. 3 samples)
  - Smoothing filter for reference samples for 8x8 prediction
  - Additional Hadamard transform for 16x16 blocks
- HEVC: 35 modes in total
  - 32x32 downto 4x4 sub-blocks
  - DC, planar or directional prediction (33 directions)
  - Same modes for chroma blocks (but no 2x2 blocks)
  - $\frac{1}{32}^{th}$ sample accuracy (bilinear)
  - Adaptive smoothing filter for most reference samples
  - Additional (permanent) boundary value smoothing



Source: Sullivan et al. (2012)

Inter prediction I

- H.264: Symmetric partitioning
  - 16x16, 16x8, 8x16 or 8x8 paritions
  - 8x4, 4x8 or 4x4 sub-partitions for 8x8 partitions
- HEVC: Adaptive (a)symmetric partitioning
  - Symmetric PB partitioning (like intra)
  - Asymmetric PB partitioning



Source: Sullivan et al. (2012)

Inter prediction II

- H.264: Limited reference granularity
  - One motion vector per (sub-)partition (2 for bi-pred.)
  - One reference picture index per partition
  - $\frac{1}{4}^{th}$ sample accuracy
    - 6-tap filter for half samples
    - Averaging for quarter samples
  - Direct mode uses MV prediction to save bits
- HEVC: Full reference granularity
  - One motion vector per PB (2 for bi-pred.)
  - One reference picture index per PB
  - $\frac{1}{4}^{th}$ sample accuracy
    - 8-tap filter for half samples
    - 7-tap filter for quarter samples
  - Merge mode: Choose one derived MV candidate based on temporal and spatial neighbours' MVs to save bits (allows direct-like modes)

Transform

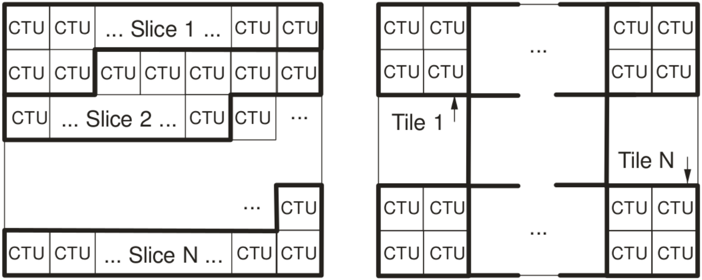
- H.264: 4x4 Integer transform (DCT approximation)
  - Used for all block partitionings and modes
  - Adaptive 8x8 integer transform in High Profile
- HEVC: H.264-like transform for each transform block (TB)
  - More transform sizes: 4x4, 8x8, 16x16, 32x32
  - TBs must be squared
  - Used for all modes except intra 4x4
    - 4x4 intra uses a DST approximation
    - Reason: Residuals tend to increase with distance from boundary
    - 1% intra-only bit rate decrease (hardly any for larger sizes)

Deblocking and additional filtering

- H.264: In-loop deblocking filter
  - Transform block boundary filtering
  - Static 4x4 grid
  - Adaptive strength from 0 to 5
- HEVC: Additional filtering
  - In-loop deblocking filter (similar to H.264)
    - 8x8 grid on PB and TB boundaries only
    - Adaptive strength from 0 to 2
  - Sample-adaptive offset (SAO)
    - Idea: Sharpen edges and remove banding
    - LUT-based addition of offset to each sample
    - Offset depends on local gradient
    - LUT depends on region (changed per CTB)

Parallelism I: Tiles

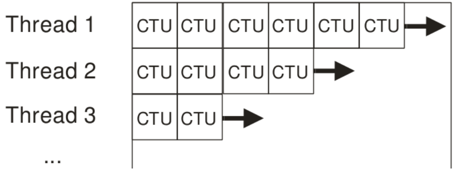
- H.264 slices: Independently decodable picture areas
- HEVC slices: Same, but can be split into multiple NALUs
- HEVC: Additional concept of tiles
  - Also independently decodable picture areas
  - About the same number of CTBs in each tile
  - A tile can span multiple slices



Source: Sullivan et al. (2012)

Parallelism II: WPP

- WPP = **W**avefront **P**arallel **P**rocessing
- Slices are divided into rows of CTBs
- Each thread processes one CTB row
- Thread n can start shortly after thread n-1
- Few dependencies (mainly entropy coder adaption)
- Not allowed in combination with tiles



Source: Sullivan et al. (2012)

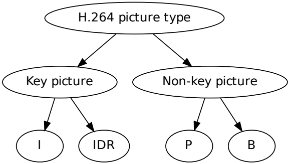
Parallelism III: CABAC

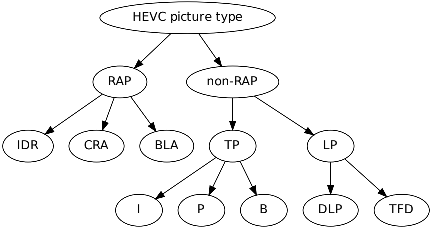
- H.264: CAVLC (faster, but less efficient) and CABAC
- HEVC: “New” CABAC
  - Same principle: multiple contexts (but fewer than in H.264)
  - Less dependencies, but still more efficient than CABAC (H.264)
  - Coefficients are always scanned on a 4x4 block basis
  - Coefficient scan pattern is mode dependent

Random access

- H.264: IDR, I, P and B pictures
- HEVC: Additionally:
  - Clean random access (CRA) pictures
    - Like IDRs, but without DPB flush
    - Some subsequent pictures may have to be discarded
    - Tagged for discard (TFD) pictures: smaller display order
    - Decodable leading pictures (DLPs) allowed
  - Broken link access (BLA) pictures
    - Originally a CRA picture with type changed to BLA
    - For splice points in concatenated bit streams
    - May also be followed by TFD pictures and DLPs
- Random access pictures (RAPs): IDRs, CRAs, BLAs

Picture type overview





IDR = Instantaneous Decoder Refresh, RAP = Random Access Picture,
CRA = Clean Random Access, BLA = Broken Link Access,
TP = Trailing Picture, LP = Leading Picture,
DLP = Decodable Leading Picture, TFD = Tagged For Discard

Miscellaneous

- Byte stream structure remains the same (NAL units)
  - NAL unit header is longer (not compatible to H.264)
  - New NAL unit types and type number changes
  - Modified NALU payload syntax which is not H.264-compliant
- New profiles, e.g., Main
  - 8 bit with 4:2:0 chroma subsampling
  - CTB sizes only from 16x16 to 64x64
- New levels: 1.0 (176x144@15) to 6.2 (8192x4320@120)
- Multiple tiers (currently Main and High) orthogonal to levels
- Temporal scalability built-in (without additional extensions)
- No more interlaced handling (optional SEI signalling)

Updates

- Update 1 (2014)
  - Range extensions (10, 12 and 16 bit color depth handling with 4:0:0, 4:2:2 and 4:4:4 chroma subsampling support)
  - Scalable extensions (multiple resolutions and quality levels – “layers”)
  - Multi-view extensions (multiple views of one video, e.g., 3-D with 2)
- Update 2 (2015)
  - Coding of depth maps (for 3-D applications)
  - 3D Main Profile for typical use case
- Update 3 (2016)
  - Range extensions for scalable video coding (new profiles)
  - Screen content coding extensions (special coding tools for screen captures)

What comes next?

- Call for evidence[4] (ongoing)
  - Is there sufficient potential for a new standard?
  - Evaluation in July 2017

→ H.266?
  - Goal: Again approx. 50% higher coding efficiency
  - Explicit support for coding of 360° videos

[4] http://www.itu.int/en/ITU-T/studygroups/2017-2020/16/Documents/201701/Video-CfE-SG16R1-AnnexH.pdf

Thank you for your attention!

Questions?