Image/Video Processing and Coding

Dr.-Ing. Henryk Richter
Institute of Communications Engineering
Phone: +49 381 498 7303, Room: W 8220
EMail: henryk.richter@uni-rostock.de
Content

- Introduction
- Colors, Image Matrix, Image Processing and Recognition structure
- Image Sampling and Quantization
- Discrete Image Characteristics
- Image Transformation
- Image Improvement
- Image Segmentation
- Features, Extraction, Descriptors
- Pattern Recognition (Basics, Systems for classification, Neural Networks)
- Data compression fundamentals
- Methods, techniques and algorithms for data compression
  - Data reduction, Coding, Decorrelation
- Image and Video coding standards and their specifics
  - JPEG, JPEG-2000
  - Video Coding: H.265
Introduction

- joint project JVT of the ITU-T VCEQ and MPEG
- nomenclature
  - ITU-T Rec. H.265
  - ISO/IEC 23008-2 HEVC
- development based on H.264/AVC
- specifically targeted at UHDTV video resolution (4k,8k)
  - >8 Bit per pixel included in initial specification (8 Bit or 10 Bit, RXext provides 14 Bit per component)
  - YCbCr subsampling 4:2:0 (in addition, 4:0:0, 4:2:2 and 4:4:4 in RXext)

Development goal: compression efficiency doubled (again) in comparison with previous standard
Key Changes in comparison to H.264

- Coding Tree Units (CTU) as replacement for the traditional 16x16 Macroblock
  - 16x16, 32x32 or 64x64 luma samples per CTU

- Prediction Units (PU)
  - adaptive subdivision of CTU sized blocks as needed

- Transform Units (TU)
  - adaptive subdivision of CTU sized blocks as needed
  - 4x4, 8x8, 16x16 and 32x32 transform sizes (quantized DCT), 4x4 quantized DST
  - transform block sizes decoupled from PU sizes

- Advanced Motion Vector prediction
  - more MV prediction candidates compared to earlier standards
  - merge mode for MV coding
  - improved „skipped“ and „direct“ motion inference

- single-stage Motion Compensation
  - 7-tap or 8-tap filters for interpolation (instead of 6-tap plus bilinear interpolation)

- Intra Prediction with 35 different modes (33 directional, DC, Plane)

- Sample-Adaptive Offset (SAO)
  - non-linear amplitude mapping by lookup-table after the deblocking filter, parameters from bitstream

- Parallel Encoding Decoding support
  - Tiles, Wavefront Parallel Processing (including Entropy decoding), Dependent Slices

- Clean Random Access (CRA)
  - Open GOP principle, start with a temporal independent picture (RAP) and discard non-decodable pictures
Profiles / Levels

- Initially only the Main Profile was specified for the first version of HEVC
  - Acknowledgement that traditionally separate services (broadcast, mobile, streaming) converge toward multipurpose receiver devices
  - Restrictions
    - Only 8 Bit video with 4:2:0 chroma sampling
    - Usage of tiles excludes wavefront parallel processing
    - Tiles must be at least 256 x 64 luma samples large

- 13 Levels for initial specification
  - Level 1-3: SDTV resolution and below
  - Level 3.1: up to 720p @ 33 FPS
  - Level 4,4.1: HDTV 1080i/p @ 30,60 FPS
  - Level 5,5.1,5.2: 4k @ 30,60,120 FPS
  - Level 6,6.1,6.2: 8k @ 32,64,128 FPS
  - Max. 6 pictures in decoded picture buffer (DPB) at maximum pixel count allowed in level, total limit of 16 pictures in DPB
High Level Syntax

- Concepts from H.264 retained

- Network Adaptation Layer (NAL)
  - encapsulation of VCL (video coding layer) units into various transport layers (RTP, ISO MP4, MPEG-2 Systems)

- NAL units classified into VCL and non-VCL
  - VCL NAL types
    - used for different picture categories (especially random access markers)
    - temporal scalability (temporal sub-layers)
  - non-VCL NAL types
    - parameter sets
    - sequence/bitstream delimiting
    - SEI messages
      - supplemental enhancement information for parameters not directly associated with bitstream decoding
      - aspect ratio, cropping, interlace support
Picture Random Access

- Traditional video coders required to start decoding with I-frames (or IDR in H.264)
  - Closed GOP concept, where each GOP starts with I/IDR pictures and implicitly invalidates the reference picture buffer immediately
- HEVC Decoders can start at different random access point (RAP) pictures
  - IDR = Instantaneous Decoder Refresh
  - CRA = Clean Random Access
  - VLA = Broken Link Access
- Clean Random Access (CRA) syntax
  - Intra pictures (CRA) at the location of a random access point (RAP) to start successful decoding without the knowledge of prior pictures in the bitstream
  - Pictures following CRAs in decoding order and precede them in display order are discarded by decoders starting with CRA pictures (TFD=tagged for discard NAL unit types)
- Broken Link Access (BLA)
  - bitstream splicing, i.e. switch from one bitstream to another
  - BLA NAL units signal a disrupted picture numbering to the decoder
Frame subdivision concepts

- **Slices**
  - Slices consist of a sequence of CTUs
  - Arbitrary number of CTUs per slice
  - Slices can be decoded independently*

- **Tiles**
  - Wimplied version of the H.264 FMO framework
  - Rectangular groups of CTUs (typically same size)
  - Independent decoding
  - Each tile may contain a variable number of slices (also possible are multiple tiles within a single slice)

- **Dependent Slices**
  - Slices that depend on a particular decoding order of other slices (e.g. Wavefront entry points for parallel processing)

* except deblocking filter
Coding Blocks and Prediction Blocks from Coding Tree Units

- CTUs >16x16 are the key feature of H.265 towards higher coding efficiency
- Quadtree decomposition of CTUs into smaller Coding Blocks (CB)

- Recursive quadtree partitioning of CBs into Transform Blocks (TB) (min. size 4x4)
- In Intra CUs, each CB may be (optionally) further subdivided into 4 quadrants, where each quadrant is assigned a distinct intra prediction mode (min. size 4x4)
- In Inter CUs, the CB may be (optionally) subdivided into two prediction blocks (PB), or into four PBs when the CB size is at the minimum allowed CB size
H.265 / HEVC system architecture

- **Input frame**
- **Subdivision into CTUs**
- **Coder control**
- **Transform/Quantizer**
  - **Decoder**
  - **Intra-prediction**
  - **Motion compensated predictor**
- **Motion estimation**
- **Deq./Inv. Transform**
- **Deblocking & SAO**
- **Entropy Coding**
  - **Prediction modes**
  - **Motion vector data**

**Control flow**
- **Quant. Transf. coeffs**

Quelle: HHI Berlin
Intra Prediction

- Smoothed predictors from directly adjacent neighbors (left, above the current block)
- Square sizes from 4x4 … 32x32 luma pixels
- Intra_Angular Prediction with 33 different directions of prediction
  - non-uniform angles in directional modes: denser mode coverage for near vertical and near horizontal prediction angles
  - for a block size of NxN, 4N+1 spatially neighboring samples are used
  - bi-linear interpolation with 1/32 sample accuracy
- Intra_Planar
  - four corner reference samples, average of two linear projections
- Intra_DC Prediction
  - average of predictor samples copied to the whole block
- additional boundary smoothing for DC, horizontal, vertical
- by selecting an index from the 3 most probable modes or 5 bit FLC
Motion Compensation - Luma

- Single-stage interpolation process
  - MPEG-4 and H.264 required
  - HEVC result is available after just two interpolation filter steps (horizontal+vertical)

- Separable horizontal and vertical filters

- 7 filter taps (half-pel positions) or 8 filter taps (quarter pel positions)

- 16 Bit word accuracy sufficient for computation
Motion Compensation - Luma (2)

- **Filter Coefficients**
  
<table>
<thead>
<tr>
<th>index</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>hfilter[i]</td>
<td>-1</td>
<td>4</td>
<td>-11</td>
<td>40</td>
<td>40</td>
<td>-11</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>qfilter[i]</td>
<td>-1</td>
<td>4</td>
<td>-10</td>
<td>58</td>
<td>17</td>
<td>-5</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

- **First stage** ($B$ is bits per pixel)

  \[
  a_{0,j} = \left( \sum_{i=-3}^{3} A_{i,j} \right) qfilter[i] \quad >> (B - 8) \\
  b_{0,j} = \left( \sum_{i=-3}^{4} A_{i,j} \right) hfilter[i] \quad >> (B - 8) \\
  c_{0,j} = \left( \sum_{i=-2}^{4} A_{i,j} \right) qfilter[1 - i] \quad >> (B - 8) \\
  d_{0,0} = \left( \sum_{j=-3}^{3} A_{0,j} \right) qfilter[j] \quad >> (B - 8) \\
  h_{0,0} = \left( \sum_{j=-3}^{4} A_{0,j} \right) hfilter[j] \quad >> (B - 8) \\
  n_{0,0} = \left( \sum_{j=-2}^{4} A_{0,j} \right) qfilter[1 - j] \quad >> (B - 8)
  \]

- **Second Stage**

  \[
  e_{0,0} = \left( \sum_{j=-3}^{3} a_{0,j} \right) qfilter[j] \quad >> 6 \\
  f_{0,0} = \left( \sum_{j=-3}^{3} b_{0,j} \right) qfilter[j] \quad >> 6 \\
  g_{0,0} = \left( \sum_{j=-3}^{3} c_{0,j} \right) qfilter[j] \quad >> 6 \\
  i_{0,0} = \left( \sum_{j=-3}^{4} a_{0,j} \right) hfilter[j] \quad >> 6 \\
  j_{0,0} = \left( \sum_{j=-3}^{4} b_{0,j} \right) hfilter[j] \quad >> 6 \\
  k_{0,0} = \left( \sum_{j=-3}^{4} c_{0,j} \right) hfilter[j] \quad >> 6 \\
  p_{0,0} = \left( \sum_{j=-2}^{4} a_{0,j} \right) qfilter[1 - j] \quad >> 6 \\
  q_{0,0} = \left( \sum_{j=-2}^{4} b_{0,j} \right) qfilter[1 - j] \quad >> 6 \\
  r_{0,0} = \left( \sum_{j=-2}^{4} c_{0,j} \right) qfilter[1 - j] \quad >> 6
  \]

- **Third stage**: optionally, the interpolation process is followed by weighted prediction (like H.264, only explicit mode supported)
- for bi-directional prediction, interpolated results are added together
- last: rounding/clipping to $[0, 2^B - 1]$ is performed

$x >> n$ operator: arithmetic right shift ($x/2^n$)
Motion Compensation - Chroma

- Chroma MC is 1/8 pel resolution in YCbCr 4:2:0
- 4 tap FIR filter, indices 5-7 are the mirrored indices 3-1
- horizontal / vertical filtering separately, depending on fractional position
- workflow in analogy to Luma MC

<table>
<thead>
<tr>
<th>index</th>
<th>-1</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter1[i]</td>
<td>-2</td>
<td>58</td>
<td>10</td>
<td>-2</td>
</tr>
<tr>
<td>filter2[i]</td>
<td>-4</td>
<td>54</td>
<td>16</td>
<td>-2</td>
</tr>
<tr>
<td>filter3[i]</td>
<td>-6</td>
<td>46</td>
<td>28</td>
<td>-4</td>
</tr>
<tr>
<td>filter4[i]</td>
<td>-4</td>
<td>36</td>
<td>36</td>
<td>-4</td>
</tr>
<tr>
<td>filter5[i]</td>
<td>-4</td>
<td>28</td>
<td>46</td>
<td>-6</td>
</tr>
<tr>
<td>filter6[i]</td>
<td>-2</td>
<td>16</td>
<td>54</td>
<td>-4</td>
</tr>
<tr>
<td>filter7[i]</td>
<td>-2</td>
<td>10</td>
<td>58</td>
<td>-2</td>
</tr>
</tbody>
</table>
Motion Compensation - reference frames

- each motion vector is assigned a reference frame index $\Delta$
- reference frames kept on encoder and decoder side
- in bi-directional prediction: two sets of motion vector parameters (list0, list1)
- sorting of reference frames by Picture Order Count (POC)
- ordering transmitted in the slice header (RPS=reference picture set)
- simplified and more robust syntax compared to H.264

$\Delta=3$, $\Delta=2$, $\Delta=1$, $\Delta=0$

4 previously decoded frames

current frame
Motion Compensation - Merge Mode Motion Inference

- Multiple motion vector prediction candidates in spatial and temporal directions
  - Generalization and extension of the concepts toward DIRECT and SKIP modes in earlier standards
  - Multi-hypothesis motion information
- Temporal motion candidates are stored with a granularity of 16x16 for memory efficiency
- Set of motion vector prediction candidates
  - spatial neighbor candidates
    - based on availability and PU location
    - limit towards the number can be signaled so that only the first candidates are retained
  - temporal candidate from collocated reference picture
    - index explicitly transmitted to decide which reference frame list
  - generated candidates
    - pre-defined list of usual motion vectors, included into merge candidate list for B-Slices
    - zero (0,0) vectors appended to the list for P-Slices
- Candidate list is pruned to remove redundant entries
- For each PU, the index into the candidate list is transmitted
- Similar multi-hypothesis approach to non-merge motion prediction (limited set)
Residual Transform

- Integer DCT approximation for 4x4, 8x8, 16x16, 32x32 transform sizes
  - One-dimensional transform in horizontal and vertical directions, followed by 7 Bit shift, along with 16 Bit clipping to fit all immediately stored data into 16 Bit
  - Each PU can be either transformed directly as 1 TB or subdivided into 4x4 TBs

Core transform $H$, as given in the standard:

<table>
<thead>
<tr>
<th>$H_{17}</th>
<th>H_{16}</th>
<th>H_{15}</th>
<th>H_{14}</th>
<th>H_{13}</th>
<th>H_{12}</th>
<th>H_{11}</th>
<th>H_{10}</th>
<th>H_{9}</th>
<th>H_{8}</th>
<th>H_{7}</th>
<th>H_{6}</th>
<th>H_{5}</th>
<th>H_{4}</th>
<th>H_{3}</th>
<th>H_{2}</th>
<th>H_{1}</th>
<th>H_{0}</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
<td>60</td>
</tr>
<tr>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
</tr>
<tr>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
<td>52</td>
</tr>
<tr>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
<td>48</td>
</tr>
<tr>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
<td>44</td>
</tr>
<tr>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
<td>36</td>
</tr>
<tr>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Residual Transform (2)

- Matrices $H^{(16)}$, $H^{(8)}$, $H^{(4)}$ can be obtained from $H$ by extracting each 2nd, 4th, 8th row/column, respectively.
- Recursive and partially factored implementation possible instead of straight-forward (and computationally expensive) matrix multiplication.

- Mode-dependent alternative transform
  - Only Intra 4x4 transform can be selectively performed by an alternative algorithm.
  - Integer DST (discrete sine transform).
  - Observation that the prediction error increases with the distance from the border samples used as predictors.
  - Approx. 1% bit rate reduction in intra-predictive coding.

\[ H = \begin{bmatrix} 29 & 55 & 74 & 84 \\ 74 & 74 & 0 & -74 \\ 84 & -29 & -74 & 55 \\ 55 & -84 & 74 & -29 \end{bmatrix} \]
Coefficient Scanning, Quantization, Entropy Coding

- Coefficient Scanning, Block coding
  - three different coefficient scanning methods: diagonal, horizontal, vertical

- Quantization
  - Applied transforms are scaled orthonormal DCT/DST
  - Frequency uniform Quantization/Reconstruction
  - QP 0...51 (like H.264), logarithmic step size

- Entropy Coding
  - CABAC (from H.264) as single supported coding method
  - Modified context modeling, significant reduction in context models compared to H.264
  - Higher use of the bypass-mode in CABAC in order to reduce the computational demands
  - Last nonzero coefficient signaling, significance map, sign bit and level encoding concepts similar to H.264
  - Significance map grouped for multiple 4x4 blocks
In-loop Filters

- Two post-processing filter steps in HEVC

- Deblocking Filter (DBF)
  - similar in concept to H.264 deblocking filter
  - low pass filtering of decoded frames for improved visual appearance and improved coding efficiency, reduction of block artifacts
  - significant reduction in complexity compared to H.264 approach
  - multi-processing friendly

- Sample adaptive offset (SAO)
  - applied after the Deblocking Filter
  - Suppression of „banding artifacts“ from strong quantization as well as „ringing artifacts“ from quantization errors of high frequency components
  - Signaling of SAO parameters flexible, ranging from one parameter specification (over the whole picture) down to a fine granularity on CU level
Deblocking Filter

- Applied on TU and PU boundaries (not necessarily the same)
- 8x8 sample grid for luma and chroma (H.264: 4x4/2x2)
- 3 different strengths for luma
  - 0  no filtering
  - 1  regular filtering for motion compensated blocks (practically the same rules as in H.264 apply)
  - 2  strong Intra filtering if any neighboring block is Intra
- Chroma shares the decisions, yet only two modes (on,off) are defined

- HEVC processing order of DBF is horizontal filtering (vertical edges) first over the whole picture, followed by vertical filtering (horizontal edges)
  - inherently multi-processing friendly
DBF Impact: 176x144, QP36, GOP-Length 32, 30 kBit/s

- DBF off
- DBF on
Sample adaptive offset

- Four gradient patterns in SAO: current pixel \( p \), neighboring pixels \( n_0, n_1 \)

- Syntax element sao_type_idx controls the SAO operation for the current coding tree blocks (CTB, one luma or chroma component of a CTU), either
  - 0 = off
  - 1 = band offset type
    - central band, side band
    - offset depends on sample amplitude (amplitude range divided into 32 bands, sample values of four consecutive bands are modified as band offsets)
  - 2 = edge offset type
    - additional parameter: direction in which the gradient strengths are to be evaluated
    - gradient evaluation and classification done in encoder and decoder
    - encoder sends the appropriate offsets to the decoder for each CTB (positive values only for 1,2, negative values only for 3,4)

- Offsets are added to considered pixels within a coding tree block (luma/chroma) via look-up tables
SAO Impact: 448x256, QP40, GOP-Length 32, 80 kBit/s

- SAO off

- SAO on

- Contrast enhanced difference image
Further Development

- RXext - Range Extensions
  - Higher pixel depth >10 Bits / sample
  - Additional chroma formats (4:2:2, 4:4:4, 4:0:0)
  - Additional Channels (e.g. Alpha)

- Scalability Extensions (SHVC)
  - temporal scalability is already part of HEVC v1
  - spatial and SNR scalability
  - multi-loop coding framework (requires decoding of current and all lower layers)

- 3D Video Extensions
  - MV-HEVC (direct sibling to H.264 MVC)
  - Multiview HEVC with modified Block-Level Tools
    - higher compression efficiency than MV-HEVC, backwards compatible to HEVC v1, yet new tools for additional views

- Hybrid Architectures
  - Legacy Base Layer (H.264, H.262) with HEVC enhancement layer
Image Quality H.264 vs. H.265

- Foreman QCIF, first picture (Intra)
  - H.264, 9768 Bits, Y-PSNR 30.8 dB (JM12.4)
  - H.265, 9208 Bits, Y-PSNR 32.4 dB (HM16.6)
Comparison of PSNR H.265 ↔ MPEG-2/4,H.264 1080p

Park Scene, 1920x1080, 24Hz
Literature


- Reference software: https://hevc.hhi.fraunhofer.de/

- Free implementation: http://x265.org