Information and Coding

Armando J. Pinho

Departamento de Electrónica, Telecomunicações e Informática

Universidade de Aveiro

ap@ua.pt

Contents

Motivation

- The main objective of using transforms in the context of data compression is to convert the original data into a new data set more simple to quantize and encode.
- Transforms are used to reduce the statistical dependencies among the original data (Ideally, the resulting coefficients should be statistically independent).
- Transforms are also used to separate the relevant information from the irrelevant, in order to permit course quantization or even removal of the irrelevant information.

Principles

- Let us consider linear transforms of finite dimension, that transform a vector $\mathbf{x}$ of dimension m into a vector $\mathbf{y}$ also of dimension m:

$$\mathbf{y} = A^*\mathbf{x}, \quad \mathbf{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{m-1} \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{m-1} \end{pmatrix}$$

where A is a $m \times m$ matrix, real or complex, and A^* is the corresponding conjugated transposed.

- We consider only invertible transforms, i.e., those for which $\mathbf{x} = S\mathbf{y}$, with $S = (A^*)^{-1}$.

Principles

- The transform coefficients can be written as

$$y_q = \mathbf{a}_q^* \mathbf{x}, \quad q = 0, 1, \ldots, m-1$$

where $\mathbf{a}_q$ is the q^{th} column of A. Usually, we call $\mathbf{a}_q$ the q^{th} analysis vector of the transform.

- Similarly, we can write

$$\mathbf{x} = \sum_{q=0}^{m-1} y_q \mathbf{s}_q$$

where $\mathbf{s}_q$ is the q^{th} column of S, the q^{th} synthesis vector.

- Therefore, a transform can be seen as a decomposition of vector $\mathbf{x}$ into a linear combination of the synthesis (or prototype) vectors.

Principles

- A transform is orthonormal if and only if

$$\begin{aligned}\langle \mathbf{a}_i, \mathbf{a}_j \rangle &= 0, \quad \forall i \neq j \\ \langle \mathbf{a}_i, \mathbf{a}_i \rangle &= \|\mathbf{a}_i\|^2 = 1, \quad \forall i\end{aligned}$$

- In this case, $AA^* = A^*A = I$, and, therefore, $S = A$ (i.e., A is a unitary matrix): the analysis and synthesis vectors are the same.

- The Parseval relation is an important property that holds for the orthonormal transforms, i.e., $\|\mathbf{x}\|^2 = \|\mathbf{y}\|^2$ (the energy is preserved among domains).

The DCT

- The DCT (Discrete Cosine Transform) is a real and orthonormal transform.

- The analysis/synthesis vectors, $\mathbf{s}_q$, are formed by equally spaced samples of a cosine function with frequencies $f_q = q/(2m)$:

$$s_{q,p} = c_q \cos\left(2\pi f_q(p + 0.5)\right)$$

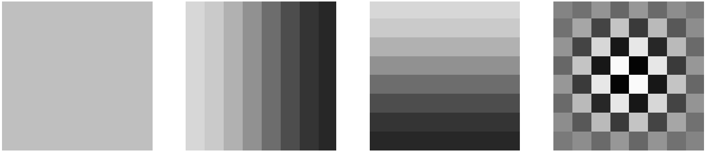
- The normalization factor, c_q, is such that $\|\mathbf{s}_q\| = 1$:

$$c_q = \begin{cases} \sqrt{\frac{1}{m}} & \text{if} \quad q = 0 \\ \sqrt{\frac{2}{m}} & \text{if} \quad q \neq 0 \end{cases}$$

The DCT

- The 2D DCT is obtained through a separable extension of the 1D version:

$$S_{q_1,q_2} = (\mathbf{s}_{q_1,q_2})_{p_1,p_2} = c_{q_1} c_{q_2} \cos\left(2\pi f_{q_1}(p_1 + 0.5)\right) \cos\left(2\pi f_{q_2}(p_2 + 0.5)\right)$$



$S_{0,0}$ $S_{0,1}$ $S_{1,0}$ $S_{7,7}$

- For calculating a single coefficient of a non-separable transform we need m^2 operations.
- In the separable case, only $2m$ operations are required.

The DCT

- The DCT has been one of the most used transforms in the context of image and video coding.
- There are several reasons for this choice:
  - It provides a good energy compaction and reduction of the correlation among the coefficients.
  - It uses only real numbers.
  - There are fast algorithms, based on the FFT (Fast Fourier Transform), that can be used for its calculation.

The DCT

- Let us see how the DCT attains energy compaction, considering, for example, the following 8×8 block of pixels:

$$\mathbf{X} = \begin{bmatrix} 183 & 160 & 94 & 153 & 194 & 163 & 132 & 165 \\ 183 & 153 & 116 & 176 & 187 & 166 & 130 & 169 \\ 179 & 168 & 171 & 182 & 179 & 170 & 131 & 167 \\ 177 & 177 & 179 & 177 & 179 & 165 & 131 & 167 \\ 178 & 178 & 179 & 176 & 182 & 164 & 130 & 171 \\ 179 & 180 & 180 & 179 & 183 & 169 & 132 & 169 \\ 179 & 179 & 180 & 182 & 183 & 170 & 129 & 173 \\ 180 & 179 & 181 & 179 & 181 & 170 & 130 & 169 \end{bmatrix}$$

The DCT

- The coefficients (rounded to the integers), resulting from applying the DCT to the block (after subtracting $2^7 = 128$ to each pixel), are:

$$\mathbf{Y} = \begin{bmatrix} 313 & 56 & -27 & 18 & 78 & -60 & 27 & -27 \\ -38 & -27 & 13 & 44 & 32 & -1 & -24 & -10 \\ -20 & -17 & 10 & 33 & 21 & -6 & -16 & -9 \\ -10 & -8 & 9 & 17 & 9 & -10 & -13 & 1 \\ -6 & 1 & 6 & 4 & -3 & -7 & -5 & 5 \\ 2 & 3 & 0 & -3 & -7 & -4 & 0 & 3 \\ 4 & 4 & -1 & -2 & -9 & 0 & 2 & 4 \\ 3 & 1 & 0 & -4 & -2 & -1 & 3 & 1 \end{bmatrix}$$

The JPEG standard

- The JPEG (Joint Photographic Experts Group) standard is a family of coding methods for images of continuous tones of grays or colors.
- The group was established in 1986, the standard was proposed in 1992 and approved in 1994 (ISO 10918-1).
- The JPEG standard comprises four coding methods: sequential, progressive, hierarchical and lossless.
- The JPEG standard is based on a number of compression techniques, such as the DCT, statistical coding and predictive coding.

The sequential mode of JPEG

- Every codec should include this mode in order to be considered JPEG-compatible (it is also known as the “baseline” mode).
- The sequential mode of JPEG comprises the following steps:
  - Calculation of the DCT.
  - Quantization of the DCT coefficients, in order to eliminate less relevant information, according to the characteristics of the human visual system.
  - Statistical coding (Huffman or arithmetic) of the quantized DCT coefficients.

The sequential mode of JPEG

- Calculation of the DCT:
  - The image is partitioned into 8×8 blocks of pixels. If the number of rows or columns is not multiple of 8, then they are internally adjusted (using padding).
  - Subtract 2^{b-1} to each pixel value, where b is the number of bits used to represent the pixels.
  - Calculate the DCT 2D of each block.

The sequential mode of JPEG

- Quantization of the DCT coefficients:
  - The DCT coefficients are quantized using a quantization matrix, previously scaled by a compression quality factor.
  - Next, the coefficients are organized in a one-dimensional vector according to a zig-zag scan.
- Statistical coding:
  - The non-zero AC coefficients are encoded using Huffman or arithmetic coding, representing the value of the coefficient, as well as the number of zeros preceding it.
  - The DC coefficient of each block is predictively encoded in relation to the DC coefficient of the previous block.

Quantization of the coefficients

- The DCT, alone, does not provide data compression.
- In fact, each $m \times m$ block of pixels is transformed into another $m \times m$ block, usually requiring higher precision for representing its elements.
- Typically, compression is obtained through the concatenation of two distinct processes:
  - Quantization of the coefficients resulting from the transformation.
  - Use of statistical coding.

Quantization of the coefficients

- Compression is obtained due to the low-pass characteristic of the human visual system.
- Because of this characteristic, generally more bits are assigned to the low frequencies (those appearing in the upper left corner of the transformed block).
- This is done using threshold coding (non-linear approximation).

Quantization of the coefficients

- Threshold coding is based on the use of one or more decision levels, such that coefficients below the thresholds are eliminated.
- This way, block to block variations can be accommodated.
- Generally, the thresholding and quantization operations are done together, through a quantization matrix

$$\tilde{y}(r,c) = \mathrm{ROUND}\left(\frac{y(r,c)}{q(r,c)}\right),$$

where $\tilde{y}(r,c)$ is the quantized version of $y(r,c)$, and $q(r,c)$ is the corresponding element of the quantization matrix, Q.

Quantization of the coefficients

- Generally, the elements of Q are 8 bit integers that determine the quantization step according to the position of each coefficient.
- Example: quantization matrix of JPEG (luminance):

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 129 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Quantization of the coefficients

- By applying this quantization matrix to the block that we have previously used as example, we obtain the matrix $\widetilde{Y}$:

$$\widetilde{Y} = \begin{bmatrix} 20 & 5 & -3 & 1 & 3 & -2 & 1 & 0 \\ -3 & -2 & 1 & 2 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

- In this example, 45 of the 64 coefficients are eliminated.

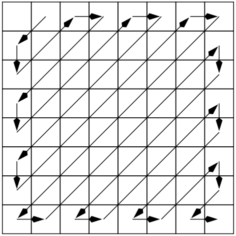
Quantization of the coefficients

- Because the sensitivity of the human eye to the colors is different from that of the luminance, JPEG provides a different quantization matrix for the chrominance components:

$$Q = \begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix}$$

Coefficient coding

- JPEG uses a zig-zag scanning of $\widetilde{Y}$ in order to encode the quantized coefficients, except for the $(0, 0)$ position, i.e., the DC coefficient.



- The objective of this scanning is to group together the zero coefficients, allowing a more efficient representation.
- This efficiency is obtained using a variant of run-length coding.

Coefficient coding

- Using again the same example, a JPEG encoder would generate the following codewords:

$$(0,5),(0,-3),(0,-1),(0,-2),(0,-3),(0,1),(0,1),(0,-1),(0,-1),$$
$$(2,1),(0,2),(0,3),(0,-2),(0,1),(0,1),(6,1),(0,1),(1,1),\mathrm{EOB}$$

$$\begin{bmatrix} 20 & 5 & -3 & 1 & 3 & -2 & 1 & 0 \\ -3 & -2 & 1 & 2 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The visual effect of the coding blocks

- The coding techniques that are based on a partition of the image into blocks are generally affected by a visual phenomenon known as the blocking artifact.

This artifact is more visible when the compression ratio is high and happens because the blocks are encoded independently (except for the DC coefficients).

Example: 8×8 DCT, 0.31 bpp.

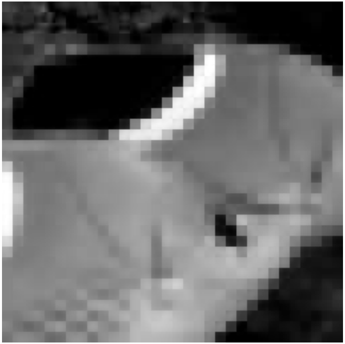
The progressive mode of JPEG

- This mode relies on encoding the DCT coefficients using several passes, such that in each pass only part of the information associated to those coefficients is transmitted.
- JPEG provides two methods for doing this:
  - Spectral selection: the coefficients are organized in spectral bands, and those corresponding to the lower frequencies are transmitted first.
  - Successive approximation: all coefficients are first transmitted using a limited precision. Afterward, additional detail is sent using more passes through the coefficients.

The progressive mode of JPEG

- Sequential *vs.* progressive:

Sequential, 1000 bytes



Progressive, 1000 bytes

The progressive mode of JPEG

- Sequential *vs.* progressive:

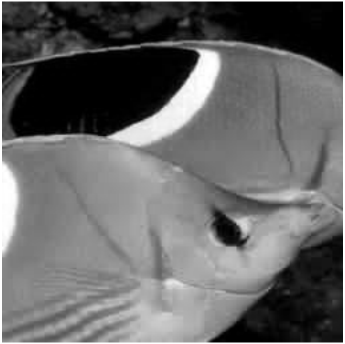
Sequential, 2000 bytes

Progressive, 2000 bytes

The progressive mode of JPEG

- Sequential *vs.* progressive:

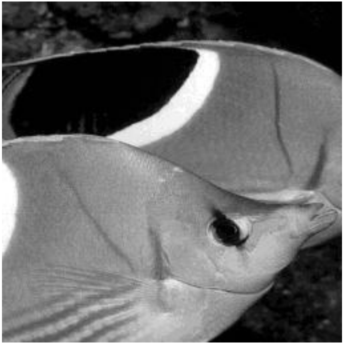
Sequential, 4000 bytes



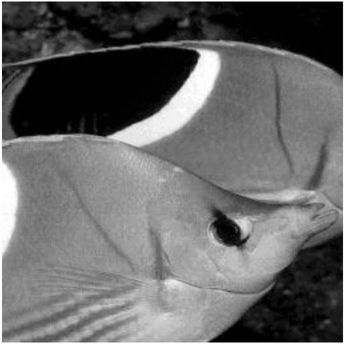
Progressive, 4000 bytes

The progressive mode of JPEG

- Sequential *vs.* progressive:



Sequential, 10023 bytes



Progressive, 10198 bytes

Contents

H.261

- H.261 (1990) is a ITU-T video coding standard (Video Codec for Audiovisual Services at p$\times$64 kbit/s) that was developed with the aim of being used
  - In video-phone applications.
  - In video-conference applications.
  - Over ISDN links at $p \times 64$ kbps, $p = 1, \ldots, 30$.
- For example, $p = 1$ (64 kbps) would be appropriated for video-phone, where the video signal was transmitted at 48 kbps and the audio signal at 16 kbps.
- Generally, video-conference required better image quality, implying typically $p \geq 6$ (384 kbps).
- For $p = 30$ we have 1.92 Mbps, which was sufficient for a video quality similar to the old VHS tapes.

H.261

- Because this standard was intended for bi-directional real-time communication, the maximum delay allowed in the coding process is 150 milliseconds.
- It allows only two frame formats: CIF (Common Intermediate Format) 352×288, and QCIF (Quarter Common Intermediate Format) 176×144...
- ...and frame-rates of 30 Hz, 15 Hz, 10 Hz and 7.5 Hz.
- Notice that, even for QCIF at 10 Hz, it is required a compression of at least 1:48 for transmission in a 64 kbps channel.

H.261

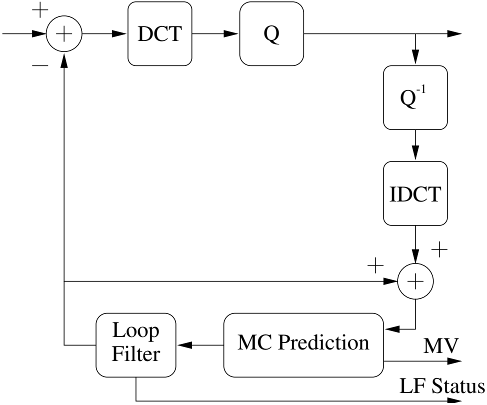
- The encoded stream has the following structure:
  - At the top, the frame.
  - Each frame is partitioned into several groups of blocks.
  - Each group of blocks is formed of several macroblocks.
  - The macroblock is the smallest region that can have a particular coding mode assigned to.
  - The macroblock is composed of four basic blocks (a basic block is 8×8) of luminance (Y) and by the corresponding 8×8 chrominance blocks (C_r and C_b).
  - A group of blocks is always composed of 33 macroblocks, organized in a 3×11 matrix.
  - Each frame is formed of 12 groups of blocks (CIF) or 3 groups of blocks (QCIF).

H.261

- The H.261 uses two compression modes:
  - Intraframe: similar to the JPEG compression, i.e., relies on DCT applied to 8×8 blocks of pixels.
  - Interframe: temporal prediction (motion compensation), followed by DCT of the prediction residuals.
- Motion compensation (MC) is performed in macroblocks, within a search area of 15 pixels around the macroblock.
- For smoothing the prediction values, it is possible to turn on/off a low-pass loop filter.
- It has 32 quantizers, one of them dedicated to the DC coefficient in intraframe mode (quantization step of 8). The others have quantization steps from 2 to 62.
- Statistical coding is performed with Huffman codes.

H.261

- Encoder:



MPEG–1

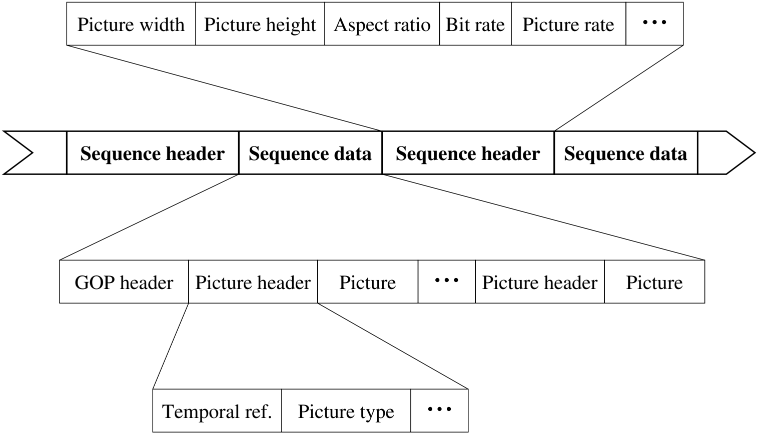
- MPEG–1 (1992) is a ISO/IEC (11172) coding standard that has been developed with the aim of storing video and audio in CD-ROMs.
- Target bitrates were around 1.5 Mbps, which was the bitrate associated to the early CD-ROM readers.
- The main objective of MPEG–1 was to provide means for encoding audio and video for interactive multimedia applications.
- For video segments having a moderate motion content, quality similar to VHS could be attained for MPEG–1 video at 1.2 Mbps.

MPEG–1

- The algorithms used in MPEG–1 are similar to those of H.261, although having some additional characteristics, such as
  - Random access (using type I frames)
  - Fast forward and reverse.
  - Backwards playing.
- Generally, the input signal is in the CCIR 601 format (576×720, for a 50 Hz frame-rate or 480×720 for 60 Hz), and is converted to SIF (Source Input Format) before encoding (luminance with $288(240) \times 352$ pixels and chrominance with $144(120) \times 176$ pixels).

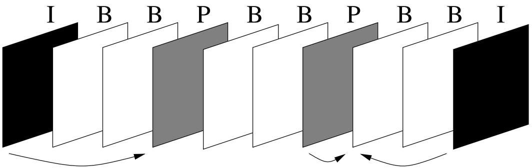
MPEG–1

Organization of the bitstream



MPEG–1

- MPEG–1 allows three types of frames:
  - Type I: encoding is similar to that of JPEG. These frames serve as entry points for random access.
  - Type P: frames encoded in predictive mode, using as reference previous frames of type I or P.
  - Type B: frames encoded in predictive mode, using both reference frames from the past and from the future (of type I or P).
- The number of I, P and B frames composing a group of frames depends on the application.



MPEG–1

- The slices provide resynchronization capabilities, in case of errors.

Picture header	Slice header	Macroblock	···	Macroblock	Slice header	Macroblock	···

Address	Type	Quantizer scale	Motion vectors	Coded block pattern	Block	···

- In summary, the main operations performed by a MPEG–1 video encoder are
  - Choose the type of frame (I, P or B).
  - Estimate the motion vector for each macroblock (only for type P and B frames).
  - Find the coding mode for each macroblock.
  - Find the appropriate quantization step for rate control.

MPEG–2

- MPEG–2 (1994) has been developed aiming applications such as
  - Transmission of television signals in standard definition formats (PAL, SECAM, NTSC).
  - High definition television (HDTV).
  - Electronic cinema.
  - Games and high quality multimedia applications.
  - …
- Some characteristics of MPEG–2 video:
  - Bitrates up to 100 Mbps.
  - More choices in terms of spatial and temporal resolution.
  - Support for interlaced video (notion of even and odd field).
  - More possibilities for the chrominance sub-sampling.
  - More coding and quantization options.
  - Support for bitstream scalability.

H.263

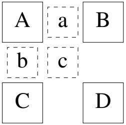
- Initially (1993), the MPEG–4 group started developing a video coding standard for bitrates $<$ 64 kbps, i.e., for very low bitrates.
- However, some time after, this line was reformulated into a much more ambitious objective: that of creating a standard for coding audiovisual objects.
- Due to the urgent need for a low bitrate standard (for example, for enabling video over the analog public telephone network or over wireless channels), the work was divided in two phases:
  - One, for the immediate development of a video coding standard for very low bitrates: recommendation H.263 (1995).
  - The other, directed to a more vast set of tools, originated the MPEG–4 standard.

H.263

- Recommendation H.263 specifies an algorithm for video coding, similar to that of H.261, for bitrates of about 22 kbps of a total of 28.8 kbps.
- The main differences between H.261 and H.263 are:
  - New formats available: sub-QCIF, 4CIF and 16CIF, in addition to those already supported by H.261, CIF and QCIF.
  - Possibility of using a motion vector per block as well as one motion vector per macroblock.
  - Half-pixel precision motion estimation and prediction of motion vectors.
  - Arithmetic coding.
  - PB-frames (bi-directional prediction, similar to that used in MPEG).

H.263

- Motion estimation with half-pixel precision, using bi-linear interpolation:



$$a = (A + B + 1) / 2$$

$$b = (A + C + 1) / 2$$

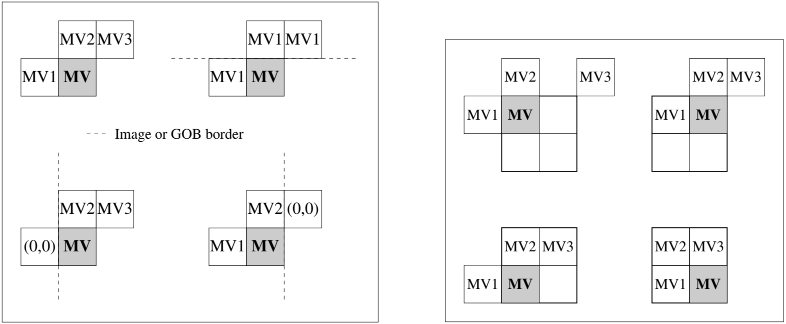
$$c = (A + B + C + D + 2) / 4$$

- In the normal mode, the motion vectors are restricted to value in $\{-16, \ldots, 15.5\}$.

- In the unrestricted mode, they may have values in $\{-31.5, \ldots, 31.5\}$ and may point to outside the frame.

H.263

- Differential coding of the motion vectors:



$$MV_x = \mathrm{Med}(MV1_x, MV2_x, MV3_x)$$
$$MV_y = \mathrm{Med}(MV1_y, MV2_y, MV3_y)$$

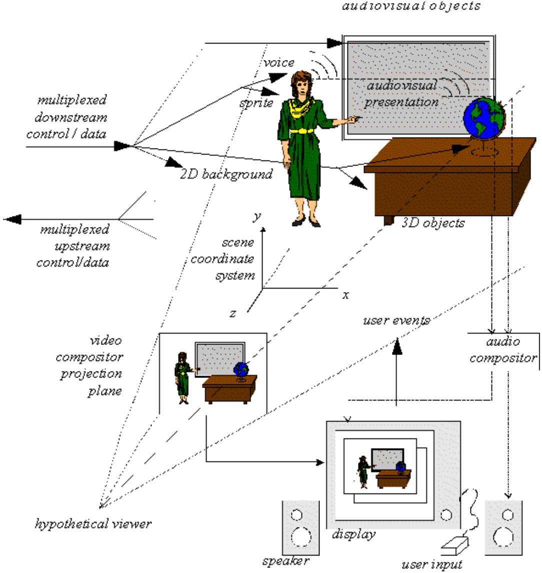
MPEG–4

- MPEG–4 (initial version in 1998) is a ISO/IEC standard providing tools for:
  - Representing audio, video or audiovisual data through media objects that can be natural (i.e., captured by a microphone or video camera) or synthetic (i.e., computer generated).
  - Describing the composition of these objects for creating composed objects and audiovisual scenes.
  - Multiplexing and synchronizing the data associated to the media objects, for transmission through the communication channels, providing an appropriate quality of service (QoS) to each object.
  - Enabling the interaction of the clients (receptor) with the audiovisual scene.

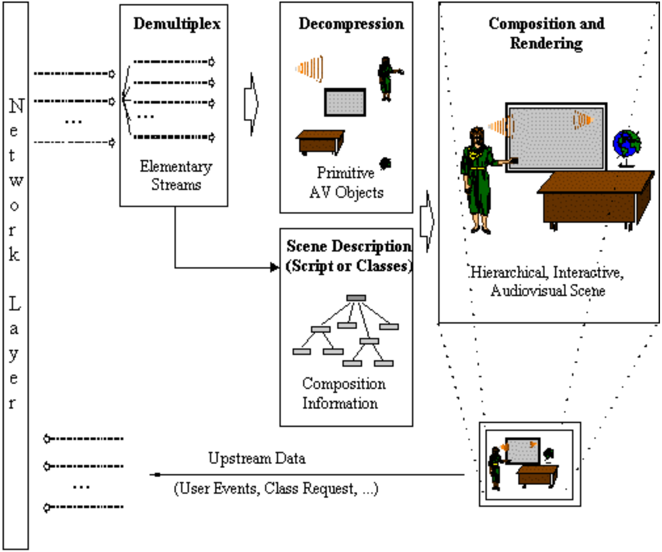
MPEG–4

- MPEG–4 defines several primitive objects for representing natural and synthetic information, as well as 2D and 3D data.
- The audiovisual scenes are composed of these media objects, hierarchically organized:
    - Images (for example, a fixed background).
    - Video objects (for example, a person talking).
    - Audio objects (for example, the voice of the person, background music, . . .).
    - Text and graphics.
    - Synthetic talking heads and the corresponding text used by the speech synthesizer; animated synthetic bodies.
    - Synthetic sound.
    - . . .

MPEG–4


audiovisual objects
voice
audiovisual presentation
sprite
multiplexed downstream control / data
2D background
3D objects
multiplexed upstream control/data
y
scene coordinate system
x
z
user events
video compositor projection plane
audio compositor
hypothetical viewer
display
speaker
user input

MPEG–4

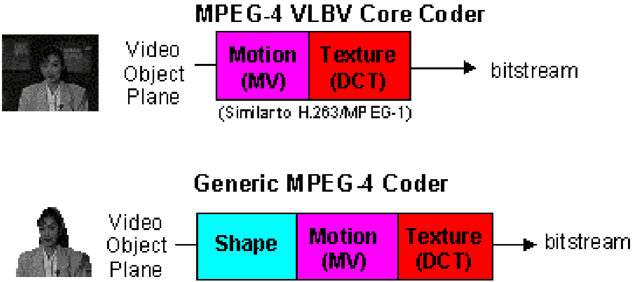

Network Layer
Demultiplex
Elementary Streams
Decompression
Primitive AV Objects
Scene Description (Script or Classes)
Composition Information
Composition and Rendering
Hierarchical, Interactive, Audiovisual Scene
Upstream Data
(User Events, Class Request, ...)

MPEG–4

- MPEG–4 provides algorithms and tools for representing visual objects (VO), originated both from natural and synthetic sources, including:
  - The compression of images and video.
  - The compression of textures for mapping in 2D or 3D meshes.
  - The compression of the information required for animating the 2D or 3D meshes.
  - Efficient random access to all types of visual objects.
  - Content-based image and video coding.
  - Scalability in terms of content, texture, image and video.
  - Fine granularity scalability (FGS), allowing the accommodation of more severe conditions (e.g., video over the Internet), avoiding the digital cutoff.
  - Temporal, spatial and SNR scalability.
  - Error protection.

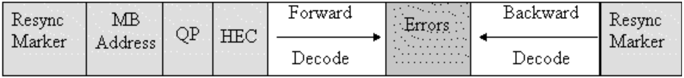
MPEG–4

- Conventional video coding is performed as in MPEG–1/2.
- In content-based coding, it is possible to encode regions with arbitrary shape, but, it this case, the shape of the object also needs to be efficiently represented.
- Shape is represented using a 8 bit transparency component or a binary mask.



MPEG–4

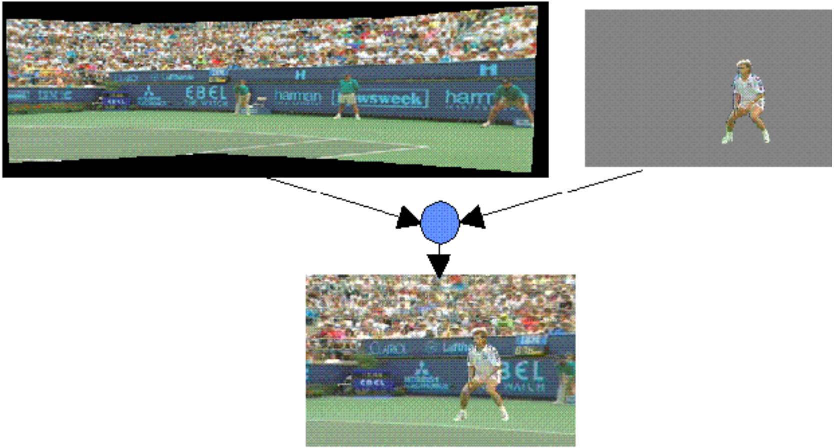
- MPEG–4 provides error protection (essential, for example, in wireless transmission), through:
  - Re-synchronization:
    - At the beginning of each GOB.
    - Periodically in the bitstream.
  - Data recovery:
    - Reversible variable length codes (RVLC).



  - Error concealment.
    - Reproduction of the block from the previous frame.

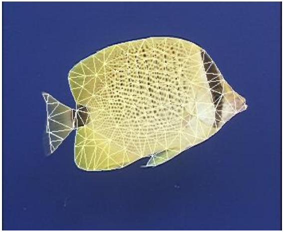
MPEG–4

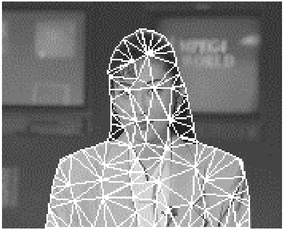
- Sprites:



MPEG–4

- MPEG–4 supports synthetic visual objects:
  - Parametric description of human heads and bodies (also body animation in Version 2).
  - Parametric description of static or dynamical meshes with texture mapping.





  - Scalable texture coding.

H.264/AVC

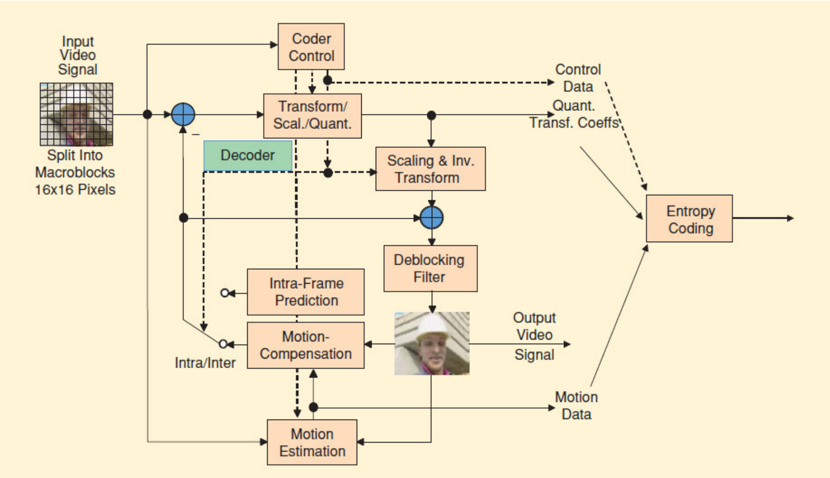
Overview

- H.264/AVC (Advanced Video Coding) was jointly developed by the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC MPEG.
- It was finalized in March 2003 and approved by the ITU-T in May 2003.
- H.264/AVC provides gains in compression efficiency of up to 50% over a wide range of bit rates and video resolutions compared to previous standards.
- The decoder complexity is about four times that of MPEG-2 and two times that of MPEG-4 Visual Simple Profile.

H.264/AVC

Overview

Block diagram of a typical encoding process of H.264/AVC
(IEEE SP Magazine, Mar 2007)



H.264/AVC

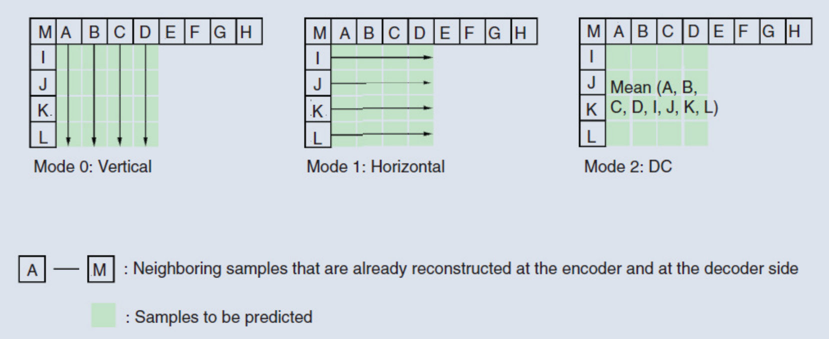
Intra prediction

- The samples of a macroblock are predicted using only information of already transmitted macroblocks of the same image.
- There are two types of Y intra prediction: INTRA_4$\times$4 and INTRA_16$\times$16.
- In INTRA_4$\times$4, the (16$\times$16) macroblock is divided into 16 4$\times$4 subblocks. There are 9 different prediction modes.
- In INTRA_16$\times$16, one out of four different prediction modes (vertical, horizontal, DC and plane) is applied to each macroblock.
- The intra prediction for the C_b and C_r components is similar to the INTRA_16$\times$16 type, but applied to 8$\times$8 blocks.

H.264/AVC

Intra prediction

Three out of nine possible intra prediction modes for INTRA_4$\times$4
(IEEE CS Magazine, 1st Quarter 2004)



H.264/AVC

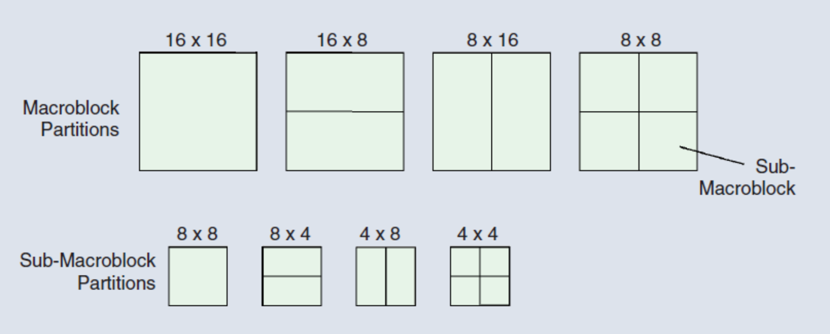
Motion-compensated prediction

- Each macroblock can be divided into smaller partitions: 16×16, 16×8, 8×16 and 8×8.
- In a P-slice, a 8×8 sub-macroblock can be further divided into blocks of 8×4, 4×8 or 4×4.
- In H.264/AVC it is possible to refer to several preceding images.
- The accuracy of the displacement vectors is 1/4 of a pixel.
- It is possible to use images containing B-slices as reference images.

H.264/AVC

Motion-compensated prediction

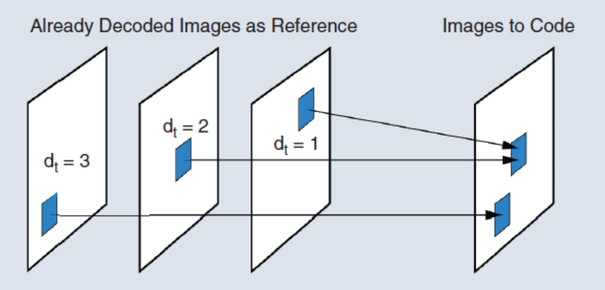
Partition of macroblock/sub-macroblock for motion-compensation
(IEEE CS Magazine, 1st Quarter 2004)



H.264/AVC

Motion-compensated prediction

Motion-compensated prediction with multiple reference images
(IEEE CS Magazine, 1st Quarter 2004)



H.264/AVC

Transform coding

- Instead of the DCT, three different integer transforms are used:

$$H_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \quad H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \quad H_3 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- They are mostly applied to 4×4 blocks, but can also be applied to 2×2 blocks.
- H_1 is applied to all prediction error blocks of Y, C_b and C_r. If the macroblock is predicted using type INTRA_16×16, then H_2 is applied in addition to H_1.
- H_3 is used for transforming the 4 DC coefficients of each chrominance component.

H.264/AVC

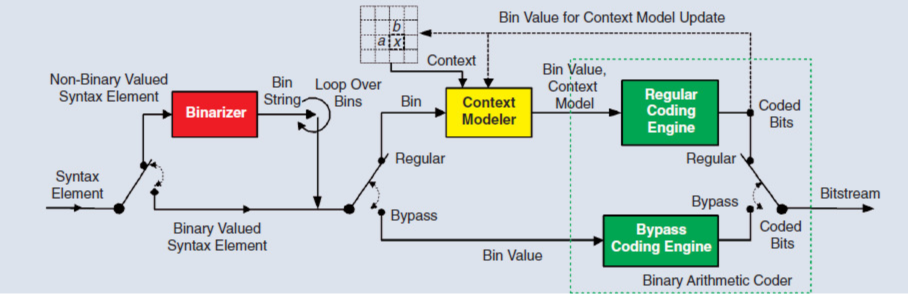
Entropy coding

- H.264/AVC provides two methods for entropy coding:
  - CAVLC, a low-complexity technique based on context-adaptive sets of variable length codes.
  - CABAC, a context-based adaptive binary arithmetic encoder.
- By incorporating context modeling, both methods offer a high degree of adaptation to the underlying source.
- CAVLC relies on 32 different VLCs. For typical coding conditions, it is 2–7% better than conventional codes.
- Typically, CABAC provides bit rate reductions of 5–15% compared to CAVLC.

H.264/AVC

CABAC

Context-based adaptive binary arithmetic coding
(IEEE CS Magazine, 1st Quarter 2004)

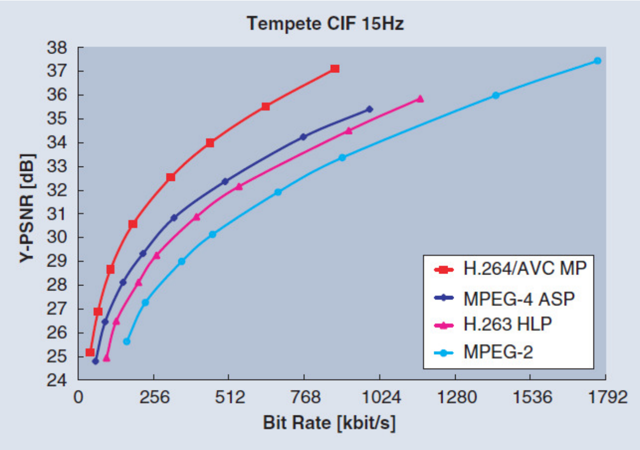


H.264/AVC

Performance

Video streaming application

(IEEE CS Magazine, 1st Quarter 2004)

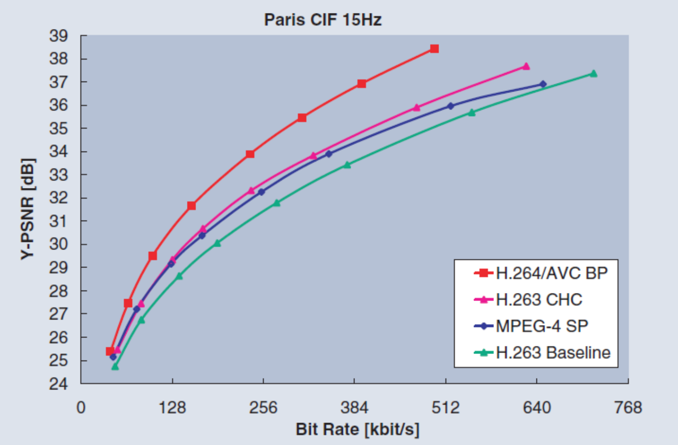


H.264/AVC

Performance

Video conferencing application

(IEEE CS Magazine, 1st Quarter 2004)



H.265/HEVC

Overview

- H.265/HEVC (High Efficiency Video Coding) was (again) the result of a collaboration between the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC MPEG.
- It is also known as the MPEG-H Part 2 and the first version was finalized in 2013.
- H.265/HEVC can provide gains in compression efficiency of about 50%, when compared to H.264/AVC.
- This is mostly attained by further exploring existing techniques, but at a cost of increasing the complexity of the encoder.
- As with H.264/AVC, H.265/HEVC is dependent of a considerable number of patents, which is preventing its wide use. . .

H.265/HEVC

Block diagram