BASICS OF MULTIMEDIA TECHNOLOGIES

Lecture slides

BME Dept. of Networked Systems and Services

2018.

Video bit rate requirements

Bit rate of active video content for uncompressed video stream (8 bits/component)

- 1080i50/4:2:2 (1080 x 1920 + 2 x 1080 x 960) x 8 x 25 = 829 Mbit/s
- 1080i50/4:2:0 (1080 x 1920 + 2 x 540 x 960) x 8 x 25 = 622 Mbit/s
- 720p50/4:2:2 (720 x 1280 + 2 x 720 x 640) x 8 x 50 = 737 Mbit/s
- 720p50/4:2:0 (720 x 1280 + 2 x 360 x 640) x 8 x 50 = 552 Mbit/s
- ITU-601/4:2:2 (SD) (576i50) (576 x 720 + 2 x 576 x 360) x 8 x 25 = 166 Mbit/s
- ITU-601/4:2:0 (SD) (576i50) (576 x 720 + 2 x 288 x 360) x 8 x 25 = 124 Mbit/s
- CIF (288 x 360 + 2 x 144 x 180) x 8 x 30 = 37,3 Mbit/s
- QCIF (144 x 180 + 2 x 72 x 90) x 8 x 30 = 9,3 Mbit/s
- SIF (288 x 352 + 2 x 144 x 176) x 8 x 25 = 30,4 Mbit/s

Size of 1 hour uncompressed video stream

- 1080i50/4:2:2 829 Mbit/s : 347 Gbyte
- 1080i50/4:2:0 622 Mbit/s : 260 Gbyte
- 720p50/4:2:2 737 Mbit/s : 308 Gbyte
- 720p50/4:2:0 552 Mbit/s : 231 Gbyte
- ITU-601/4:2:2 (SD) (576i50) 166 Mbit/s : 70 Gbyte
- ITU-601/4:2:0 (SD) (576i50) 124 Mbit/s : 51 Gbyte
- CIF 37,3 Mbit/s :15 Gbyte
- QCIF 9,3 Mbit/s : 3.8 Gbyte

Fundamentals of video compression

Types of redundancy:

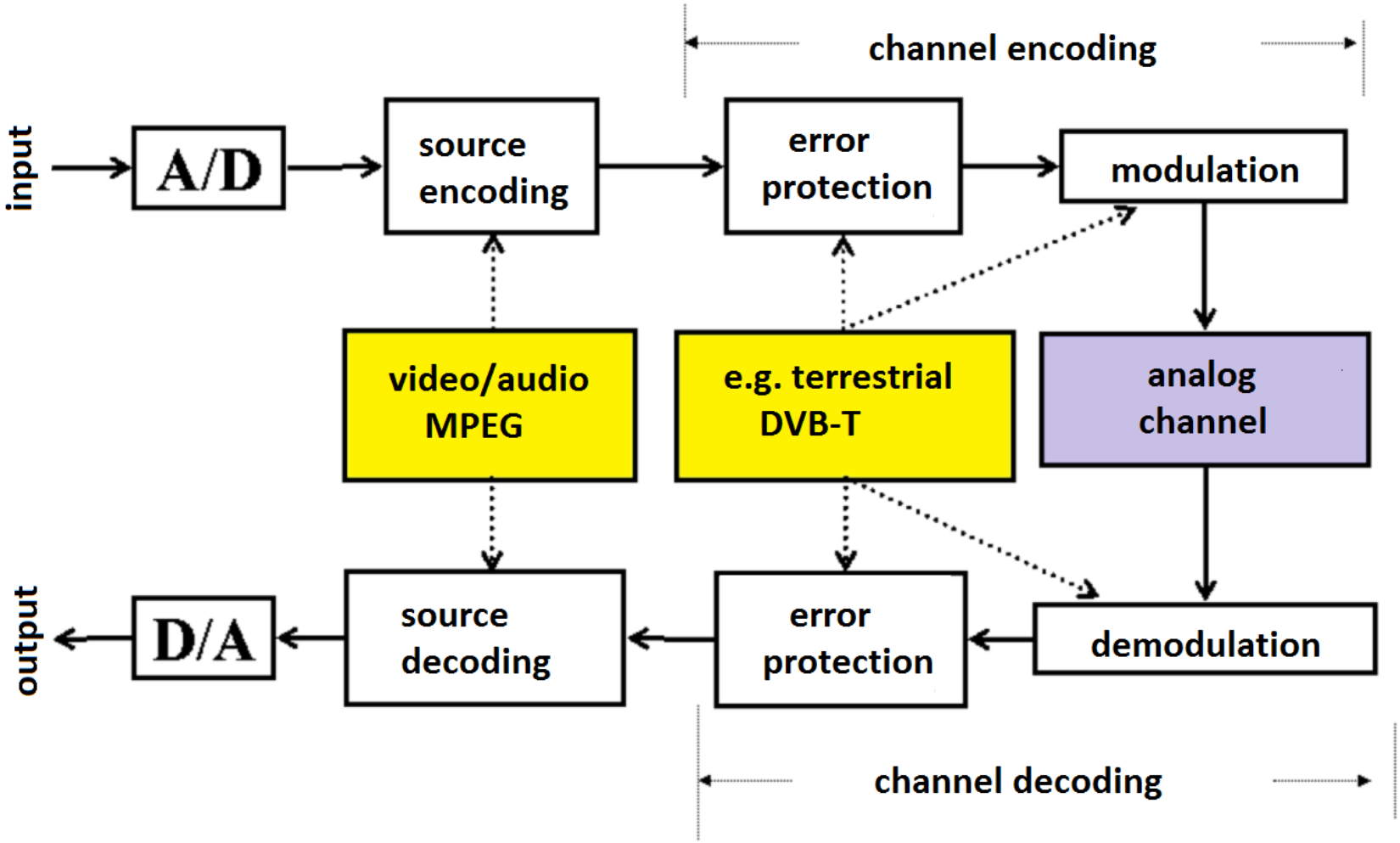
- Statistical redundancy:
  - Spatial redundancy: intra-frame redundancy: correlation among pixels within one frame
  - Temporal redundancy: inter-frame redundancy: correlation among the content of consequent frames
- Psychovisual redundancy: perceptually unimportant information, i.e. what the human visual system (HVS) can not see
- Coding Redundancy: the uncompressed image usually is code with each pixel by a fixed length. Using variable length code schemes such as Huffman coding and arithmetic coding may produce compression

Fundamentals of video compression

Important properties of HVS in the aspect of compression

- Spatial resolution of HVS is 3-5 times higher for brightness (black and white images) than for colors
- For stationary and slowly changing video content higher resolution is required
- For stationary and slowly changing video content HVS is more sensitivity for change in video quality
- For scene change and rapidly moving content low quality is less disturbing

Digital channel model

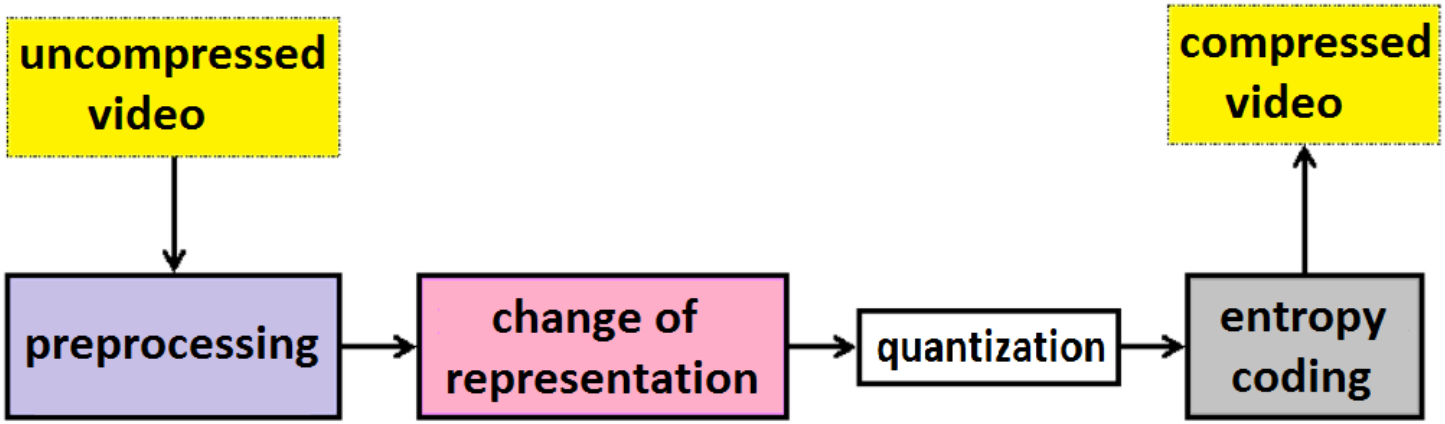


Elements of digital channel model

Source encoding

- The properties of source (video/audio) and the sink (auditory system/HVS) are taken into consideration: reducing the source redundancy by keeping only the relevant information, so that the sink (e.g. HVS) „doesn't notice any change"
  - Change of representation: in the new representation space less redundancy (more compact representation), e.g. prediction, transform coding, motion compensation
  - Irreversible coding: changing the accuracy of representation by removing irrelevant information, e.g. quantization, spatial/temporal subsampling
  - Reversible coding: finding an efficient code-assignment, reducing statistical redundancy, e.g. Variable Length Coding, Run-length coding

Source encoding model

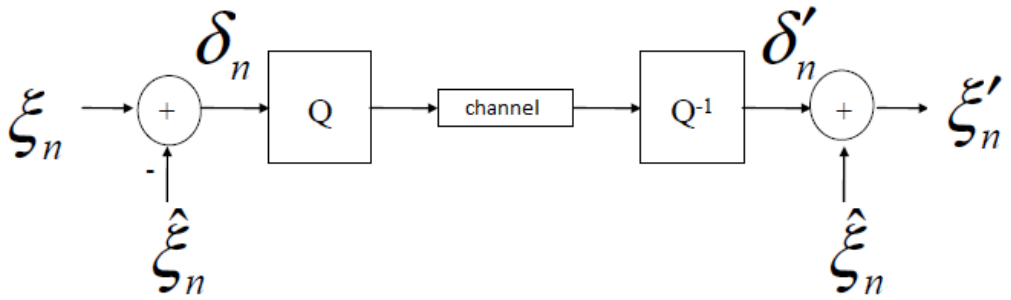


- prefilterin
- format conversion

- prediction
- transform coding
- sub-band coding

- Run-length coding
- Variable length coding

Differential quantization



- ξ_n: source sample
- $\hat{\xi}_n$: estimated/predicted source sample
- $\delta_n = \xi_n - \hat{\xi}_n$: differential/error signal
- $\delta'_n = \delta_n + \epsilon_n$: quantized error (effect of quantization: additive noise)
- $\xi'_n = \xi_n + \epsilon_n$

- Resultant SNR : $SNR_p = \frac{\mathbb{E}[\xi_n^2]}{\mathbb{E}[\epsilon_n^2]}$
- $SNR_p = \frac{\mathbb{E}[\xi_n^2]}{\mathbb{E}[\epsilon_n^2]} = \frac{\mathbb{E}[\xi_n^2]}{\mathbb{E}[\delta_n^2]} \frac{\mathbb{E}[\delta_n^2]}{\mathbb{E}[\epsilon_n^2]}$
- **Prediction gain**: $G_p = \frac{\mathbb{E}[\xi_n^2]}{\mathbb{E}[\delta_n^2]}$
- Optimal quantizer SNR: $SNR_q = \frac{\mathbb{E}[\delta_n^2]}{\mathbb{E}[\epsilon_n^2]}$
- $\mathbf{SNR_p = G_p\ SNR_q}$

Linear prediction (for 1D signal)

Goal: minimizing the energy of differential signal δ_n in a mean square sense (LMS):

$$\min : \mathbb{E}[|\xi_n - \hat{\xi}_n|^2],$$

Let the estimation be a linear combination of past samples

$$\hat{\xi}_n = \mathbf{a}^T \boldsymbol{\xi}_{n-1},$$

where $\mathbf{a} = [a_1, \ldots, a_N]$ and $\boldsymbol{\xi}_{n-1} = [\xi_{n-1}, \ldots, \xi_{n-N}]$, so that

$$\mathbb{E}[|\xi_n - \hat{\xi}_n|^2] = \mathbb{E}[\xi_n^2 - 2\xi_n \mathbf{a}^T \boldsymbol{\xi}_{n-1} + \mathbf{a}^T \boldsymbol{\xi}_{n-1} \boldsymbol{\xi}_{n-1} \mathbf{a}]$$

Let's make use that the autocorrelation of the source : $r_\xi(m) = \mathbb{E}[\xi_n \xi_{n-m}]$

Linear prediction (for 1D signal)

$$\mathbb{E}[|\xi_n - \hat{\xi}_n|^2] =$$

$$= \mathbb{E}[\xi_n^2 - 2\xi_n \mathbf{a}^T \boldsymbol{\xi}_{n-1} + \mathbf{a}^T \boldsymbol{\xi}_{n-1} \boldsymbol{\xi}_{n-1} \mathbf{a}] =$$

$$= r_\xi(0) - 2\mathbf{a}^T \begin{bmatrix} r_\xi(1) \\ r_\xi(2) \\ \vdots \\ r_\xi(N) \end{bmatrix} + \mathbf{a}^T \begin{bmatrix} r_\xi(0) & r_\xi(1) & \dots & r_\xi(N-1) \\ r_\xi(1) & r_\xi(0) & \dots & r_\xi(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_\xi(N-1) & r_\xi(N-2) & \dots & r_\xi(0) \end{bmatrix} \mathbf{a}$$

Linear prediction

$$\mathbf{r} = \begin{bmatrix} r_\xi(1) \\ r_\xi(2) \\ \vdots \\ r_\xi(N) \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} r_\xi(0) & r_\xi(1) & \dots & r_\xi(N-1) \\ r_\xi(1) & r_\xi(0) & \dots & r_\xi(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_\xi(N-1) & r_\xi(N-2) & \dots & r_\xi(0) \end{bmatrix}$$

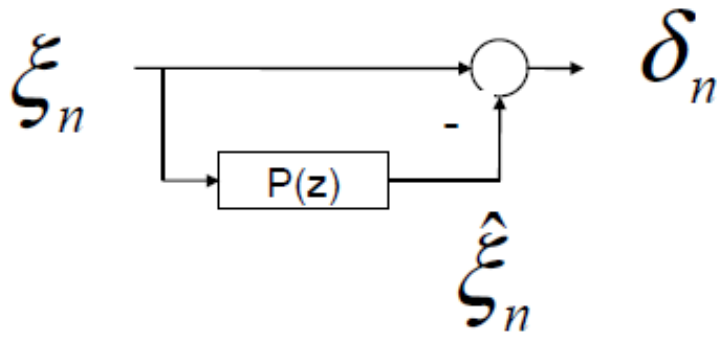
$$\min : \mathbb{E}[|\xi_n - \hat{\xi}_n|^2] = r_\xi(0) - 2\mathbf{a}^T\mathbf{r} + \mathbf{a}^T\mathbf{R}\mathbf{a}$$

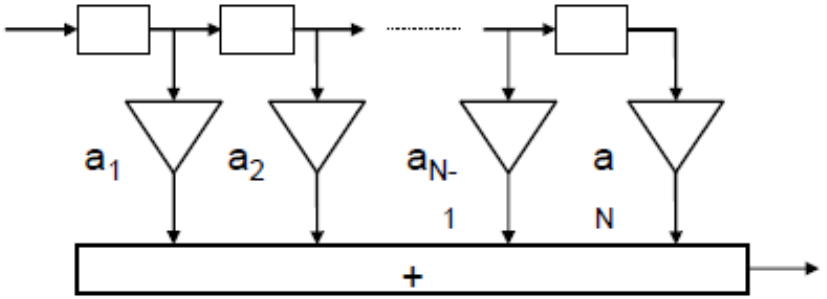
$$\frac{\partial \mathbb{E}[|\xi_n - \hat{\xi}_n|^2]}{\partial \mathbf{a}} = -2\mathbf{r} + 2\mathbf{R}\mathbf{a} = 0$$

$$\mathbf{a} = \mathbf{R}^{-1}\mathbf{r}$$

Linear prediction

Linear prediction is realized by a FIR filter

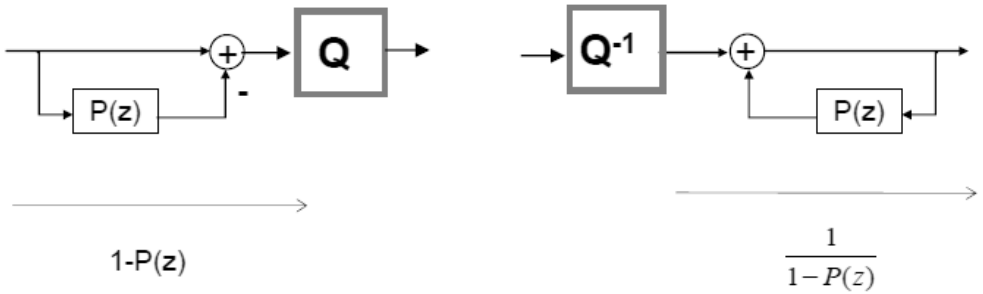




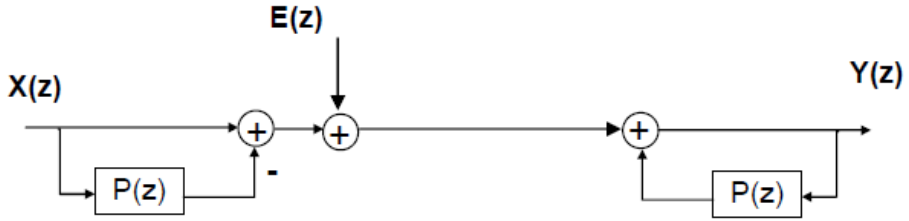
$$P(z) = \sum_{i=1}^{N} a_i z^{-i}$$

Differentiation and quantization

Feed-forward prediction



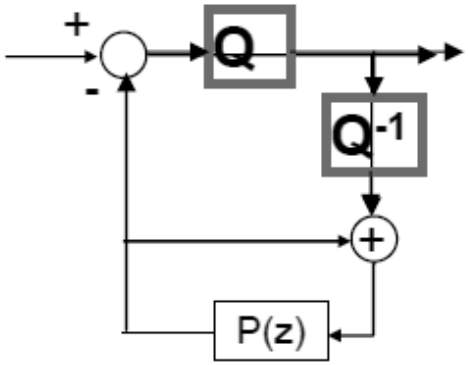
Effect of quantization as additive noise

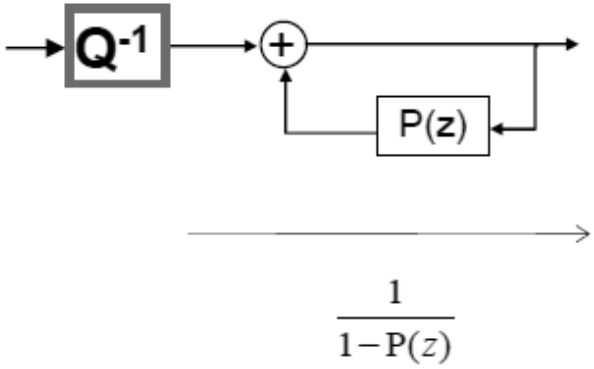


$$Y(z) = X(z) + \frac{1}{1-P(z)} E(z)$$

Feedback prediction loop

Quantizer, inverse quantizer and ”decoder” inside the loop

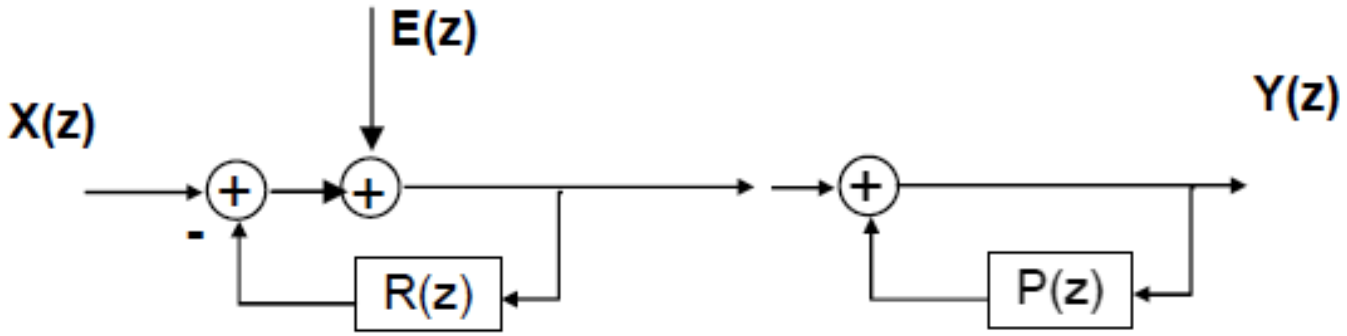




$$R(z) = \frac{P(z)}{1 - P(z)}$$

Feedback prediction loop

Effect of quantization as additive noise



$$Y(z) = X(z) + E(z)$$

Signal at the output of encoder block

$$\frac{1}{1+R(z)}X(z) + \frac{1}{1+R(z)}E(z) = [1-P(z)][X(z)+E(z)]$$

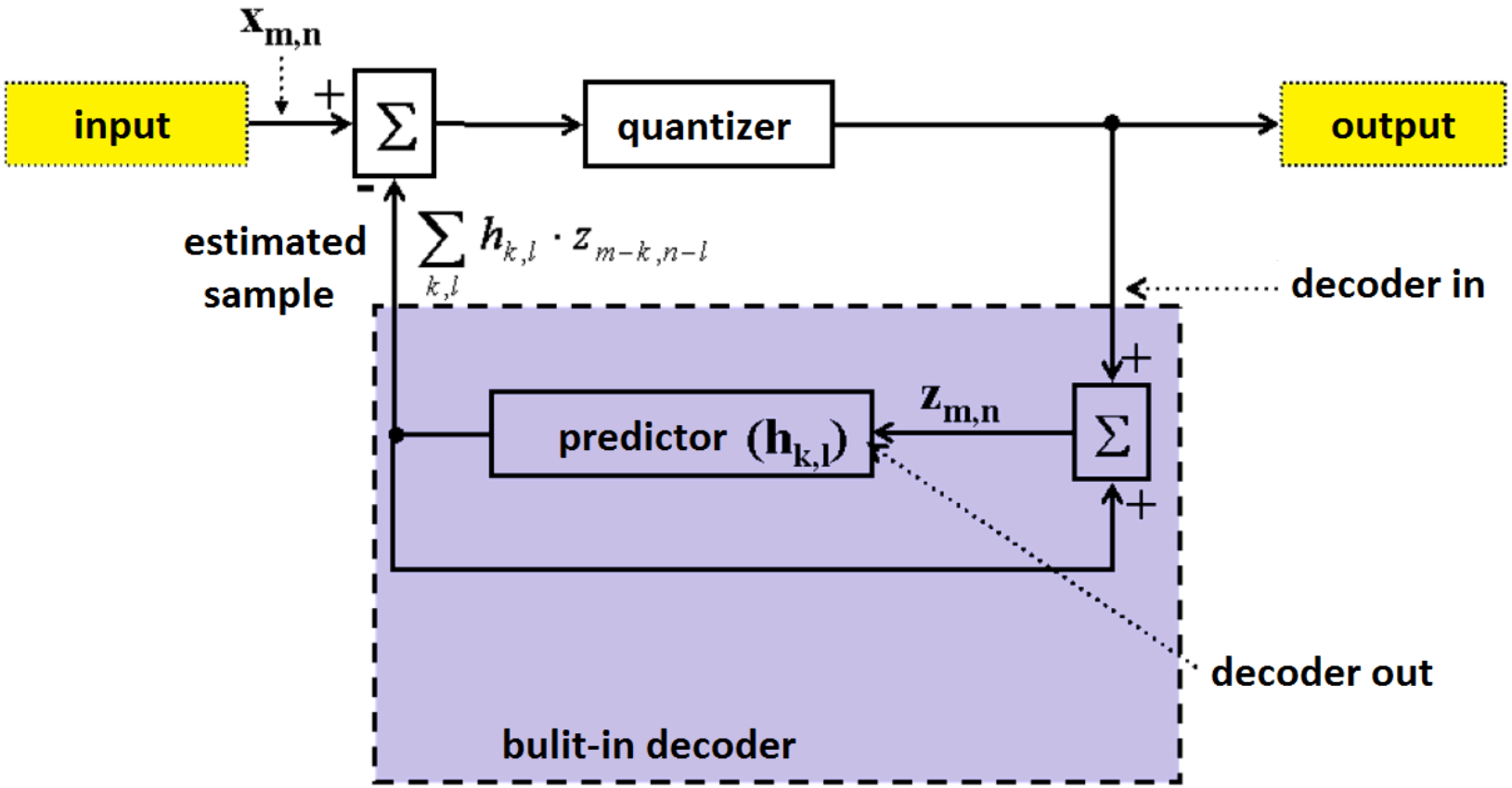
the transfer function of the decoder

$$\frac{1}{1-P(z)}$$

Linear prediction for video

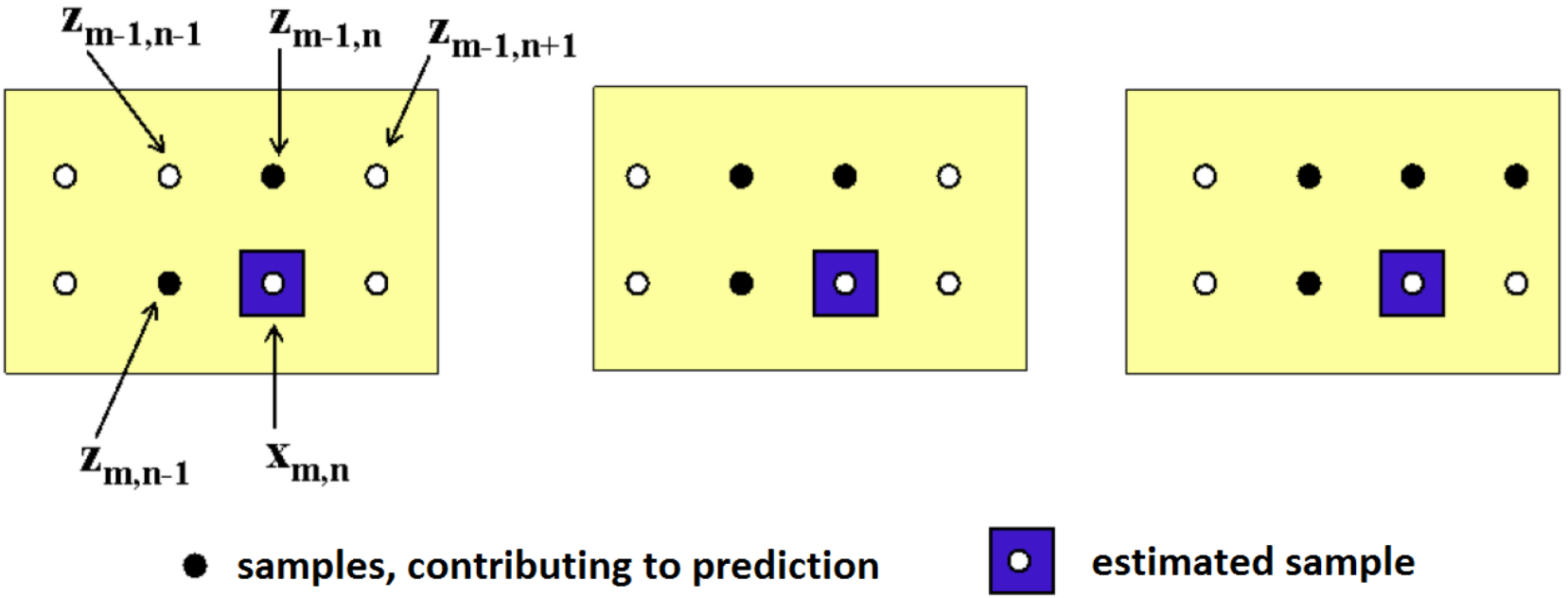
- can be applied between pixels of an image (spatial prediction) or between consequent images (temporal prediction)
- instead of storing the actual sample/pixel $x_{m,n}$ the difference from the estimation is coded
- the estimated pixel is obtained as the linear combination of adjacent samples
- at the decoder side: only difference signal is available (along with the coefficients of linear combination)
- in order to avoid accumulation of quantization error the encoder contains the decoder model (feedback prediction loop)
- estimation of the actual pixel: based on the samples, decoded previously

Block scheme of predictive encoder I.



Block scheme of predictive encoder II.

Frequently used (2D) prediction schemes (PNG)



Transform coding I.

- correlation between adjacent samples of natural images is high
- aim: transform the 2D blocks of images into a new representation space (change of basis), where redundancy (correlation) between samples is lower
- thus: transform coding is usually block-based: $N \times N$ sized blocks of pixels are transformed separately
- transform: linear operation: non-overlapped blocks of pixels are multiplied with a *transform matrix*
- elements of the resulting matrix are usually less correlated
- further coding steps: done in this transform space

Transform coding II.

1D transform

- input vector: $\mathbf{x} = [x(0)\ x(1) \ldots x(N-1)]^T$
- output (coefficient) vector: $\mathbf{y} = [y(0)\ y(1) \ldots y(N-1)]^T$
- relation of input and output vectors:
  - $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{A}$ is the transform matrix
  - $\mathbf{x} = \mathbf{B}\mathbf{y}$, where $\mathbf{B}$ is the inverse transform matrix, $\mathbf{B} = \mathbf{A}^{-1}$

Transform coding II.

- For unitary transformations ($\mathbf{A}^H\mathbf{A} = \mathbf{A}\mathbf{A}^H = \mathbf{E}$):

$$\mathbf{B} = \mathbf{A}^{-1} = \mathbf{A}^H$$

- For orthogonal transformations (with real valued transform matrix $\mathbf{A}^H\mathbf{A} = \mathbf{A}\mathbf{A}^H = \mathbf{E}$):

$$\mathbf{B} = \mathbf{A}^{-1} = \mathbf{A}^H$$

  thus the columns of the transform matrix A form an orthonormal basis
- in image compression/processing the frequently applied transformations are unitary/orthogonal

Transform coding III.

– For unitary transforms the inverse transform matrix can be written in the form

$$\mathbf{x} = \sum_{k=0}^{N-1} y(k)\mathbf{a}_k^*,$$

where $\mathbf{a}_k^*$ are the columns of $\mathbf{A}^H$, forming the **vector basis** of the transform.

– Similarly, in case of orthogonal transform

$$\mathbf{x} = \sum_{k=0}^{N-1} y(k)\mathbf{a}_k$$

where $\mathbf{a}_k$ are the columns of $\mathbf{A}^T$.

– Thus inverse transform: series expansion of the input vector $\mathbf{x}$ as the sum of the basis vectors $\mathbf{a}_k^*$ (or $\mathbf{a}_k$) weighted by the coef. vector $y(k)$

Transform coding IV.

2D linear transform

– The $N \times N$ sized input matrix is transformed by multiplying with a transfrom hypermatrix, reading

$$\mathbf{Y} = Y[k,l] = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x[m,n] A(k,m;l,n)$$

while the inverse transform in case of orthogonal transform reads

$$\mathbf{X} = x[m,n] = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} Y[k,l] A^T(k,m;l,n)$$

Transform coding V.

2D linear transform

- The $N \times N$ sized input blocks of an image are non-overlapping, and fill the entire image
- Again: physical interpretation of inverse transform: expanding the original image block as the weighted sum of *basis images (matrices)* (series expansion)
- Basis images: $A^T(k, m; l, n)$, or with different notation: $A^T_{k,l}(m, n)$
- In the transform domain/space the elements of coefficient matrix are less correlated (than in the input image block)
- Furher steps of encoding (quantization, entropy coding) are performed on the coefficient matrix

Transform coding VI.

2D separable transform

- if the hypermatrix of a 2D transform $A(k, m; l, n)$ can be written as a product $A(k, m; l, n) = A_1(k, m)\ A_2(l, n)$ the 2D transform is separable: the 2D transform is equivalent to consequent row-wise and column-wise 1D transformations
- if $A(k, m; l, n)$ is unitary/orthogonal: $A_1(k, m)$ and $A_2(l, n)$ are also unitary/orthogonal, and usually also $\mathbf{A}_2 = \mathbf{A}_1^T$ holds
  - in this case $\mathbf{Y} = \mathbf{AXA}^T$ holds, and
  - if the transform is orthogonal: $\mathbf{X} = \mathbf{A}^T\mathbf{YA}$ holds.

Transform properties

Properties

- In case of unitary/orthogonal transforms the Parseval's theorem (conservation of energy) holds
- $\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}|x[m,n]|^2 = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1}|y[k,l]|^2$
  - Let $\mathbf{E}_y = \mathbf{Y} - \hat{\mathbf{Y}}$ denote an error in the transform space (e.g. due to quantization)
  - Let$\mathbf{E}_x = \mathbf{X} - \hat{\mathbf{X}}$ denote the error after the inverse transform, compared the unaltered, original input matrix
- Due to Parseval's theorem and linearity of transform ($\mathbf{E}_y = \mathbf{A}\mathbf{E}_x\mathbf{A}^T$) $||\mathbf{E}_x||^2 = ||\mathbf{E}_y||^2$ holds, therefore if the energy of error is minimized in the transform domain, it means a minimized error energy in the spatial domain as well

General comments on unitary/orthog. transforms

- The transform itself is theoretically lossless: consequent forward and inverse transform return the original input (aside from numeric errors)
- Transformed representation is more compact: energy of input signal is distributed amongst a few coefficients
- Efficiency of transform: number of significant coefficients
- If inverse transform is performed using significant coefficients the output differs from the original, but the discrepancy is not considerable

Types of transforms applied

- Optimal transform: zero correlation between the coefficient matrix (after the linear transform)
- Equivalent statement: the representation of the input data is the most compact
- Achieving optimal transform: the input signal is transformed into the subspace, spanned by the eigenvectors of its autocorrelation matrix
- I.e. basis vectors: eigenvectors of input correlation matrix $\rightarrow$ basis depends on the actual input
- In practice: fixed-base transforms are applied
- These are sub-optimal, but suitable for real-time application
- Otimal transform: KLT (Karhunen-Love Transform)

Types of transforms applied II.

Transforms with non-harmonic basis

- DWHT
- DHT
- DST
- ...

Transforms with harmonic basis

- DFT
- DCT
- DST
- Wavelet

Discrete Hadamard transform

Non-harmonic basis

- Real, orthogonal transform
- Transform matrix of 2×2-sized transform:

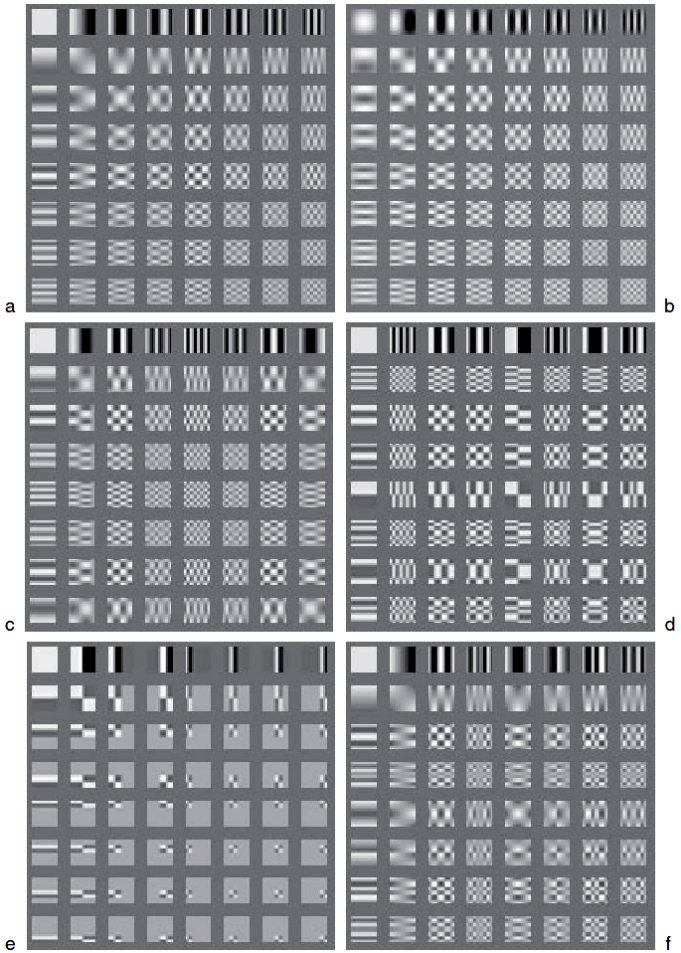
$$\mathbf{H}_{2\times 2} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- For larger block-sizes the transform matrix $\mathbf{H}$ is calculated recursively:

$$\mathbf{H}_{2N\times 2N} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{H}_{N\times N} & \mathbf{H}_{N\times N} \\ \mathbf{H}_{N\times N} & -\mathbf{H}_{N\times N} \end{bmatrix}$$

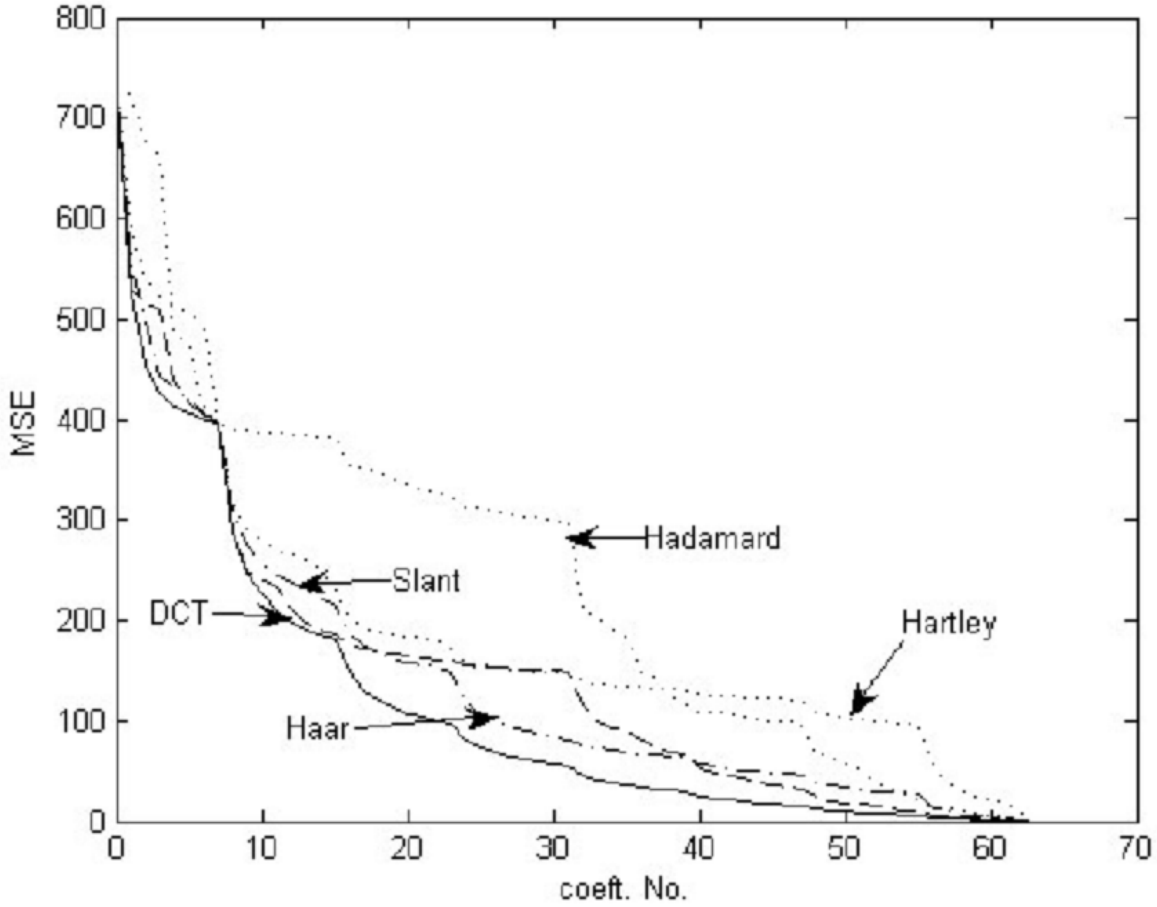
- The formulation of 2D DHT
  - $\mathbf{y} = \mathbf{H}\mathbf{x}\mathbf{H}^T$ and
  - $\mathbf{x} = \mathbf{H}^T\mathbf{y}\mathbf{H}$

2D basis images



a: DCT, b: DST, c: Hartley, d: Hadamard, e: Haar, és f: Slant transformations

MSE as the function of transform type



(Cameraman image)

DCT transform

Advantages

- Natural images are unccorrelated almost perfectly
- Quasi-optimal (nearly optimal) transform
- Most of the energy of input image is contained by low frequency coefficients
- Results in real coefficients
- Can be computed efficiently (efficient algorithms exist)

DCT transform

1D DCT and IDCT

– DCT

$$X(k) = \sqrt{\frac{2}{N}}\,\alpha(k)\,\sum_{n=0}^{N-1} x(n)\,\cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \qquad k = 0,\ldots,N-1$$

– IDCT

$$x(n) = \sqrt{\frac{2}{N}}\,\sum_{n=0}^{N-1} \alpha(k)\,X(k)\,\cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \qquad n = 0,\ldots,N-1$$

– where

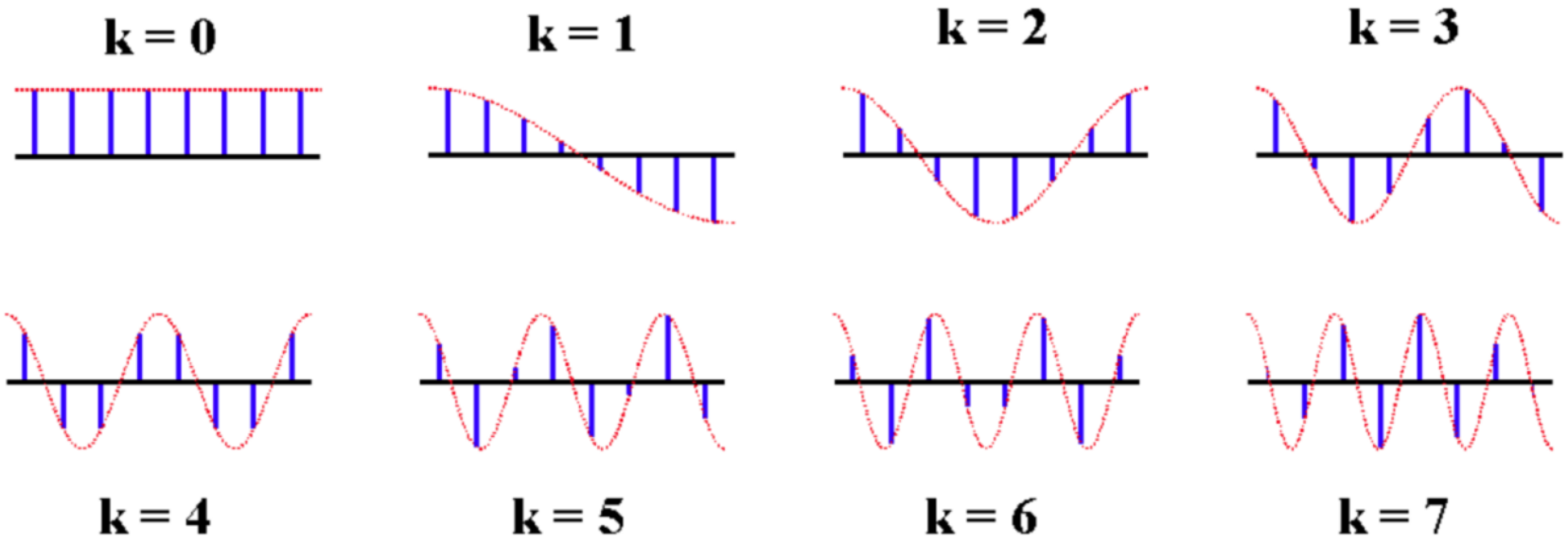
$$\alpha(k) = \begin{cases} \frac{1}{\sqrt{2}} & \text{ha } k \equiv 0 \\ 1 & \text{ha } k \neq 0. \end{cases}$$

1D DCT and IDCT

Terminology

- DC coefficient: $X(0)$
- AC coefficient$X(1), \ldots X(N-1)$
- The k-th basis vector:
  $\sqrt{\frac{2}{N}}\alpha(k)\cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \quad n=0,\ldots,N-1$

Basis vectors in case of $N = 8$



- The input vector is constructed as the linear combination of harmonic components (cosine function) with increasing frequency
- Weighting factors: DCT coefficients
- If input signal is smooth, i.e. „changes slowly”, the spectral representation is dominated by the DC and low frequency AC coefficients

2D DCT and IDCT

– 2D DCT

$$X(k,l) = \frac{2}{N}\,\alpha(k)\alpha(l)\sum_{n=0}^{N-1}\sum_{m=0}^{N-1} x(n,m)\,\cos\left[\frac{\pi}{N}\left(m+\frac{1}{2}\right)k\right]\,\cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \quad k=0,\ldots,N-1$$

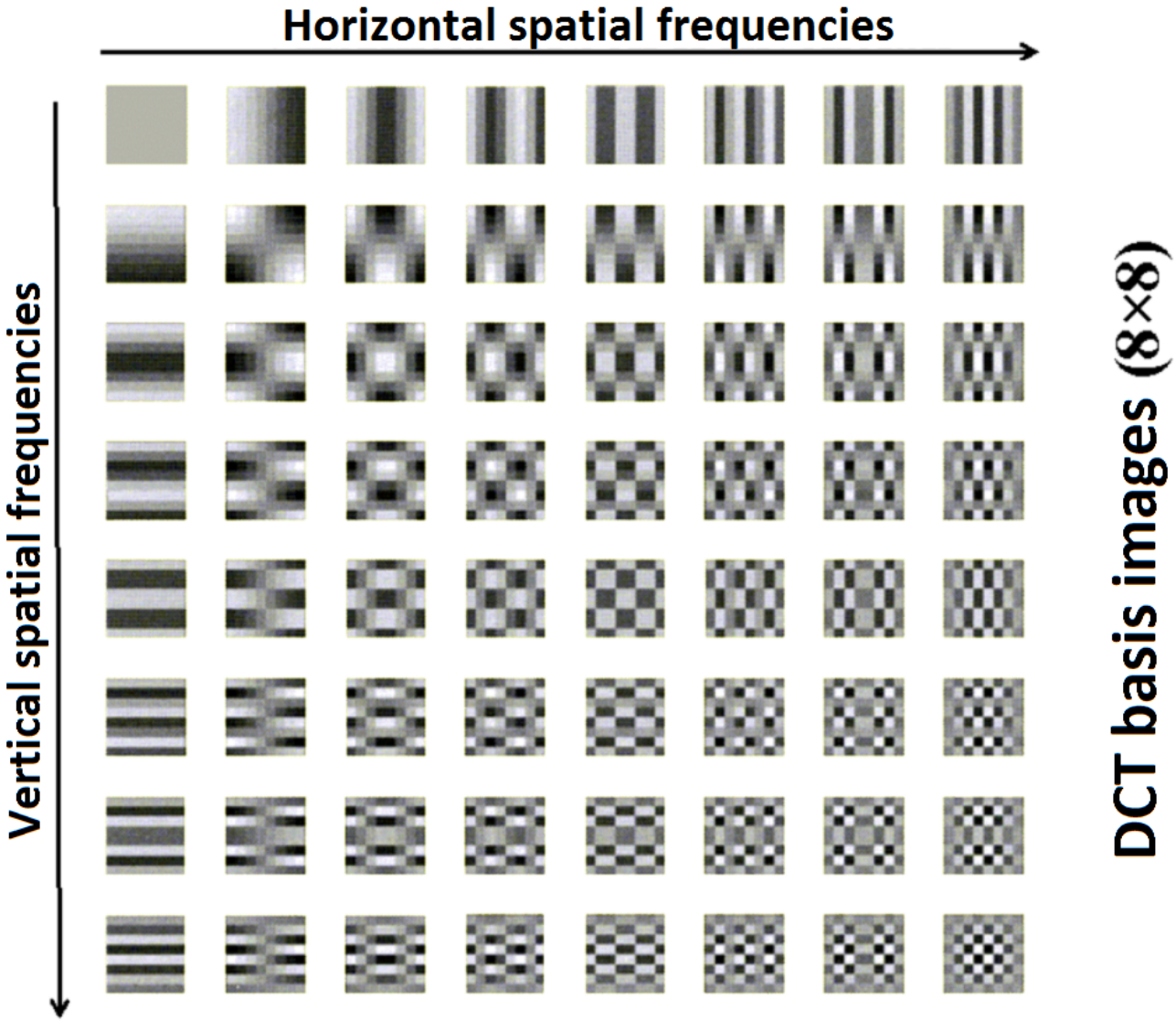
– 2D IDCT

$$x(m,n) = \frac{2}{N}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1}\alpha(k)\alpha(l)\,X(k,l)\,\cos\left[\frac{\pi}{N}\left(m+\frac{1}{2}\right)k\right]\,\cos\left[\frac{\pi}{N}\left(n+\frac{1}{2}\right)k\right] \quad k=0,\ldots,N-1$$

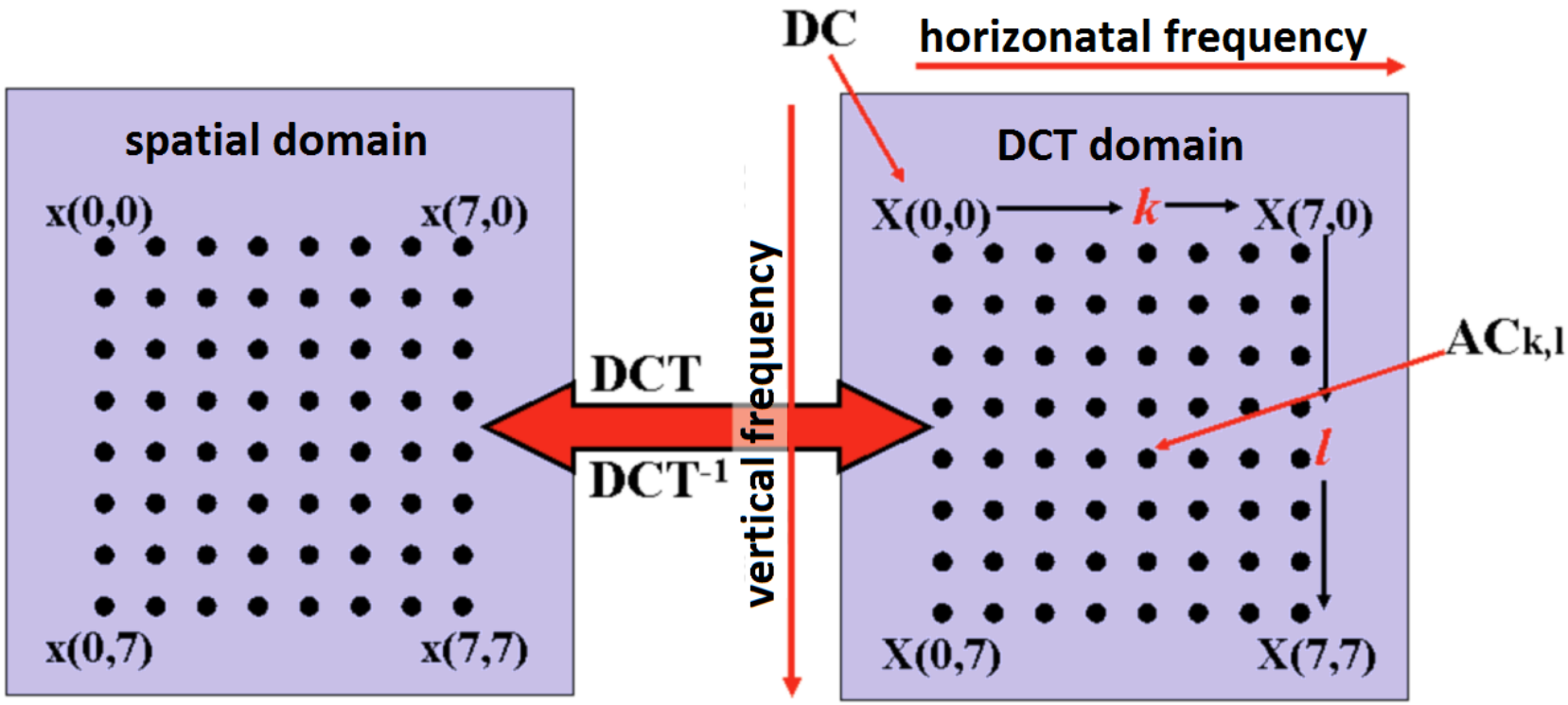
– where

$$\alpha(k),\alpha(l) = \begin{cases} \frac{1}{\sqrt{2}} & \text{ha } k,l \equiv 0 \\ 1 & \text{ha } k,l \neq 0. \end{cases}$$

2D DCT illustration I.



2D DCT illustration II.



2D DCT illustration III.

8×8 pixel matrix

$$\begin{bmatrix} 83 & 87 & 92 & 90 & 89 & 91 & 47 & 95 \\ 98 & 81 & 77 & 96 & 71 & 44 & 58 & 49 \\ 27 & 43 & 65 & 40 & 64 & 99 & 61 & 55 \\ 83 & 21 & 45 & 51 & 59 & 80 & 48 & 63 \\ 87 & 94 & 56 & 62 & 41 & 74 & 75 & 57 \\ 98 & 82 & 68 & 79 & 54 & 46 & 52 & 60 \\ 72 & 42 & 70 & 84 & 69 & 50 & 97 & 67 \\ 76 & 53 & 85 & 88 & 73 & 66 & 78 & 86 \end{bmatrix}$$

DCT

8×8 coefficient matrix

$$\begin{bmatrix} 248 & 18 & -12 & 2 & 3 & -7 & 0 & 0 \\ 10 & 9 & 8 & 1 & -4 & 1 & 0 & 0 \\ -6 & 2 & 1 & 0 & 2 & 4 & 2 & -1 \\ 2 & 1 & -1 & 1 & -3 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 & 2 & 0 & 0 & -1 \\ 1 & -2 & 1 & 0 & 1 & -2 & 3 & 0 \\ 0 & 0 & 0 & -1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Calculation of 2D DCT

2D DCT: 2D separable transform

- $\mathbf{A} = A[k, m] = \sqrt{\frac{2}{N}}\, \alpha(k) \cos\left[\frac{\pi}{N}\left(m + \frac{1}{2}\right) k\right]$ denotes the transform matrix of 1D DCT
- $\mathbf{X} = x[m, n]$ is the 2D image matrix (input matrix)
- Then $\mathbf{Y}_c = \mathbf{AX}$ is the column-wise DCT and $\mathbf{Y}_{c,r} = \mathbf{AY}_c^T$ is the row-wise DCT of the input
- Thus $\mathbf{Y}_{c,r} = \mathbf{A}(\mathbf{XA})^T = \mathbf{AX}^T\mathbf{A}^T$
- and the 2D DCT of the input reads as $\mathbf{Y}_{k,l} = \mathbf{Y}_{c,r}^T$

Quantization of Coefficient matrix

Quantization matrix

- Qunatization is performed block-wise (block by block) by the quantization matrix
- Quantized matrix: $X^*(k,l) = \text{round}\left(\frac{X(k,l)}{W(k,l)}\right)$, where $W(k,l)$ is the quantization factor of the coefficient (k,l)
- Elements of quantization matrix: chosen based on the properties of HVS
- Rounding: result is rounded to the nearest integer (data loss)

Quantization of Coefficient matrix

Quantization based on the properties of HVS

- HVS is less sensitive to small, details: energy of tiny details contribute mainly to high frequency DCT coefficients
- Quantization: Relevant coefficients (DC and low frequencies) are quantized less, high frequency components quantized more rough
- Typical quantizer matrix: high frequency components larger $\rightarrow$ after division and rounding the high frequency components become close to zero

$$W(k,l)=\begin{bmatrix} 8 & 16 & 19 & 22 & 26 & 27 & 29 & 34 \\ 16 & 16 & 22 & 24 & 27 & 29 & 34 & 37 \\ 19 & 22 & 26 & 27 & 29 & 34 & 34 & 38 \\ 22 & 22 & 26 & 27 & 29 & 34 & 37 & 40 \\ 22 & 26 & 27 & 29 & 32 & 35 & 40 & 48 \\ 26 & 27 & 29 & 32 & 35 & 40 & 48 & 58 \\ 26 & 27 & 29 & 34 & 38 & 46 & 56 & 69 \\ 27 & 29 & 35 & 38 & 46 & 56 & 69 & 83 \end{bmatrix}$$

Quantization of Coefficient matrix

Quantization based on the properties of HVS

- Bitrate control: multiplying $W(k, l)$ with a constant
- Besides quantization DC components are coded differentially from the adjacent, previous DC component (simple spatial linear prediction in MPEG and JPEG)

Quantization of Coefficient matrix I.

DCT coefficients

$$\begin{bmatrix} 284 & 18 & -12 & 2 & 3 & -7 & 0 & 0 \\ 10 & 9 & 8 & 1 & -4 & 1 & 0 & 0 \\ -6 & 2 & 1 & 0 & 2 & 4 & 2 & -1 \\ 2 & 1 & -1 & 1 & -3 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 & 2 & 0 & 0 & -1 \\ 1 & -2 & 1 & 0 & 1 & -2 & 3 & 0 \\ 0 & 0 & 0 & -1 & 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

/

Quantizer matrix (W(k,l))

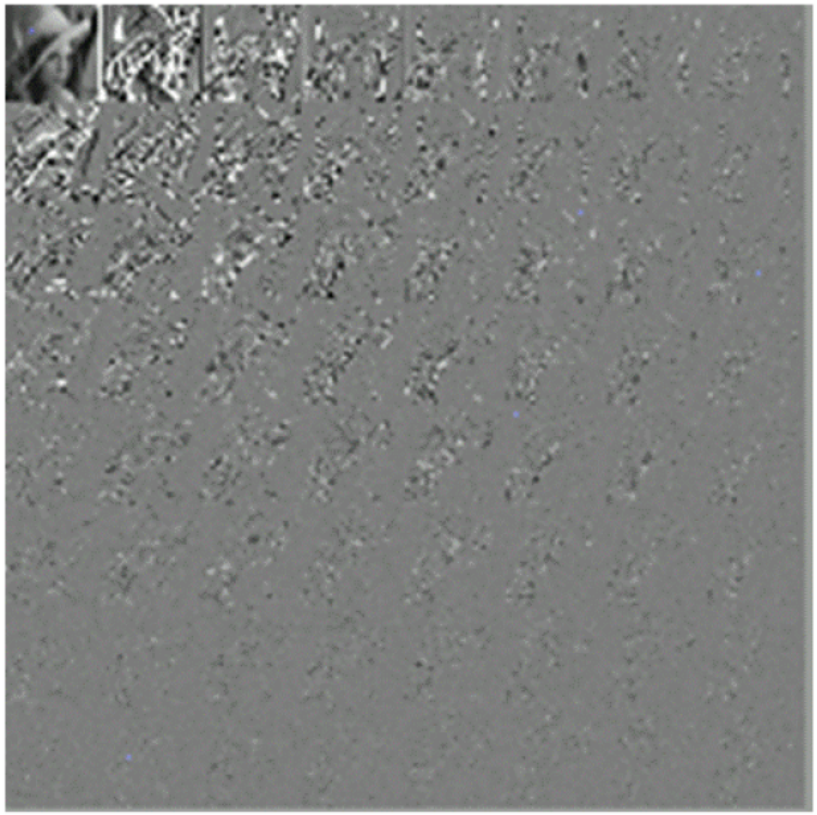
$$\begin{bmatrix} 8 & 16 & 19 & 22 & 26 & 27 & 29 & 34 \\ 16 & 16 & 22 & 24 & 27 & 29 & 34 & 37 \\ 19 & 22 & 26 & 27 & 29 & 34 & 34 & 38 \\ 22 & 22 & 26 & 27 & 29 & 34 & 37 & 40 \\ 22 & 26 & 27 & 29 & 32 & 35 & 40 & 48 \\ 26 & 27 & 29 & 32 & 35 & 40 & 48 & 58 \\ 26 & 27 & 29 & 34 & 38 & 46 & 56 & 69 \\ 27 & 29 & 35 & 38 & 46 & 56 & 69 & 83 \end{bmatrix}$$

=

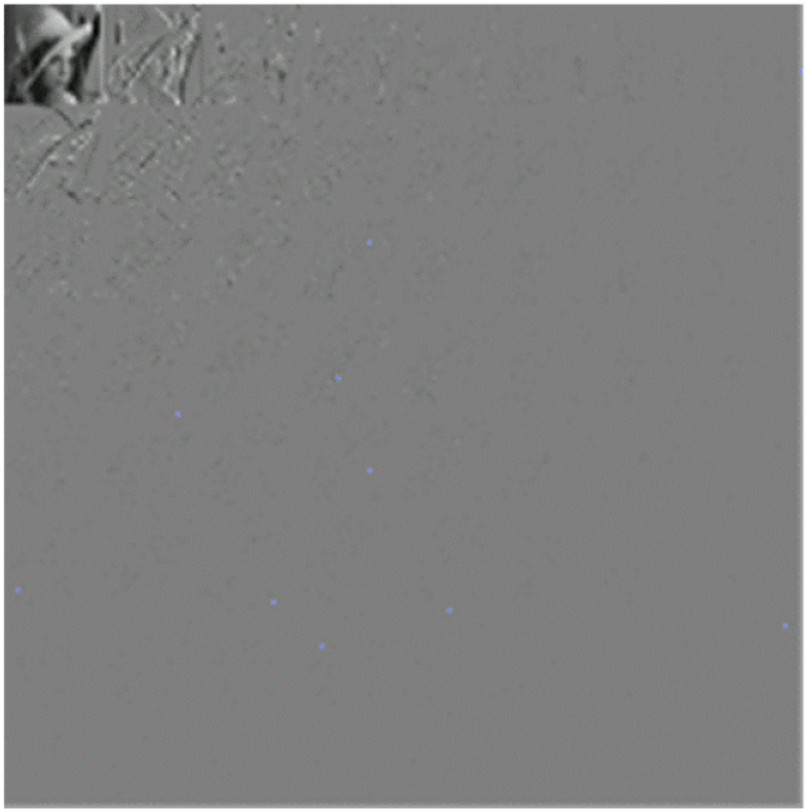
After quantization

$$\begin{bmatrix} 35 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Quantization of Coefficient matrix II.



DCT transform
Original DCT coefficients

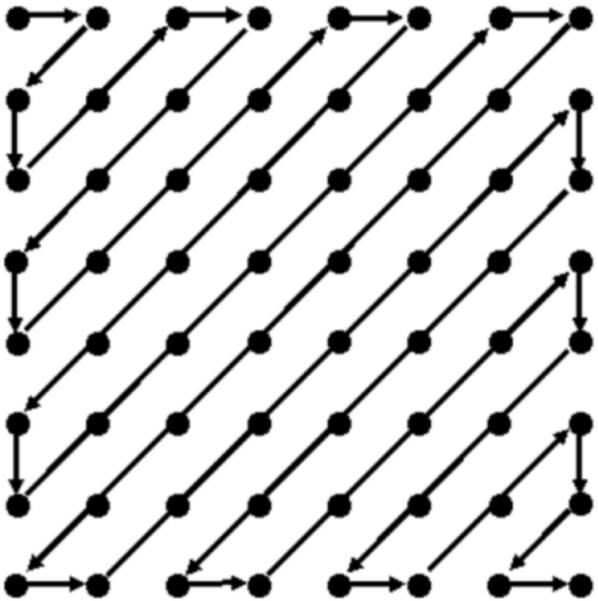


DCT transform
Quantized coefficients

Ordering quantized coefficients

- After quantization: most of AC coefficients become 0
- only non-zero coefficients are stored, transmitted (but their positions have to be known)
- Solution: zig-zag ordering:
  - aim: ordering coefficient in increasing spatial frequency
  - DC coeff. handled separately: lossless coding, differentially to the DC coef. of the previous block (DC coef.: average brightness of the block, changes slowly from block by block)
  - Zig-zag ordered AC coeff.: multiple zeros next to each other in the zig-zag vector
  - Only number of zeros is stored: Run-length coding (RLC)
  - Run-length coding: [Number of zeros; Next non-zero value] pairs are constructed from the zig-zag vector
  - No more non-zero values left: End of Block (EOB) sign

Zig-zag ordering



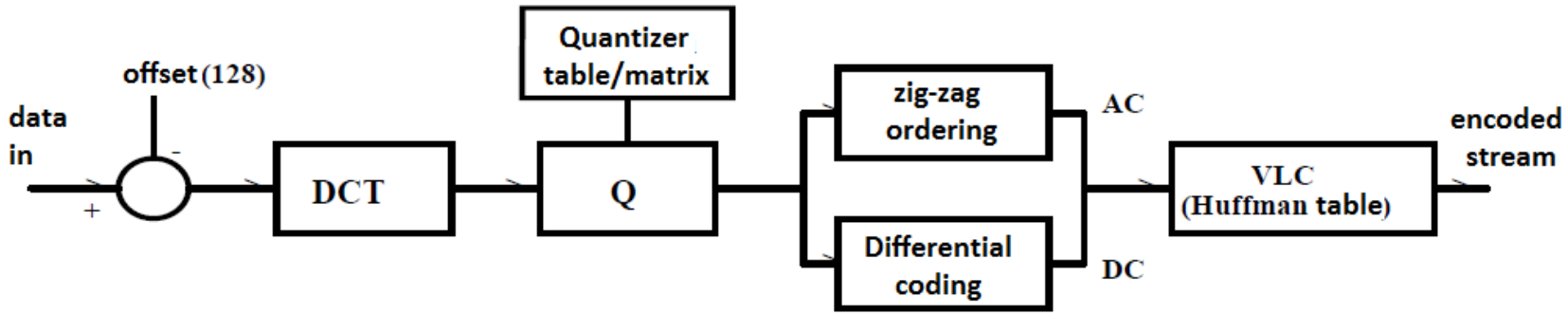
$$\mathbf{X} = \begin{bmatrix} 110 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -3 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Run-level pairs

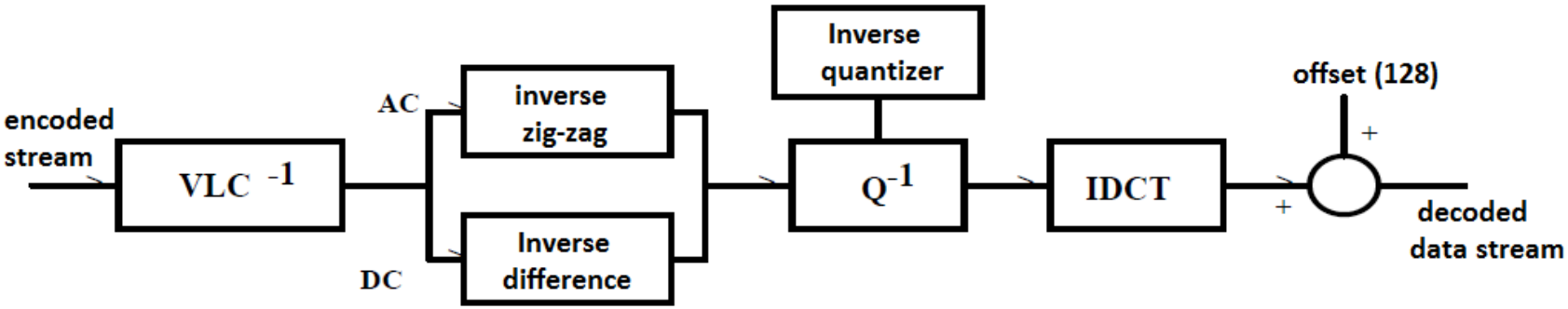
(0,– 2)
(0,1)
(1,– 3)
(2,1)
(0,– 1)
EOB

43

Block diagram of JPEG encoder/decoder



DCT based sequential JPEG encoder



DCT based sequential JPEG decoder

Steps of sequential JPEG coding

- Chroma subsampling (4:4:4, 4:2:2, 4:2:0)
- Block splitting: after subsampling, each channel are split into 8×8 non-overlapping
- DCT of the blocks
- Quantization of DCT blocks with user-defined quantizer matrices and quantizer scale
- Max. 4 types of quantizer matrices (e.g. different for each component)
- Components encoded separately

Steps of sequential JPEG encoding

- Previous quantized DC coefficient is used to predict the current quantized DC coefficient.
- Zig-zag ordering of AC components, constructing run-length pairs
- VLC coding of differential DC coefficients and RLC pairs (+end of block symbol)
- VLC coding: Huffman + Variable Length Integer coding

- Differential (predicted) DC values are classified based on their amplitude: Size ; Amplitude pairs are constructed
- Size symbol is encoded by Huffman coding , Amplitude symbol is VLI coded (variable length integer)
- Size symbol: number of bits of the following VLI value
- VLI:for positive amplitudes simple binary representation of amplitude without sign
- VLI for negative amplitudes: ones' complement

JPEG VLI table, example

Size	Amplitude	VLI
1	-1, 1	0, 1
2	-3, -2, 2, 3	00, 01, 10, 11
3	-7...-4, 4...7	000...011, 100...111
4	-15...-8 , 8...15	0000..0111, 1000..1111
5	-31...-16 , 16...31	.....
6	-63...-32 , 32...63	.....
7	-127...-64 , 64...127	.....
8	-255...-128 , 128...255	.....
9	-511...-256 , 256...511	.....
10	-1023..511, 511..1023	.....
11	-2047...-1024, 1024...2047	00000000000...01111111111 10000000000...11111111111

Steps of sequential JPEG encoding

- Maximum no. of Huffman tables: 2 DC and 2 AC tables
- Using default quant. matrixes, Huffman tables the encoder and decoder symmetrical in computational need
- Data required for decoding: stored in the bitstream header: e.g. non-default quantizer matrices, Huffman tables
- Computation need of decoding slightly lower than encoding
- Decoded image quality: depends on the quantizer matrix, quantizer scale and precision of DCT and IDCT transforms
- No actual bitrate control (only global)

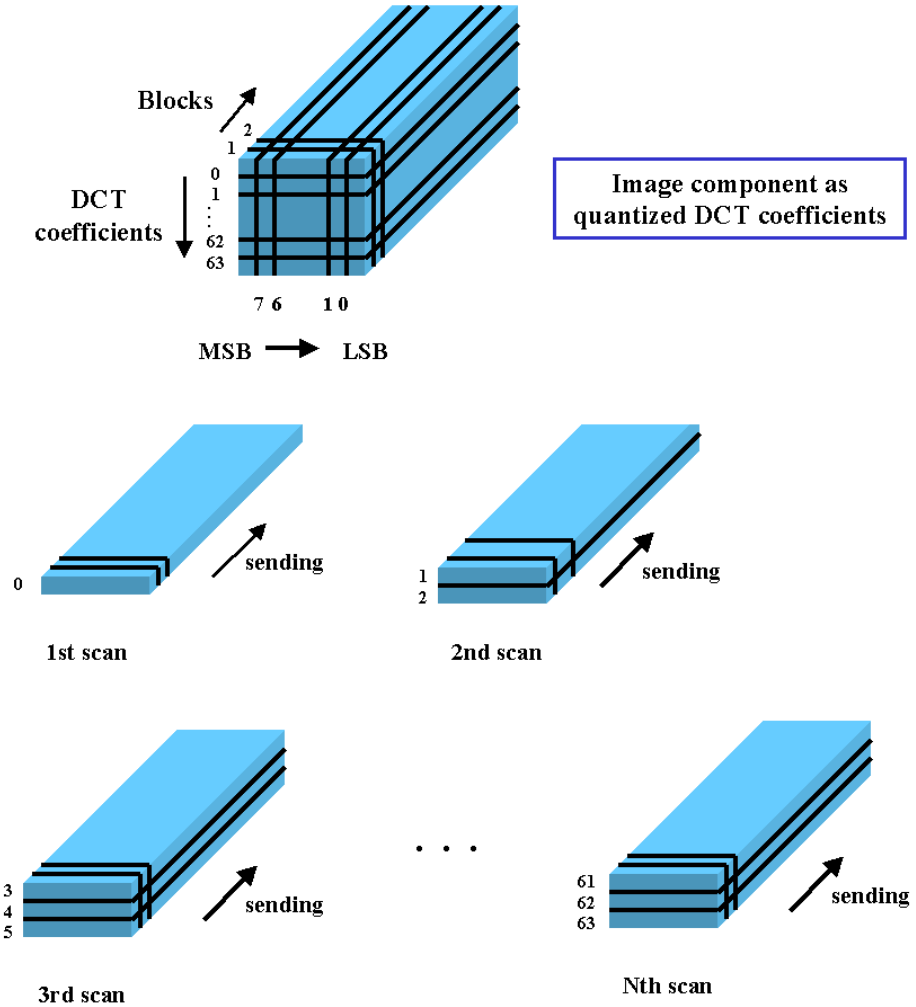
JPEG standard

- maximal image size 65535 x 65535,
- maximum number of components (e.g. RGB or YCbCr): 255
- color space not specified (color profiles may be embedded)
- coder and encoder: symmetric structure
- image quality - bitrate compromise
- 4 JPEG modes:
  - **DCT based sequential mode**,
  - **DCT based progressive mode**,
  - Lossless mode,
  - Hierarchical mode (DCT, DPCM).

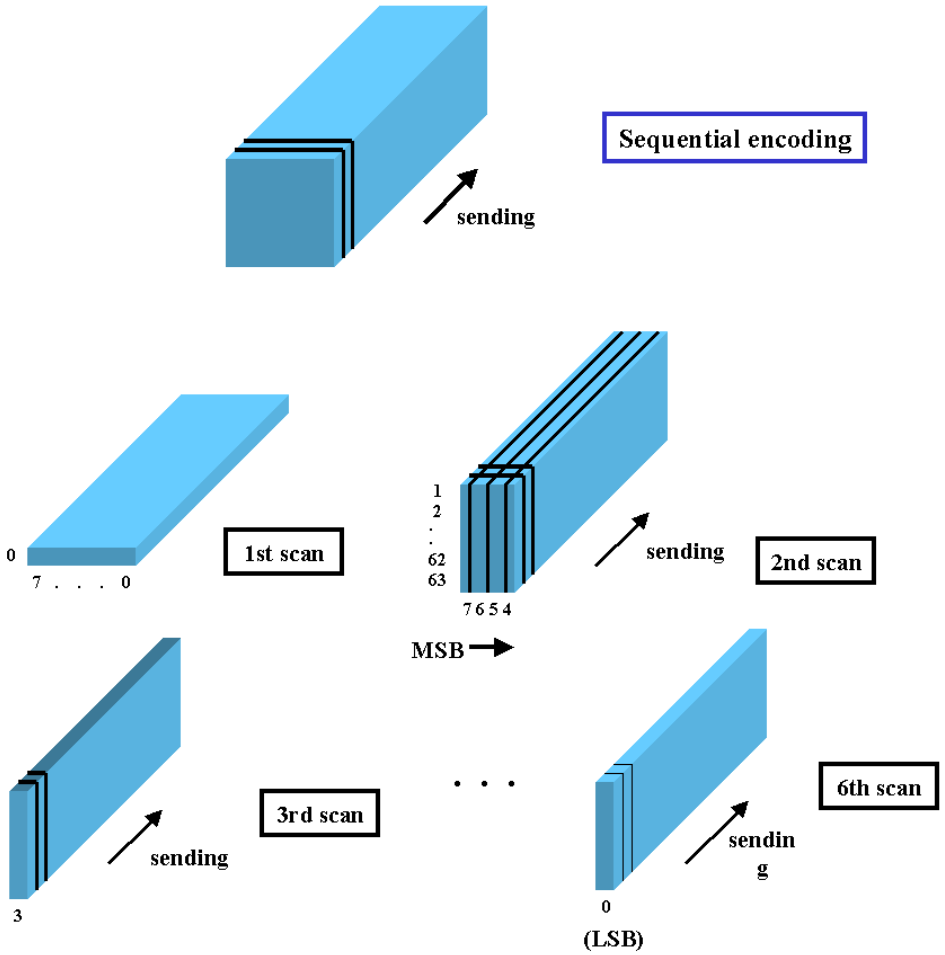
Progressive JPEG encoding

- Goal: display low quality image and successively improve.
- Image is transmitted in multiple steps (from rougher version to detailed version)
- Base layer and correction layers
- Two ways to successively improve image:
  - Spectral selection: Send DC component and first few AC coefficients first, then gradually some more ACs.
  - Successive approximation: send DCT coefficients MSB (most significant bit) to LSB (least significant bit). (Effectively, it is sending quantized DCT coefficients frist, and then the difference between the quantized and the non-quantized coefficients with finer quantization stepsize.)

Progressive JPEG encoding



Progressive JPEG encoding



Progressive encoding : successive approximation

JPEG artifacts - ringing/blocking

MPEG standards

- very strictly defined bitstream, and decoder function, but standards does not define how encoding is to be performed (although a reference implementation is provided in ISO/IEC-11172-5)
- MPEG-1 (ISO 11172 /1993/): designed to compress VHS-quality raw digital video (SIF quality) and CD audio down to 1.5 Mbit/s (26:1 and 6:1 compression ratios respectively)
- MPEG-2 (ISO 13818 /1994-95/): Designed for Digital Video Broadcasting, (DVB), 2-8 Mb/s distribution quality; Studio technologies, 18-50 Mb/s contribution quality; Encoding Digital Versatile Disc (DVD), 3-7 Mb/s (Variable Bitrate!) (better than PAL quality); Extended version of MPEG-1, encoding principles are common.

MPEG standards

- MPEG-4 (ISO/IEC 14496): Initially, MPEG-4 was aimed primarily at low bit-rate video communications; however, its scope as a multimedia coding standard was later expanded. *Two most important parts*:
  - MPEG-4 Part 2: standard video encoding: H.263, implementation: DivX, and XVid
  - MPEG-4 Part 10: H.264/AVC

MPEG-1 encoding

- DCT based motion compensated hybrid encoding type
- MPEG system is asymmetric: computational complexity of encoding and decoding sides differ
- Basis property of MPEG video is its layered structure
- Layer structure: 6 nested layers, where lower layers can not be decoded without decoding upper layers

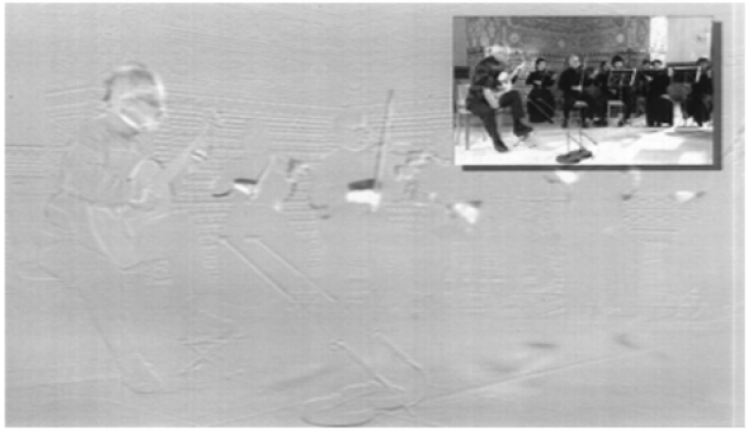
Motion prediction, motion compensated prediction

- subsequent frames of videos are strongly correlated
- except for scene changes and rapid motion (e.g. sport) the content does not change much from frame to frame
- thus: temporal prediction is feasible
- taking movements into consideration: basis of prediction is the corresponding part of previous frame: motion prediction
- efficiency of decreasing redundancy: depends on accuracy of motion prediction
- result of motion estimation: motion vectors: complex and time-consuming task to find
- strategy: definition of cost functions (e.g. mean squared error, mean absolute error) and minimizing it with some compromises
- in case of offline encoding: more accurate motion estimation algorithms can be used

Difference between frames (in case of slowly moving content)

frame N.

frame (N+1).



frame N.- frame (N+1).

Difference between frames (change of scene)

frame N.

frame (N+1).

frame (N+1).

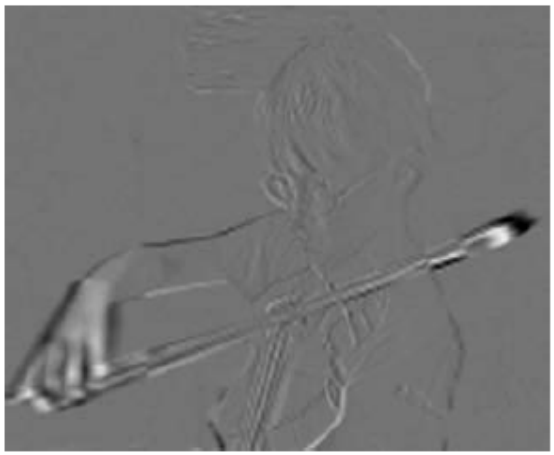
-

frame N.

Apparent motion of pixels: optical flow

Frame 1

Frame 2



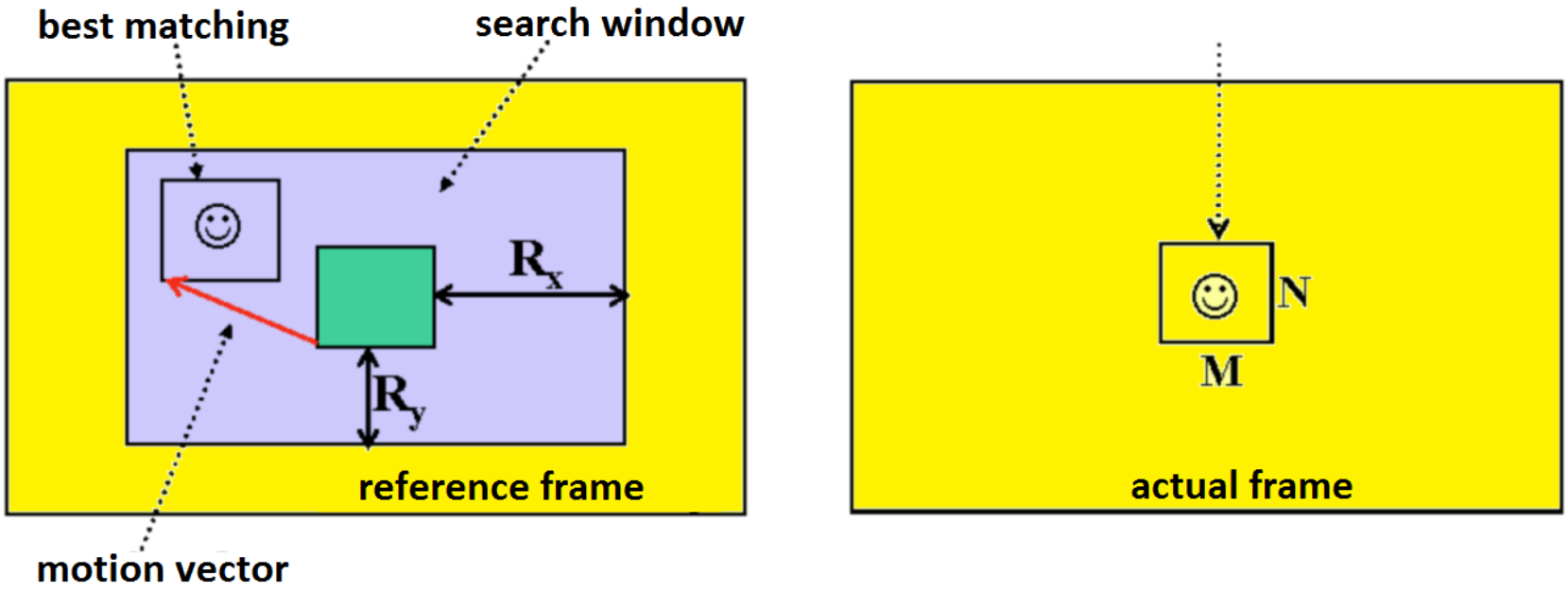
Difference

Optical flow

Block-based motion estimation I.

- fitting to the $N \times N$ block-based DCT encoding block based motion prediction and compensation is feasible
- aim: assigning an $N \times N$ sized reference block to each block of the actual frame
- difference of the blocks in pixels: motion vector of the given block
- choosing the optimal reference/predictor block: minimization a properly chosen cost function (discrepancy between the difference between actual and predictor block) in a search window around the actual block

Block-based motion estimation II.



Block-based motion estimation III.

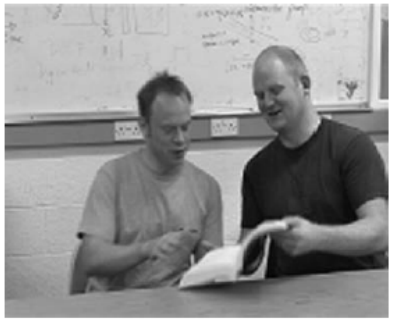
Block-matching algorithms

- Full search: accurate, but huge computational need
- Pixel-recursive search: searching starts from an initial estimation (e.g. motion vector in previous frame/adjacent blocks)
- Logarithmic search: search is performed on subsampled versions of the frame, with increasing resolution (first on a lower resolution, then this is estimation is refined recursively with a higher resolution

Block-based motion estimation IV.

- Motion estimation is performed on the Y component, the obtained motion vectors are applied for Cb and Cr blocks as well
- Unit of motion estimation/compensation: **Macroblock (MB)**
- For each non-intra block at least one motion vector can be assigned, motion within the MB can not be estimated (present as prediction error/ residual)
- Motion vectors are usually coded differentially, compared to the previous MB's motion vector in the block raster sequence (efficient, because neighbouring blocks usually move into the same direction)
- Determination of motion vectors: 1, 1/4 or 1/8 pixel (pels) accuracy, depending on the implementation of encoder: half pel motion vectors are determined by motion estimation performed on an interpolated (increased resolution) version of the frame.

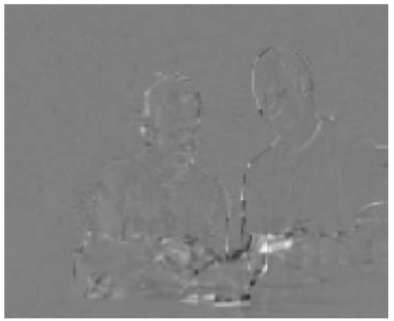
Motion estimation example



Frame 1

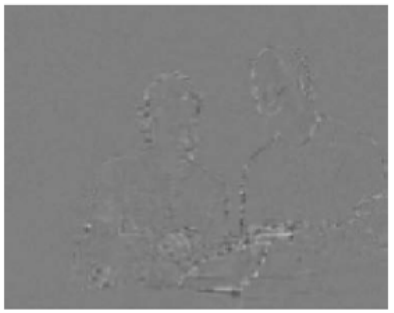
Residual : 16 × 16 block size

Frame 2



Residual : 8 × 8 block size

Residual : no motion compensation



Residual : 4 × 4 block size

MPEG picture types

I (intra) frame coded independently, compressed using only intraframe coding, including all data, required for decoding

P (predictive) frame stores only the difference in image from the frame (either an I-frame or P-frame) immediately preceding it (this reference frame is also called the anchor frame)

B (bidirectional) frame are similar to P-frames, except they can make predictions using both the previous and future I or P frames (i.e. two anchor frames). B frames can not be reference frames for prediction.

Choosing MPEG picture types

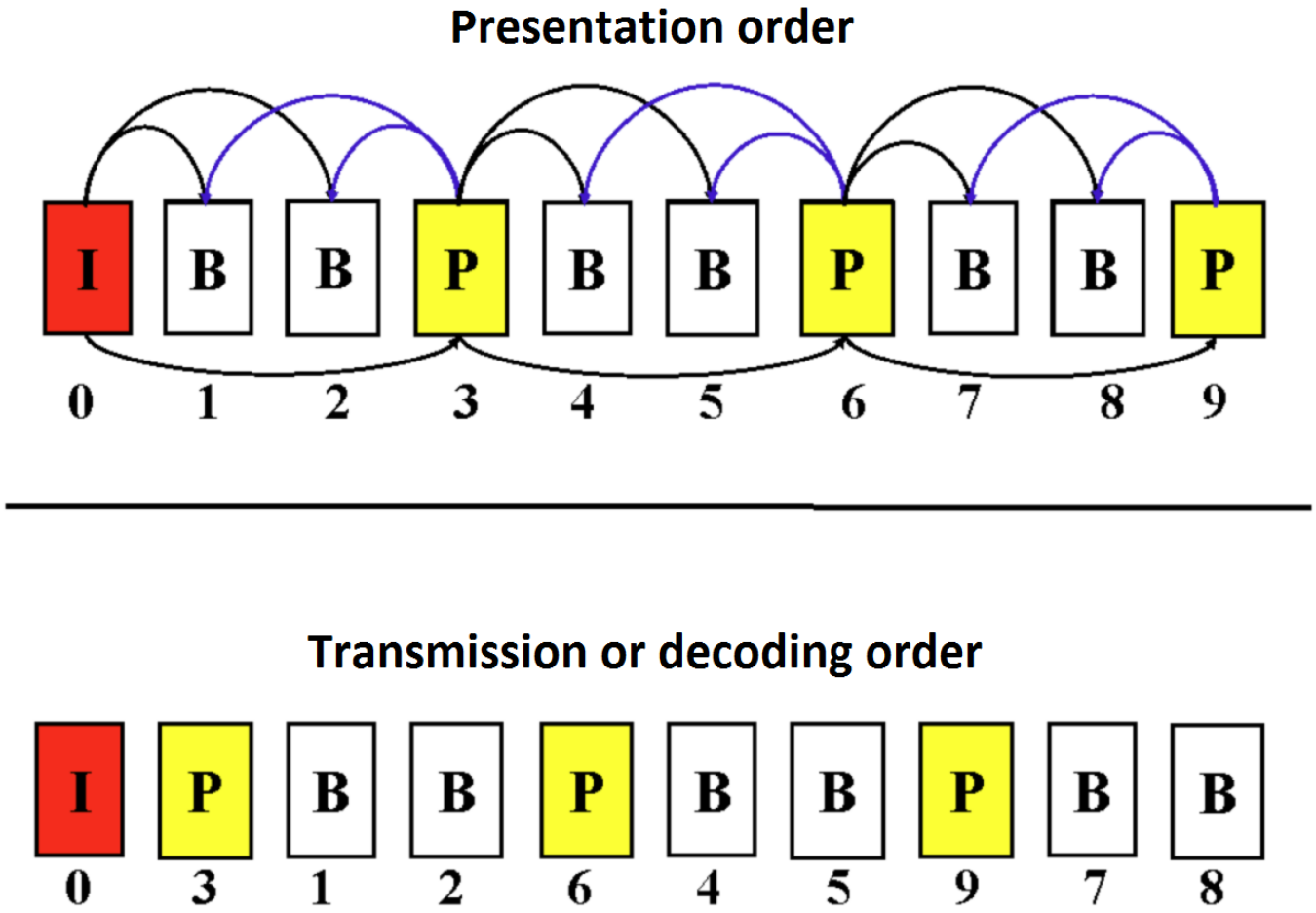
- Global compression rate is between 10:1 and 50:1
- IPB structure is not defined by standard, employing B frames is not necessary: a sequence may consist of merely I, or I and P frames
- Increasing the number of B frames increases the computational cost, required memory and latency at the encoder side
- High compression rate: high number of B, low number of I frames
- Low latency, low hardware cost: no B frames
- Fast random access (editability, start of decoding): high number of I, low number of P and B frames
- Frequently used pic. order/GOP structure for video broadcasting: IBBPBBPBBPBB (Long GOP - EU)
- Frequently used pic. order for studios: III.... (only I frames)

MB types

- Macroblock types
  - in I frames: all macroblocks coded independently: intra MB
  - in P frames: macroblocks may be I and P MB
  - in B frames: macroblocks may be encoded intra (I), from a future reference, from a past reference or both (for this latter case the anchor macroblock is the average of past and future reference)
- in case of P and B macroblocks decision has to be made
  - coded / non coded macroblock (all values = 0)
  - motion compensated/not motion compensated
- Optimal decision: check all possibilities: computational expensive
- Instead: special decision strategies

MPEG picture orders

Due to the presence of B frames two picture orders are defined, since anchor frames of B pictures have to be decoded before the actual B frame: **decoding order** and **presentation order**



MPEG layer structure

Sequence layer : any self-contained bitstream, for example a coded movie or advertisement

Group of Picturs (GOP) : sequence of frames, consisting at least one intra-coded frame. Unit of random access

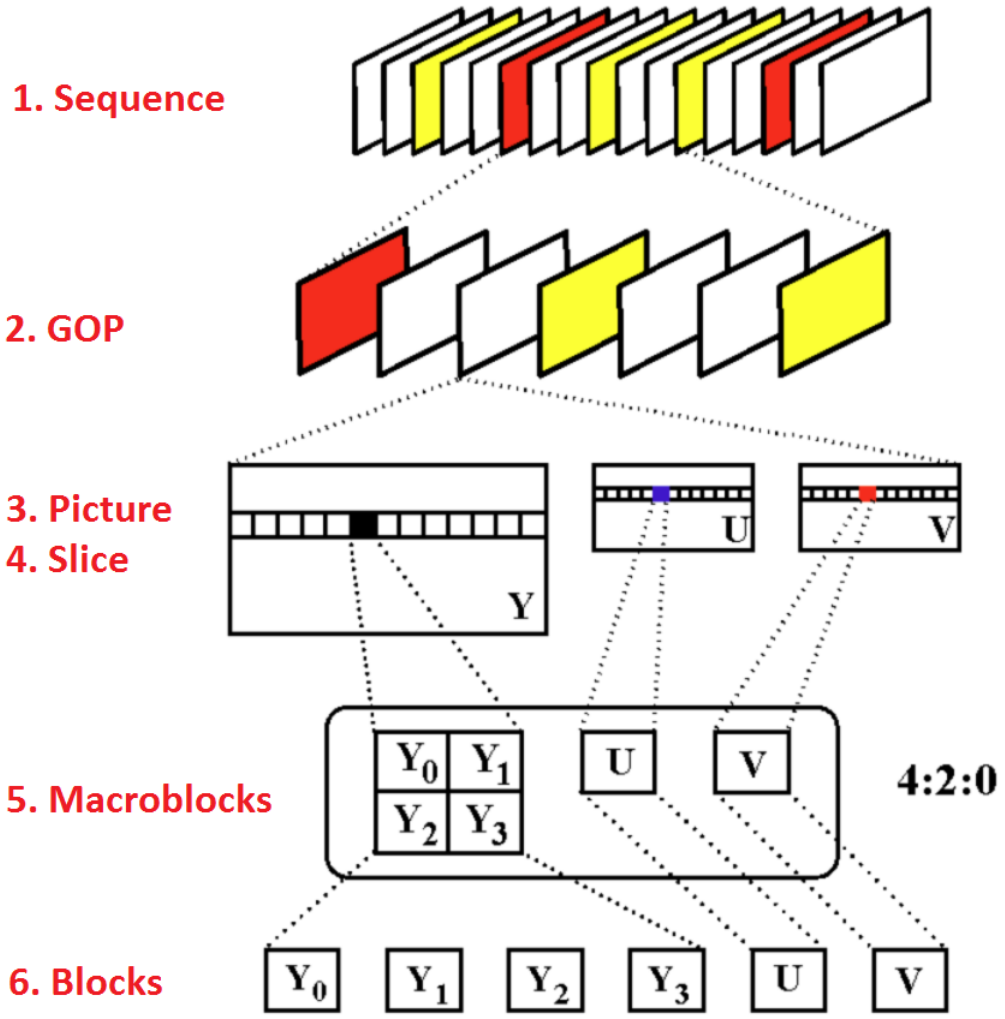
Picture layer Encoded data of one frame

Slice layer continuous sequence of raster ordered macroblocks, most often on a row basis in typical video applications. Unit of re-synchronization (lowest layer, where decoder is able to „revive” in case of bit-error)

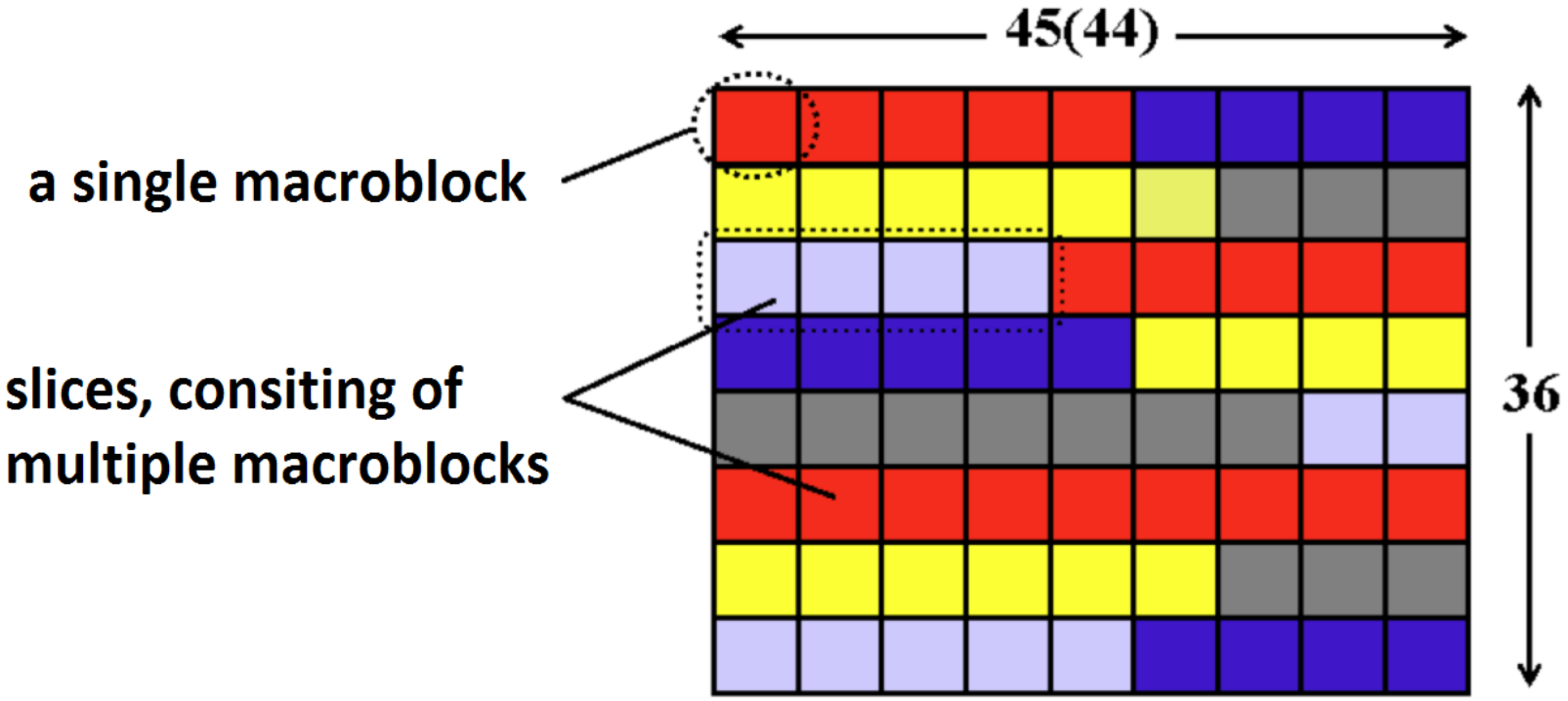
Macroblock layer 16×16 and 8×8 sized blocks of luma (Y) and chroma (Cb, Cr) components respectively, unit of motion compensation (for a sampling structure of 4:2:0 1 MB = 6 blocks)

Block layer 8×8 size block within a MB, unit of DCT encoding

MPEG layer structure



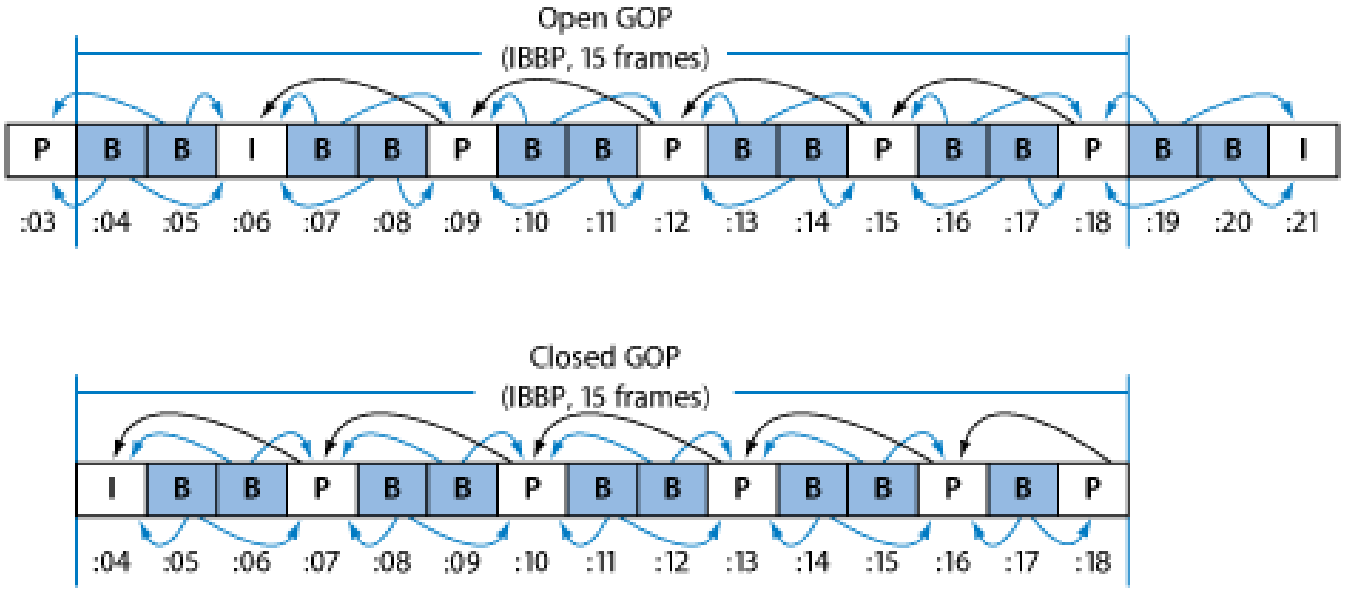
MPEG picture structure



MPEG-1/2 EU SD picture: (45) 44 x 36 macroblocks

Group of Pictures (GOP) types

- Closed GOP: first B frame does not have an anchor frame in the previous GOP: feasible in case of e.g. a scene change in the beginning of the GOP
- Open GOP: first B frame in the GOP has an anchor frame in the previous GOP (i.e. last I or P frame in the previous GOP serves as a past reference for the first B frame in the actual GOP)



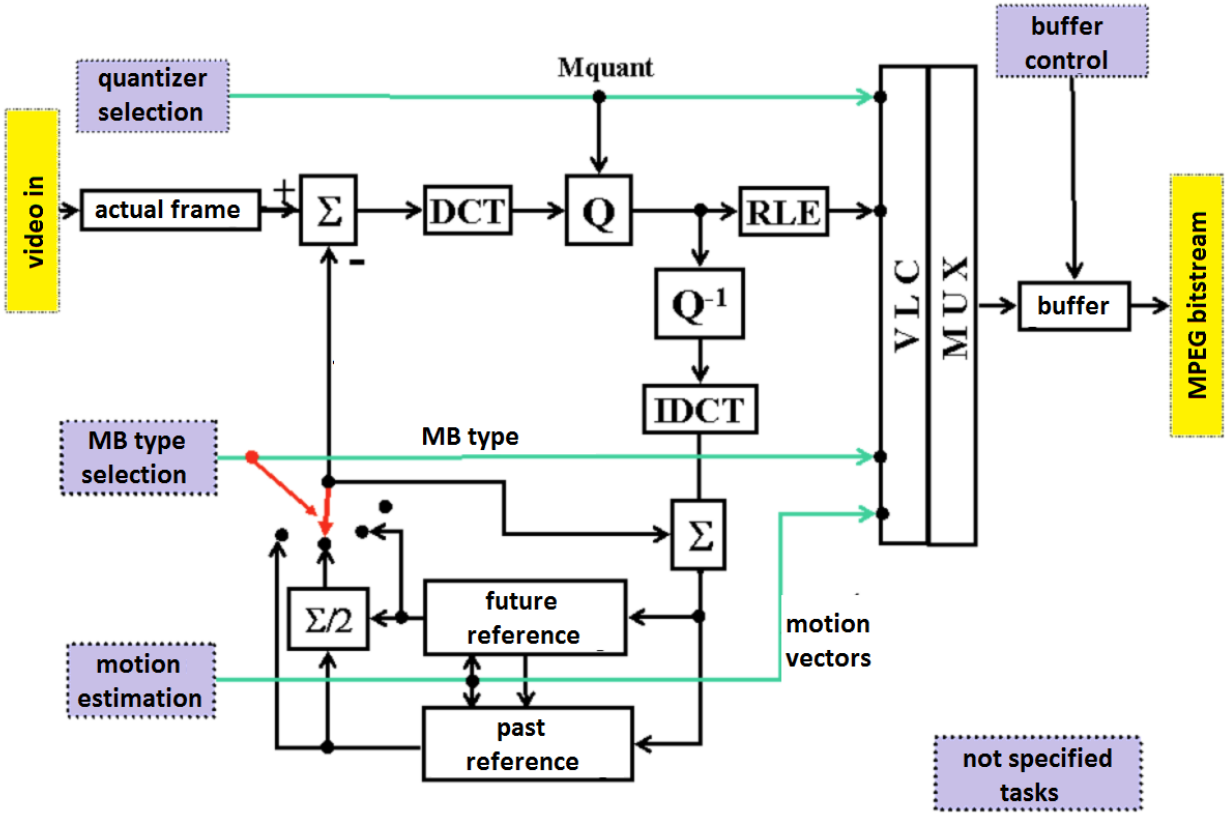
Steps of MPEG encoding

- DCT transform coding of the MB (intra MB type) or the residual error MB of motion prediction (P or B)
- Quantization of the DCT coefficients: quantization steps are obtained as the quantization matrix and quantizer scale
- Quantization matrix: perceptually based for I pictures (same as for JPEG), and constant for P and B frames (since residuals are usually noise-like, high-pass filtered)
- Quantizer scale factor: default set in slice header, but may be redefined for each macroblock (realizing bitrate control)

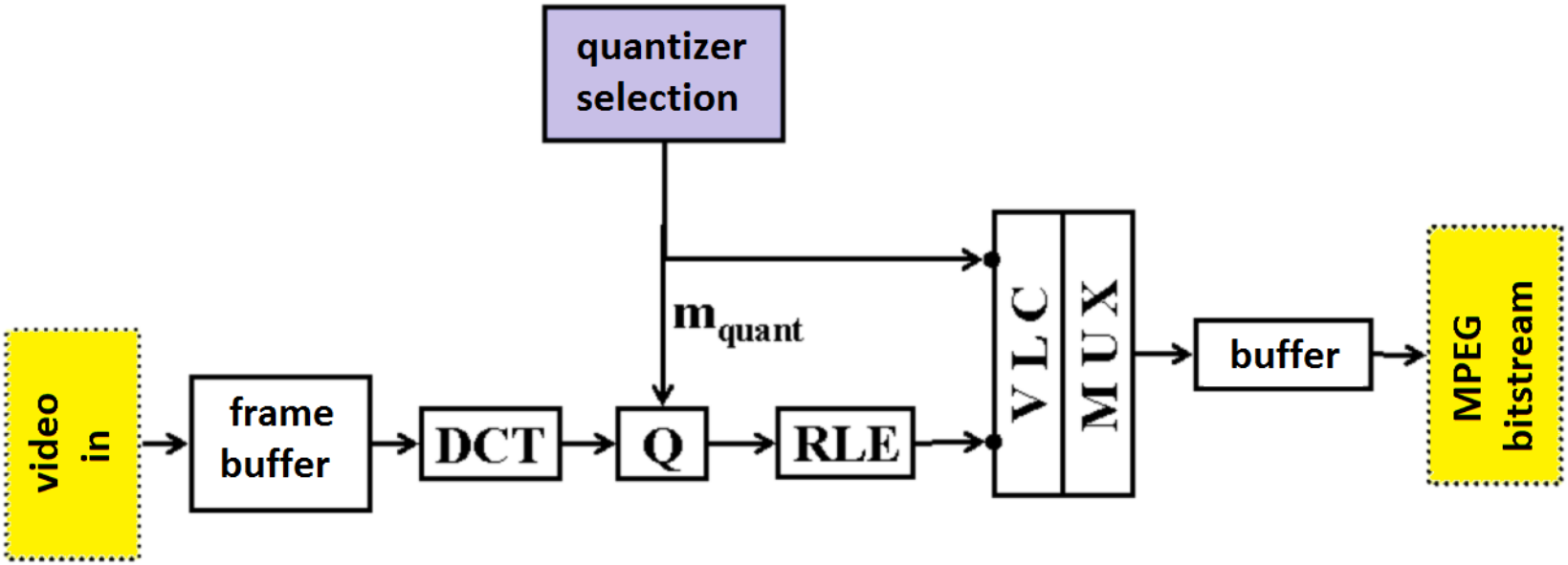
MPEG video encoder

- 2 reference buffers: one for future and one for storing past reference/frames
- for I frames: after displayed/presented I frames pushed into the reference buffer
- for P frames: inverse prediction, presentation and storing in the reference buffer

- for B frames: only presentation and discarding



MPEG I-only (studio) encoder



MPEG rate control I.

Constant bitrate (CBR)

- CBR: rate at which a codec's output data should be consumed is constant.
- Number of bits at the output of the decoder over unit time varies, because
  - Compression rate for I,P,B types is different (B pics are represented on lower number of bits than I frames)
  - Content varies over a single image, complex blocks require higher number of bits at the same image quality
  - VLC encoding produces variable number of bits over unit time
- Most transmission channel require a constant bit-rate data source
- Solution: buffer at the output of the encoder; from the buffer the channel reads out the bits in a constant rate

Steps of bit-rate control

- the number of allocated bits per GOP is given (global quality is predefined), aim: defining the number of bits per I, P, B pictures
- definition of default quantizer scales based on the complexity of the actual macroblocks (local quality control)
- modifying the quantizer scale based on the buffer state

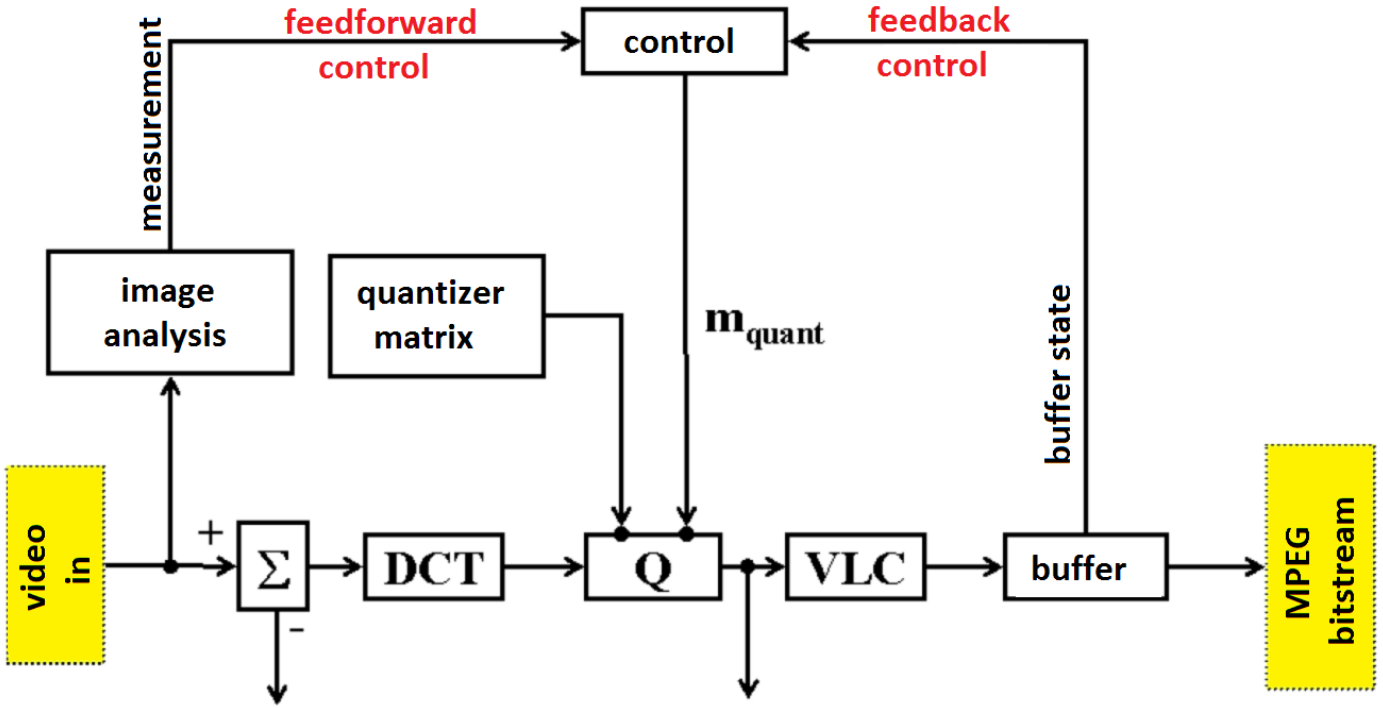
MPEG bit-rate control II.

Variable bitrate (VBR)

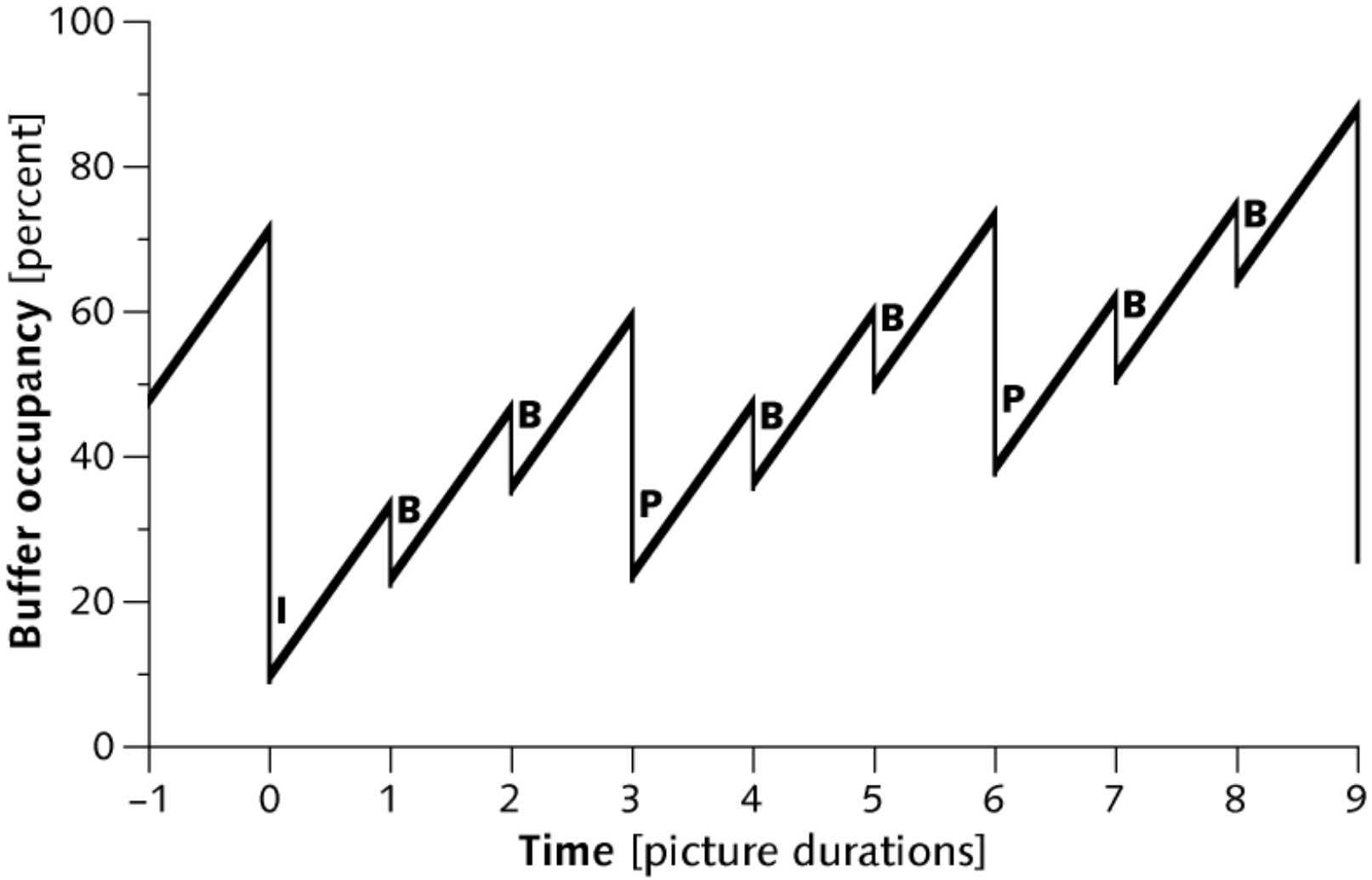
- Two basic types:
  - Open loop control: no output buffer, or output buffer can be considered infinitely large (e.g. DVD player, memory, etc.)
  - Feedback control: output bitrate is smoothed using output buffer control before sending data to the transmission channel (bitrate remains under a pre-defined maximum bitrate)
- Types of open loop VBR:
  - Constant scale factor encoding: does not ensure optimal quality
  - Constant quality encoding: scale factor is adapted to the perceived image quality

MPEG bit-rate control

Possibilites: Open loop/feedforward control with global (frame level) and local (MB level) image analysis, or feedback control by applying a video buffering verifier model



MPEG Video Buffering Verifier (VBV)



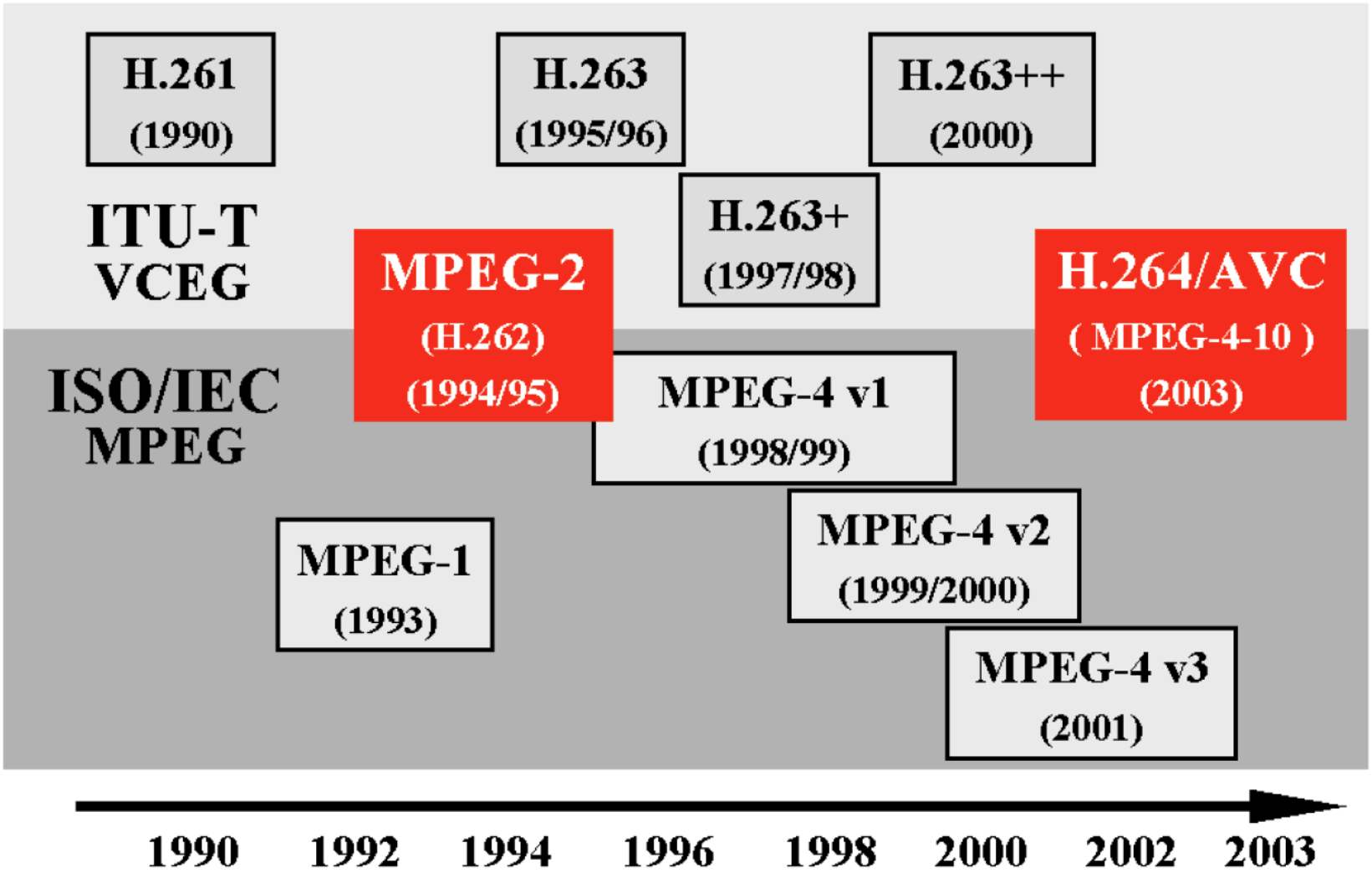
Basic parameters of MPEG-1 encoding

- chroma subsampling structure: 4:2:0
- frame rates: 24, 25,29.97, 30 frame/sec
- maximum image size: 4095x4095
- only progressive scanning is supported
- user defined quantization matrices
- I,P,B frame
- achievable compression factor :1:25 - 1:30

Limitations of MPEG-1

- MPEG-1 Layer 3 (MP3) audio encoding: max two channels
- only progressive scan
- encoding is optimal for one parameter group: *Constrained Parameter Bitstream (CPB)*
  - 352x288 @ 25fps, vagy 352x240 @ 30fps
  - optimal output bitrate: 1-1.5 Mb/s
  - general MPEG-1 parameters $\equiv$ Constrained Parameters Bitstream (CPB)
- In system level only Program Stream (PS) is defined

History of video compression 1990-2003



MPEG standards (again)

- very strictly defined bitstream, and decoder function, but standards does not define how encoding is to be performed (although a reference implementation is provided in ISO/IEC-11172-5)
- MPEG-1 (ISO 11172 /1993/): designed to compress VHS-quality raw digital video (SIF quality) and CD audio down to 1.5 Mbit/s (26:1 and 6:1 compression ratios respectively)
- MPEG-2 (ISO 13818 /1994-95/): Designed for Digital Video Broadcasting, (DVB), 2-8 Mb/s distribution quality; Studio technologies, 18-50 Mb/s contribution quality; Encoding Digital Versatile Disc (DVD), 3-7 Mb/s (Variable Bitrate!) (better than PAL quality); Extended version of MPEG-1, encoding principles are common.

Differences between MPEG-1 and MPEG-2

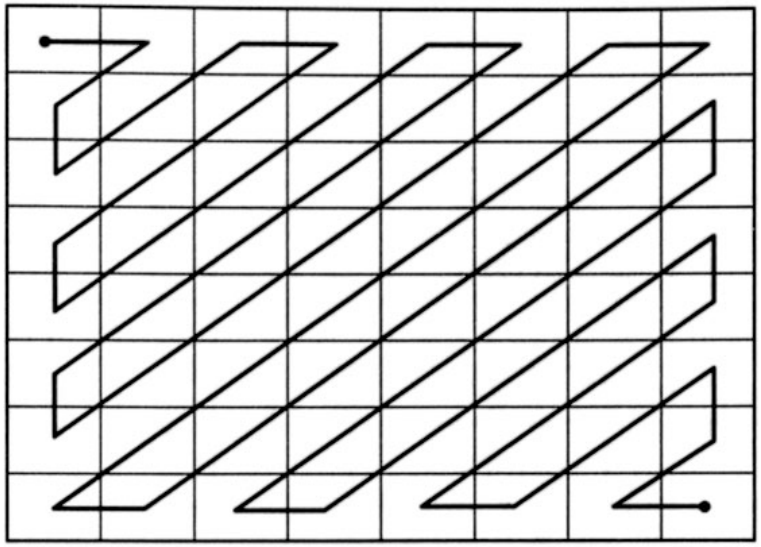
- Opposed to MPEG-1, MPEG-2 was designed for the wide range of applications (MPEG-1: SIF only)
- Based on MPEG-1, with upward compatibility
- introduction of coding tools for interlaced format
  - alternate scan for interlaced pictures (instead of zig-zag scanning)
  - frame-field based adaptive DCT
  - field based motion compensation methods
- bitrates higher than MPEG-1, larger image sizes are supported
- Only half-pel motion estimation accuracy
- 4:2:2, 4:4:4 structures are supported (besides default 4:2:0, only option for MPEG-1)
- Profile-level structure
- Scalable coding is supported (SNR, spatial scalability, etc.)

MPEG-2 interlaced motion compensation

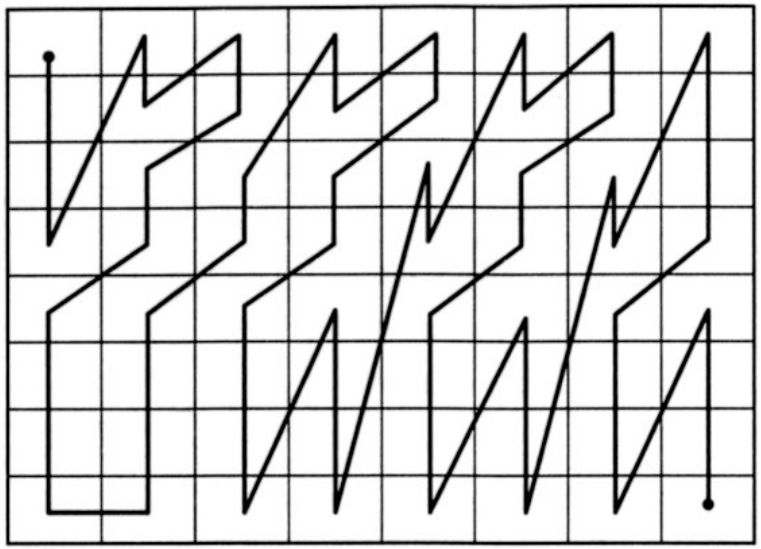
- From frame to frame (same as for MPEG-1)
- Field to field
- Field to frame
- 16x8 motion compensation
- Dual Prime prediction

Motion Compensation Mode	Use in Field Pictures?	Use in Frame Pictures?
Frame Prediction for Frame Pictures	No	Yes
Field Prediction for Field Pictures	Yes	No
Field Prediction for Frame Pictures	No	Yes
Dual-Prime for P-Pictures	Yes	Yes
16 × 8 MC for Field-Pictures	Yes	No

MPEG-2 alternate scanning



Zigzag scan



Alternate scan

– scanning/ordering redefined, suitable for interlaced scanning

MPEG-2 scalable video

Basic idea: base-layer+enhancement-layer, base layer: basic video quality, decodable by lower capabilities

- SNR scalability
  - DCT coefficients are transmitted with increasing accuracy (quantization is finer in enhancement layer)
- Spatial scalability
  - image is transmitted with increasing resolution (e.g. base-layer is only SIF, etc.)
- Temporal scalability
  - video is transmitted with increasing frame rate
- Hybrid scalability
  - arbitrary combination of the forementioned
- Data partitioning
  - partitioning of DCT coefficient matrix

MPEG-2 Profile-Level structure

- Different classes and levels of parameters are defined in order to classify the capabilities and capacities of the decoder required for decoding an MPEG bitstream
- **Profile**: chroma sampling formats, supported frame types and other tools of encoding are classified
- **Level**: video resolution, frame rate, bitrate and other sequence parameters are classified
- Main area of application by Levels
  - Low: videoconference
  - Main: SDTV
  - 422P: studio
  - High: HDTV

MPEG-2 Profile-Level properties

A given Profile/level decoder is capable of decoding video streams on its left in the table above. For professional and consumer applications the important areas are denoted by yellow background

Levels

	Low	Main	High-1440	High	422P
Horizontal size	352	720	1440	1920	720
Vertical size	288	576	1152	1152	608
Frame rate (Hz)	30	30	60	60	30
Bitrate (Mbit/s)	4	15	60	100	50

Profiles

	Simple	Main	SNR	Spatial	High	422P
YUV	4:2:0	4:2:0	4:2:0	4:2:0	4:2:0 4:2:2	4:2:0 4:2:2
Frame/field types	I, P	I, P, B	I, P, B	I, P, B	I, P, B	I, P, B
Scalability	no	no	yes	yes	yes	yes

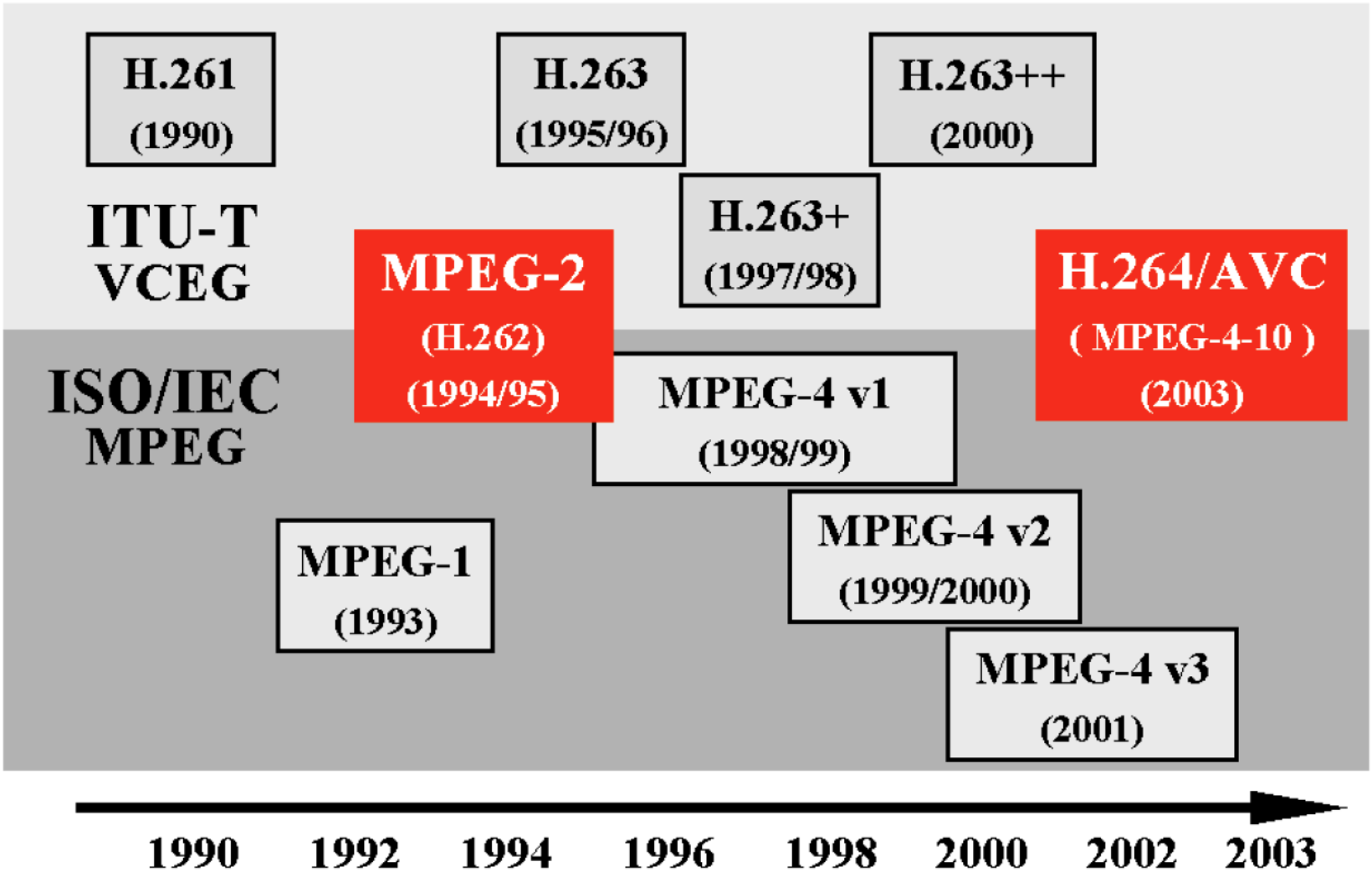
MPEG-2 Profile-Level properties

Profile / *@Level*	MPEG-1 **CPB**	Simple (no B pictures)	Main (MP)	4:2:2 (422P)
High (HL)			1920×1152 60 Hz 80 Mb/s	1920×1088 60 Hz 300 Mb/s
High-1440 (H14)			1440×1152 60 Hz 47 Mb/s	
Main (ML)		720×576 30 Hz 15 Mb/s	720×576 30 Hz 15 Mb/s	720×608 30 Hz 50 Mb/s
Low (LL)			352×288 30 Hz 4 Mb/s	
MPEG-1 CPB †*max 99 Kpx*	768×576† 30 Hz 1.856 Mb/s			

MPEG-2 Profile-Level properties

Profile@Level	Image columns (N_C)	Image rows (N_R)	Frame rate, Hz	Luma rate [samples/s]	Bit rate [Mb/s]	VBV size [KBytes]
422P@HL	1920	1088	60	62,668,800	300	5,760
MP@HL	1920	1088	60	62,668,800	80	1,194
MP@H-14	1440	1088	60	47,001,600	60	896
422P@ML	720	608	60	11,059,200	50	1,152
MP@ML	720	576	30	10,368,000	15	224
MP@LL	352	288	30	3,041,280	4	58

History of video compression (1990-2003)



MPEG standards (again)

- MPEG-4 (ISO/IEC 14496): Initially, MPEG-4 was aimed primarily at low bit-rate video communications; however, its scope as a multimedia coding standard was later expanded. *Two most important parts*:
  - MPEG-4 Part 2: standard video encoding: H.263, implementation: DivX, and XVid
  - MPEG-4 Part 10: H.264/AVC

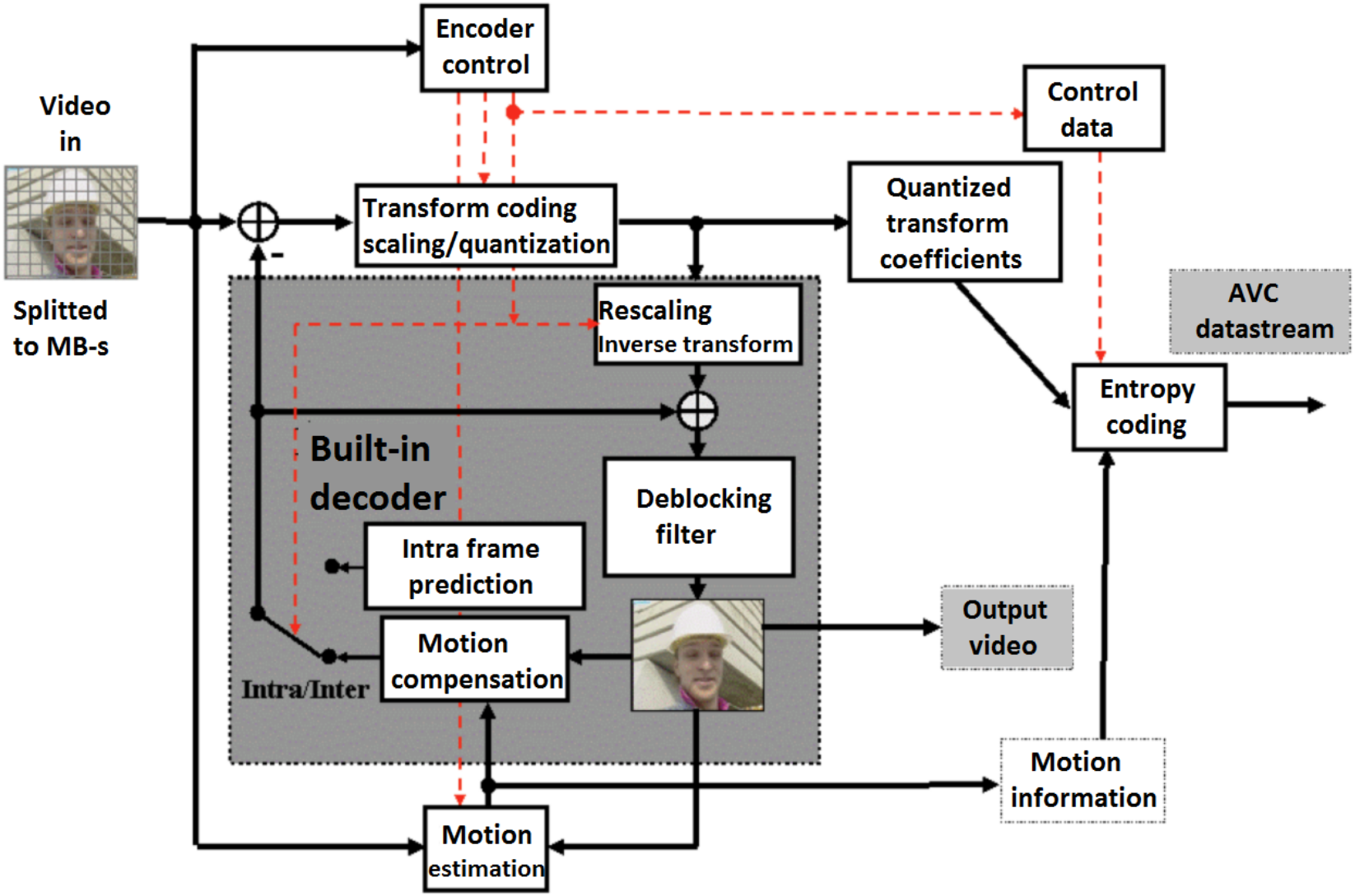
MPEG-4 (Part 10) H.264/AVC

- Co-developed by the ITU-T Video Coding Experts Group (VCEG) and ISO Moving Picture Expert Group (MPEG) (Together: Joint Video Team): therefore known both as ITU-T H.264, and ISO/IEC MPEG-4 AVC
  - better video quality than MPEG-2
  - increased compression ratio
  - improved prediction(intra- and inter-)
  - improved motion estimation accuracy
  - improved entropy coding
- Profile-Level structure
- Wide range of applications (videocalls, internet-media, DVB), first introduced for "entertainment" purposes' (SD and lower resolutions)
- Fidelity Range Extensions (FREXT) extended the standard for professional applications (studio, digital cinema, etc)

FRExt - Fidelity Range Extensions

- New profile-requirements:
  - Higher bit depth than 8 bits
  - Supporting 4:2:2 and 4:4:4 structures
  - Video-keying and transparency (alpha channel) support for studio applications
  - High bit-rate support
  - Possibility of lossless compression (linear PCM or entropy coding without transform coding)
  - Avoiding numeric errors due to color space conversion (YCgCo color space)
  - RGB component support (opposed to YCbCr)
- 8×8-sized transform coding
- Nowadays High Profile of FRExt is more important than original Main profile of AVC, because of the increased compression ratio at almost the same computational complexity and hardware requirement

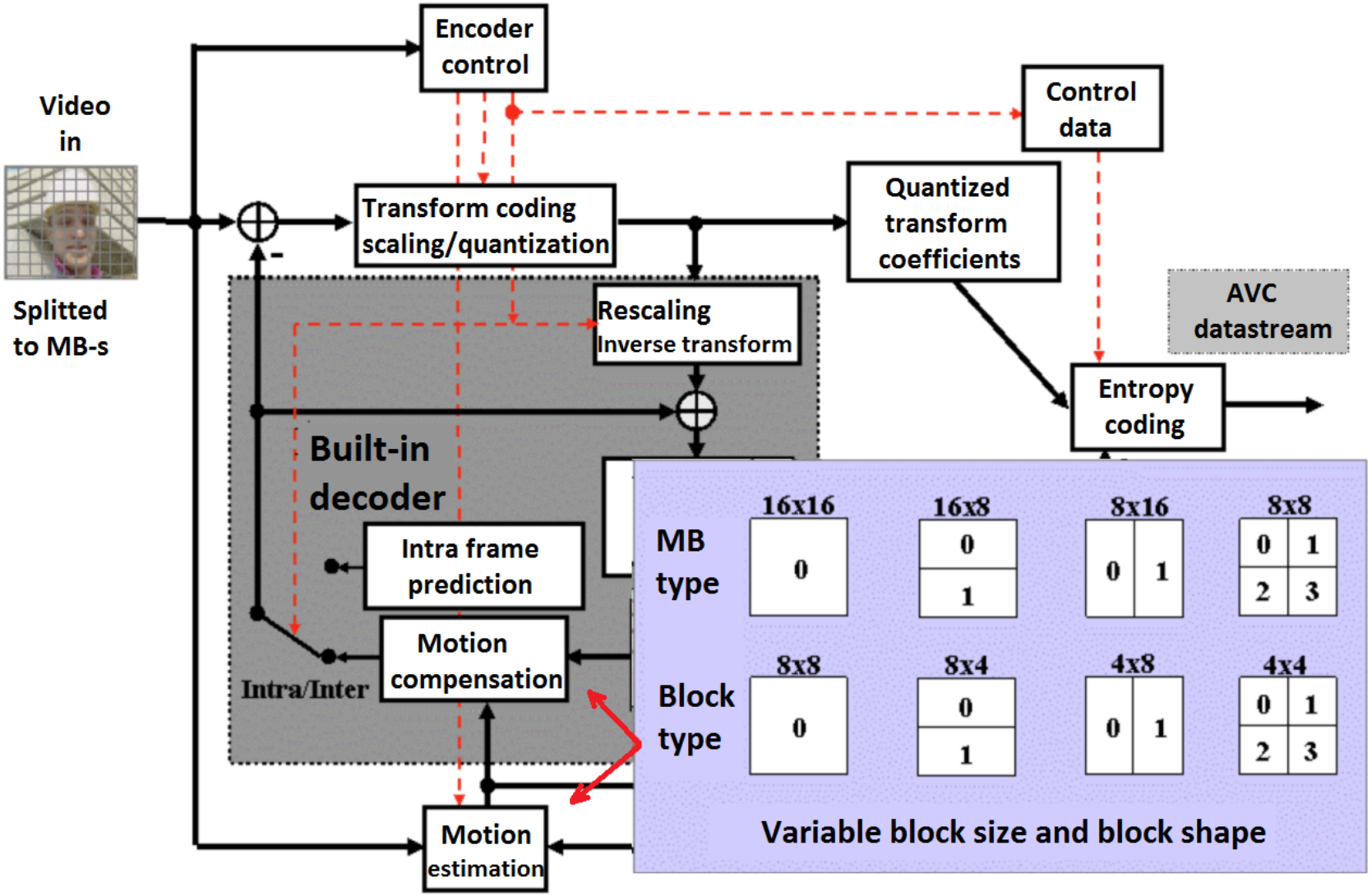
MPEG-4 H.264/AVC encoder



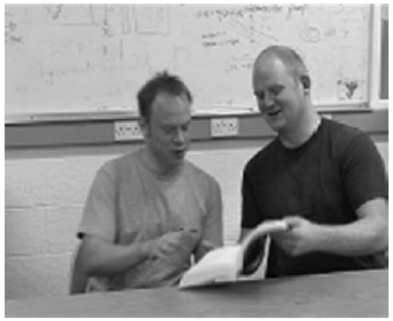
MPEG-4 H.264/AVC details

- **Motion compensation**: MB are size of 16×16, units of motion compensation are sub Macroblocks, with the size of 16×16, 16×8, 8×16, 8×8, 4×8, 4×8 or 4×4 pixels
- **Transform block-size**: Prediction error/residual is transformed with 4×4 or 8×8 (FRExt) sized blocks
- **Transform coding**: Integer-DCT transform instead of DCT

MPEG-4 H.264/AVC MB sizes



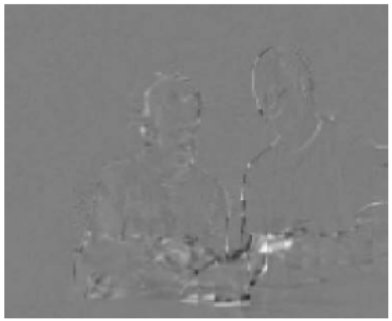
Block based motion estimation (again)



Frame 1

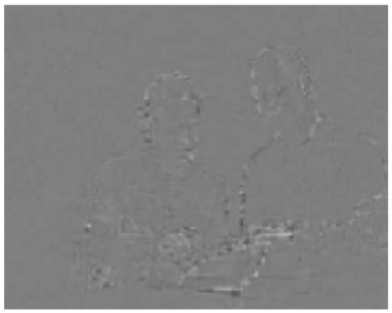
Residual : 16 × 16 block size

Frame 2



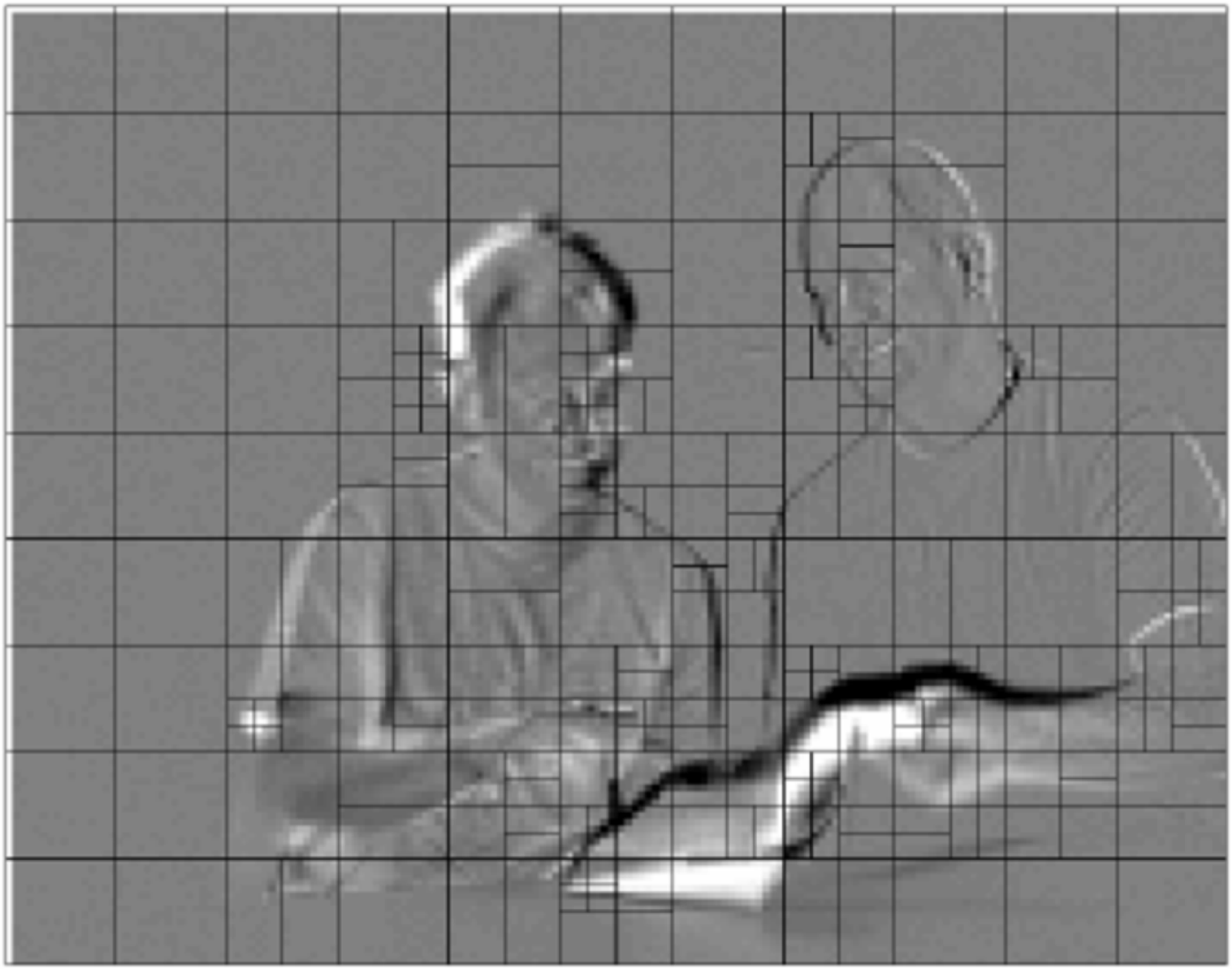
Residual : 8 × 8 block size

Residual : no motion compensation



Residual : 4 × 4 block size

MPEG-4 H.264/AVC MB sizes



MPEG-4 H.264/AVC details

- **Video layers**: Sequence - Picture - Slice - MB - sub-MB - block - sample
- **Deblocking filter**: decreases significantly the artifacts if the prediction error by smoothing the block edges. The perceived quality of the decoded video increases, thus the required bitrate for the same video quality is decreased by 5-10%
- **Two frame types**: reference and non-reference frame
- Within a frame: I,P,B type slices
- Within slice: I,P,B type MBs (e.g. intra MB in a P slice is possible)
- New entropy coding methods (CABAC, CAVLC)

MPEG-4 H.264/AVC details

Weighted prediction and motion compensation I.

- H.264/AVC allows the application of multiple reference pictures for P and B type predictions
- While in MPEG-2 the anchor frame for prediction was the adjacent/previous I or P frame (or for B prediction the previous and the next frame I or P frame), in H.264 multiple reference frame may be used for prediction
- Number of reference/anchor frames is limited by the actual Level
- Reference picture selection is MB based.

MPEG-4 H.264/AVC MB sizes

Example

A macroblock in frame *n* is shown in Figure 6.17. The macroblock is divided into two partitions, each consisting of 8 × 16 luma samples and corresponding chroma samples. The left partition (A)

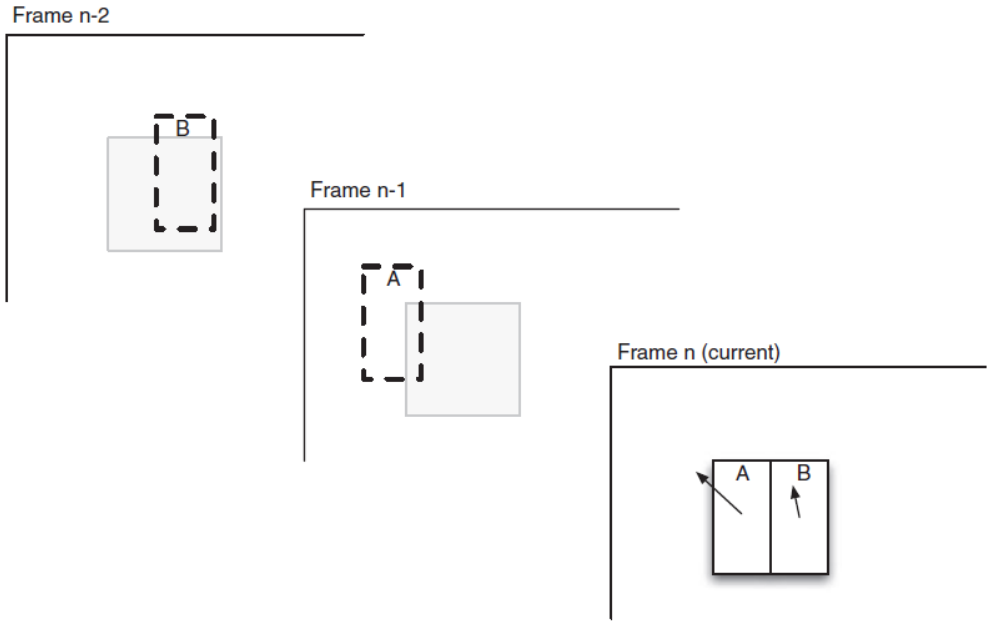


Figure 6.17 P macroblock prediction example

is predicted from a region in the previous frame, frame *n−1*, and the right partition (B) is predicted from a region in frame *n−2*. Partition A has a motion vector (−6.5, −5.75), i.e. the reference region is offset by −6.5 samples in the x direction (left) and −5.75 samples in the y direction (up). Partition B has motion vector (−1.25, −4), i.e. −1.25 samples in the x direction (left) and −4 samples in the y direction (up).

MPEG-4 H.264/AVC motion compensation with multiple reference

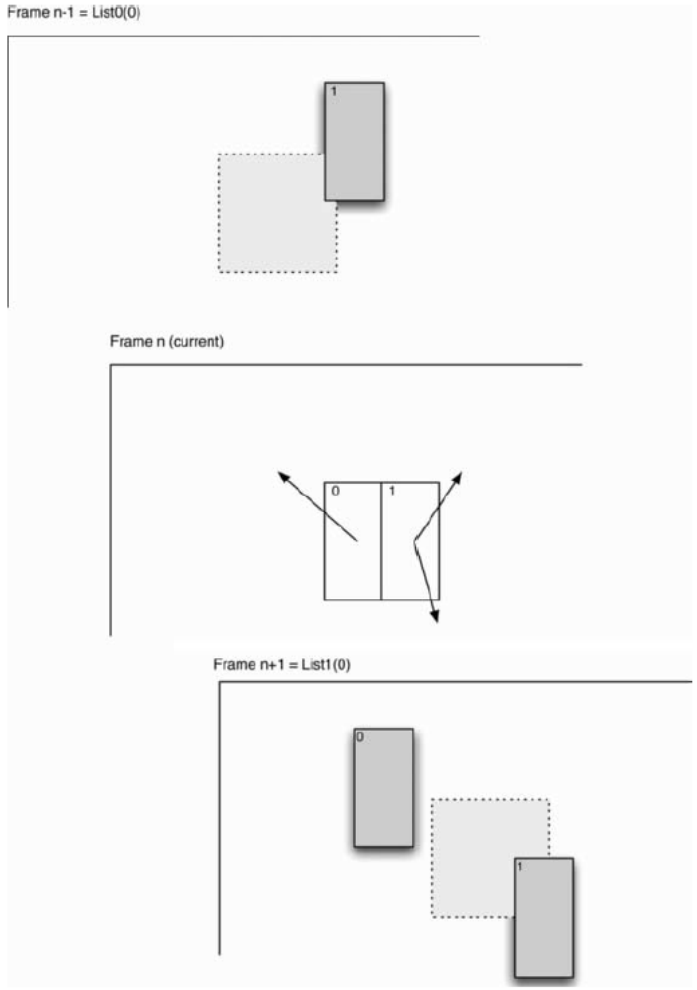
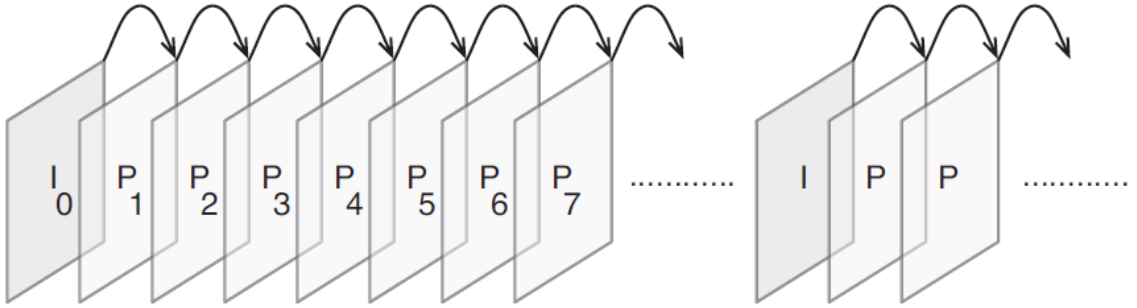


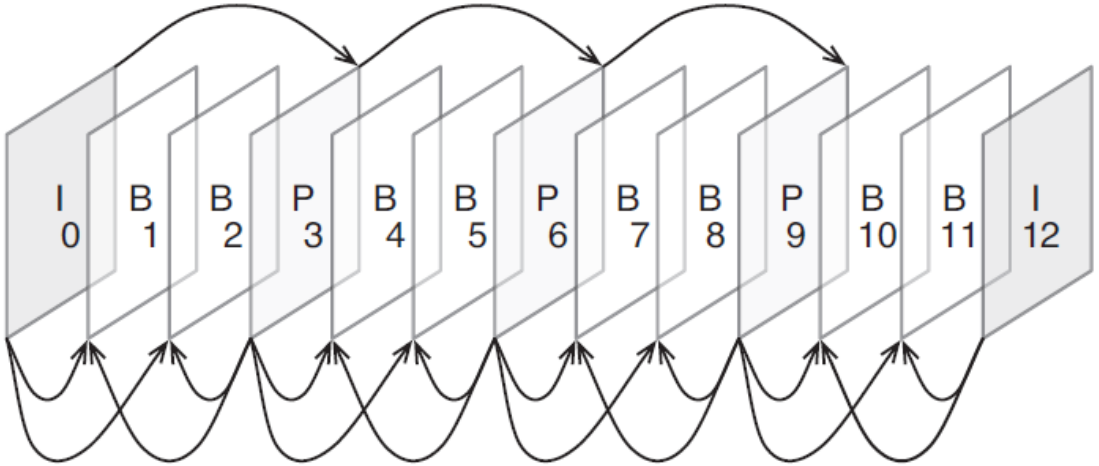
Figure 6.40 Inter prediction example, B slice

MPEG-4 H.264/AVC prediction structures I.

– Low latency structure

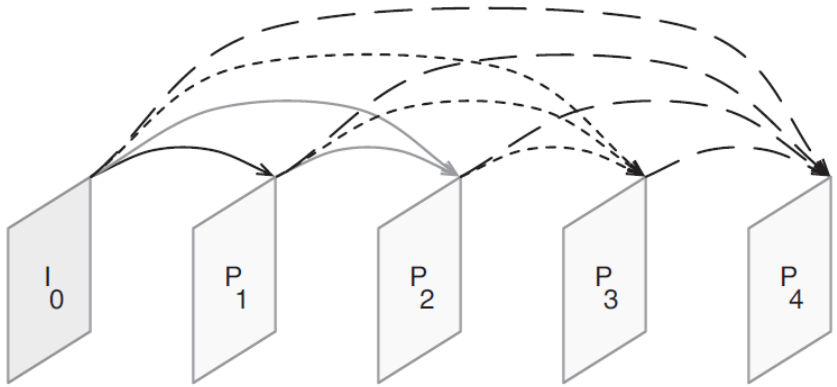


– Classic GOP structure

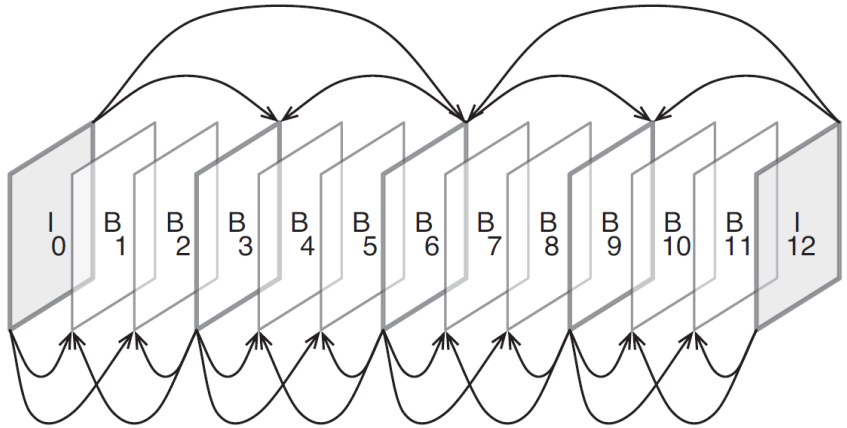


MPEG-4 H.264/AVC prediction structures II.

– Multiple reference structure



– Hierarchic GOP structure



– that compression factor is increased by new structures

MPEG-4 H.264/AVC details

Weighted prediction and motion compensation II.

- The reference picture can be weighted by arbitrary scale and offset factors, with in **explicit mode** the weighting factor(s) determined by the encoder and transmitted in the slice header. The motion compensated prediction is performed by the linear combination of these weighted refrence frames
- If **implicit prediction** is used, the weighting factors are calculated based on the relative temporal positions of the list 0 and list 1 reference frames. A larger weighting factor is applied if the reference frame is temporally close to the current frame and a smaller factor is applied if the reference frame is temporally further away from the current frame.
- For example, weighted prediction may be particularly effective in coding of fade transitions where one scene fades into another.

MPEG-4 H.264/AVC details

Weighted prediction and motion compensation III.

- For P and B slices also the motion vectors are differentially coded
- Motion vectors are coded by their difference from the median of the 3 adjacent motion vectors, or encoded by scaled prediction
- The encoder may omit the transmission of motion vectors as well, and ask the decoder to estimate the missing motion vector from the adjacent, available vectors

Steps of interprediction (example)

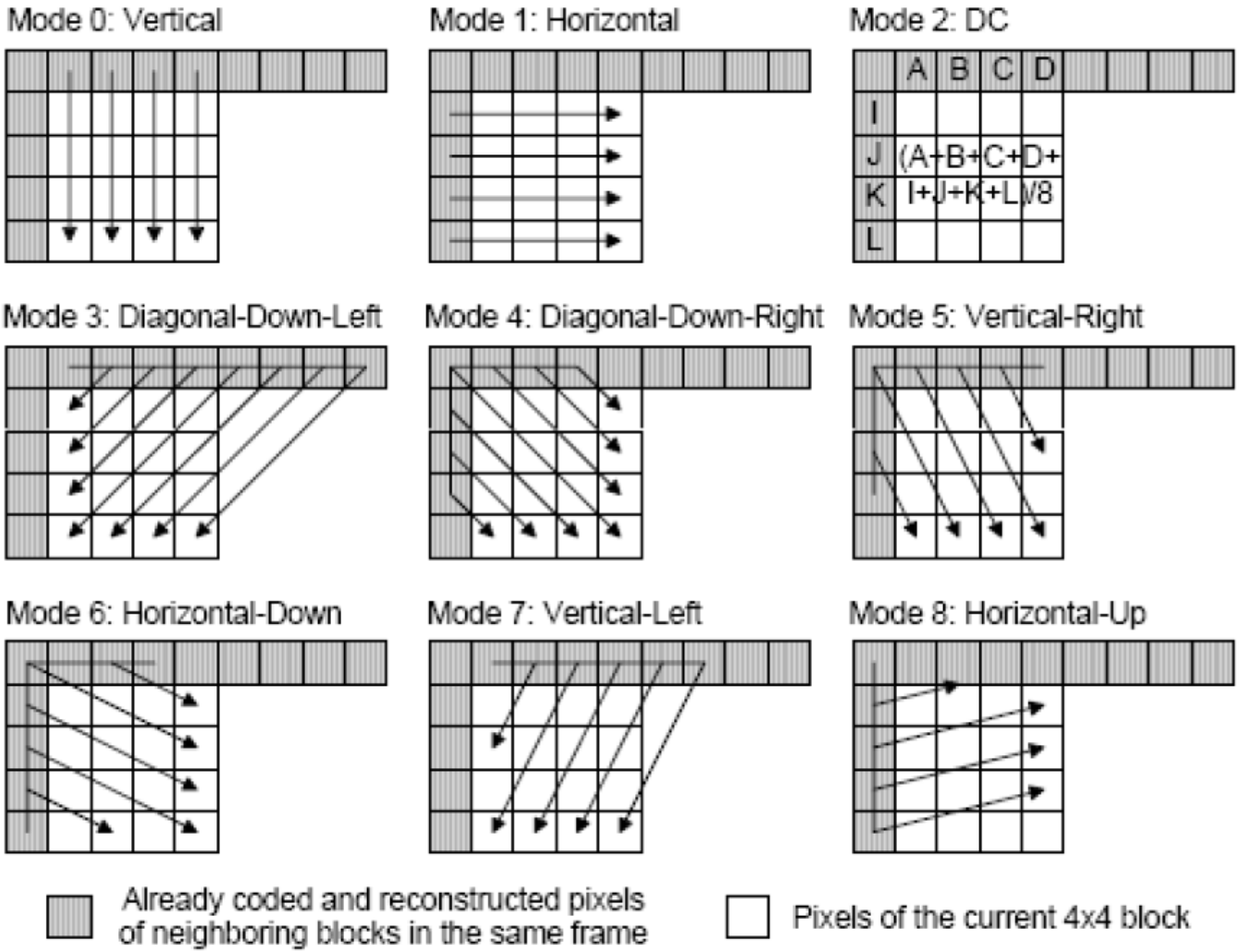
- Interpolation of the images to 1/4 or 1/8 pixels
- Choosing interprediction parameters:
  - Choosing reference frames
  - Choosing P or B prediction types
  - Choosing sub-MB sizes
  - Type of weighted prediction
- Definition of motion vectors
- Prediction of motion vectors

MPEG-4 H.264/AVC details

Intra prediction

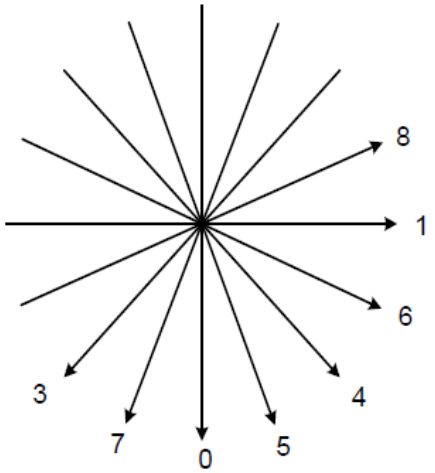
- Within an I slice the entire MB (16x16 pels) or 4x4, or 8x8 blocks are coded predictively from previously-coded data in the same slice.
- Prediction is performed from the adjacent pixels of the actual block in several direction of prediction
- for 16x16 intra coding the direction of prediction may be horizontal, vertical, DC and planar
- for 4x4 intra prediction 9 different direction of prediction is possible, where
- intra prediction of H.264/AVC increases the compression factor significantly, opposed to MPEG-1/2, where only the DC coefficients of adjacent blocks were coded differentially/predictively in case if intra MBs

H.264/AVC 4x4 (and 8x8) intra prediction

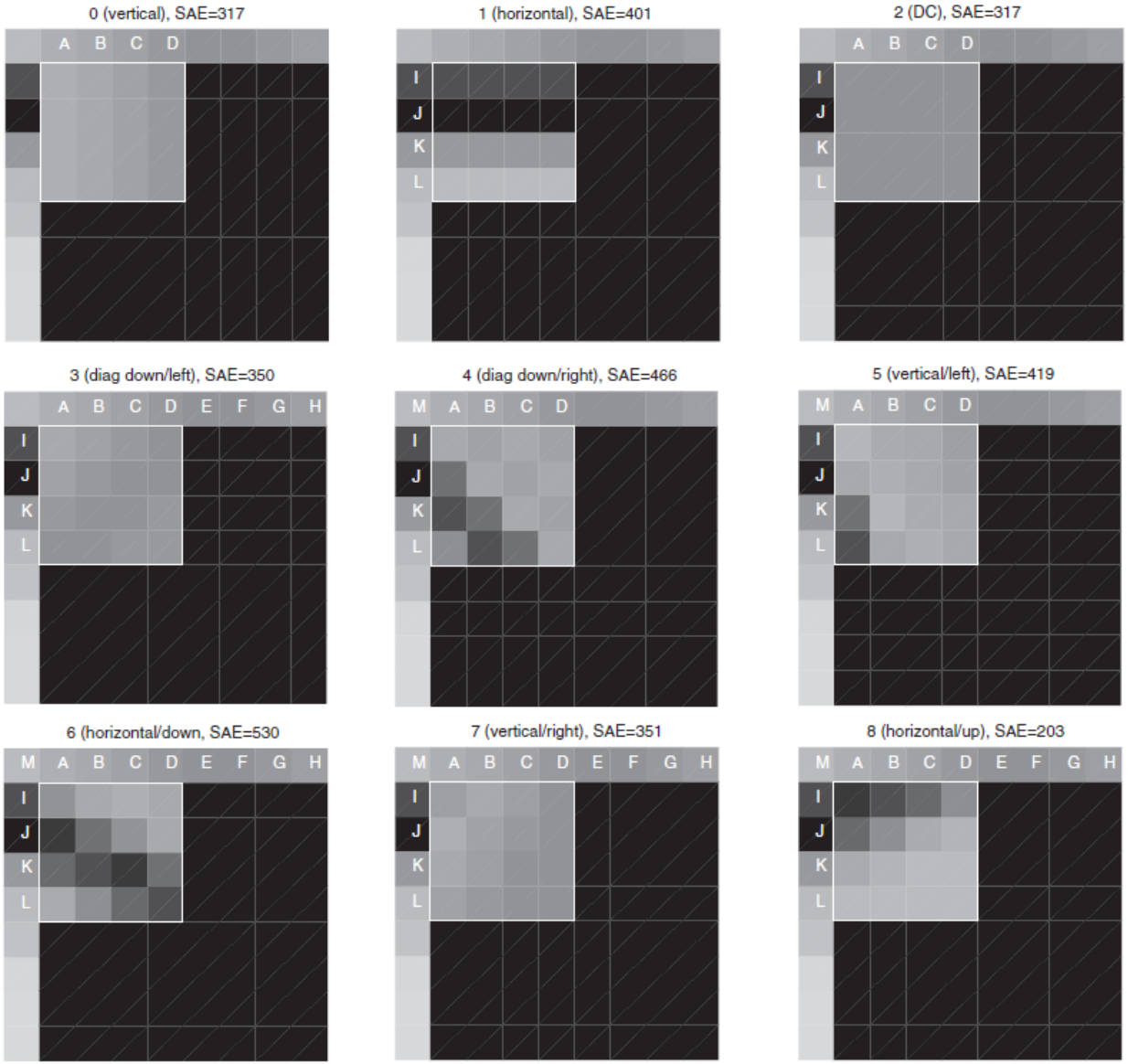


H.264/AVC 4x4 intra prediction

Intra4x4PredMode[luma4x4BlkIdx]	Name of Intra4x4PredMode[luma4x4BlkIdx]
0	Intra_4x4_Vertical (prediction mode)
1	Intra_4x4_Horizontal (prediction mode)
2	Intra_4x4_DC (prediction mode)
3	Intra_4x4_Diagonal_Down_Left (prediction mode)
4	Intra_4x4_Diagonal_Down_Right (prediction mode)
5	Intra_4x4_Vertical_Right (prediction mode)
6	Intra_4x4_Horizontal_Down (prediction mode)
7	Intra_4x4_Vertical_Left (prediction mode)
8	Intra_4x4_Horizontal_Up (prediction mode)



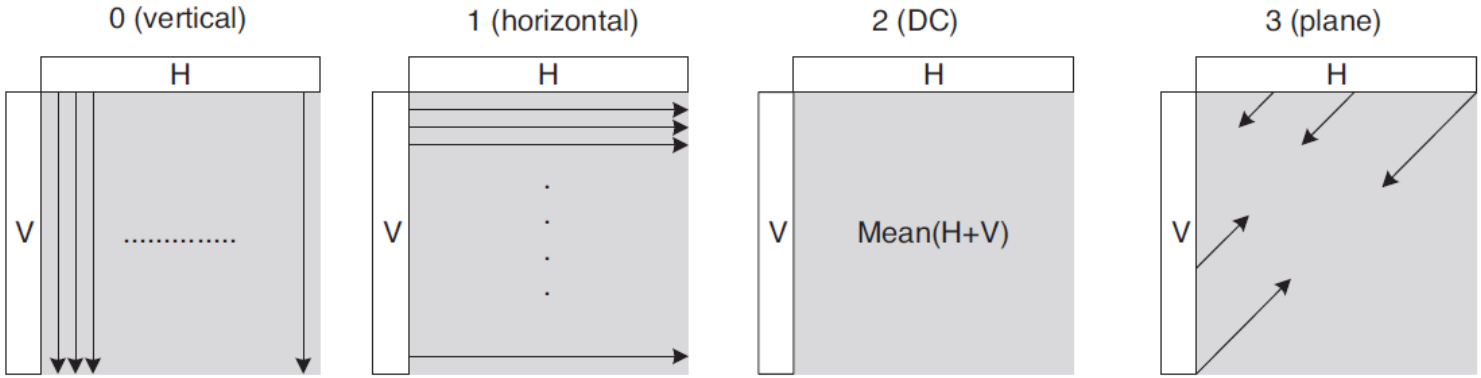
H.264/AVC 4x4 intra-prediction



4x4 intra-prediction example

```
Intra_4x4_Vertical prediction mode
pred4x4[ x, y ] = p[ x, −1 ], with x, y = 0..3


Intra_4x4_Horizontal prediction mode
pred4x4[ x, y ] = p[ −1, y ], with x,y = 0..3


Intra_4x4_DC prediction mode
pred4x4[ x, y ] = ( p[ 0, −1 ] + p[ 1, −1 ] + p[ 2, −1 ] + p[ 3, −1 ] +
p[ −1, 0 ] + p[ −1, 1 ] + p[ −1, 2 ] + p[ −1, 3 ] )


Intra_4x4_Diagonal_Down_Left prediction mode
pred4x4[ x, y ] = ( p[ x + y, −1 ] + 2 * p[ x + y + 1, −1 ] + p[ x + y + 2, −1 ] )


Intra_4x4_Diagonal_Down_Right prediction mode
If x is greater than y,
pred4x4[ x, y ] = ( p[ x − y − 2, −1] + 2 * p[ x − y − 1, −1 ] + p[ x − y, −1 ] )
Otherwise if x is less than y,
pred4x4[ x, y ] = ( p[ −1, y − x − 2 ] + 2 * p[ −1, y − x − 1 ] + p[ −1, y − x ] )
```

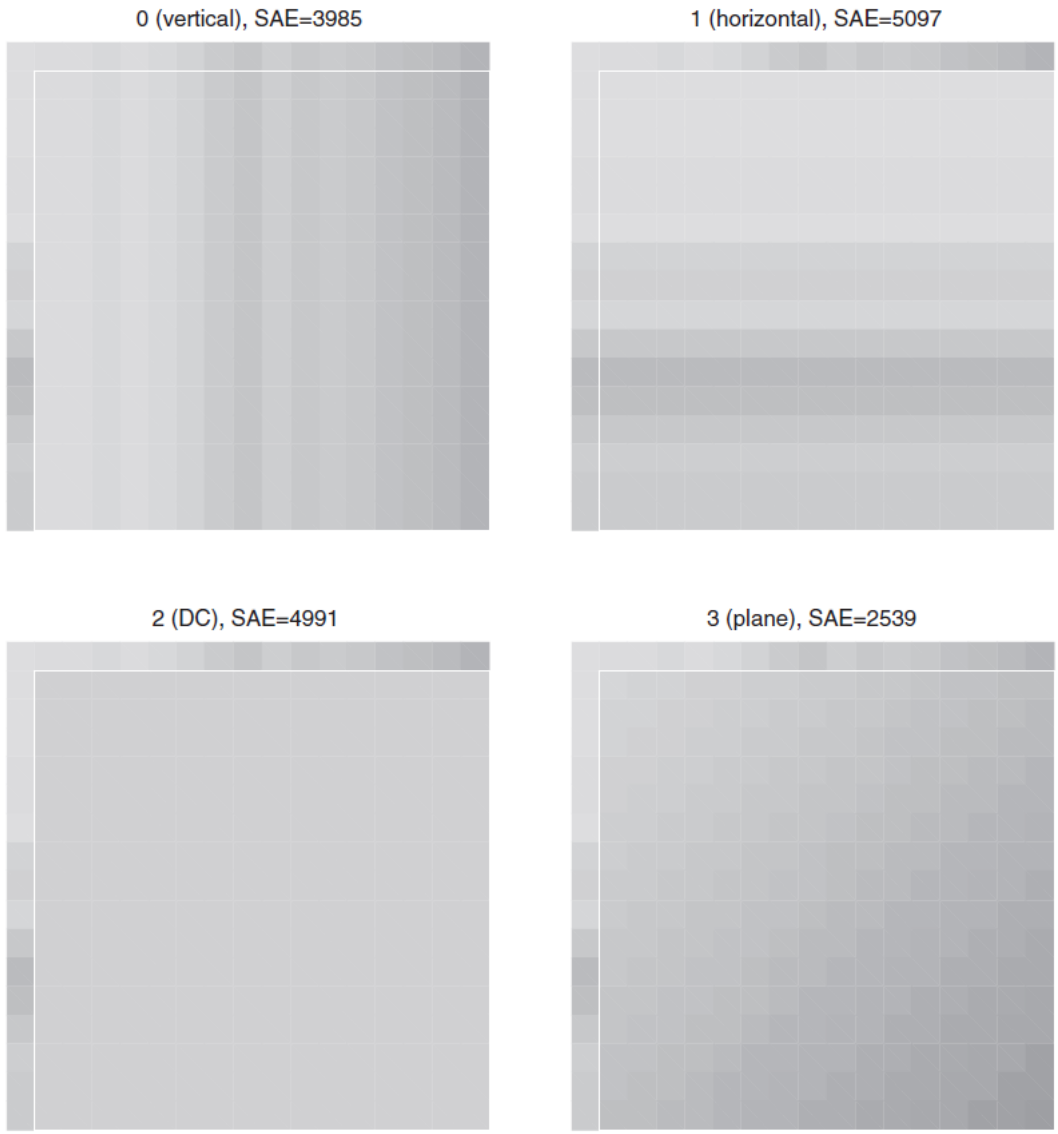
H.264/AVC 16x16 intra prediction



- vertical extrapolation: Extrapolation from upper samples (H)
- horizontal extrapolation: Extrapolation from left samples (V)
- DC: Mean of upper and left-hand samples ($H + V$)
- Plane: A linear 'plane' function is fitted to the upper and left-hand samples H and V. This works well in areas of smoothly-varying luminance.

In case of 16×16 intra prediction the DC coefficients of the 4×4 integer transform (16 coefficients) are further Hadamard transformed

H.264/AVC 16x16 intra-prediction example



Discrete Hadamard transform (again)

Non-harmonic basis

- Real, orthogonal transform
- Transform matrix of 2×2-sized transform:

$$\mathbf{H}_{2\times 2} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- For larger block-sizes the transform matrix $\mathbf{H}$ is calculated recursively:

$$\mathbf{H}_{2N\times 2N} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{H}_{N\times N} & \mathbf{H}_{N\times N} \\ \mathbf{H}_{N\times N} & -\mathbf{H}_{N\times N} \end{bmatrix}$$

- The formulation of 2D DHT
  - $\mathbf{y} = \mathbf{H}\mathbf{x}\mathbf{H}^T$ and
  - $\mathbf{x} = \mathbf{H}^T\mathbf{y}\mathbf{H}$

MPEG-4 H.264/AVC details

Integer transform

- DCT transform of MPEG-1/MPEG-2 applied idealized trigonometric basis functions
- it is not guaranteed that the decoder uses the same basis functions (numerically, e.g. the same accuracy, same rounding factors) as the encoder: error may accumulate
- solutions in MPEG-1/MPEG-2: defining tolerancy thresholds

MPEG-4 H.264/AVC details

Integer transform

In H.264/AVC the transform and quantization processes are designed to minimize computational complexity and to avoid encoder/decoder mismatch, achieved by

- using a core transform, an integer transform, that can be carried out using integer or 16 bits fixed-point arithmetic and
- integrating a normalization step with the quantization process to minimize the number of multiplications required to process a block of residual data.
- Integer transform matrix: obtained from the 4×4 sized 1D DCT transform matrix by rescaling, rounding to the nearest integer and orthonormalizing its values
- Implementation requires only addition/subtraction and shifting

MPEG-4 H.264/AVC details

Integer transform

- In 16x16 mode intra prediction the DC coefficients of integer transformed blocks are further Hadamard transformed (in 4x4 block size)
- The effective quantizer stepsize is logarithmic function of quantizer scale (opposed to simple linear relation, used in MPEG-1/MPEG-2), fitting better to HVS properties
- In FRExt profile the encoder can choose between 4x4 and 8x8 transform sizes in MB level

MPEG-4 H.264/AVC details

4x4 integer transform matrix

Rounded integer approximation of 4x4 sized 1D DCT matrix

$$\mathbf{T}_{4\times4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \qquad \mathbf{D}_{st} = \begin{bmatrix} 1/2 & & & \\ & 1/\sqrt{10} & & \\ & & 1/2 & \\ & & & 1/\sqrt{10} \end{bmatrix}$$

where multiplication with $\mathbf{D}_{st}$ orthonormalizes the elements of $\mathbf{T}_{4\times4}$ ($D_{st}[i,i] = \frac{1}{\sqrt{\sum_j (T[i,j])^2}}$)

Therefore, it can be written that:

$$\mathbf{A}_1 = \mathbf{D}_{st}\mathbf{T}_{4\times4} \qquad \mathbf{Y} = \mathbf{A}_1\mathbf{X}\mathbf{A}_1^T$$

MPEG-4 H.264/AVC details

4x4 integer transform matrix

In practice, left product with the diagonal matrix $\mathbf{D}_{st}$ is realized by element-wise multiplication by a matrix $\mathbf{S}_t$, obtained as the dyadic product of the vector, containing the diagonal elements of $\mathbf{D}_{st}$

$$\mathbf{Y} = \mathbf{D}_{st}\,\mathbf{T}_{4\times4}\,\mathbf{X}\,(\mathbf{D}_{st}\,\mathbf{T}_{4\times4})^T = \mathbf{D}_{st}\,\mathbf{T}_{4\times4}\,\mathbf{X}\,\mathbf{T}_{4\times4}^T\,\mathbf{D}_{st}^T = (\,\mathbf{T}_{4\times4}\,\mathbf{X}\,\mathbf{T}_{4\times4}^T)\star(\mathbf{d}_{st}\,\mathbf{d}_{st}^T)$$

thus

$$\mathbf{Y} = (\mathbf{T}_{4\times4}\,\mathbf{X}\,\mathbf{T}_{4\times4}^T)\star\,\mathbf{S}_t$$

where

$$\mathbf{S}_t = \mathbf{d}_{st}\mathbf{d}_{st}^T$$

MPEG-4 H.264/AVC details

4x4 integer transform matrix

Re-quantization of the transform coefficients are calculated as a right shift by 15 bits

$$\mathbf{Y}_q = \frac{1}{2^{15}} \left[(\mathbf{T}_{4\times4} \, \mathbf{X} \, \mathbf{T}^T_{4\times4}) \star \mathbf{W}_t \right]$$

$$\mathbf{W}_t = \frac{2^{15}}{Q_{step}} \mathbf{S}_t$$

The constant factor 2^{15} is chosen as a compromise between higher accuracy and limited arithmetic precision.

MPEG-4 H.264/AVC details

4x4 integer inverse transform matrix

$$\mathbf{T}_{4\times4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix}$$

$$\mathbf{D}_{si} = \begin{bmatrix} 1/2 & & & \\ & \sqrt{2/5} & & \\ & & 1/2 & \\ & & & \sqrt{2/5} \end{bmatrix}$$

$$\mathbf{A}_2 = \mathbf{D}_{si}\mathbf{T}_{4\times4} \qquad \mathbf{X} = \mathbf{A}_2^T\mathbf{Y}\mathbf{A}_2$$

MPEG-4 H.264/AVC details

4x4 integer inverse transform matrix

$$\mathbf{A}_2 = \mathbf{D}_{si}\mathbf{T}_{4\times4} \qquad \mathbf{X} = \mathbf{A}_2^T\mathbf{Y}\mathbf{A}_2$$

$$\mathbf{X} = (\mathbf{D}_{si}\ \mathbf{T}_{4\times4})^T\mathbf{Y}(\mathbf{D}_{si}\ \mathbf{T}_{4\times4}) = \mathbf{T}_{4\times4}^T\ \mathbf{D}_{si}^T\ \mathbf{Y}\ \mathbf{D}_{si}\ \mathbf{T}_{4\times4} = \mathbf{T}_{4\times4}^T\ (\mathbf{Y}\star\mathbf{S}_i)\ \mathbf{T}_{4\times4}$$

where

$$\mathbf{S}_i = \mathbf{d}_{si}\mathbf{d}_{si}^T$$

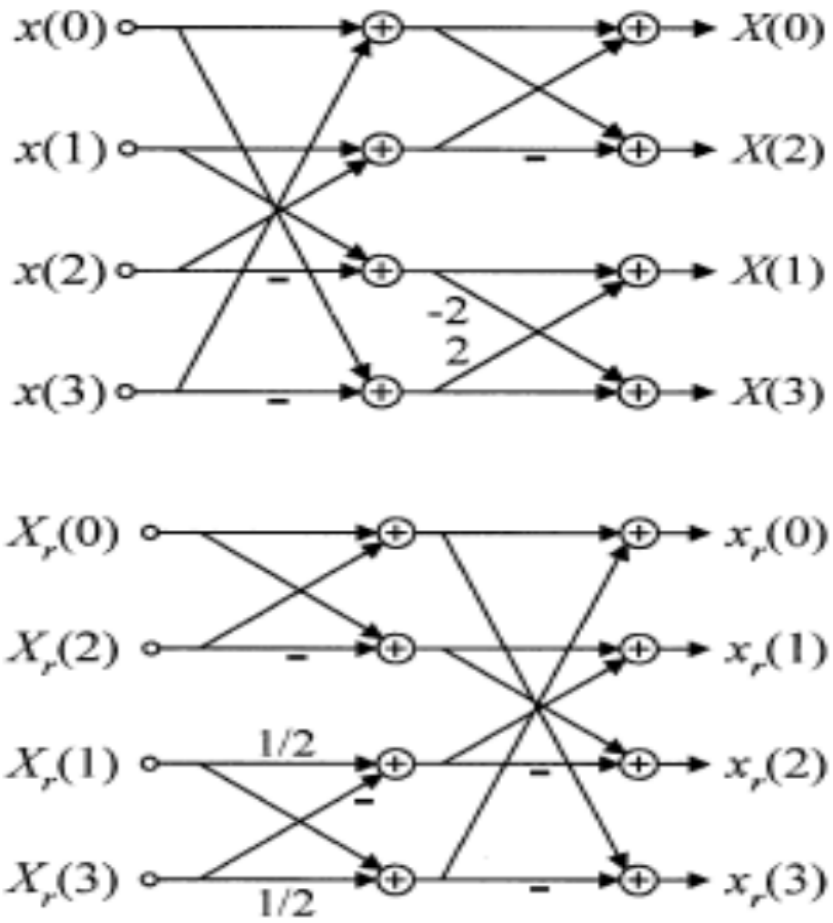
Inverse quantization is realized as the followings

$$\mathbf{X} = \frac{1}{2^6}\ \left[\mathbf{T}_{4\times4}^T\ (\mathbf{Y}_q\star\mathbf{W}_i)\ \mathbf{T}_{4\times4}\right]$$

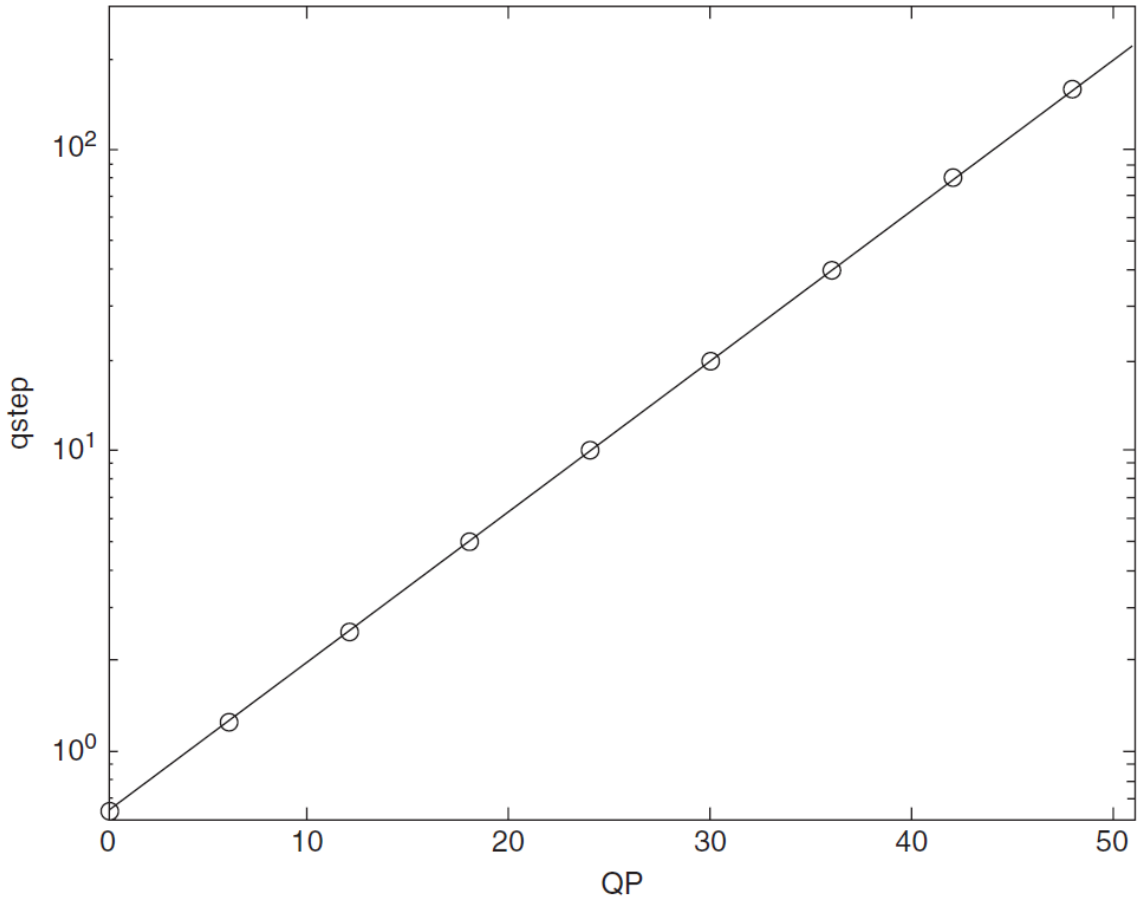
where

$$\mathbf{W}_i = 2^6\ Q_{step}\mathbf{S}_i$$

H.264/AVC 16x16 integer core transform implementation



Relation of Q_{step} and QP



Q_{step} : effective quantizer step size

QP : quantization parameter (as controlled by e.g. bitrate control system or the user)

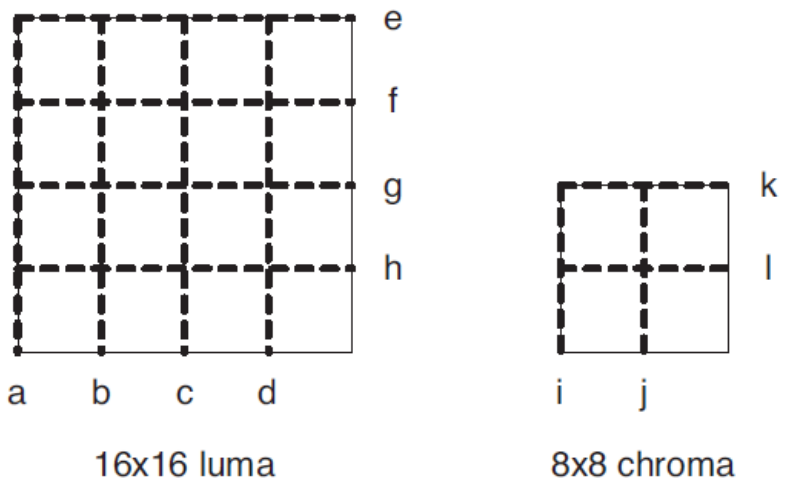
MPEG-4 H.264/AVC details

De-blocking filter

- Deblocking filter in the prediction loop decreases the "blocking" artifacts by spatial low-pass filtering the block edges
- Filter coefficients: standardized
- Obviously, the reference for prediction is the filtered image (to avoid error accumulation)
- Filter strength is defined by the MB type (I,P,B), the quantization scale and several other factors
- Basically: larger quantization steps $\rightarrow$ higher filter strength applied

H.264/AVC De-blocking filter

– Edge filtering order in a macroblock



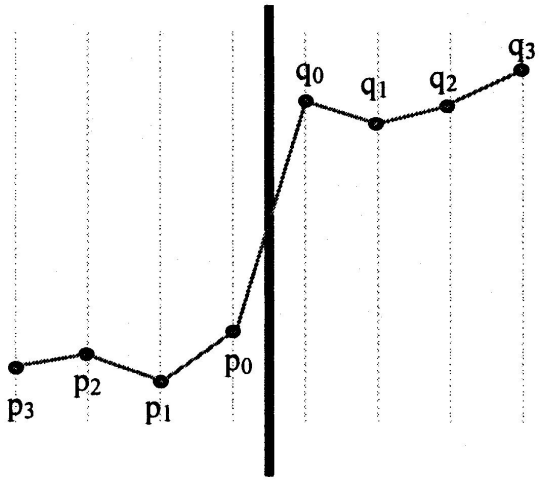
– Not the entire picture is low-pass filtered (data loss would occur)
– Not every edge has to be filtered with the same strength (blocking artifact should separated from real edge in video content)
– Filtering: in two steps
  - Classification of edges - filter decision
    - ≪ "Boundary strength"
    - ≪ gradient of image samples across the boundary
  - filtering the pixels at the boundary

H.264/AVC De-blocking filter II.

– Boundary Strength parameter

p or q is intra coded **and** boundary is a macroblock boundary	Bs = 4, strongest filtering
p or q is intra coded and boundary is **not** a macroblock boundary	Bs = 3
neither p or q is intra coded; p or q contain coded coefficients	Bs = 2
neither p or q is intra coded; neither p or q contain coded coefficients; p and q have different reference frames **or** a different number of reference frames **or** different motion vector values	Bs = 1
neither p or q is intra coded; neither p or q contain coded coefficients; p and q have same reference frame and identical motion vectors	Bs = 0, no filtering

– Gradient between adjacent blocks

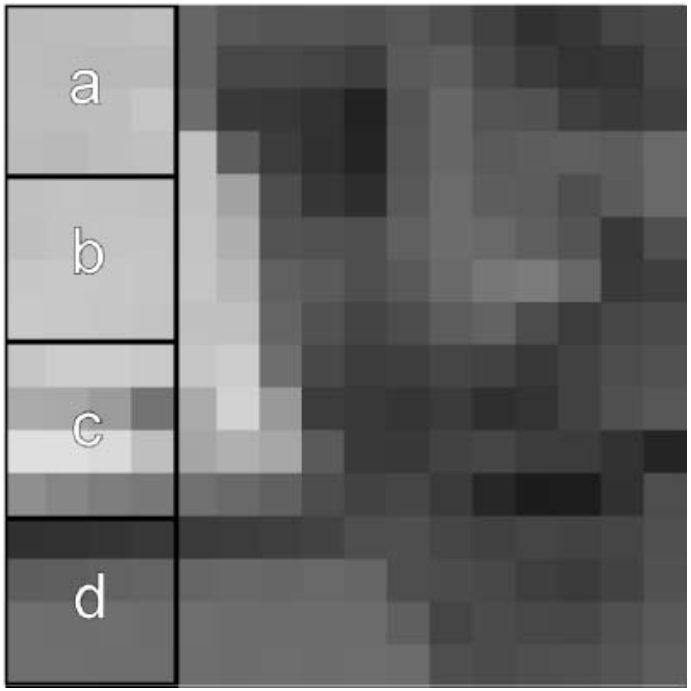


– Condition of filtering: only if $|p_0 - q_0|$, with $|p_1 - p_0|$ and $|q_1 - q_0|$ are all LOWER than a predefined threshold α and β respectively

– if gradient is larger: the jump on the block edge is not blocking artifact, but actual content

H.264/AVC De-blocking filter III.

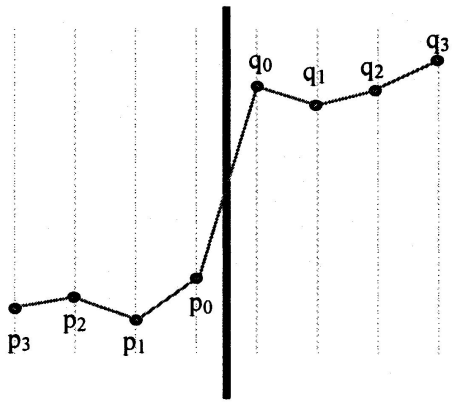
Filter decision example:



- gradient at edges $a - b$ and $b - c$ blokk the gradient is small
- blocking would be clearly visible $\rightarrow$ deblocking turn on
- $c - d$ transition: large gradient: most probably the edge is the part of the image
- here deblocking is turned off

H.264/AVC De-blocking implementation

- $BS \in 1, 2, 3$:
  - from p_1, p_0, q_1, q_0 (4 long FIR filter) P_0, Q_0 are calculated
  - if $|p_2 - p_0| < \beta$: p_2, p_1, p_0, q_0 involved to the calculation of P_1
  - if $|q_2 - q_0| < \beta$: p_0, q_0, q_1, q_2 involved to the calculation of Q_1
- $BS = 4$
  - if $p_2 - p_0 < \beta$ and $p_0 - q_0 < \frac{\alpha}{4}$
    $\ll$ P_0, P_1, P_2 calculated with 5,4,5 long filters respectively
  - if $q_2 - q_0 < \beta$ and $p_0 - q_0 < \frac{\alpha}{4}$
    $\ll$ Q_0, Q_1, Q_2 calculated with 5,4,5 long filters respectively
  - else: P_0 ill. Q_0 calculated with 3 long filters



H.264/AVC effect of deblocking filter

QP=32, no filter

QP=32, with filter

MPEG-4 H.264/AVC details

New color space: YCgCo (FRExt)

- Calculation of YCbCr component (e.g. from RGB components) requires a linear transform, where multiplication with coefficients are floating-point operations: it can not be always performed with optimal precision
- In case of integer transform coding the floating point representation of YCbCr components is the inevitable source of rounding errors
- New color space: new Y component and Cg (Green chroma), Co (Orange Chrome) components
- Derivation:
  $Y = \frac{1}{2}\left(G + \frac{(R+B)}{2}\right)$, $Cg = \frac{1}{2}\left(G - \frac{(R+B)}{2}\right)$, $Co = \frac{(R-B)}{2}$