# AY 2019/2020
# EE6403 Distributed Multimedia Systems

## Part 2 Text and Image Compression & Standards

Dr Yap Kim Hui

Room: S2-B2b-53
Tel:  6790 4339
Email: ekhyap@ntu.edu.sg

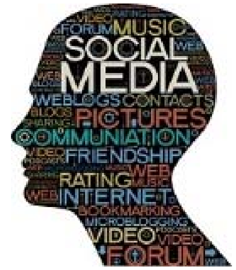

# Section II
# Media Compression and Standards

# This Section

- Text Compression
- Image & Video Compression Basics
- Image Coding/Compression:
  - Transform coding
    - Discrete Cosine Transform (DCT)
    - Karhunen-Loeve Transform (KLT)
  - Predictive/differential coding
  - Vector Quantization (VQ)
  - Singular Value Decomposition (SVD)
- JPEG standard

# Text Compression

# Text




- Text represents letters, numbers and special characters in coded form, such as American Standard Code for Information Interchange (ASCII) or American National Standard Institute (ANSI) code.
- Such codes allow computers to process and store text far more efficiently.
- Text conveys essential and precise information.

# Text Compression


- Text compression must be lossless.
- Two types of coding
  - Static coding: text to be compressed has known characteristics.
  - Dynamic coding: the characteristics of text being transferred may vary from one transfer to the other.
- Huffman coding
  - A type of Variable Length Coding (VLC).
  - Higher frequency patterns (symbols) are assigned shorter codewords.
  - Average number of bits used to represent each pattern/ symbol will be reduced.

# Huffman Coding (I)

Example:

String of character:

AAAABBCD

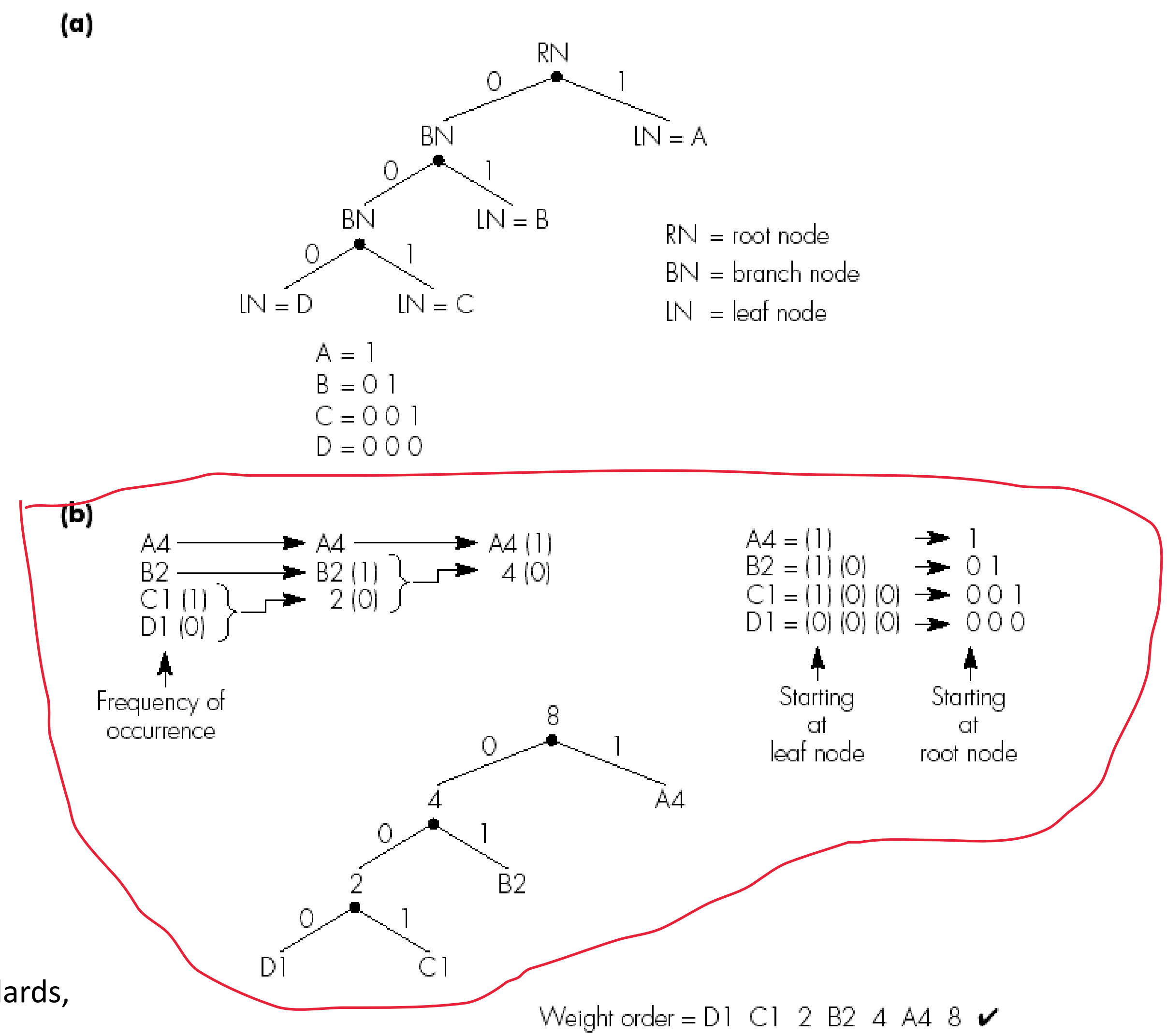



Figure source:

Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

# Huffman Coding (II)

- A Huffman code tree is a binary tree with branches assigned value 0 or 1.
- Tree structure
  - Root node: the root of tree.
  - Branch node: the point at which a branch divides.
  - Leaf node: the termination point of a branch.
- At each branch node, 0 and 1 are assigned to the left and right branches accordingly.
- The codeword used for each character/symbol are determined by tracing the path from the root node to the leaf node.

# Exercise 2.1: Text Coding

(a) In a simple messaging system, only the first eight Roman alphabets (A, B, C, D, E, F, G, and H) are used to encode and convey the textual information. The probabilities of occurrences for these alphabets (symbols) in the messages are given in the table below:

| Symbol | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| Probability of occurrences | 0.05 | 0.10 | 0.20 | 0.15 | 0.15 | 0.15 | 0.10 | 0.10 |

(i) Design a suitable set of Huffman codewords to encode the alphabets based on the probabilities given in the table above. Your answer should clearly outline all the steps and calculations involved.

(8 marks)

# Exercise 2.1: Text Coding


(ii) The following messages are to be encoded and transmitted:

(Message X): BCCDEEFG
(Message Y): AGHCDAAE

The messages are to be encoded using two different schemes:

(1) Uncompressed scheme: 3-bit binary codewords, and
(2) Compressed scheme: Huffman codewords designed in part (a)(i).

Calculate the compression ratios that can be achieved for Messages X and Y, respectively, by using the compressed scheme over the uncompressed scheme. Hence, discuss and explain the effectiveness of using Huffman coding to compress Messages X and Y, respectively.

(6 marks)

# Solution


FORUM MUSIC
SOCIAL
MEDIA
WEBLOGS CONTACTS
PICTURES
COMMUNIATION
FRIENDSHIP
RATING WEB
INTERNET
BOOKMARKING
MICROBLOGGING
VIDEO VIDEO
FORUM

# Exercise 2.2: VQ and Huffman Coding

In a simple image compression scheme, vector quantization (VQ) is used to compress a class of 2-bit grayscale images, with gray levels of 0, 1, 2 and 3. A given image is partitioned into $2 \times 2$ blocks (subimages), and encoded using $2 \times 2$ codevectors from a generated codebook. The codebook used in this VQ scheme, together with the associated probabilities of occurrences for the codevectors, is given as follows:

| Symbols | $s_0$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ |
|---|---|---|---|---|---|---|---|---|
| Probabilities of occurrences | 0.2 | 0.2 | 0.15 | 0.15 | 0.1 | 0.1 | 0.05 | 0.05 |
| Codevectors | $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ | $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ | $\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$ | $\begin{bmatrix} 3 & 3 \\ 3 & 3 \end{bmatrix}$ | $\begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$ | $\begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$ | $\begin{bmatrix} 0 & 0 \\ 2 & 2 \end{bmatrix}$ | $\begin{bmatrix} 2 & 2 \\ 0 & 0 \end{bmatrix}$ |

# Exercise 2.2: VQ and Huffman Coding

(b) Huffman coding is used to encode the symbols or indices of the codevectors. Design a suitable set of Huffman codewords to encode the symbols based on the probability distribution given in the table above. Your answer should clearly outline all the steps and calculations involved.

(6 marks)

(c) Based on the designed Huffman codeword set in part 1(b), compute the average number of bits that is required to represent a symbol. Hence, calculate the average compression ratio of this VQ scheme.

(4 marks)

# Solution

# Image & Video Compression Basics

# Source & Channel Coding


- Claude Shannon introduced a concept called entropy to measure the information content of a source.
- He asked 2 questions:
  - How can a communication system efficiently transmit the information that a source produces?
  - How can a communication system achieve reliable communication over a noisy channel?
- These are known as source coding and channel coding, respectively.
- Source coding is related to compression and channel coding is related to error control coding.

# Need for Compression


- There would be no multimedia today without the dramatic progress that occurred in compression algorithms and their implementations.
- Compression is necessary for two reasons:
  - To reduce the storage volume of media (audios, images and videos, etc.).
  - To reduce the bit rate required to transmit them over the network.

比特率是单位时间播放连续的媒体如压缩后的音频或视频的比特数量。在这个意义上讲，它相当于术语数字带宽消耗量，或吞吐量。

# Why Possible to Compress?


- Image and video can be compressed because of two types of redundancies:
  - Statistical Redundancy
    - Spatial redundancy
    - Temporal redundancy
    - Coding redundancy
  - Psycho-visual Redundancy
    - Frequency masking
    - Color masking

# Spatial Redundancy

- Spatial redundancy refers to the statistical correlation between pixels within an image (or more specifically, within a small image neighborhood).
- It is also called intraframe redundancy.

# Temporal Redundancy

- Temporal redundancy refers to the statistical correlation between pixels from successive frames in a video sequence.
- Therefore, it is also called interframe redundancy.

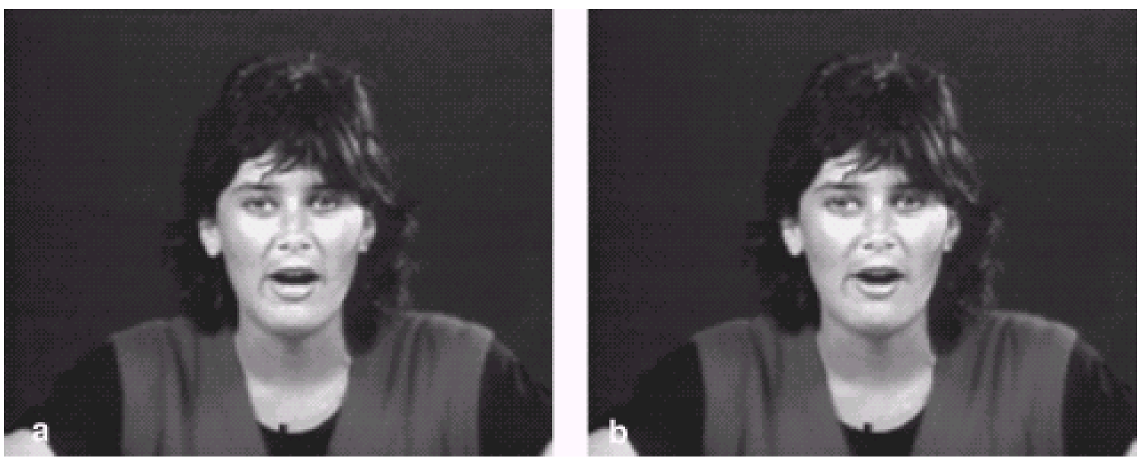



FIGURE 1.5 (a) The 21st frame, and (b) 22nd frame of the "Miss America" sequence.

Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

# Coding Redundancy


- Coding redundancy focus on the representation of information, i.e., coding itself.
- Consider the following example:

**TABLE 1.1**
**An Illustrative Example**

| Symbol | Occurrence Probability | Code 1 | Code 2 |
|---|---|---|---|
| $a_1$ | 0.1 | 000 | 0000 |
| $a_2$ | 0.2 | 001 | 01 |
| $a_3$ | 0.5 | 010 | 1 |
| $a_4$ | 0.05 | 011 | 0001 |
| $a_5$ | 0.15 | 100 | 001 |

**Figure source:**
**Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000**

- uniform-length coding: 3 bits/symbol
- variable-length coding

$$= 4 \times 0.1 + 2 \times 0.2 + 1 \times 0.5 + 4 \times 0.05 + 3 \times 0.15 = 1.95 \textit{ bits per symbol}$$

# Psychovisual Redundancy

- Frequency masking
  - Human is less sensitive to noise or distortion in high change fast frequency components and vice versa.
- Color masking
  - Human is more sensitive to luminance (brightness) components than chrominance (color) components.

# General Image & Video Compression Systems

# Source Encoding


- Source encoding is based on the content of the original signal and hence it is also termed as semantic-based coding.
- The compression that can be achieved using source coding may be very high, when compared to strict entropy coding.
- In general, source encoding may operate either in a lossless or lossy mode.
- Source encoding techniques can be classified into three popular types, namely transform coding, differential coding, and vector quantization. Compression

# General Image and Video Compression Systems

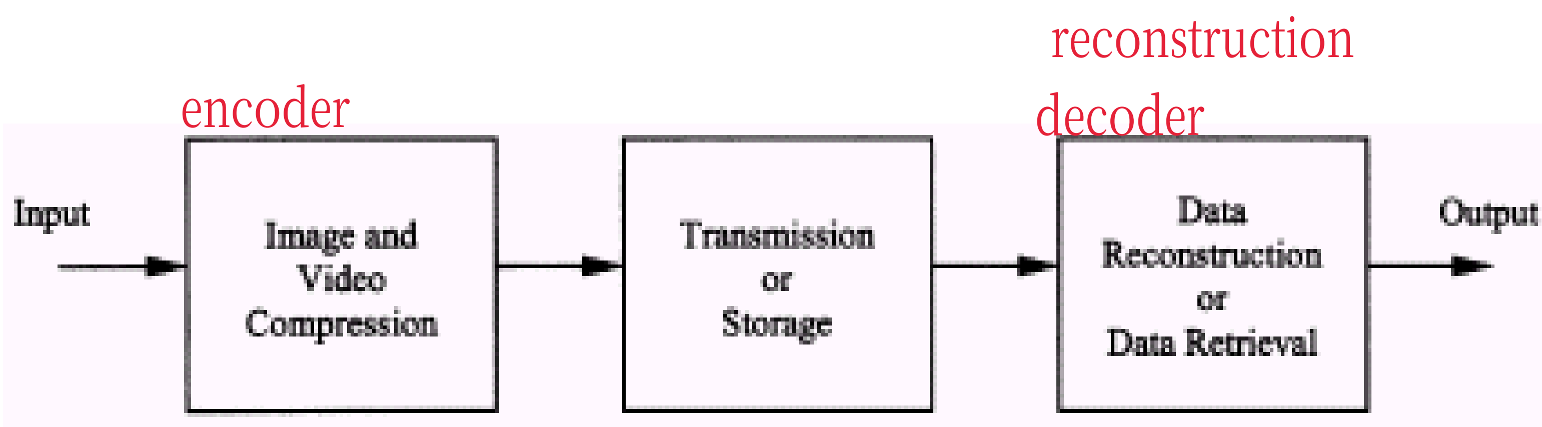



FIGURE 1.1 Image and video compression for visual transmission and storage.

**Figure source:**
**Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000**

# Lossy and Lossless Compression


- Lossless compression:
  - Use in important data (e.g. medical images)
  - Reconstructed image is identical to the original image after decompression.

- Lossy compression:
  - For media such as images or video, it is not necessary to display more information than the human ear or eye can perceive. Thus the compression techniques may discard data with little perceived difference by humans.
  - Reconstructed image is not identical to the original image after decompression.

# Transform-based Coding/Compression

# Transform Coding


- Why transform coding?
  - To convert the data into a form which is more suitable for compression.
- Transformation produces good properties for compression:
  - 促进
    Facilitate reduction of irrelevant information so that transform coefficients can be quantized according to their statistical properties.
  - Transformation offers energy compaction. 压缩 related to DCT
  - Transformation offers redundancy reduction (i.e., reduce the correlation between transform coefficients).
- A reversible process
  - Original signal can be obtained by applying the inverse transform.

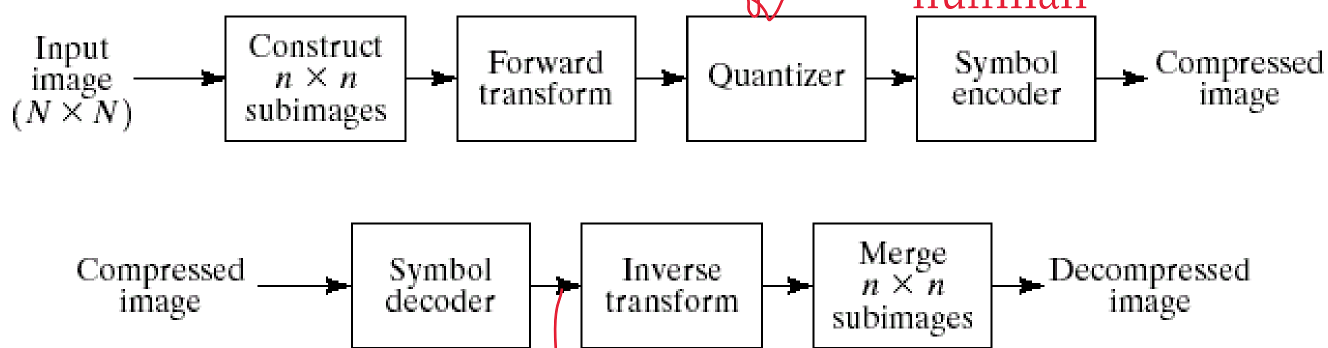
no information loss

# Transform-based Image Compression

digital represent finite pixel presentation   finite number of bits/ level (Quantization)

- A typical transform-based image compression system consists of a transformer, a quantizer, and a coder

apply VLC like huffman




should have de-Quantization

a
b

**FIGURE 8.28** A transform coding system: (a) encoder; (b) decoder.

Figure source:
R. C. Gonzalez and R. E. Woods, Digital Image Processing, 2nd edition, Prentice Hall, 2002.

# Transformer

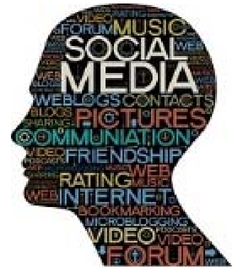

- Apply a one-to-one transformation to the input image data.
- The transformed output is in a representation which is more suitable for efficient compression than the raw image data.
- Unitary mappings such as DCT packs the signal energy into a small number of coefficients.
- Some well-known transforms:
  - Discrete Fourier Transform (DFT)
  - Discrete Cosine Transform (DCT)
  - Discrete Wavelet Transform (DWT)
  - Etc.

# Quantizer


- Generate a limited number of symbols from the transformed coefficients.
- An irreversible many-to-one mapping, causing information loss.
- It can be performed by scalar or vector quantizers.
  - Scalar quantization: element-by-element quantization of data. one-to-one
  - Vector quantization: quantization of a block (vector) at a time.

    block-by-block comparison

# Coder




- Assign a codeword or binary bitstream to each symbol at the output of the quantizer.
- May employ fixed-length or variable-length codes.
- Variable Length Coding (VLC) or entropy coding assigns codewords in such a way as to minimize the average length of the binary representation of the symbols.
- This is achieved by assigning shorter codewords to more probable symbols, which is the fundamental principle of entropy coding (e.g. Huffman coding).

# Popular Transforms


- Different transforms have their respective pros and cons.
- Some popular transforms:
  - Discrete Fourier Transform (DFT)
  - Discrete Cosine Transform (DCT)
  - Discrete Wavelet Transform (DWT)
  - Karhunen-Loeve Transform (KLT) achieve best compression
  - Etc.

# Discrete Cosine Transform (DCT)

# Discrete Cosine Transform (DCT)

$$s_{ij} \xrightarrow{\text{2D-DCT}} S_{uv}$$

$$S_{uv} = \alpha(u)\alpha(v)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} s_{ij} \cos\frac{(2i+1)u\pi}{2N} \cos\frac{(2j+1)v\pi}{2N} \qquad u, v = 0, \ldots, N-1$$

total have N*N coefficients

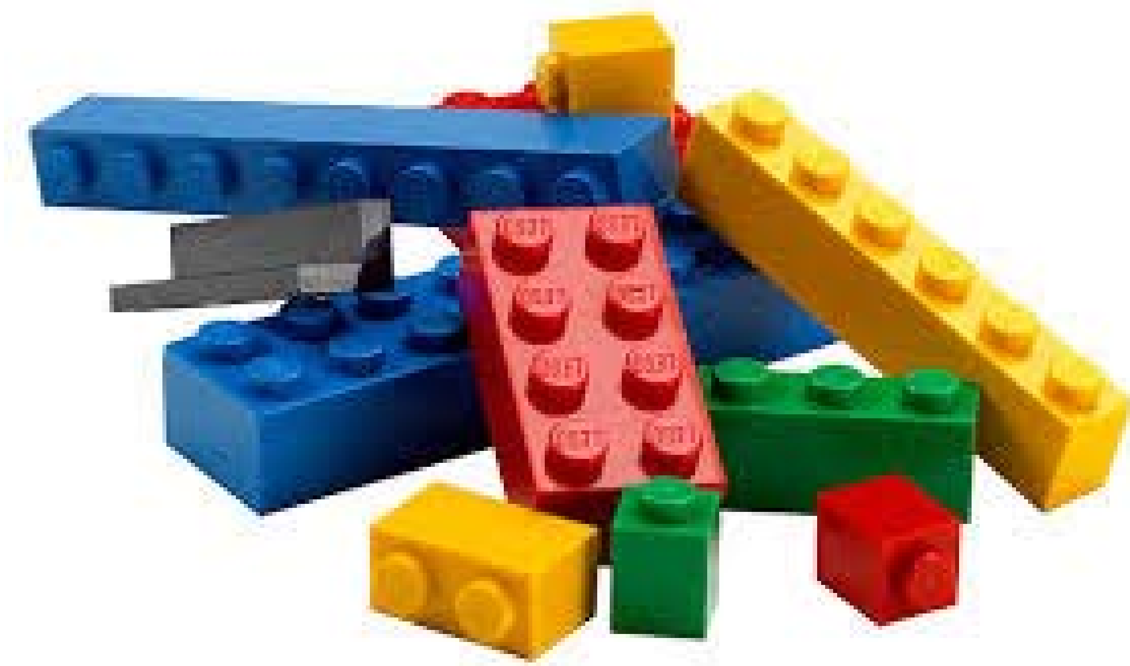
$$\alpha(k) = \begin{cases} \sqrt{\dfrac{1}{N}} & \text{for } k = 0 \\ \sqrt{\dfrac{2}{N}} & \text{for } k = 1, 2, \ldots, N-1 \end{cases}$$

# Why DCT?


- 2D-DCT is a popular transform used in transform-based image compression.
- It can offer the following:
  - Energy compaction for transform coefficients
  - Redundancy reduction amongst transform coefficients
- Pro: basis functions are fixed and not image-dependent.
- Con: compression is not as effective as some other transforms.

after 2D-DCT, redundancy reduced. cannot tell surrounding coef. based on one coeff.
for freq. change larger than original every freq. is useful, they are not similar

# “Lego Functions”



# Transform and Basis Functions


- A transform can often be described by a change of basis functions.
- A transform involves a change of perspective on the image signal.
- Basis functions are analogous to “Lego functions”

# Change of Basis for DCT

large coe. low freq.(freq. change slowly), use more bits to represent


small coe. here

after DCT ,cause good quality
called energy compaction

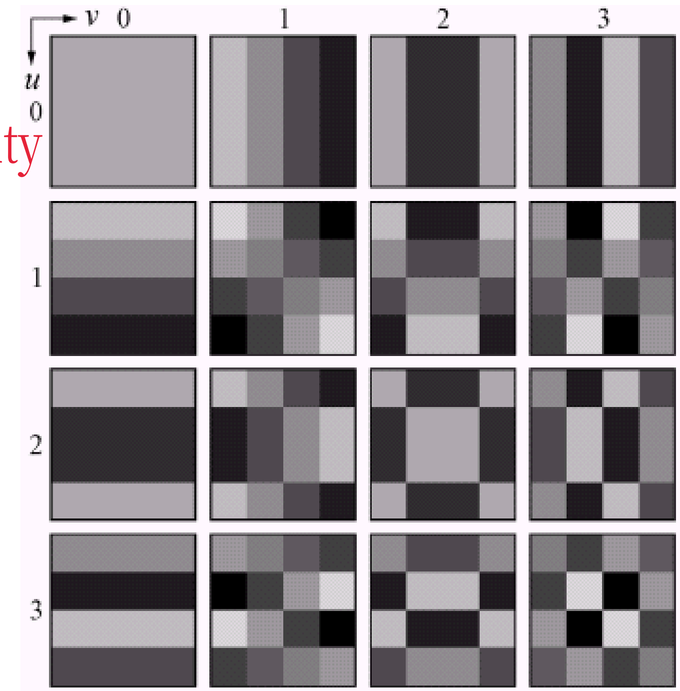



**FIGURE 8.30** Discrete-cosine basis functions for $N = 4$. The origin of each block is at its top left.

Figure source:
R. C. Gonzalez and R. E. Woods, Digital Image Processing, 2nd edition, Prentice Hall, 2002.

# Exercise 2.3: 2D-DCT


Two-dimensional Discrete Cosine Transform (2D-DCT) is a reversible transform used in the baseline JPEG image compression standard. The 2D-DCT of an N×N data matrix is given as

$$S_{uv} = \alpha(u)\alpha(v)\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} s_{ij} \cos\frac{(2i+1)u\pi}{2N}\cos\frac{(2j+1)v\pi}{2N} \qquad u, v = 0, \ldots, N-1$$

where

$$\alpha(k) = \begin{cases} \sqrt{\dfrac{1}{N}} & \text{for } k = 0 \\ \sqrt{\dfrac{2}{N}} & \text{for } k = 1, 2, \ldots, N-1 \end{cases}$$

(a) Show that the 2D-DCT transform can be computed using a two-stage 1D-DCT transform. Calculate the 2D-DCT transform of a 4×4 image block given as below:

$$s = \begin{bmatrix} 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \\ 20 & 20 & 10 & 10 \end{bmatrix}$$

(12 marks)

# Solution

看纸质课件

# Karhunen-Loeve Transform (KLT)

# Karhunen-Loeve Transform (KLT)

- An important method in transform-based image compression.
- Rotate the ambient coordinates of the data to reveal subspace in which the data resides.
- Also known as principal component analysis (PCA), eigenvalue decomposition (EVD).

# Pros and Cons

- Pros:
  - Optimal in terms of energy compaction, coefficient decorrelation and dimensionality reduction.
- Cons:
  - image-dependent, i.e. different images may require different basis functions.
  - requires estimation of image covariance matrix.
  - computationally intensive as it requires eigen-decomposition.

# KLT Procedure (I)

1. Partition an image into $N \times N$ image blocks/subimages.
2. Represent each $N \times N$ block lexicographically to form vector $\boldsymbol{x}_{ni}$
3. Perform zero-mean centering:

$$\boldsymbol{x} = \boldsymbol{x}_{ni} - \overline{\boldsymbol{x}}_n$$

4. Compute the covariance matrix:

$$\boldsymbol{C} = E[\boldsymbol{x}\boldsymbol{x}^T] = E[(\boldsymbol{x}_{ni} - \overline{\boldsymbol{x}}_n)(\boldsymbol{x}_{ni} - \overline{\boldsymbol{x}}_n)^T]$$

In practice, this is estimated using sample covariance matrix:

$$\boldsymbol{C} = \frac{1}{N_T - 1} \sum_{i=1}^{N_T} (\boldsymbol{x}_{ni} - \overline{\boldsymbol{x}}_n)(\boldsymbol{x}_{ni} - \overline{\boldsymbol{x}}_n)^T$$

$$\overline{\boldsymbol{x}}_n = \frac{1}{N_T} \sum_{i=1}^{N_T} \boldsymbol{x}_{ni}$$

# KLT Procedure (II)

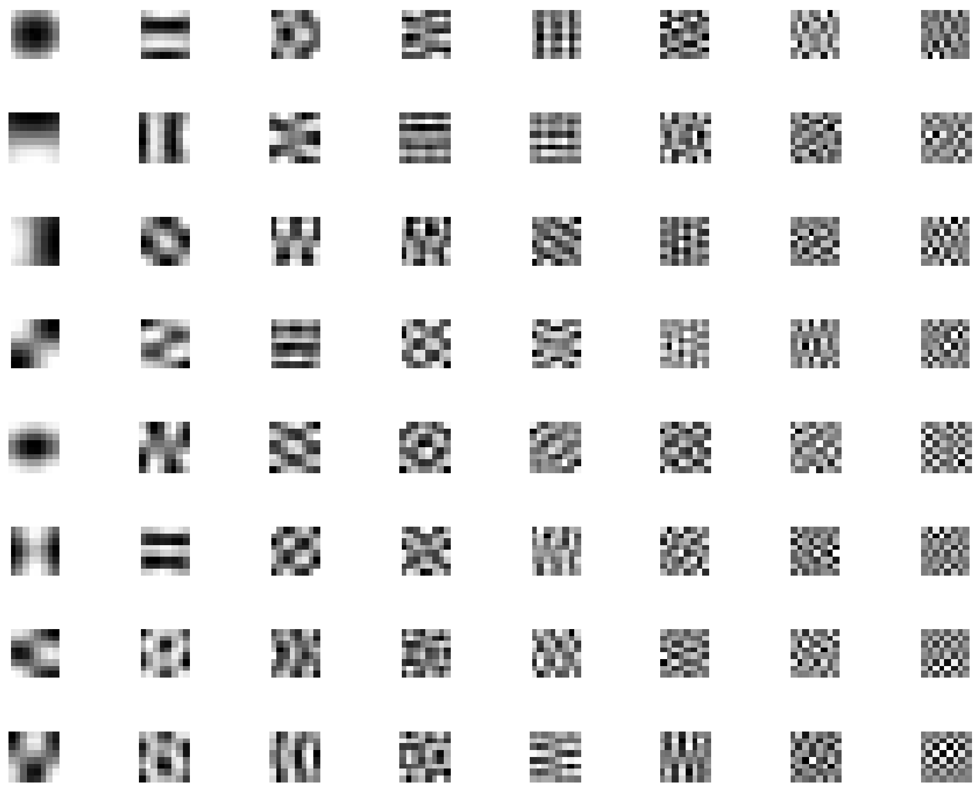
5. Perform eigen-decomposition on matrix $\boldsymbol{C}$

$$\boldsymbol{C}\boldsymbol{p}_i = \lambda_i \boldsymbol{p}_i, \qquad i = 1,\ 2,\ \ldots,\ M \ \text{ where } M = N^2$$

6. The KLT basis functions are the eigenvectors $\boldsymbol{p}_i$ of the covariance matrix $\boldsymbol{C}$.
7. The dominant basis functions/images are those eigenvectors with large eigenvalues.

Note that the eigenvectors $\boldsymbol{p}_i$ can be mapped into basis functions/images through lexicographical ordering.

# Sample KLT Basis Functions/ Images



Source: https://www.slideserve.com/dore/direction-adaptive-klt-for-image-compression

# Synthesis & Analysis

- Synthesis

$$\boldsymbol{x} = \sum_{i=1}^{M} c_i \boldsymbol{p}_i$$

- Analysis

$$c_i = \boldsymbol{p}_i^T \boldsymbol{x}, \qquad i = 1,\ 2,\ \ldots,\ M$$

- Note:

$$E[c_i c_j] = \begin{cases} \lambda_i & i = j \\ 0 & i \neq j \end{cases}$$

# Low-Rank Modeling

- Assuming that the eigenvalues are all distinct and arranged in decreasing order, namely:

$$\lambda_1 > \lambda_2 > \ldots > \lambda_M$$

- The data vector $\boldsymbol{x}$ can be approximated using low-rank model as

$$\hat{\boldsymbol{x}} = \sum_{i=1}^{r} c_i \boldsymbol{p}_i, \quad \text{where } r < M$$

- The transformation is also known as subspace decomposition.

# Exercise 2.4: KLT

In this question, Karhunen-Loeve Transform (KLT) is used to perform image compression. An 18×18 image is partitioned into eighty-one 2×2 subimages, and ordered lexicographically to form eighty-one 4×1 vectors. The covariance matrix of the vectors is given by:

$$\boldsymbol{C} = \begin{bmatrix} 5 & 5.5 & 4.5 & 7.5 \\ 5.5 & 7.25 & 8.25 & 10.5 \\ 4.5 & 8.25 & 14.25 & 15 \\ 7.5 & 10.5 & 15 & 21 \end{bmatrix}$$

(a) With the aid of a diagram, explain briefly the main components involved in a transform-based image compression scheme. Your answer should clearly outline the objective of each component in the scheme.

(6 marks)

# Exercise 2.4: KLT

(b) The covariance matrix $C$ has four eigenvectors given by:

$$\boldsymbol{v}_1 = \begin{bmatrix} 0.6454 \\ -0.6916 \\ 0.3071 \\ -0.1043 \end{bmatrix}, \quad \boldsymbol{v}_2 = \begin{bmatrix} 0.1704 \\ 0.4808 \\ 0.4828 \\ -0.7118 \end{bmatrix}, \quad \boldsymbol{v}_3 = \begin{bmatrix} 0.6946 \\ 0.3752 \\ -0.6138 \\ 0.0034 \end{bmatrix}, \quad \boldsymbol{v}_4 = \begin{bmatrix} 0.2681 \\ 0.3871 \\ 0.5439 \\ 0.6946 \end{bmatrix}$$

Calculate the corresponding eigenvalues for $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{v}_3$ and $\boldsymbol{v}_4$.

(4 marks)

# Exercise 2.4: KLT

(c) Determine the suitable number of principal components (basis images) that should be used to represent the 2×2 subimages in this image.

A 2×2 subimage after zero-mean centering is given by:

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

Based on the number of chosen principal components, calculate the square error between the original subimage and its representation.

(7 marks)

(d) Discuss briefly the limitations of KLT in practical image compression schemes.

(3 marks)

# Solution

# Differential/Predictive Coding

# Differential/Predictive Coding

- In a small image neighborhood, pixel values tend to be similar.
- Hence, instead of coding the actual pixel values, it is more efficient to code the difference between pixel values, namely, the difference between the actual pixel value and its predicted value is encoded.
- Suitable for signals in which successive samples are similar, e.g. video (successive frames are similar), audio (consecutive samples are similar).

# Motivation

a
b c

**FIGURE 8.20**
(a) The prediction error image resulting from Eq. (8.4-9).
(b) Gray-level histogram of the original image.
(c) Histogram of the prediction error.

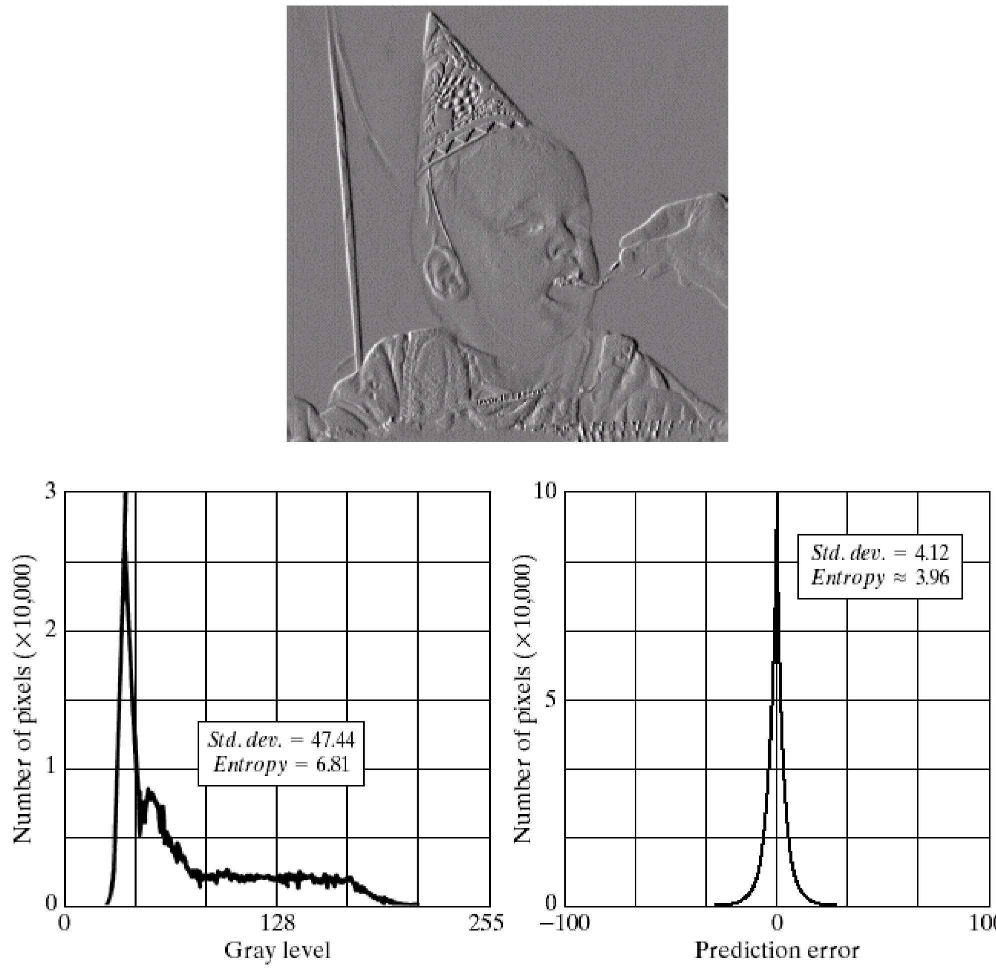



Figure source:
R. C. Gonzalez and R. E. Woods, Digital Image Processing, 2nd edition, Prentice Hall, 2002.

# Predictive Coding

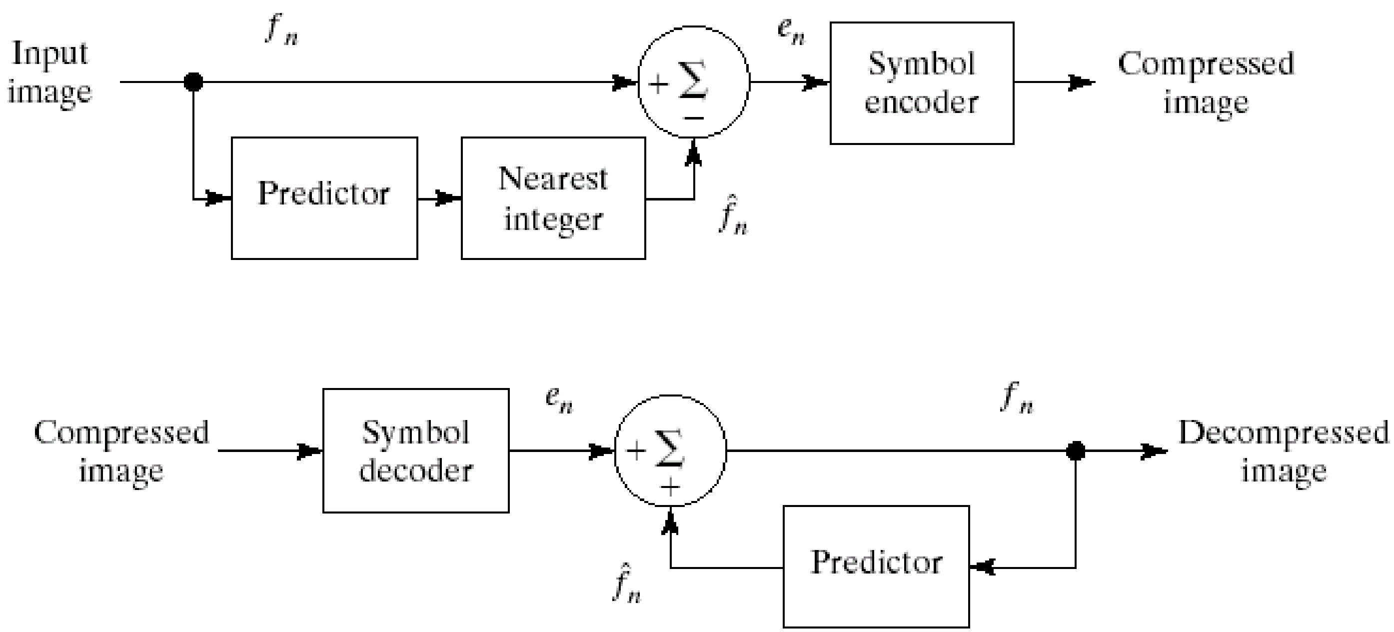



**FIGURE 8.19** A lossless predictive coding model: (a) encoder; (b) decoder.

Figure source:
R. C. Gonzalez and R. E. Woods, Digital Image Processing, 2nd edition, Prentice Hall, 2002.

# Encoder

- A predictor uses neighboring pixel(s) to predict the current/actual pixel to be encoded.
- The predicted value is rounded to nearest integer.
- The prediction error (actual value – predicted value) is computed and coded using a symbol encoder (e.g. Huffman coder).
- The output is stored as compressed bitstream.

# Decoder

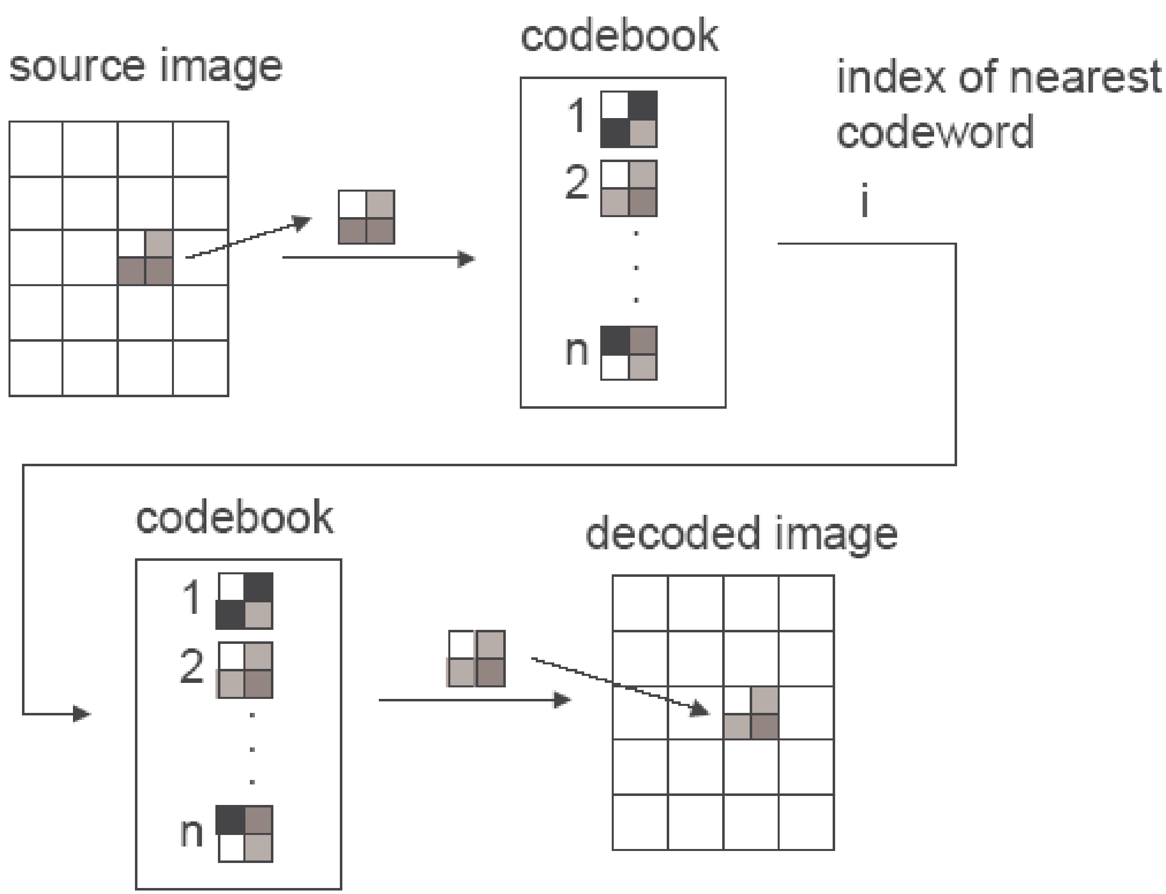
- The compressed bitstream undergoes symbol decoding to obtain the prediction error.
- A predictor uses neighboring pixel(s) from decoded image so far to predict the current pixel value.
- The prediction error is then added to predicted value to reconstruct the current pixel value.
- The decompressed/reconstructed image is formed pixel-by-pixel.

# Vector Quantization (VQ)

# Motivation

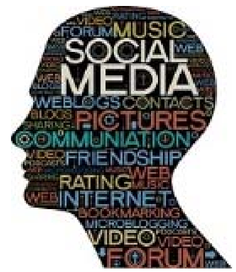

- Pixels within a small image block are usually correlated.
- Hence, instead of coding every pixel in the block, we can represent each image block with some commonly seen blocks, and refer to them through their indices.
- Commonly seen blocks: codewords or codevectors
- Collection of these codewords = codebook

# Vector Quantization Overview




Source: Alex Mohr , SUNY

# Encoder




- The image is partitioned into blocks (often square blocks).
- These blocks are also called vectors (due to lexicographical ordering).
- A table called codebook is used to find the best-match codeword (pattern) for each of the blocks.
- The indices for these block are stored into compressed bitstream.
- As the codebook is present at both the encoder and decoder, only indices need to be stored.
- When no exact match is found in the codebook, the block is coded with the nearest pattern, and this will appear as distortion in the decoded image at the decoder.

# Decoder


- The indices of the codewords are extracted from the compressed bitstream.
- Using the codebook, the codewords (patterns) are re-assembled to reconstruct the image.

# Design Issues


- Block size
- Number of codevectors
- Training data
- Distance measure: e.g. Euclidean distance
- Codebook design

# Exercise 2.5: VQ

In a simple image compression scheme, vector quantization (VQ) is used to compress a class of 2-bit grayscale images, with gray levels of 0, 1, 2 and 3. A given image is partitioned into $2 \times 2$ blocks (subimages), and encoded using $2 \times 2$ codevectors from a generated codebook. The codebook used in this VQ scheme, together with the associated probabilities of occurrences for the codevectors, is given as follows:

| Symbols | $s_0$ | $s_1$ | $s_2$ | $s_3$ | $s_4$ | $s_5$ | $s_6$ | $s_7$ |
|---|---|---|---|---|---|---|---|---|
| Probabilities of occurrences | 0.2 | 0.2 | 0.15 | 0.15 | 0.1 | 0.1 | 0.05 | 0.05 |
| Codevectors | $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$ | $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ | $\begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$ | $\begin{bmatrix} 3 & 3 \\ 3 & 3 \end{bmatrix}$ | $\begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}$ | $\begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$ | $\begin{bmatrix} 0 & 0 \\ 2 & 2 \end{bmatrix}$ | $\begin{bmatrix} 2 & 2 \\ 0 & 0 \end{bmatrix}$ |

# Exercise 2.5: VQ


(a) Explain briefly the main steps involved in a VQ-based image compression scheme. Discuss various issues involved in the generation of the codebook.

(6 marks)

(b) Huffman coding is used to encode the symbols or indices of the codevectors. Design a suitable set of Huffman codewords to encode the symbols based on the probability distribution given in the table above. Your answer should clearly outline all the steps and calculations involved.

(6 marks)

(c) Based on the designed Huffman codeword set in part 1(b), compute the average number of bits that is required to represent a symbol. Hence, calculate the average compression ratio of this VQ scheme.

(4 marks)

# Exercise 2.5: VQ


(d) The VQ scheme is used to compress an image given as follows:

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 2 |
| 1 | 2 | 2 | 2 | 3 | 3 |
| 2 | 2 | 2 | 2 | 2 | 3 |

Given that the Euclidean distance is used as the distance metric, show the reconstructed image after decompression, and calculate the square error for the reconstructed image.

(4 marks)

# Solution

# Singular Value Decomposition (SVD)

# Singular Value Decomposition (SVD)


- An important tool in linear algebra and signal processing
- SVD of an mxn matrix $\mathbf{A}$:

$$\mathbf{A} = \mathbf{U\Sigma V}^T$$

U is an mxm orthogonal matrix

V is an nxn orthogonal matrix

$\mathbf{\Sigma} = diag(\sigma_1, \sigma_2, ..., \sigma_r)$, an $\mathrm{m} \times \mathrm{n}$ matrix

with $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r \geq 0$ and r is the rank of the matrix

# How to perform SVD?

- **U** is an mxm orthogonal matrix whose column vectors are formed by eigenvectors of $\mathbf{AA}^{\mathrm{T}}$, normalized to unit length, and ordered based on decreasing corresponding eigenvalues.
- **V** is an nxn orthogonal matrix whose column vectors are formed by eigenvectors of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$, normalized to unit length, and ordered based on decreasing corresponding eigenvalues.
- **Σ** is an mxn matrix where the singular values

$$\sigma_i = \sqrt{\lambda_i}, \ i = 1, 2, \ldots, r$$

$\lambda_i, i = 1, 2, \ldots, r$ are nonzero eigenvalues of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ or $\mathbf{AA}^{\mathrm{T}}$ in decreasing value.

# Selected SVD Properties

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$$

$$\mathbf{A}\mathbf{v}_i = \sigma_i \mathbf{u}_i \quad i = 1, \ldots, r$$

$$\mathbf{A}^T\mathbf{u}_i = \sigma_i \mathbf{v}_i \quad i = 1, \ldots, r$$

# Exercise 2.6: SVD

Find the SVD of

$$\mathbf{A} = \begin{bmatrix} \sqrt{2} & 0.5 & 0.5 \\ \sqrt{2} & -0.5 & -0.5 \end{bmatrix}$$

# Solution

# Low Rank Modeling

- Original SVD:

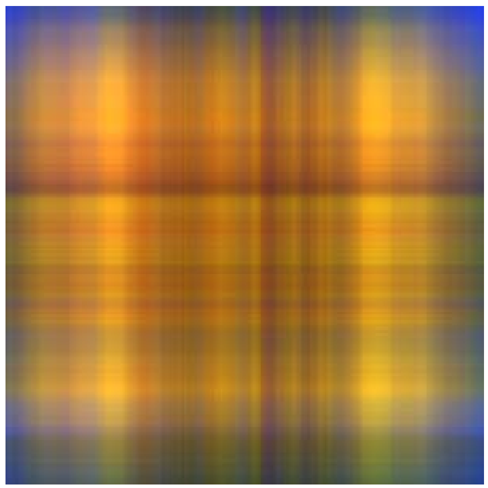
$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

- Low-rank modeling:

$$\hat{\mathbf{A}} = \sum_{i=1}^{p} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \qquad \text{where } p < r$$

# Image Compression Using SVD

Original image



p=10

p=20

p=80

Picture source: http://www.scenicnursery.com/archives/sunflower.jpg

# Exercise 2.7: SVD and Compression

In a simple image compression scheme, Singular Value Decomposition (SVD) is used to compress an image $\mathbf{A}$ given as:

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 3 \\ 4 & 2 & 4 \end{bmatrix}$$

The SVD of $\mathbf{A}$ is given as:

$$\mathbf{A} = \mathbf{U\Sigma V}^{\mathrm{T}}$$

(a) Write down the characteristic equation used to compute the singular values in the matrix $\mathbf{\Sigma}$. Hence, show that the singular values are given by $\sigma_1 = 7.0425$ and $\sigma_2 = 0.6350$.

(5 marks)

# Exercise 2.7: SVD and Compression

(b) Find the matrix $\mathbf{U}$ in the SVD of $\mathbf{A}$. Given that the matrix $\mathbf{V}$ is given by

$$\begin{bmatrix} 0.6325 & -0.6325 & 0.4472 \\ 0.3162 & -0.3162 & -0.8944 \\ 0.7071 & 0.7071 & 0 \end{bmatrix},$$

verify that the SVD of $\mathbf{A}$ is given by $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}}$.

(6 marks)

# Exercise 2.7: SVD and Compression

(c) Determine a suitable rank to be used in the low-rank representation of image block **A**, and justify your answer. Based on the chosen rank, calculate the total squared error between the original image **A** and its low-rank representation.

(5 marks)

# Solution

# JPEG

# JPEG


- JPEG is a popular image compression standard.
- Address both lossy and lossless image compression.
- Compression ratio ranges from 10:1 to 20:1.
- Four modes of operation:
  - sequential DCT-based mode
  - progressive DCT-based mode
  - lossless mode
  - hierarchical mode
- We will focus on sequential DCT-based mode or baseline JPEG as it is the most popular mode in JPEG.

# Baseline JPEG In A Slide

- Main idea of baseline JPEG:
  - Image contents change slowly across images

    (this means that lower spatial frequency components are more

    strong dominant than higher frequency components).
  - Humans are more sensitive to lower frequency components.
  - JPEG uses the frequency analysis (DCT) to perform energy compaction, emphasize on low frequency information, and then performs compression using entropy coding.

# Main Stages in Baseline JPEG


- Image/block processing
- Forward DCT
- Quantization
- Entropy encoding
- Frame building

# JPEG Encoder

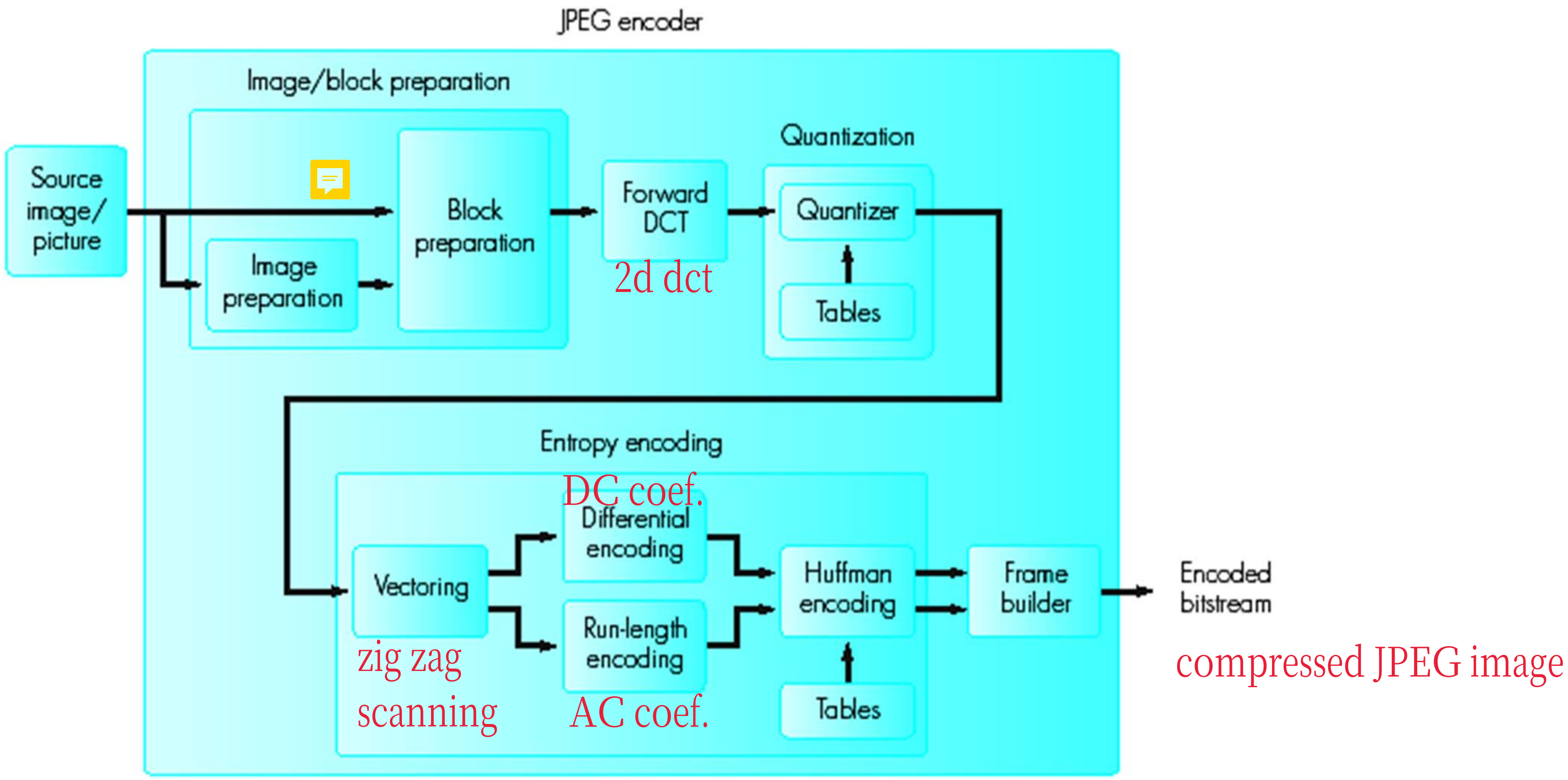



Figure source:
Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

# Image Partitioning

- Image is first partitioned into numerous 8x8 pixel blocks.

32 pixel by 32 pixel image  produce 16 8 by 8 pixel block

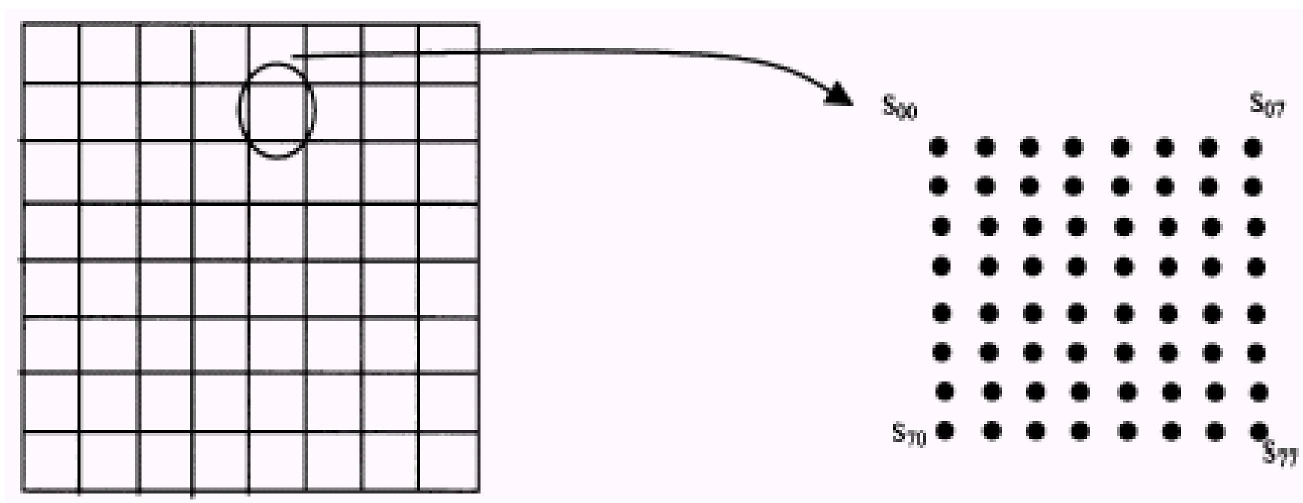
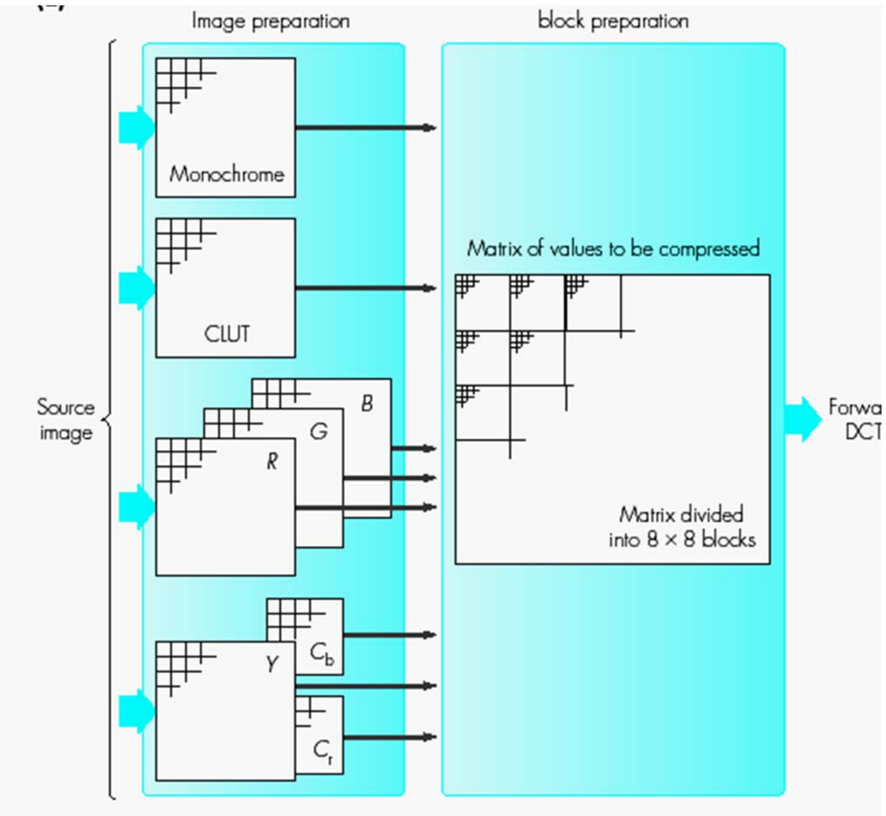



**FIGURE 7.4** Partitioning to 8 × 8 blocks.

Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

# Image Preparation



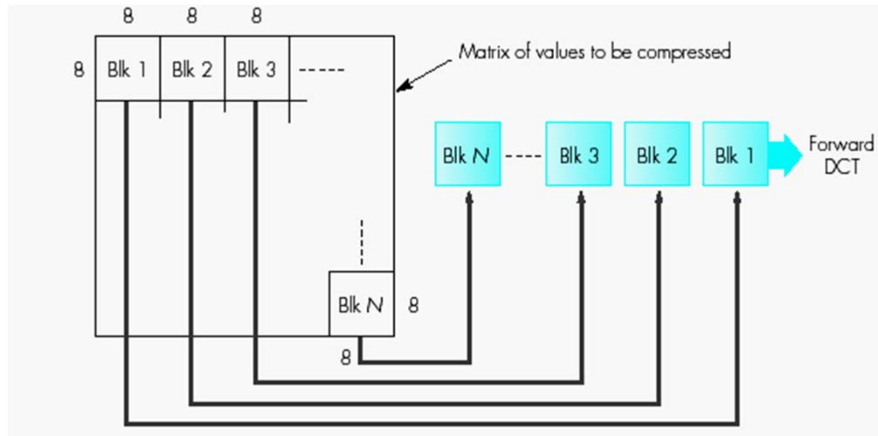




Figure source:
Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

# Forward DCT


For JPEG

- Forward 2D-DCT is applied to each 8x8 pixel block.
- Forward and inverse 2D-DCT of 8x8 block are given as follows:

$$\text{FDCT:} \quad S_{uv} = \frac{1}{4} C_u C_v \sum_{i=0}^{7} \sum_{j=0}^{7} s_{ij} \cos\frac{(2i+1)u\pi}{16} \cos\frac{(2j+1)v\pi}{16}$$

$$\text{IDCT:} \quad s_{ij} = \frac{1}{4} \sum_{u=0}^{7} \sum_{v=0}^{7} C_u C_v S_{uv} \cos\frac{(2i+1)u\pi}{16} \cos\frac{(2j+1)v\pi}{16}$$

$$C_u C_v = \begin{cases} \frac{1}{\sqrt{2}} & for\ u, v = 0 \\ 1 & otherwise \end{cases}$$

where $s_{ij}$ is the value of the pixel at position $(i,j)$ in the block, and $S_{uv}$ is the transformed $(u,v)$ DCT coefficient.

Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

# Recap: Why 2D-DCT?


- To convert image data into a form which is more suitable for compression.
- 2D-DCT yields energy compaction.
- 2D-DCT reduces the correlation between transform coefficients (redundancy reduction).

# Recap: Change of Basis in DCT

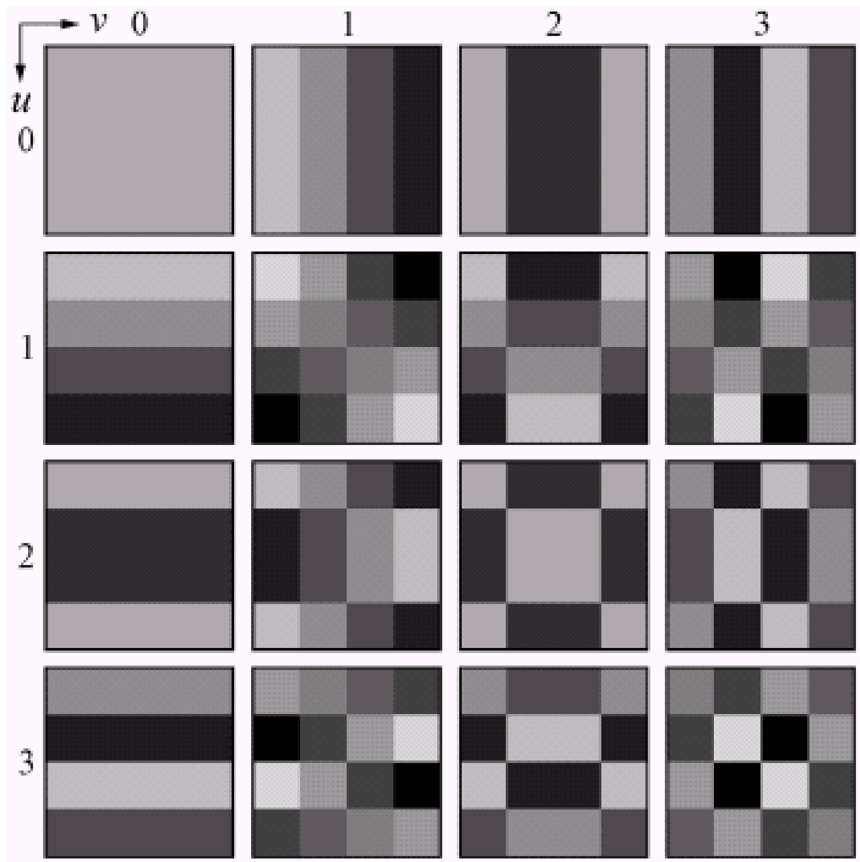
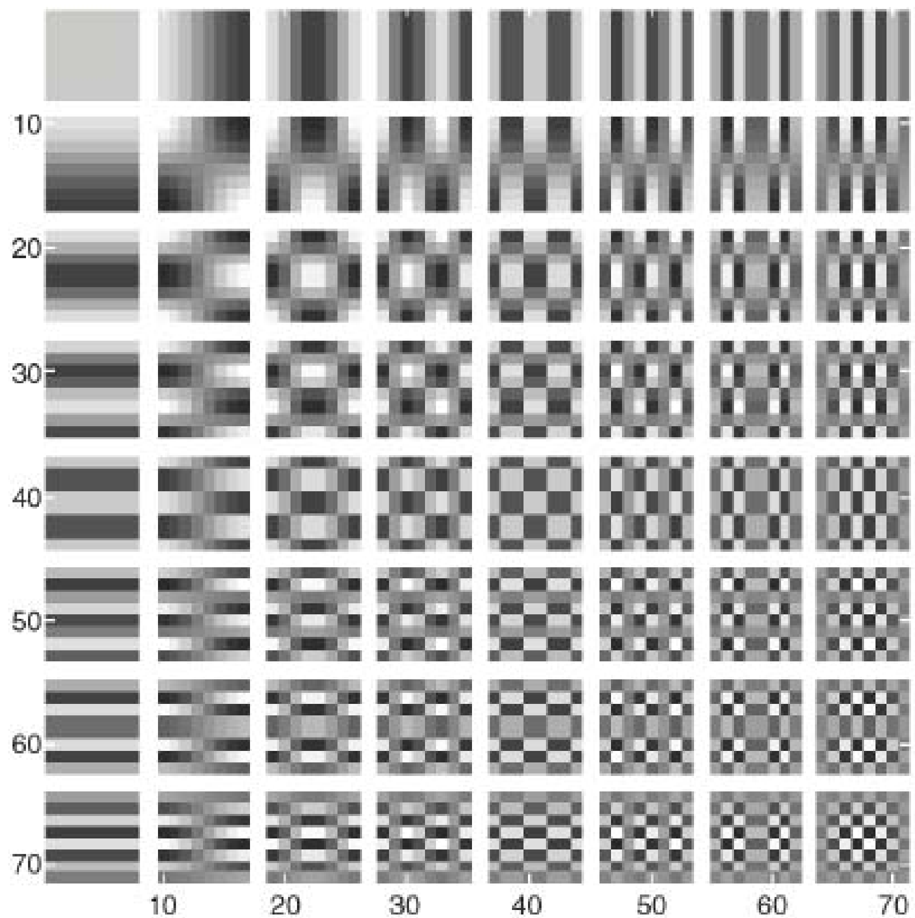



**FIGURE 8.30** Discrete-cosine basis functions for $N = 4$. The origin of each block is at its top left.

# JPEG DCT Basis Function

color white, value large

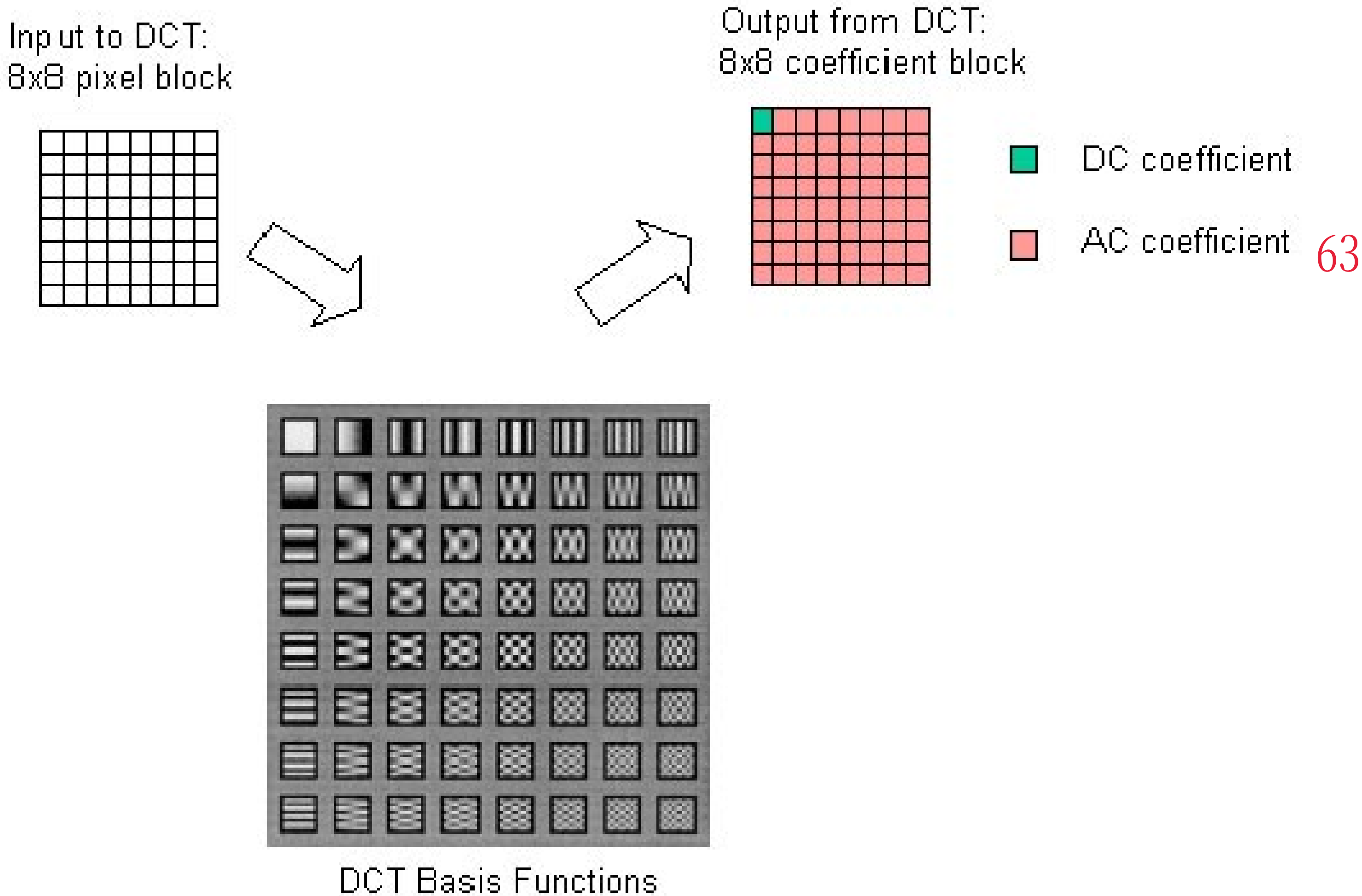



change from slow to fast,横向看

Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

**FIGURE 4.6** When N = 8: a set of the 64 basis images of the DCT.

# DCT & Change of Basis (I)




**Source: eetimes.com**

# DCT & Change of Basis (II)

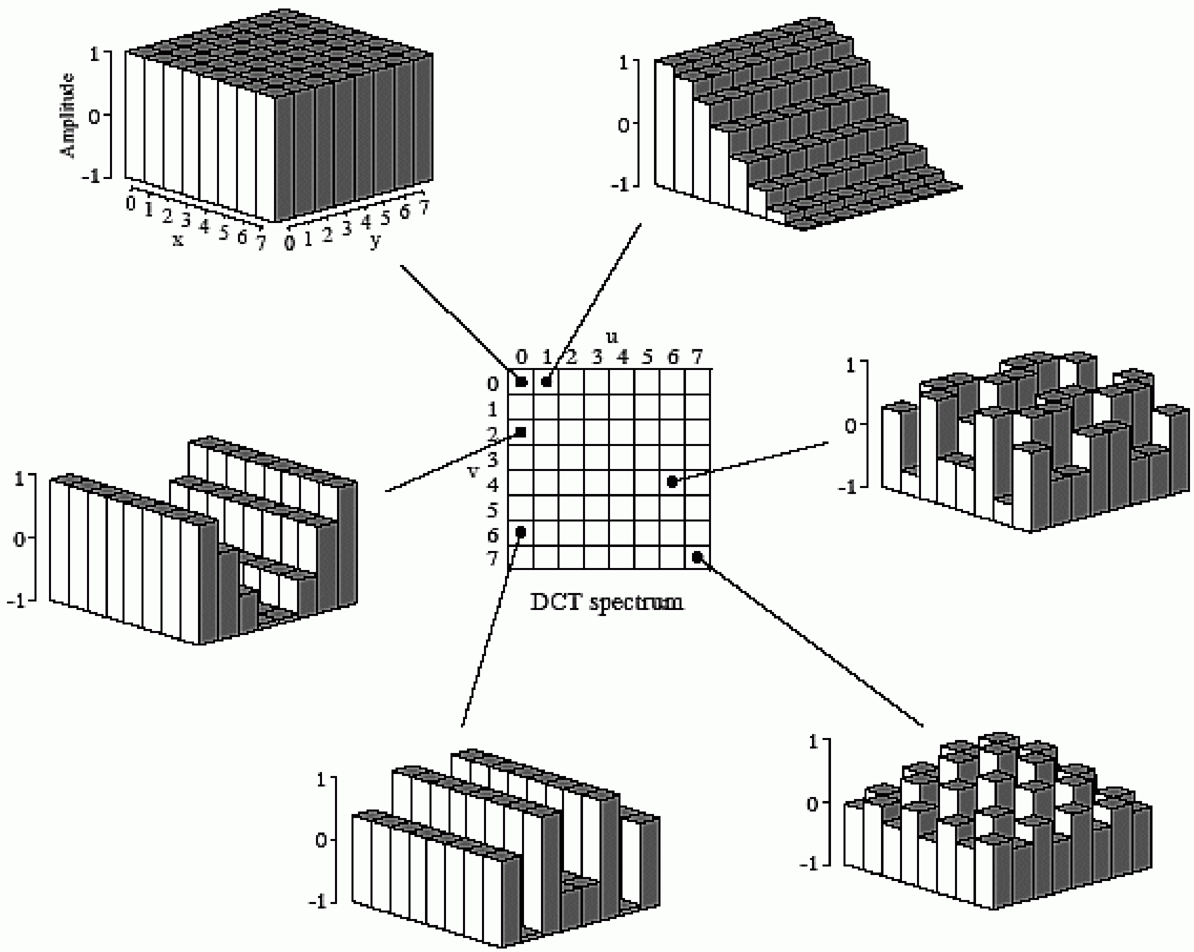
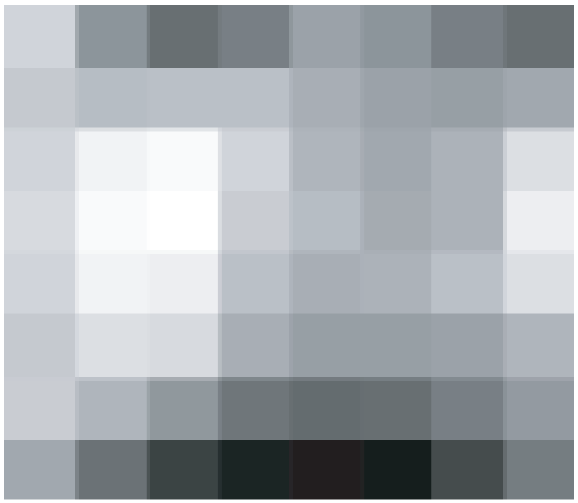



FIGURE 27-10
The DCT basis functions. The DCT spectrum consists of an 8×8 array, with each element in the array being an amplitude of one of the 64 basis functions. Six of these basis functions are shown here, referenced to where the corresponding amplitude resides.

**Source: dspguide.com**

# DCT & Change of Basis (III)



PICTURE MATRIX

$$\begin{bmatrix} 40 & 24 & 15 & 19 & 28 & 24 & 19 & 15 \\ 38 & 34 & 35 & 35 & 31 & 28 & 27 & 29 \\ 40 & 47 & 49 & 40 & 33 & 29 & 32 & 43 \\ 42 & 49 & 50 & 39 & 34 & 30 & 32 & 46 \\ 40 & 47 & 46 & 35 & 31 & 32 & 35 & 43 \\ 38 & 43 & 42 & 31 & 27 & 27 & 28 & 33 \\ 39 & 33 & 25 & 17 & 14 & 15 & 19 & 26 \\ 29 & 16 & 6 & 1 & -4 & 0 & 7 & 18 \end{bmatrix}$$

DCT COEFFICENTS

$$\begin{bmatrix} 239 & 32 & 27 & -12 & 3 & -5 & 3 & 1 \\ 34 & -3 & -19 & 6 & 3 & 0 & -1 & 1 \\ -70 & 2 & 8 & 23 & 9 & 6 & -1 & -1 \\ 5 & 0 & -6 & 11 & -2 & 0 & -1 & 1 \\ -17 & -3 & 6 & 6 & 3 & -1 & 0 & 0 \\ 2 & 4 & 2 & 2 & 1 & -2 & 0 & 1 \\ -3 & 0 & 0 & -1 & -1 & -1 & 0 & 0 \\ 1 & -1 & 3 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

**Source: cnx.org**

# DCT Coefficients

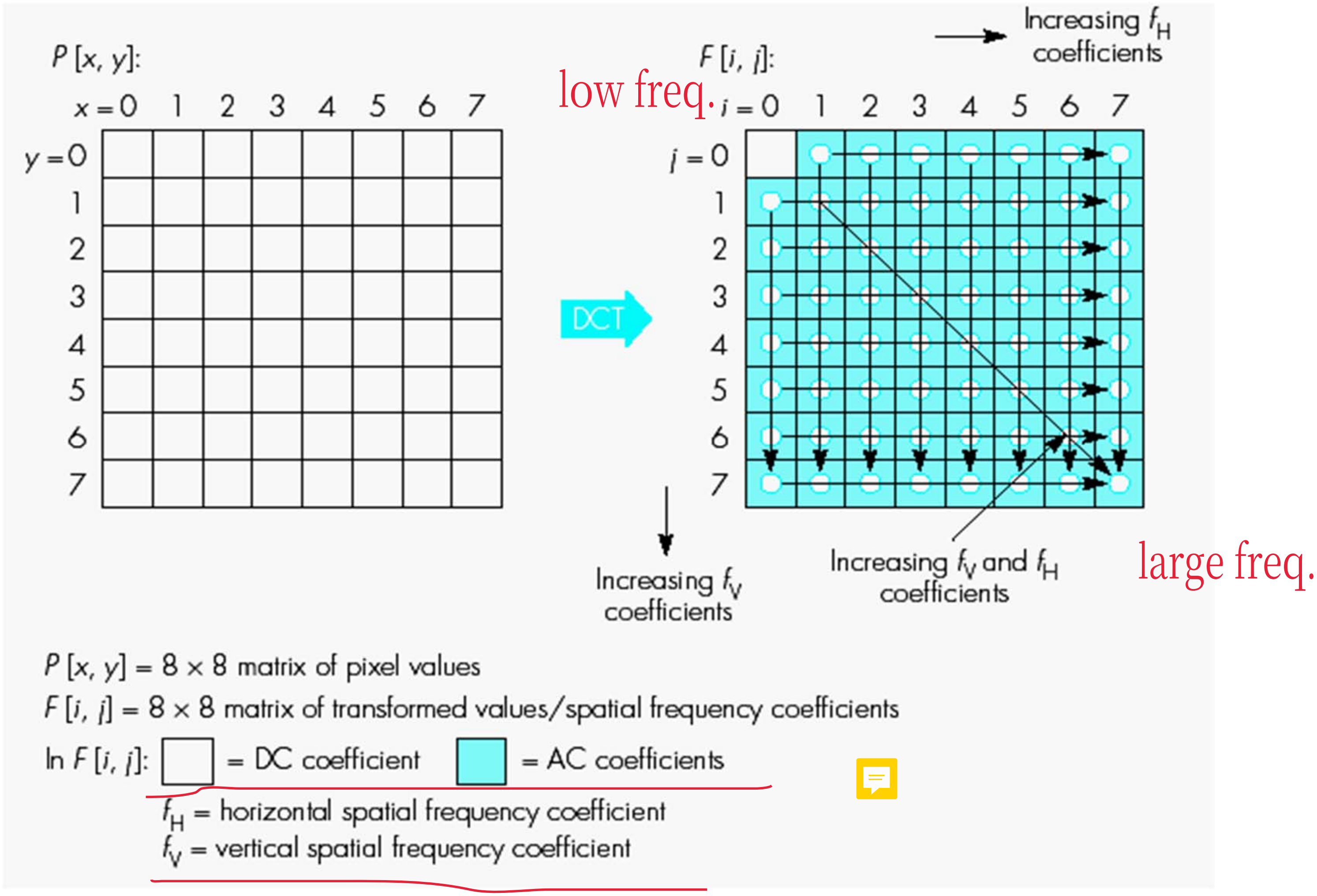



Figure source:
Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

# Quantization


finite number of levels

- As a finite number of bits is used to represent coefficient values, quantization is required to map the continuous value into discrete value.
- The coefficients are quantized using quantization tables.
- Information loss occurs during quantization.

# Quantization Tables


- Human is more sensitive to DC and low AC coefficients.
- Hence, more emphasis/care should be taken in quantizing them.
- Smaller step sizes (quantization values) are used to quantize DC and low AC coefficients to reduce quantization error.
- Step sizes/values in quantization table are a compromise between level of compression and information loss.

# Quantization

- Quantization of the transform DCT coefficients is given by:

$$S_{quv} = round\left(\frac{S_{uv}}{Q_{uv}}\right)$$

where the $S_{quv}$ is the quantized DCT coefficient, $S_{uv}$ is the DCT coefficient and $Q_{uv}$ is the quantization step size.

**TABLE 7.1**
**Two Examples of Quantization Tables Used by JPEG**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 | 120 | 101 |
| 72 | 92 | 95 | 98 | 112 | 100 | 103 | 99 |

Luminance quantization table

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |

Chrominance quantization table

Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

# Quantization

DCT coefficients

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 120 | 60 | 40 | 30 | 4 | 3 | 0 | 0 |
| 70 | 48 | 32 | 3 | 4 | 1 | 0 | 0 |
| 50 | 36 | 4 | 4 | 2 | 0 | 0 | 0 |
| 40 | 4 | 5 | 1 | 1 | 0 | 0 | 0 |
| 5 | 4 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Quantizer

Quantized coefficients

| | | | | | | |
|---|---|---|---|---|---|---|
| 12 | 6 | 3 | 2 | 0 | 0 | 0 |
| 7 | 3 | 2 | 0 | 0 | 0 | 0 |
| 3 | 2 | 0 | 0 | 0 | 0 | 0 |
| 2 | 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 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 10 | 10 | 15 | 20 | 25 | 30 | 35 | 40 |
| 10 | 15 | 20 | 25 | 30 | 35 | 40 | 50 |
| 15 | .20 | 25 | 30 | 35 | 40 | 50 | 60 |
| 20 | 25 | 30 | 35 | 40 | 50 | 60 | 70 |
| 25 | 30 | 35 | 40 | 50 | 60 | 70 | 80 |
| 30 | 35 | 40 | 50 | 60 | 70 | 80 | 90 |
| 35 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
| 40 | 50 | 60 | 70 | 80 | 90 | 100 | 110 |

Quantization table

Figure source:
Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

# Entropy Encoding


- Entropy encoding comprises 4 steps:
  - Vectoring (zig-zag scanning)
  - Differential Encoding
  - Run-length encoding
  - Huffman encoding

- Vectoring
  - Zig-zag scanning
  - Represent the quantized 2D matrix in 1D vector.
  - Aim to exploit the presence of large number of zeros in the quantized matrix.

# Zig-zag Scanning

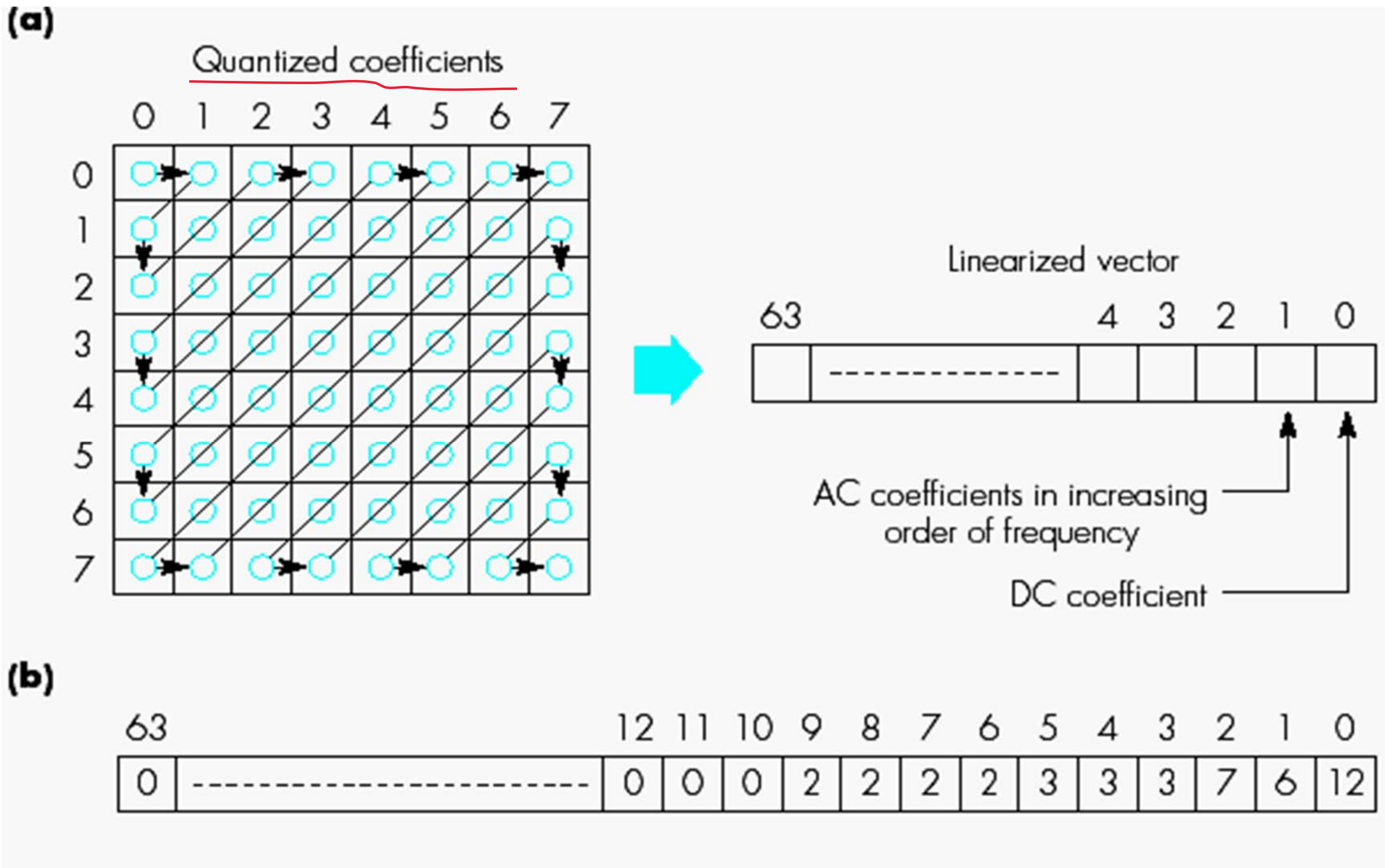



Figure source:
Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

# Differential Encoding of DC Coefficients

- DC coefficient is treated separately from the other 63 AC coefficients.
- DC coefficient reflects the average intensity of the pixel block.
- DC coefficients are encoded using differential/predictive coding.
- The difference (prediction error) between DC coefficients of current block and previous block is encoded.
- Differential coding is used as average intensity between 2 consecutive blocks is similar.

# Run-length Coding


- The AC coefficients are encoded using run-length coding.
- Run-length coding exploits the presence of a large number of zeros in the AC values by trying to group zeros together.
- Run-length coding syntax uses a string made up of pairs of values.
- Each pair has (skip, value) format:
  - Skip – number of zeros in the run
  - Value – the next non-zero coefficient

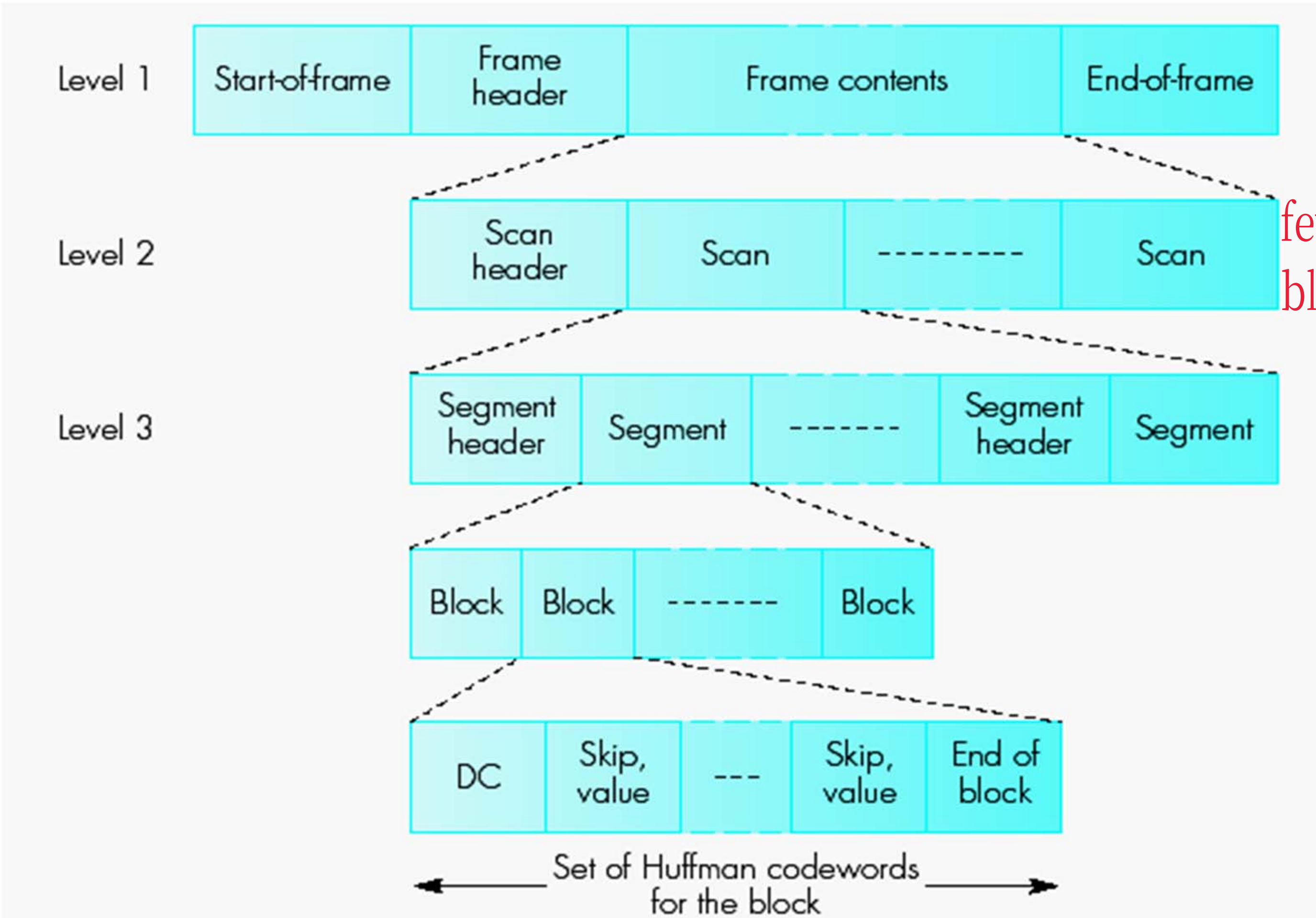
# Huffman Encoding


- After differential coding and run-length coding, some symbols/patterns occur more often than the others.
- Hence, Huffman coding is used to compress these symbols.
- High compression can be achieved by replacing frequent patterns with shorter codewords, and vice versa.
- Huffman coding is used in both differential coding and run-length coding.

# Frame Building

- JPEG define sythax for bitstream relating to image/frame.
- The role of frame builder is to encapsulate all the information relating to an encoded image in this format.

# JPEG Bitstream Format




few rows of 8 by 8 block

Figure source:
Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

# JPEG Decoder

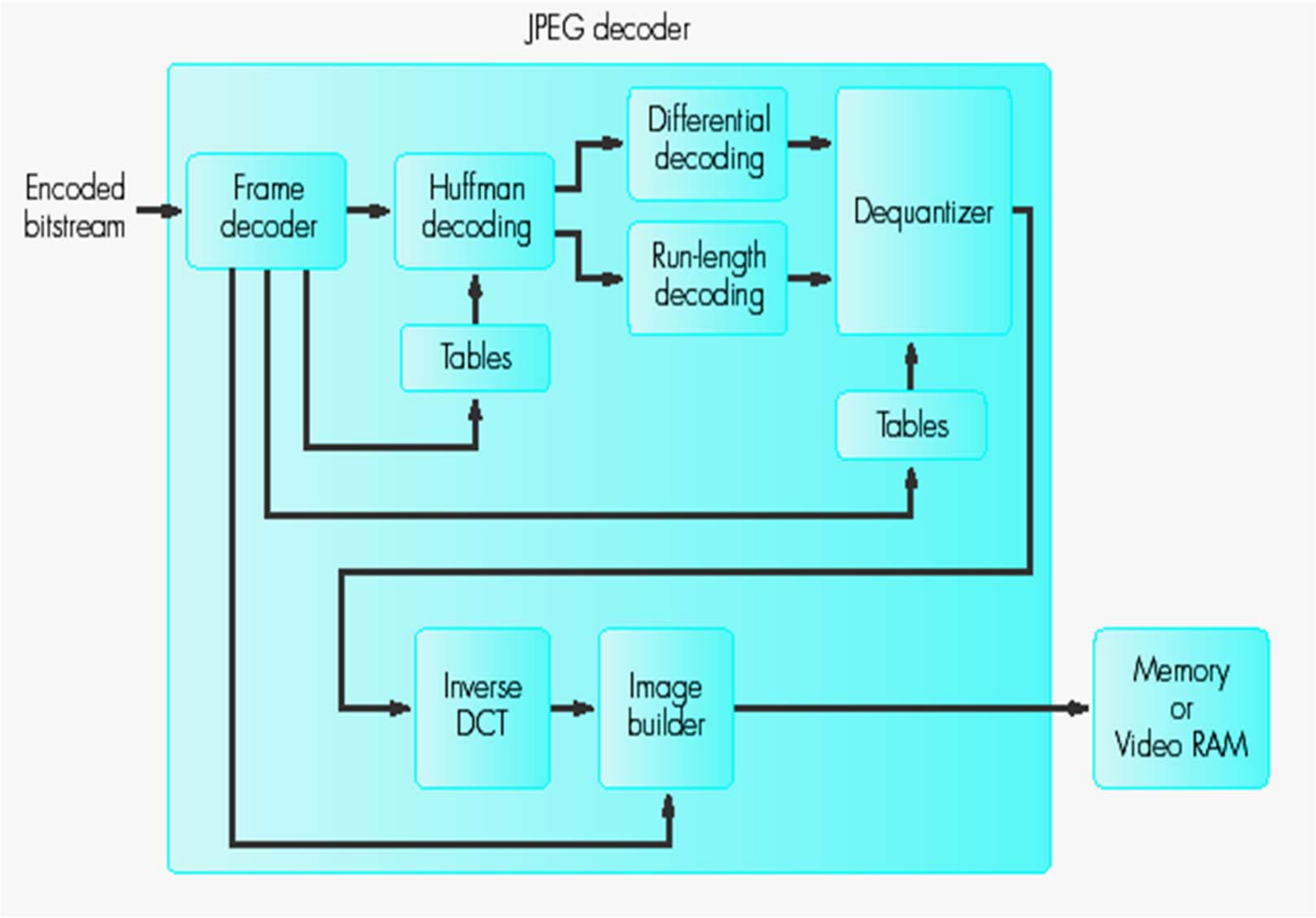
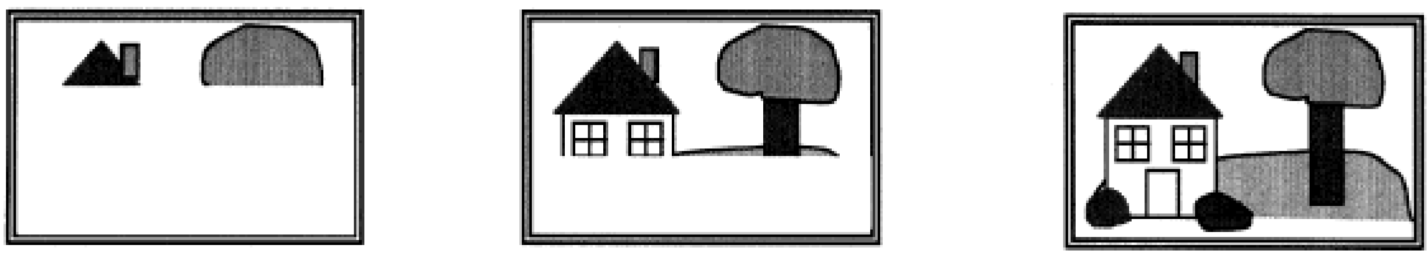



Figure source:
Fred Halsall, Multimedia communications: applications, networks, protocols and standards, Addison-Wesley, 2001

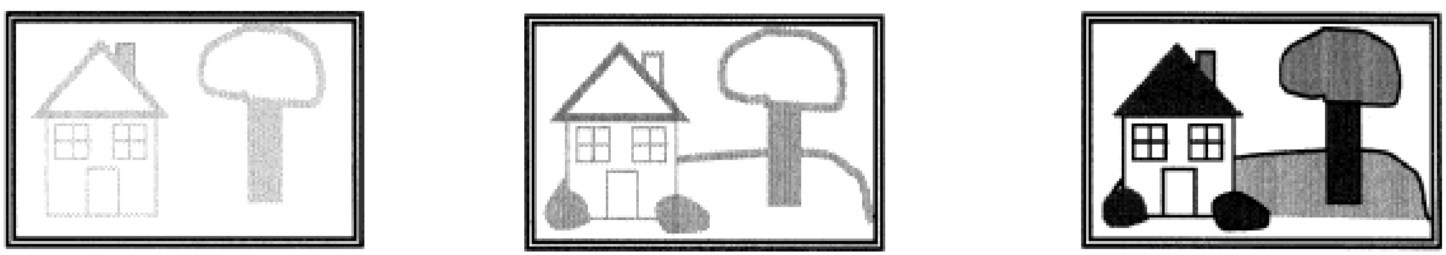
# JPEG Decoder


- JPEG decoder extracts the control information and tables, and passes them to the image builder.
- Huffman decoding is used to obtain the DC and AC symbols.
- Differential decoding is used to obtain the DC coefficients.
- Run-length decoding is used to obtain the AC coefficients.
- The DC and AC coefficients are put together and dequantized using quantization tables.
- Inverse DCT is used to transform DCT coefficients to pixel block.
- Image builder reconstructs the image from all the pixel blocks.

# Sequential vs Progressive Modes



(a) Sequential coding



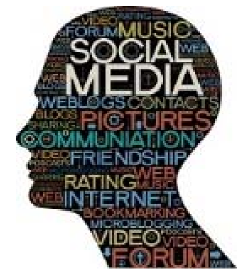
(b) Progressive coding

Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

# Sequential DCT-based JPEG

- In sequential DCT-based mode, an image is first partitioned into blocks of 8x8 pixels.
- The blocks are processed from left to right and top to bottom.
- 2D-DCT is applied to each 8x8 pixel block and the resulting DCT coefficients are quantized.
- Finally, the quantized DCT coefficients are entropy encoded and output as part of the compressed bitstream.

# Progressive DCT-based JPEG




- In progressive DCT-based mode, the block partitioning and DCT transform is similar to sequential DCT-based mode.
- However, in progressive mode, the DCT coefficients are encoded by a multiple scanning process.
- In each scan, the quantized DCT coefficients are encoded by either spectral selection or successive approximation.
  - Spectral selection: the coefficients are divided into multiple spectral (frequency) bands. In each scan, a specified band is encoded.
  - Successive approximation: a specified number of more significant bits of the coefficients are first encoded, followed by the less significant bits.

# Progressive DCT-based JPEG

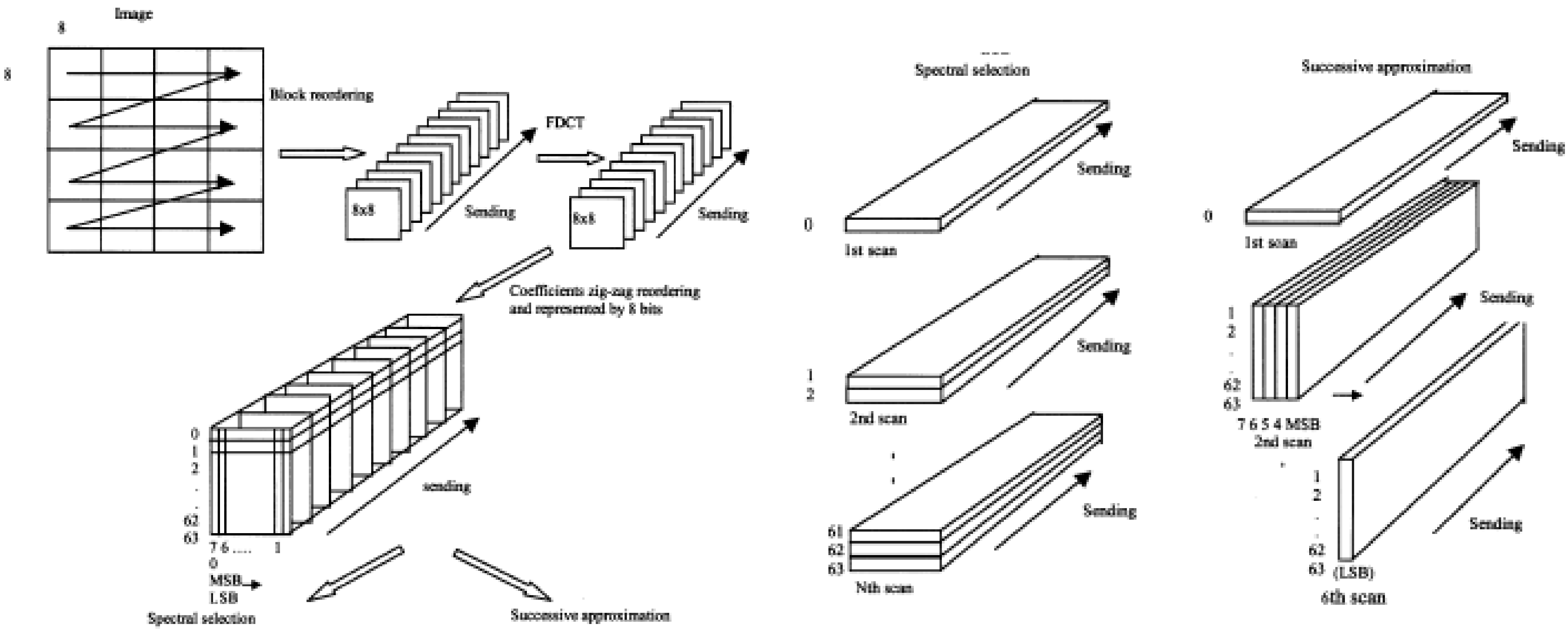



Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

# Lossless Mode JPEG

- Employ spatially-based coding instead of DCT-based coding.
- Each pixel is coded using predictive coding, where the predicted value is obtained from one-dimensional or two-dimensional predictors.

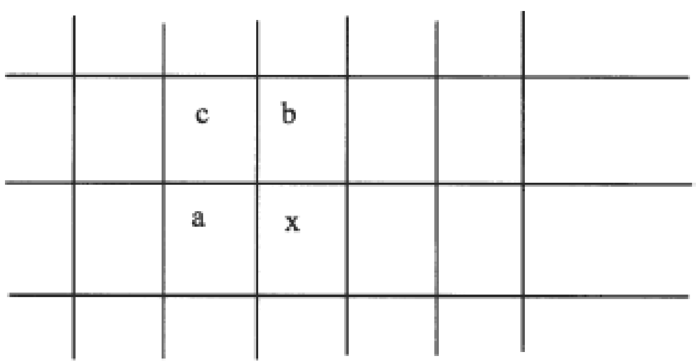



Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

**TABLE 7.6**
**Predictors for Lossless Coding**

| Selection-Value | Prediction |
|---|---|
| 0 | No prediction (hierarchical mode) |
| 1 | Px = a |
| 2 | Px = b |
| 3 | Px = c |
| 4 | Px = a+b-c |
| 5 | Px = a + ((b-c)/2)[a] |
| 6 | Px = b + ((a-c)/2)[a] |
| 7 | Px = (a+b)/2 |

[a] Shift right arithmetic operation.

# Hierarchical Mode JPEG


- Hierarchical mode JPEG addresses applications with multiresolution requirement.
- The input image is first decomposed into a sequence of frames, such as the pyramid. Each frame is obtained through a down-sampling process, i.e., low-pass filtering followed by subsampling.
- The first frame (the lowest resolution) is encoded as a non-differential frame. The following frames are encoded as differential frames, where the differential is with respect to the previously coded frame.

# Hierarchical Coding Mode

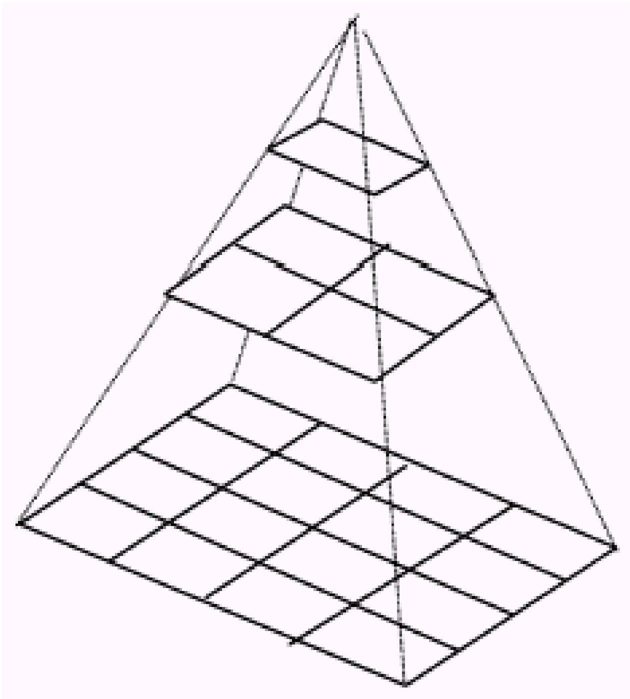


**FIGURE 7.2** Hierarchical multiresolution encoding.

Figure source:
Yun Q. Shi and Huifang Sun, Image and video compression for multimedia engineering: fundamentals, algorithms, and standards, CRC Press, 2000

# Exercise 2.8: JPEG

(c) In an image compression scheme similar to baseline JPEG, a user decides to partition an image into $4 \times 4$ subimages, perform $4 \times 4$ two-dimensional discrete cosine transform (2-D DCT), and quantize the transformed coefficients using the $4 \times 4$ quantization table $Q(u, v)$ given as follows:

$$Q(u,v) = \begin{bmatrix} 10 & 15 & 20 & 25 \\ 15 & 20 & 25 & 30 \\ 20 & 25 & 30 & 35 \\ 25 & 30 & 35 & 40 \end{bmatrix}$$

(i) If the user would like to increase the compression ratio of the scheme, suggest a suitable new quantization table to achieve the objective. What is the impact of the new quantization table on the reconstruction error of decompressed images? Justify your answers clearly.

(6 marks)

# Exercise 2.8: JPEG


(ii) The scheme is used to compress two $4 \times 4$ subimages $f(i,j)$ and $g(i,j)$ given as follows:

$$f(i,j) = \begin{bmatrix} 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 \\ 20 & 20 & 20 & 20 \\ 20 & 20 & 20 & 20 \end{bmatrix}, \quad g(i,j) = \begin{bmatrix} 10 & 50 & 10 & 50 \\ 50 & 10 & 50 & 10 \\ 10 & 100 & 10 & 100 \\ 100 & 10 & 100 & 10 \end{bmatrix}$$

change quickly
work better

Compare qualitatively the compression ratios that can be achieved for the two images $f(i,j)$ and $g(i,j)$. Justify your answer clearly.

(4 marks)

# Solution


FORUM MUSIC
SOCIAL
MEDIA
WEBLOGS CONTACTS
PICTURES
COMMUNIATION
FRIENDSHIP
RATING WEB
INTERNET
BOOKMARKING
MICROBLOGGING
VIDEO VIDEO
FORUM

# Summary


- This section covers the following:
  - Text Compression
  - Image & Video Compression Basics
  - Image Coding/Compression:
    - Transform coding
      - Discrete Cosine Transform (DCT)
      - Karhunen-Loeve Transform (KLT)
    - Predictive/differential coding
    - Vector Quantization (VQ)
    - Singular Value Decomposition (SVD)
  - JPEG standard