

# Image Compression

# Image Compression

2

- Everyday an enormous amount of information is stored, processed, and transmitted
  - Financial data
  - Reports
  - Inventory
  - Cable TV
  - Online Ordering and tracking
- Because much of this information is graphical or pictorial in nature, the storage and communications requirements are immense.
- Image compression addresses the problem of reducing the amount of data requirements to represent a digital image.

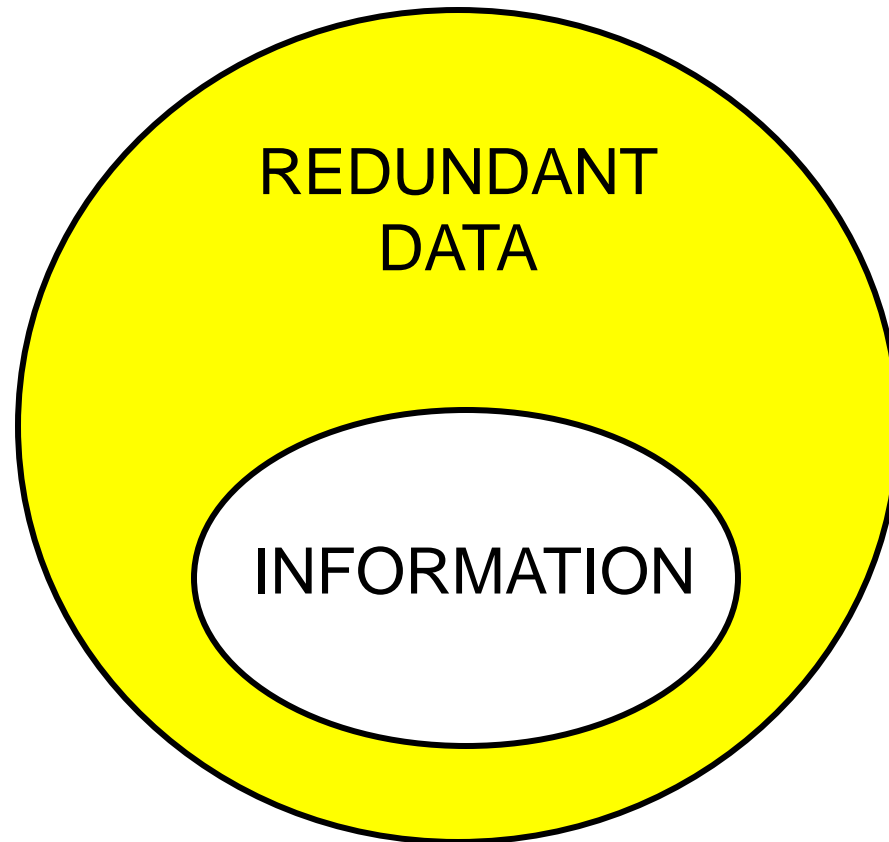
# Fundamentals

3

- The term data compression refers to the process of reducing the amount of data required to represent a given quantity of information
- Data  $\neq$  Information
- Various amount of data can be used to represent the same information
- Data might contain elements that provide no relevant information : *data redundancy*
- Data redundancy is a central issue in image compression. It is not an abstract concept but mathematically quantifiable entity

# Information vs Data

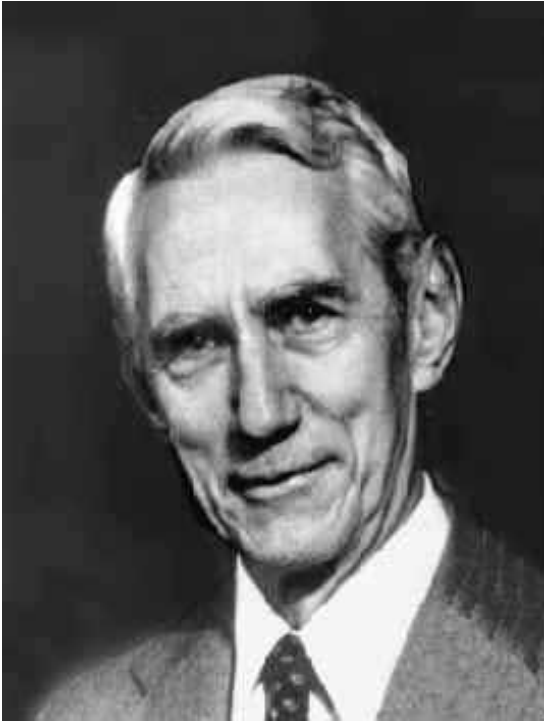
4



**DATA = INFORMATION + REDUNDANT DATA**

# Measuring Information (not assessed)

5



Claude Shannon

1916-2001

Founder of information theory

The entropy of a source is a simple measure of the information content. For any discrete probability distribution, the value of the entropy function (H) is given by:-

$$H = \sum_{i=1}^q p_i \log_r \frac{1}{p_i}$$

(r=radix = 2 for binary)

The units of entropy are bits/symbol.

We can compare the performance of our compression method with the calculated source entropy.

Where the source 'alphabet' has q symbols of probability  $p_i$  ( $i=1..q$ ).

# Why Need Compression?

6

- Savings in storage and transmission
  - ▣ multimedia data (esp. image and video) have large data volume
  - ▣ difficult to send real-time uncompressed video over current network
  
- Accommodate relatively slow storage devices
  - ▣ they do not allow playing back uncompressed multimedia data in real time
    - 1x CD-ROM transfer rate  $\sim 150$  kB/s
    - 320 x 240 x 24 fps color video bit rate  $\sim 5.5$  MB/s
  - => 36 seconds needed to transfer 1-sec uncompressed video from CD

# Example: Storing An Encyclopedia

7

- ▣ 500,000 pages of text (2kB/page)      ~ 1GB => 2:1 compress
  - ▣ 3,000 color pictures (640×480×24bits)      ~ 3GB => 15:1
  - ▣ 500 maps (640×480×16bits=0.6MB/map)      ~ 0.3GB => 10:1
  - ▣ 60 minutes of stereo sound (176kB/s)      ~ 0.6GB => 6:1
  - ▣ 30 animations with average 2 minutes long  
(640×320×16bits×16frames/s=6.5MB/s)      ~ 23.4GB => 50:1
  - ▣ 50 digitized movies with average 1 minute long  
(640×480×24bits×30frames/s = 27.6MB/s)      ~ 82.8GB =>  
50:1
- Require a total of 111.1GB storage capacity if without compression
- Reduce to 2.96GB if with compression

# Relative Data Redundancy and Compression Ratio

8

## Relative Data Redundancy

$$R_D = 1 - \frac{1}{C_R}$$

## Compression Ratio

$$C_R = \frac{n_1}{n_2}$$

$n_1$  and  $n_2$  denote the number of information carrying units in two data sets that represent the same information

## Types of data redundancy

1. Coding redundancy
2. Interpixel redundancy
3. Psychovisual redundancy



# Coding Redundancy

9

- Recall from the histogram calculations

$$p(r_k) = \frac{h(r_k)}{n} = \frac{n_k}{n}$$

where  $p(r_k)$  is the probability of a pixel to have a certain value  $r_k$

If the number of bits used to represent  $r_k$  is  $l(r_k)$ , then

$$L_{av} = \sum_{k=0}^{L-1} l(r_k)(p(r_k))$$

# Coding Redundancy

10

## □ Example:

$$L_{av} = \sum_{k=0}^7 l(r_k)(p(r_k))$$

$$= 2(0.19) + 2(0.25) + 3(0.16) + \dots + 6(0.02)$$

$$= 2.7 \text{ bits}$$

$$C_R = \frac{3}{2} = 1.11$$

$$R_D = 1 - \frac{1}{1.11} = 0.099$$

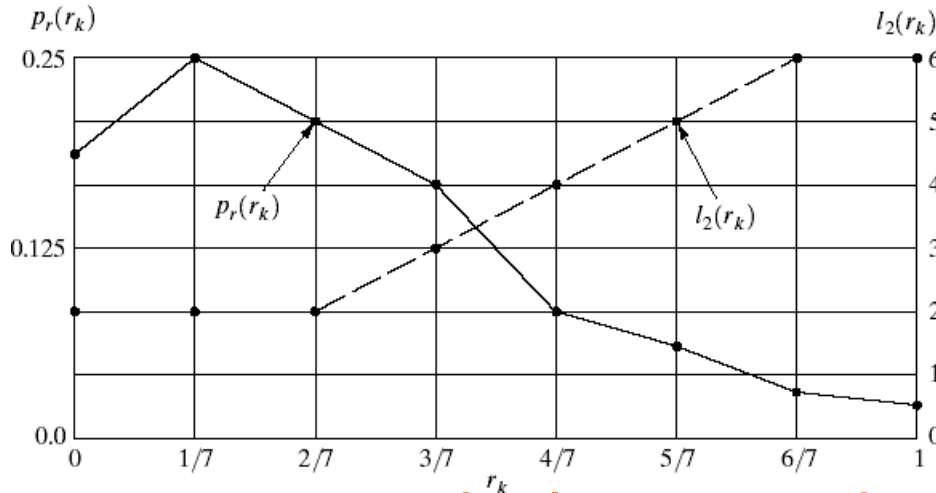
$r_k$	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(r_k)$
$r_0 = 0$	0.19	000	3	11	2
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

# Coding Redundancy

11

$r_k$	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(r_k)$
$r_0 = 0$	0.19	000	3	11	2
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

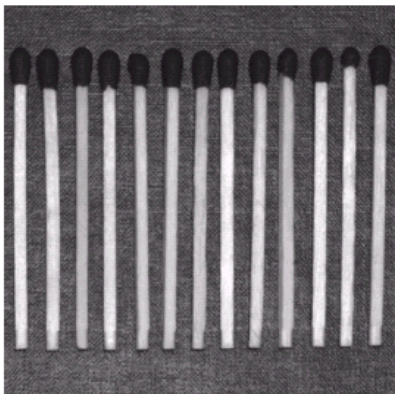
## Variable-Length Coding



Graphic representation of the fundamental basis of data compression through variable-length coding.

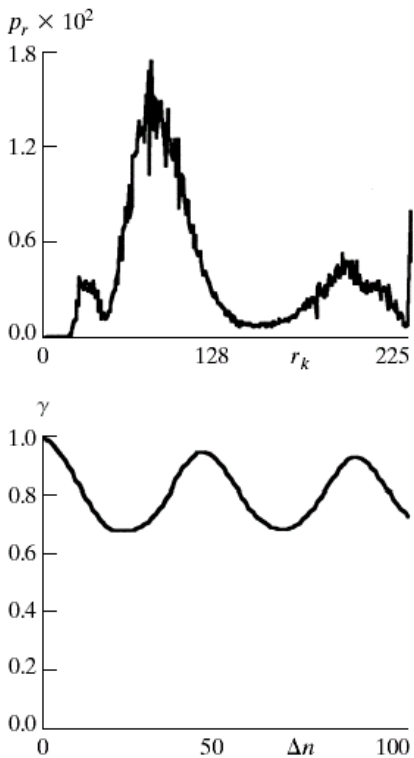
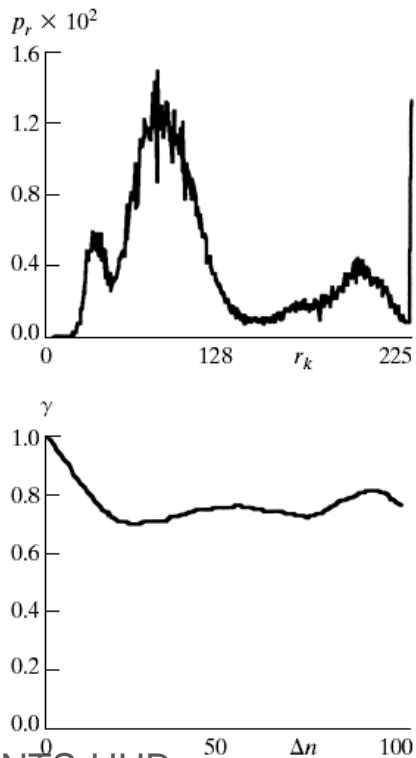
Concept: assign the longest code word to the symbol with the least probability of occurrence.

# Interpixel Redundancy



a	b
c	d
e	f

**FIGURE 8.2** Two images and their gray-level histograms and normalized autocorrelation coefficients along one line.



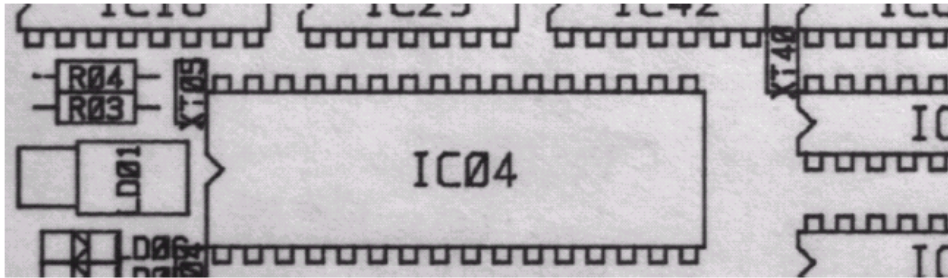
Here the two pictures have Approximately the same Histogram.

Interpixel redundancy:  
Parts of an image are highly correlated.

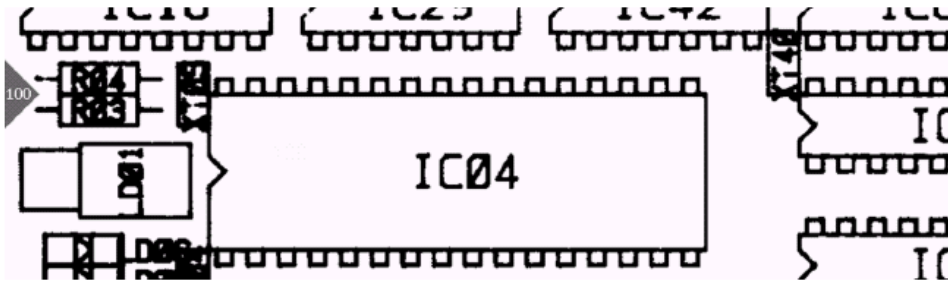
In other words, we can predict a given pixel from its neighbor.

# Run Length Coding

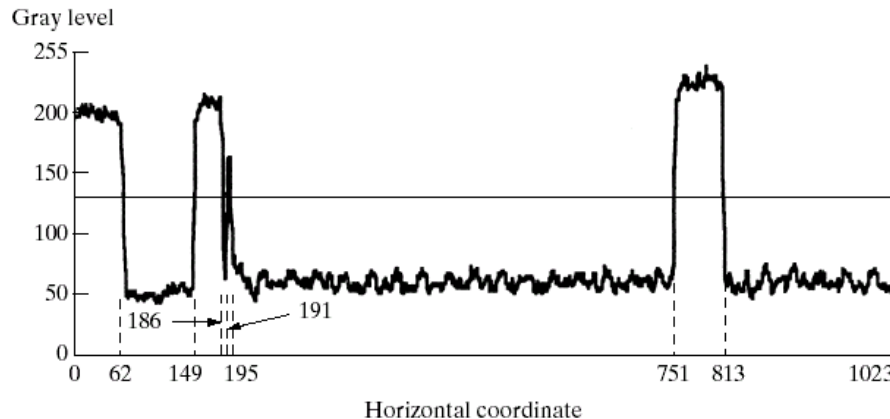
13



The gray scale image  
of size 343x1024 pixels



Binary image  
 $= 343 \times 1024 \times 1 = 351232$  bits



Line No. 100

Run length coding

Line 100: (1,63) (0,87) (1,37) (0,5) (1,4) (0,556) (1,62) (0,210)

Total 12166 runs, each run use 11 bits  $\rightarrow$  Total = 133826 Bits

# Irrelevant Redundancy

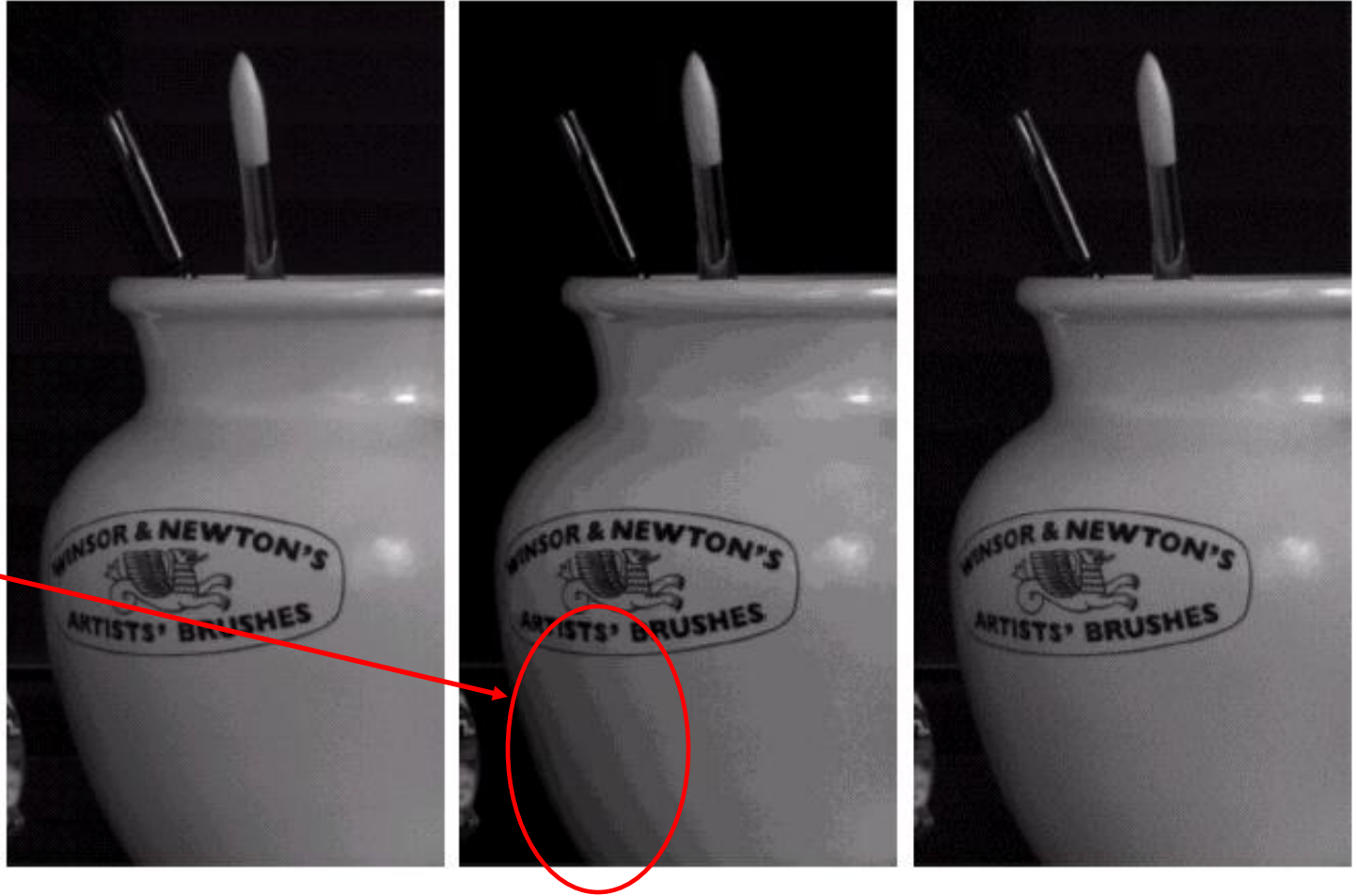
14

8-bit gray scale  
image

4-bit gray scale  
image

4-bit IGS  
image

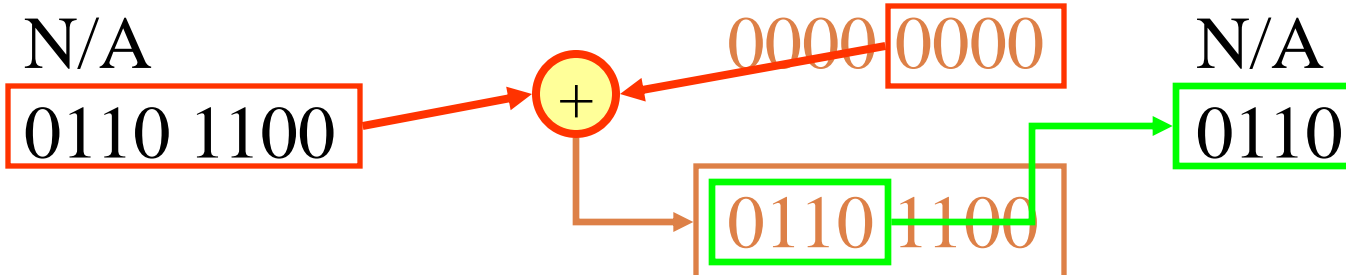
False  
contours



The eye does not response with equal sensitivity to all visual information.

# Improved Gray Scale Quantization

15

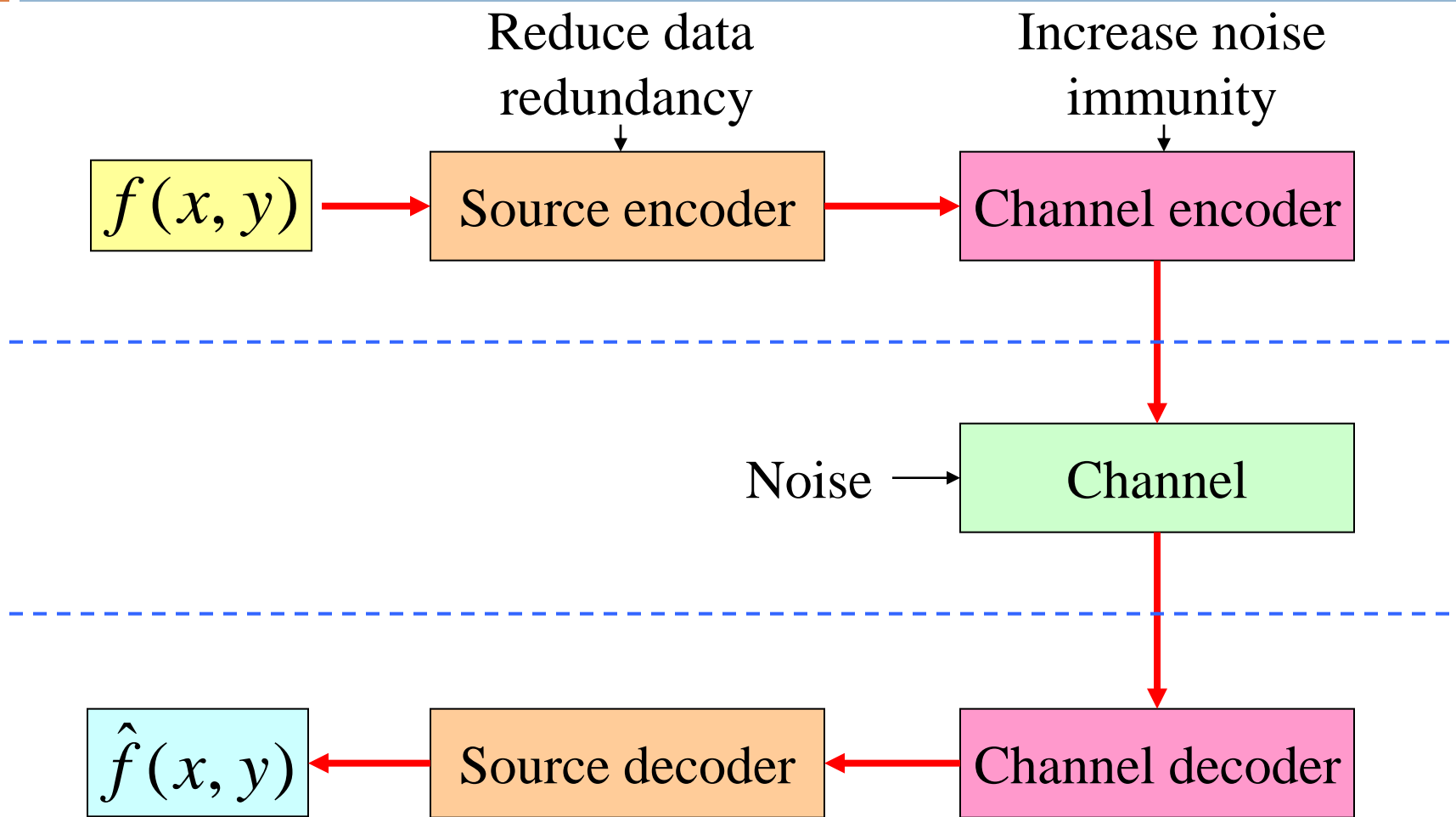
Pixel	Gray level	Sum	IGS Code
i-1	N/A	0000 0000	N/A
i	0110 1100		0110
i+1	1000 1011	1001 0111	1001
i+2	1000 0111	1000 1110	1000
i+3	1111 0100	1111 0100	1111

## Algorithm

1. Add the least significant 4 bits of the previous value of Sum to the 8-bit current pixel. If the most significant 4 bit of the pixel is 1111 then add 0000 instead. Keep the result in Sum
2. Keep only the most significant 4 bits of Sum for IGS code.

# Image Compression Models

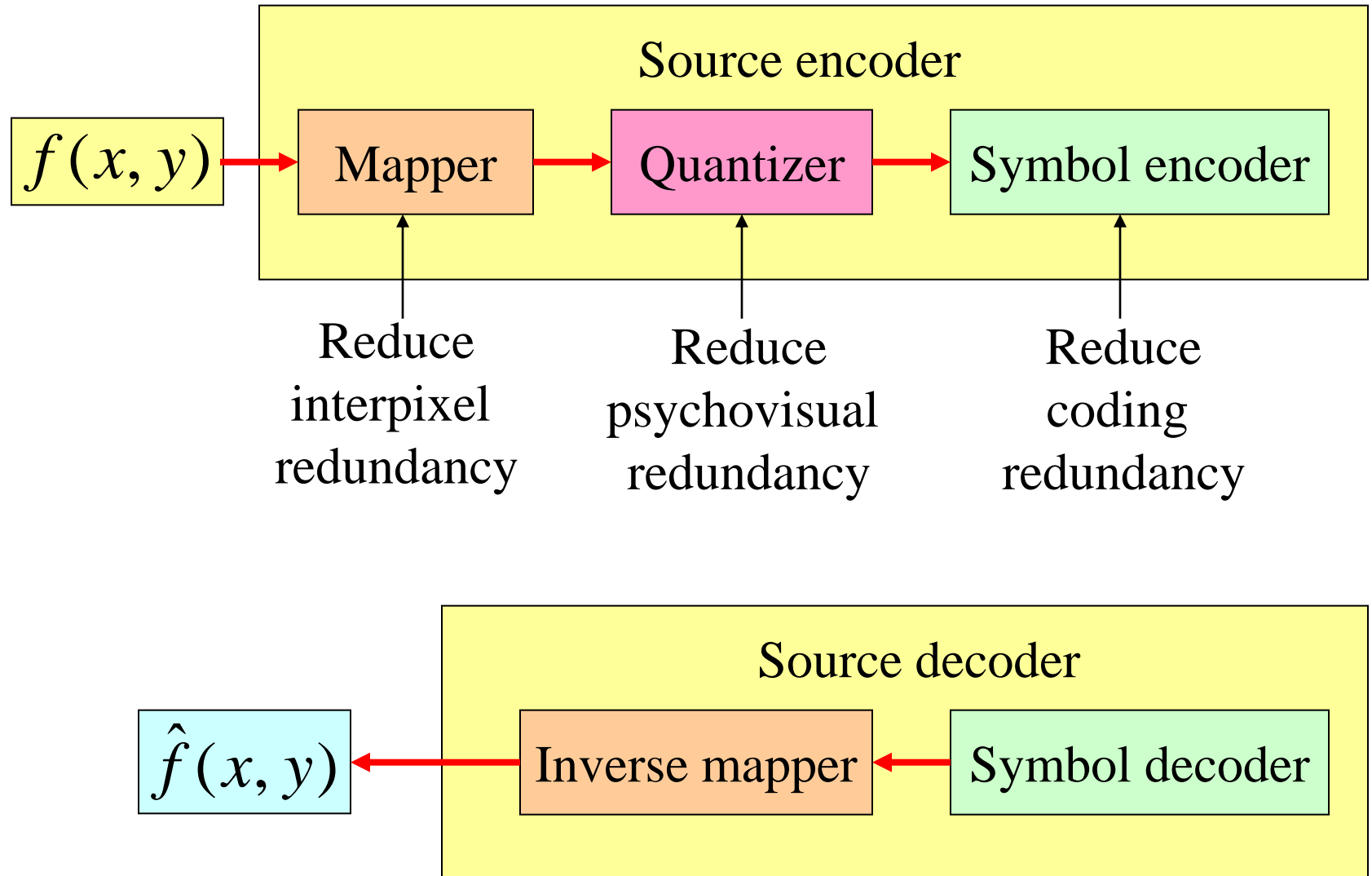
16





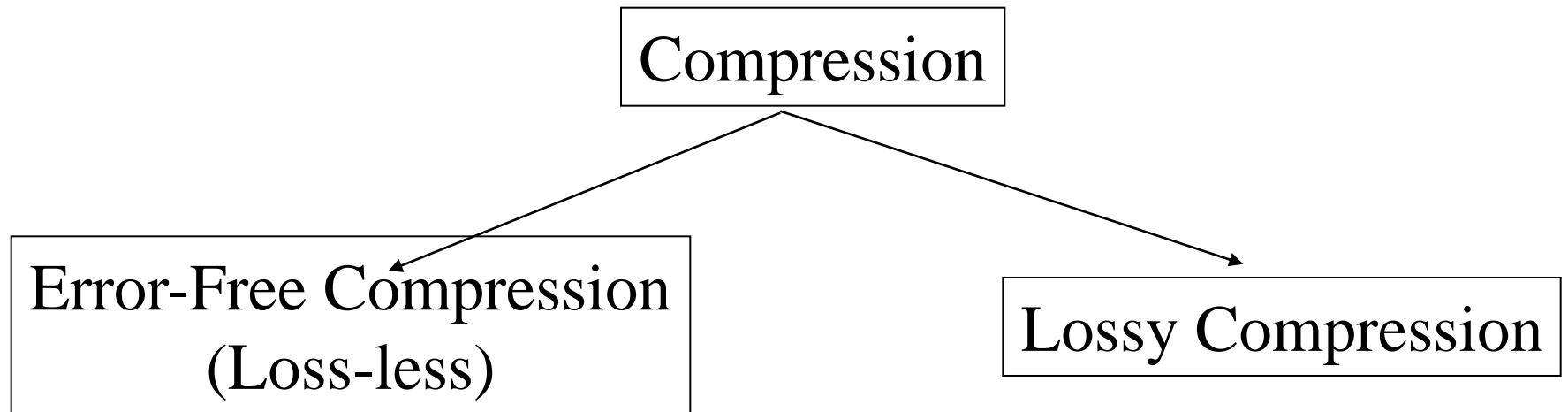
# Source Encoder and Decoder Models

17



# Compression Types

18



# Lossless and Lossy Compression

19

- Lossless compression (reversible) produces an exact copy of original.
- Lossy compression (irreversible) produces an approximation of original.
- Lossy compression is used on image, video and audio files where imperceptible (or “tolerable”) losses to quality are exchanged for much larger compression ratios.
- Lossless compression usually achieves much less compression than lossy compression.
- It can be difficult to get a lossless compression ratio of more than 2:1 for images, but most lossy image compression can usually achieve 10:1 without too much loss of quality.
- Increasing lossy compression beyond specified limits can result in unwanted compression artefacts (characteristic errors introduced by compression losses).

# Measuring “Quality”

20

- How do we measure the “quality” of lossily compressed images?
- Measurement methods
- Objective:- impartial measuring methods
- Subjective:- based on personal feelings
- We need definitions of “quality” (“degree of excellence”?) and to define how we will compare the original and decompressed images.



# Fidelity Criteria

21

- A closely related objective fidelity criterion is the mean square signal to noise ratio of the compressed-decompressed image

$$SNR_{ms} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}(x, y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2}$$

## Subjective Fidelity Criterion: Human Rating

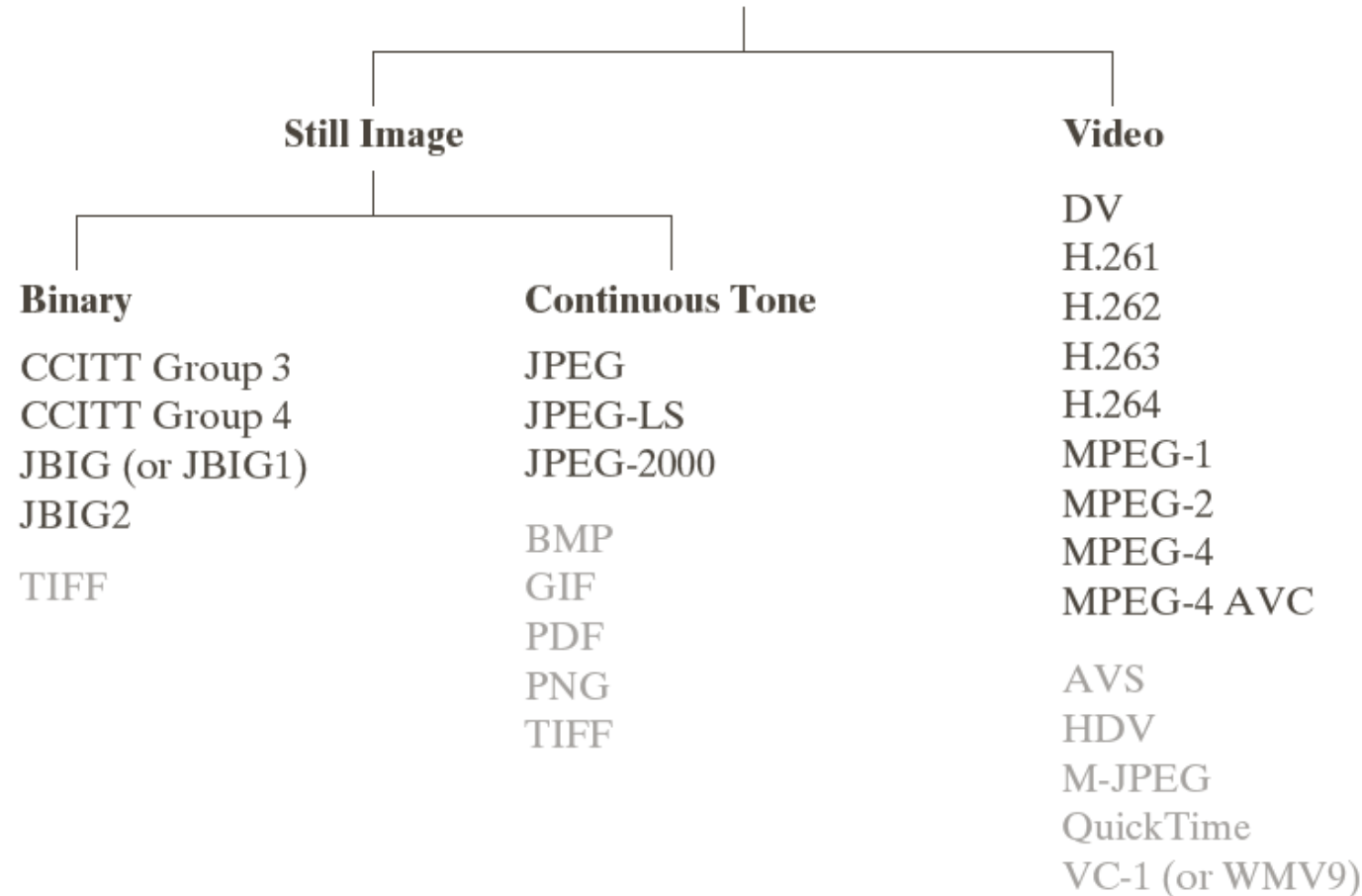
Value	Rating	Description
1	Excellent	An image of extremely high quality, as good as you could desire.
2	Fine	An image of high quality, providing enjoyable viewing. Interference is not objectionable.
3	Passable	An image of acceptable quality. Interference is not objectionable.
4	Marginal	An image of poor quality; you wish you could improve it. Interference is somewhat objectionable.
5	Inferior	A very poor image, but you could watch it. Objectionable interference is definitely present.
6	Unusable	An image so bad that you could not watch it.

# What About Content?

22

- Does image or video content affect quality perception?
- Can very poor image quality be offset by interesting content?

## Image Compression Standards, Formats, and Containers



**FIGURE 8.6** Some popular image compression standards, file formats, and containers. Internationally sanctioned entries are shown in black; all others are grayed.

Name	Organization	Description
<i>Bi-Level Still Images</i>		
CCITT Group 3	ITU-T	Designed as a facsimile (FAX) method for transmitting binary documents over telephone lines. Supports 1-D and 2-D run-length [8.2.5] and Huffman [8.2.1] coding.
CCITT Group 4	ITU-T	A simplified and streamlined version of the CCITT Group 3 standard supporting 2-D run-length coding only.
JBIG or JBIG1	ISO/IEC/ ITU-T	A <i>Joint Bi-level Image Experts Group</i> standard for progressive, lossless compression of bi-level images. Continuous-tone images of up to 6 bits/pixel can be coded on a bit-plane basis [8.2.7]. Context sensitive arithmetic coding [8.2.3] is used and an initial low resolution version of the image can be gradually enhanced with additional compressed data.
JBIG2	ISO/IEC/ ITU-T	A follow-on to JBIG1 for bi-level images in desktop, Internet, and FAX applications. The compression method used is content based, with dictionary based methods [8.2.6] for text and halftone regions, and Huffman [8.2.1] or arithmetic coding [8.2.3] for other image content. It can be lossy or lossless.
<i>Continuous-Tone Still Images</i>		
JPEG	ISO/IEC/ ITU-T	A <i>Joint Photographic Experts Group</i> standard for images of photographic quality. Its lossy <i>baseline coding system</i> (most commonly implemented) uses quantized discrete cosine transforms (DCT) on $8 \times 8$ image blocks [8.2.8], Huffman [8.2.1], and run-length [8.2.5] coding. It is one of the most popular methods for compressing images on the Internet.
JPEG-LS	ISO/IEC/ ITU-T	A lossless to near-lossless standard for continuous tone images based on adaptive prediction [8.2.9], context modeling [8.2.3], and Golomb coding [8.2.2].
JPEG-2000	ISO/IEC/ ITU-T	A follow-on to JPEG for increased compression of photographic quality images. Arithmetic coding [8.2.3] and quantized discrete wavelet transforms (DWT) [8.2.10] are used. The compression can be lossy or lossless.

Name	Organization	Description
<i>Video</i>		
DV	IEC	<i>Digital Video</i> . A video standard tailored to home and semiprofessional video production applications and equipment—like electronic news gathering and camcorders. Frames are compressed independently for uncomplicated editing using a DCT-based approach [8.2.8] similar to JPEG.
H.261	ITU-T	A two-way videoconferencing standard for ISDN ( <i>integrated services digital network</i> ) lines. It supports non-interlaced $352 \times 288$ and $176 \times 144$ resolution images, called CIF ( <i>Common Intermediate Format</i> ) and QCIF ( <i>Quarter CIF</i> ), respectively. A DCT-based compression approach [8.2.8] similar to JPEG is used, with frame-to-frame prediction differencing [8.2.9] to reduce temporal redundancy. A block-based technique is used to compensate for motion between frames.
H.262	ITU-T	See MPEG-2 below.
H.263	ITU-T	An enhanced version of H.261 designed for ordinary telephone modems (i.e., 28.8 Kb/s) with additional resolutions: SQCIF ( <i>Sub-Quarter CIF</i> $128 \times 96$ ), 4CIF ( $704 \times 576$ ), and 16CIF ( $1408 \times 512$ ).
H.264	ITU-T	An extension of H.261–H.263 for videoconferencing, Internet streaming, and television broadcasting. It supports prediction differences within frames [8.2.9], variable block size integer transforms (rather than the DCT), and context adaptive arithmetic coding [8.2.3].
MPEG-1	ISO/IEC	A <i>Motion Pictures Expert Group</i> standard for CD-ROM applications with non-interlaced video at up to 1.5 Mb/s. It is similar to H.261 but frame predictions can be based on the previous frame, next frame, or an interpolation of both. It is supported by almost all computers and DVD players.
MPEG-2	ISO/IEC	An extension of MPEG-1 designed for DVDs with transfer rates to 15 Mb/s. Supports interlaced video and HDTV. It is the most successful video standard to date.
MPEG-4	ISO/IEC	An extension of MPEG-2 that supports variable block sizes and prediction differencing [8.2.9] within frames.
MPEG-4 AVC	ISO/IEC	MPEG-4 Part 10 <i>Advanced Video Coding</i> (AVC). Identical to H.264 above.



Name	Organization	Description
<i>Continuous-Tone Still Images</i>		
BMP	Microsoft	<i>Windows Bitmap</i> . A file format used mainly for simple uncompressed images.
GIF	CompuServe	<i>Graphic Interchange Format</i> . A file format that uses lossless LZW coding [8.2.4] for 1- through 8-bit images. It is frequently used to make small animations and short low resolution films for the World Wide Web.
PDF	Adobe Systems	<i>Portable Document Format</i> . A format for representing 2-D documents in a device and resolution independent way. It can function as a container for JPEG, JPEG 2000, CCITT, and other compressed images. Some PDF versions have become ISO standards.
PNG	World Wide Web Consortium (W3C)	<i>Portable Network Graphics</i> . A file format that losslessly compresses full color images with transparency (up to 48 bits/pixel) by coding the difference between each pixel's value and a predicted value based on past pixels [8.2.9].
TIFF	Aldus	<i>Tagged Image File Format</i> . A flexible file format supporting a variety of image compression standards, including JPEG, JPEG-LS, JPEG-2000, JBIG2, and others.
<i>Video</i>		
AVS	MII	<i>Audio-Video Standard</i> . Similar to H.264 but uses exponential Golomb coding [8.2.2]. Developed in China.
HDV	Company consortium	<i>High Definition Video</i> . An extension of DV for HD television that uses MPEG-2 like compression, including temporal redundancy removal by prediction differencing [8.2.9].
M-JPEG	Various companies	<i>Motion JPEG</i> . A compression format in which each frame is compressed independently using JPEG.
Quick-Time	Apple Computer	A media container supporting DV, H.261, H.262, H.264, MPEG-1, MPEG-2, MPEG-4, and other video compression formats.
VC-1 WMV9	SMPTE Microsoft	The most used video format on the Internet. Adopted for HD and <i>Blu-ray</i> high-definition DVDs. It is similar to H.264/AVC, using an integer DCT with varying block sizes [8.2.8 and 8.2.9] and context dependent variable-length code tables [8.2.1]—but no predictions within frames.

# Error-Free Compression

26

- Some applications require no error in compression (medical, business documents, etc..)
- $C_R=2$  to 10 can be expected.
- Make use of coding redundancy and inter-pixel redundancy.
- Ex: Huffman codes, LZW, Arithmetic coding, 1D and 2D run-length encoding, Loss-less Predictive Coding, and Bit-Plane Coding.

# Error-Free Compression: Huffman Coding

Source character frequency statistics are used to allocate codewords for output.

Compression can be achieved by allocating shorter codewords to the more frequently occurring characters.

## Step 1: Source reduction

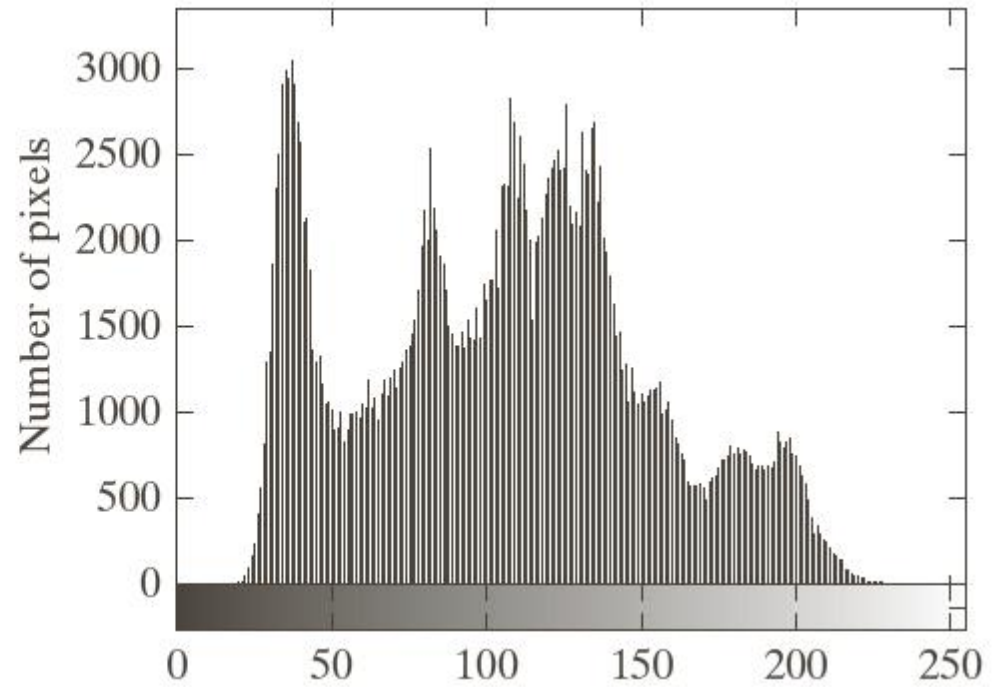
Original source		Source reduction				
Symbol	Probability	1	2	3	4	
$a_2$	0.4	0.4	0.4	0.4	0.6	
$a_6$	0.3	0.3	0.3	0.3		
$a_1$	0.1	0.1	0.2	0.3	0.4	
$a_4$	0.1	0.1				
$a_3$	0.06	0.1	0.1	0.1		
$a_5$	0.04					

# Error-Free Compression: Huffman Coding

## Step 2: Code assignment procedure

Original source			Source reduction			
Sym.	Prob.	Code	1	2	3	4
$a_2$	0.4	1	0.4	1	0.4	1
$a_6$	0.3	00	0.3	00	0.3	00
$a_1$	0.1	011	0.1	011	0.2	010
$a_4$	0.1	0100	0.1	0100	0.1	011
$a_3$	0.06	01010	0.1	0101	0.3	01
$a_5$	0.04	01011				

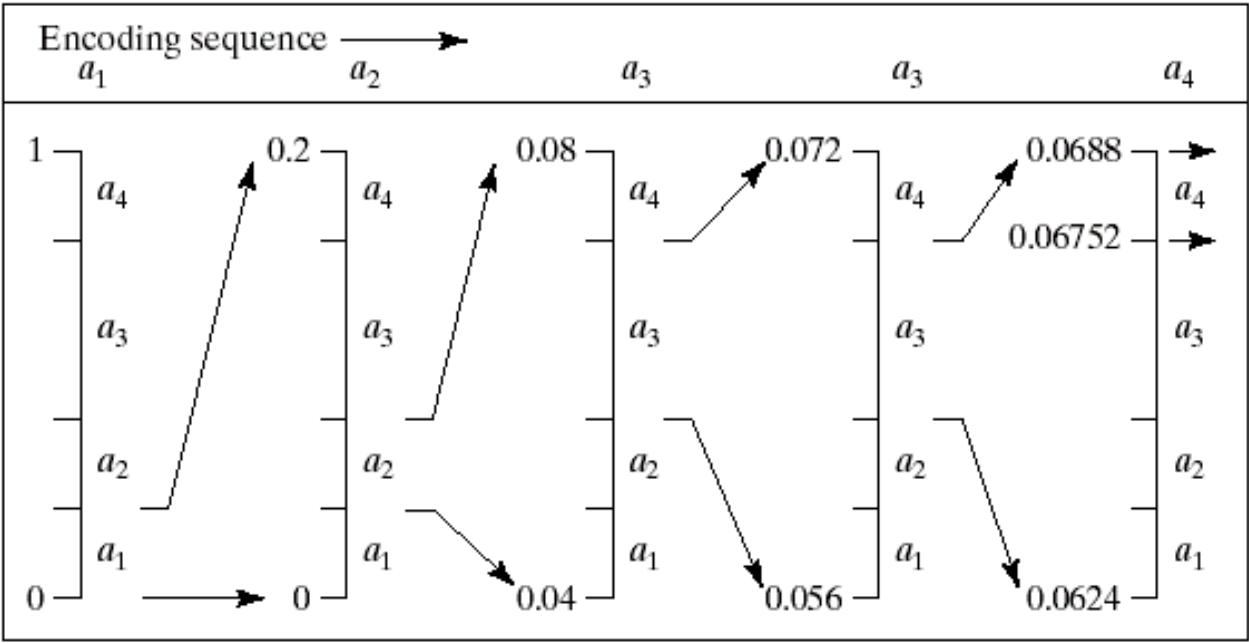
The code is instantaneous uniquely decodable without referencing succeeding symbols.



# Arithmetic Coding

**Nonblock code:** one-to-one correspondence between source symbols and code words does not exist.

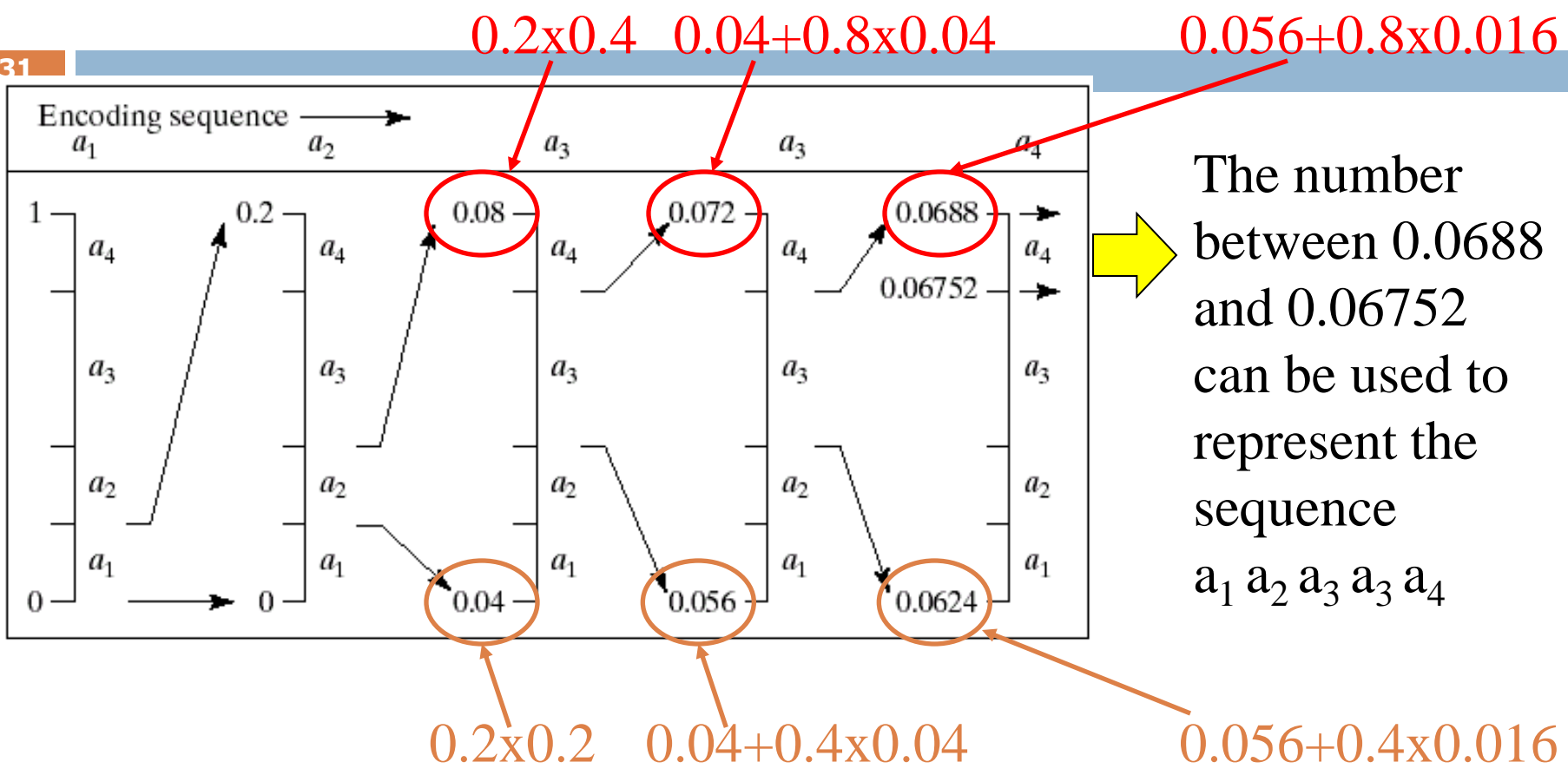
**Concept:** The entire sequences of source symbols is assigned a single arithmetic code word in the form of a number in an interval of real number between 0 and 1.



**FIGURE 8.13**  
Arithmetic coding procedure.

# Arithmetic Coding Example

31



The number between 0.0688 and 0.06752 can be used to represent the sequence  $a_1 a_2 a_3 a_3 a_4$

Source Symbol	Probability	Initial Subinterval
$a_1$	0.2	[0.0, 0.2)
$a_2$	0.2	[0.2, 0.4)
$a_3$	0.4	[0.4, 0.8)
$a_4$	0.2	[0.8, 1.0)

# Fixed Length: LZW Coding

32

- ❑ Error Free Compression Technique
- ❑ Remove Inter-pixel redundancy
- ❑ Requires no priori knowledge of probability distribution of pixels
- ❑ Assigns fixed length code words to variable length sequences
- ❑ Included in GIF and TIFF and PDF file formats



## □ Coding Technique

- ▣ A codebook or a dictionary has to be constructed
- ▣ For an 8-bit monochrome image, the first 256 entries are assigned to the gray levels 0,1,2,...,255.
- ▣ As the encoder examines image pixels, gray level sequences that are not in the dictionary are assigned to a new entry.
- ▣ For instance sequence 255-255 can be assigned to entry 256, the address following the locations reserved for gray levels 0 to 255.

# LZW Coding

34

## □ Example

Consider the following 4 x 4 8 bit image

39 39 126 126

39 39 126 126

39 39 126 126

39 39 126 126

Dictionary Location	Entry
0	0
1	1
.	.
255	255
256	-
511	-

Initial Dictionary

# LZW Coding

35

39 39 126 126  
39 39 126 126  
39 39 126 126  
39 39 126 126

- Is 39 in the dictionary.....Yes
- What about 39-39.....No
- Then add 39-39 in entry 256
- And output the last recognized symbol...39

Dictionary Location	Entry
0	0
1	1
.	.
255	255
256	39-39
511	-

# LZW Coding

Lempel-Ziv-Welch coding : assign fixed length code words to variable length sequences of source symbols.

Currently Recognized Sequence	Pixel Being Processed	Encoded Output	Dictionary Location (Code Word)	Dictionary Entry
	39			
39	39	39	256	39-39
39	126	39	257	39-126
126	126	126	258	126-126
126	39	126	259	126-39
39	39			
39-39	126	256	260	39-39-126
126	126			
126-126	39	258	261	126-126-39
39	39			
39-39	126			
39-39-126	126	260	262	39-39-126-126
126	39			
126-39	39	259	263	126-39-39
39	126			
39-126	126	257	264	39-126-126
126		126		

24 Bits

9 Bits

# LZW Coding Algorithm

37

0. Initialize a dictionary by all possible gray values (0-255)

1. **Input** current pixel

2. **If** the current pixel combined with previous pixels  
form **one of existing dictionary entries**

**Then**

2.1 **Move to** the next pixel and **repeat** Step 1

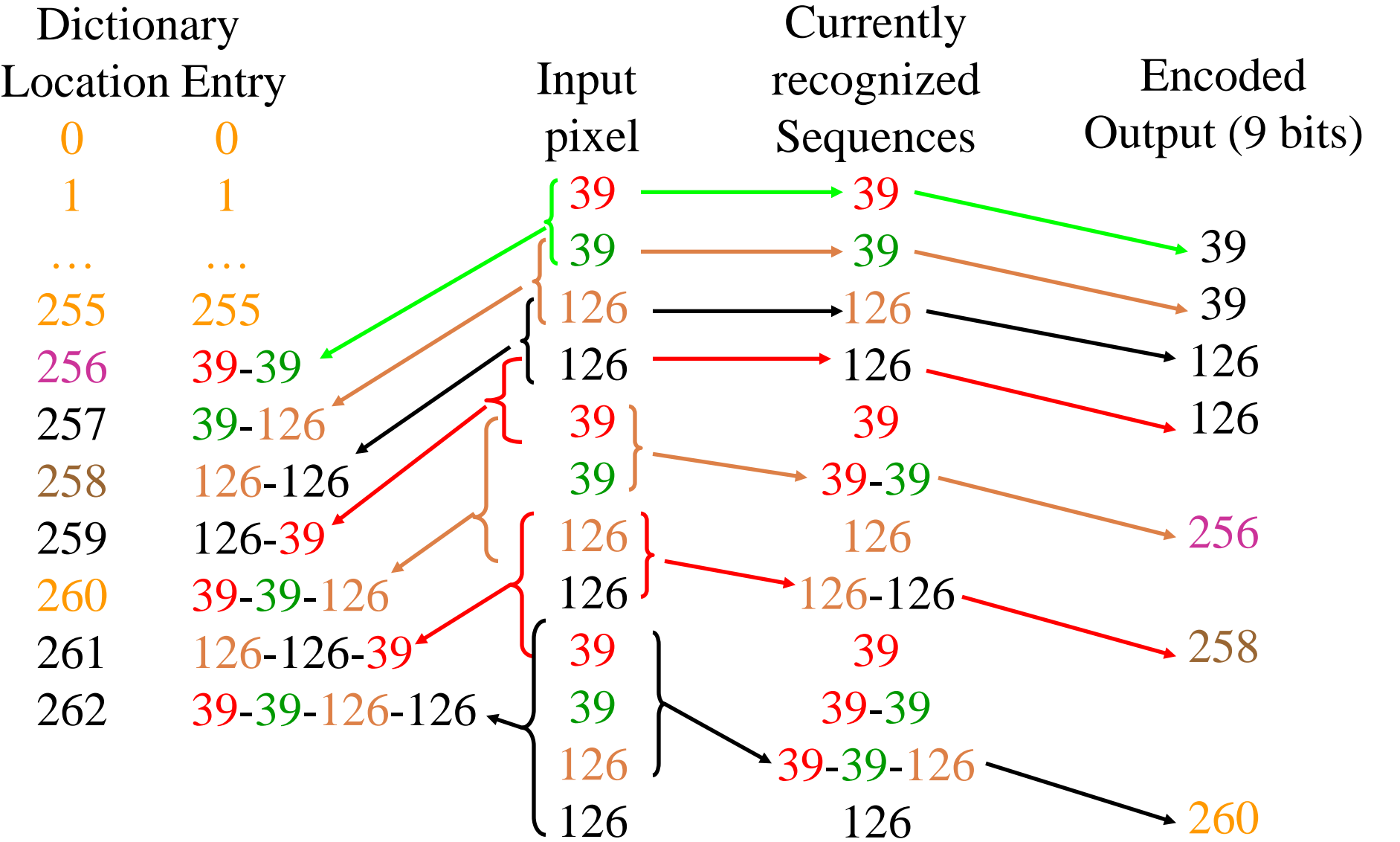
**Else**

2.2 Output the dictionary location of the currently recognized  
sequence (**which is not include the current pixel**)

2.3 Create a new dictionary entry by appending the currently  
recognized sequence in 2.2 with the current pixel

2.4 **Move to** the next pixel and **repeat** Step 1

# LZW Coding Example



# Bit-Plane Coding

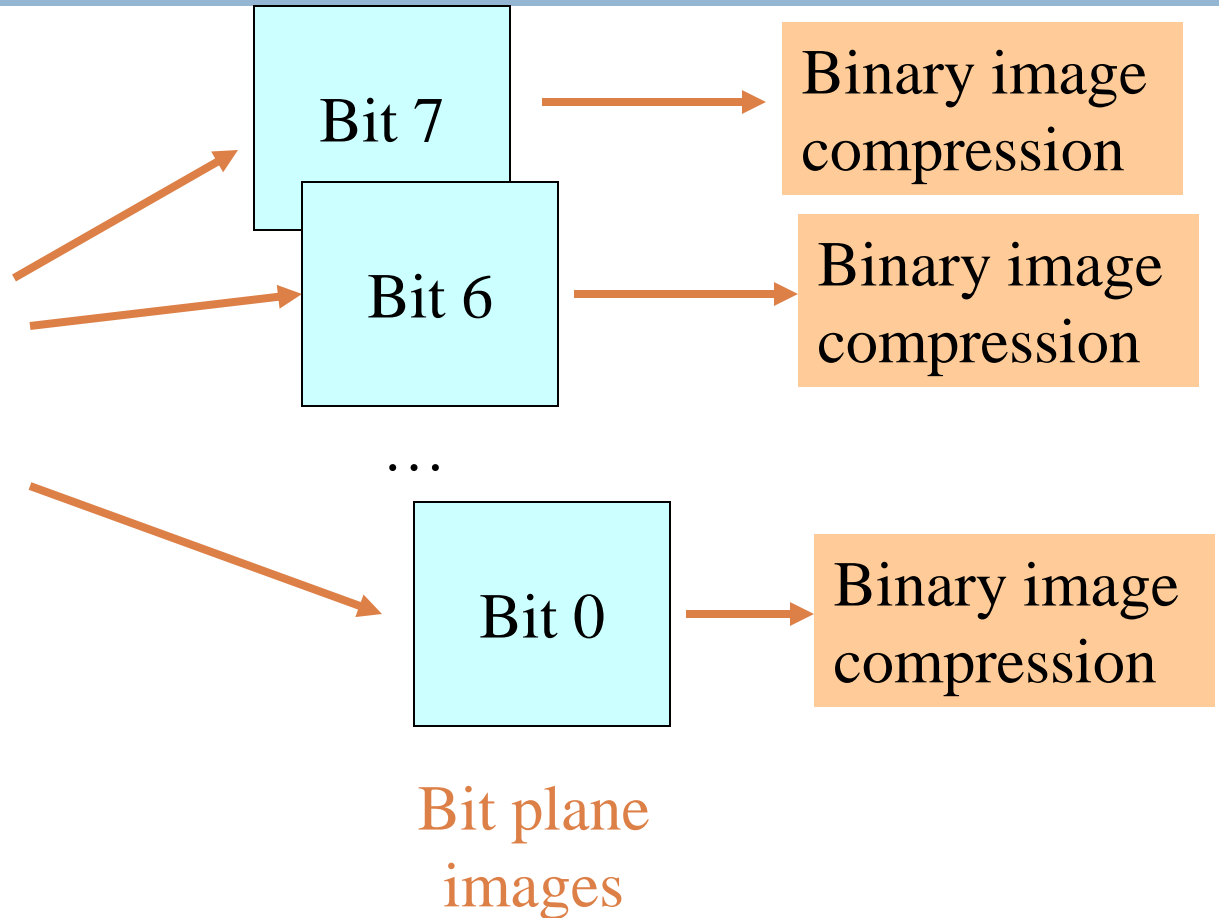
39

- An effective technique to reduce inter pixel redundancy is to process each bit plane individually
- The image is decomposed into a series of binary images.
- Each binary image is compressed using one of well known binary compression techniques.

# Bit-Plane Coding

40

Original image



Example of binary image compression: Run length coding



# Bit Planes



Original gray  
scale image

Bit 7



Bit 6



Bit 5



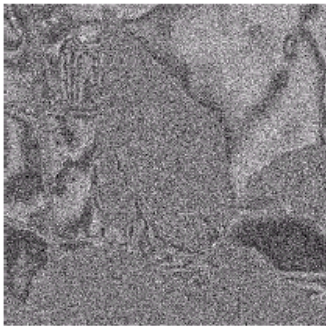
Bit 4



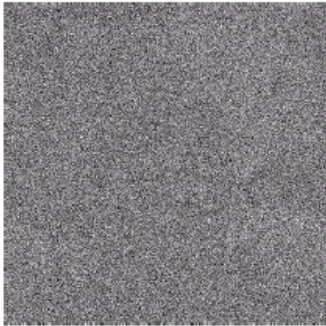
Bit 3



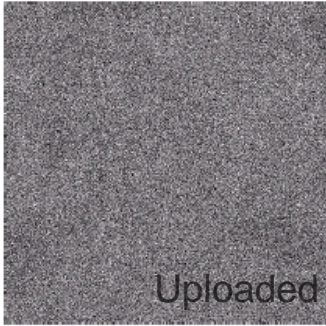
Bit 2



Bit 1



Bit 0



# Gray-coded Bit Planes

42 Original  
bit planes

$a_7$



Bit 7

$g_7$



Bit 7

$a_6$



Bit 6

$g_6$



Bit 6

$a_5$



Bit 5

$g_5$



Bit 5

$a_4$



Bit 4

$g_4$



Bit 4

Gray code:

$$g_i = a_i \otimes a_{i+1}$$

for  $0 \leq i \leq 6$

and

$$g_7 = a_7$$

$a_i$  = Original bit planes

$\otimes$  = XOR



# Gray-coded Bit Planes (cont.)

43

$a_3$



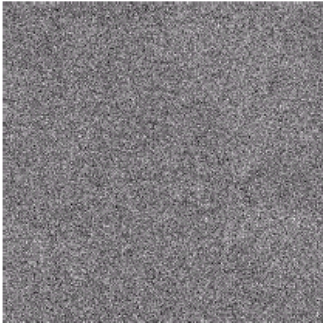
Bit 3

$a_2$



Bit 2

$a_1$



Bit 1

$a_0$

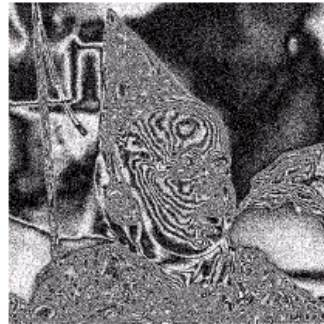


Bit 0



Bit 3

$g_3$



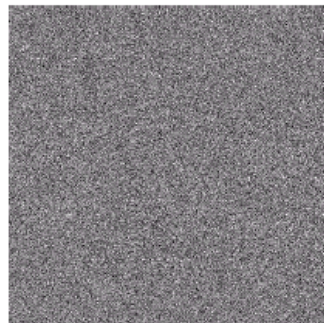
Bit 2

$g_2$



Bit 1

$g_1$



Bit 0

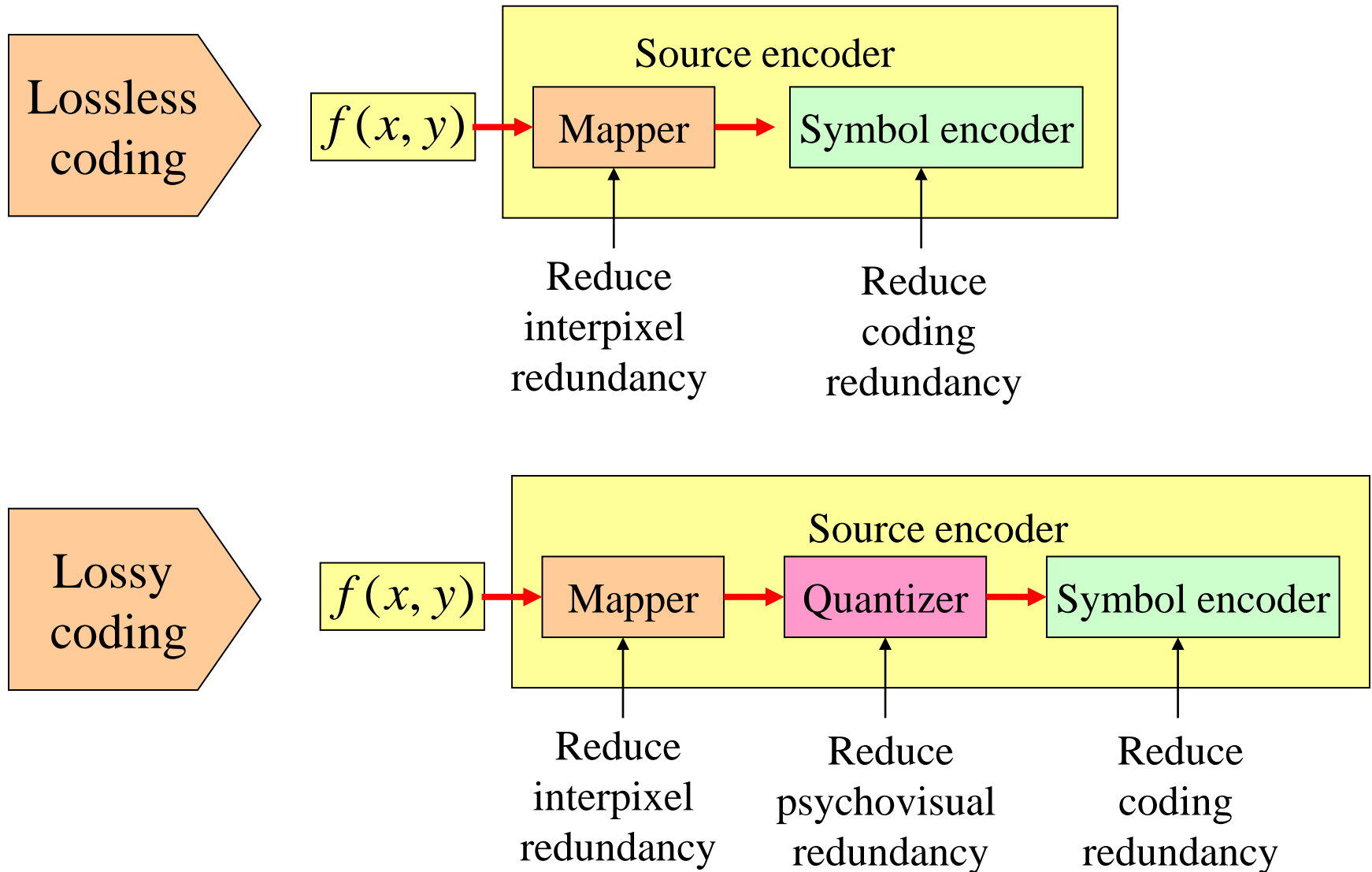
$g_0$

There are less 0-1 and 1-0 transitions in grayed code bit planes.

Hence gray coded bit planes are more efficient for coding.

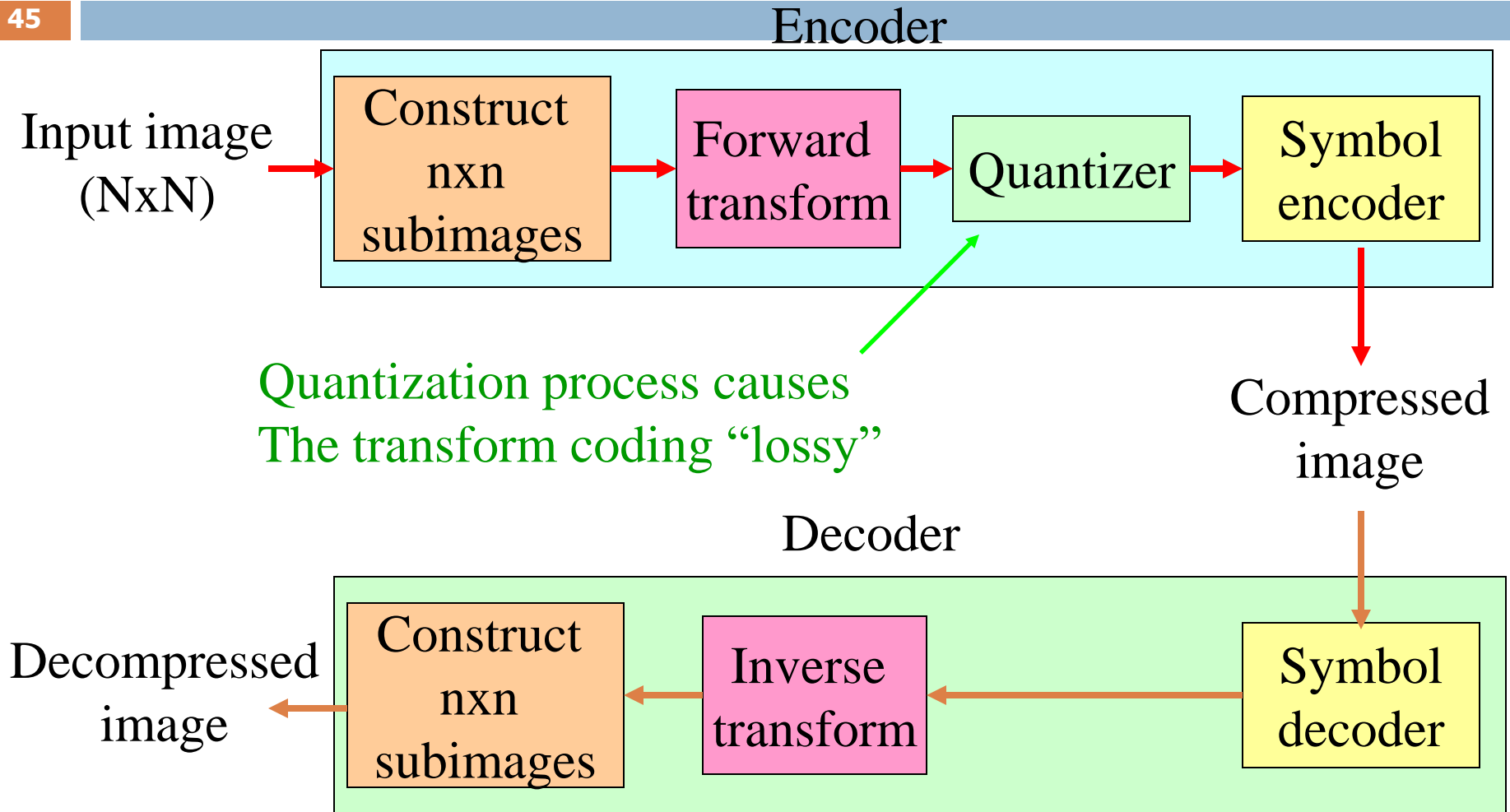
# Lossless VS Lossy Coding

44



# Transform Coding (for fixed resolution transforms)

45



Examples of transformations used for image compression: DFT and DCT

# *Transform Coding (for fixed resolution transforms)*

46

3 Parameters that effect transform coding performance:

1. Type of transformation
2. Size of subimage
3. Quantization algorithm

Forward transform:

$$T(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) g(x, y, u, v)$$

where  $g(x, y, u, v)$  = forward transformation kernel or basis function

$T(u, v)$  is called the transform coefficient image.

Inverse transform:

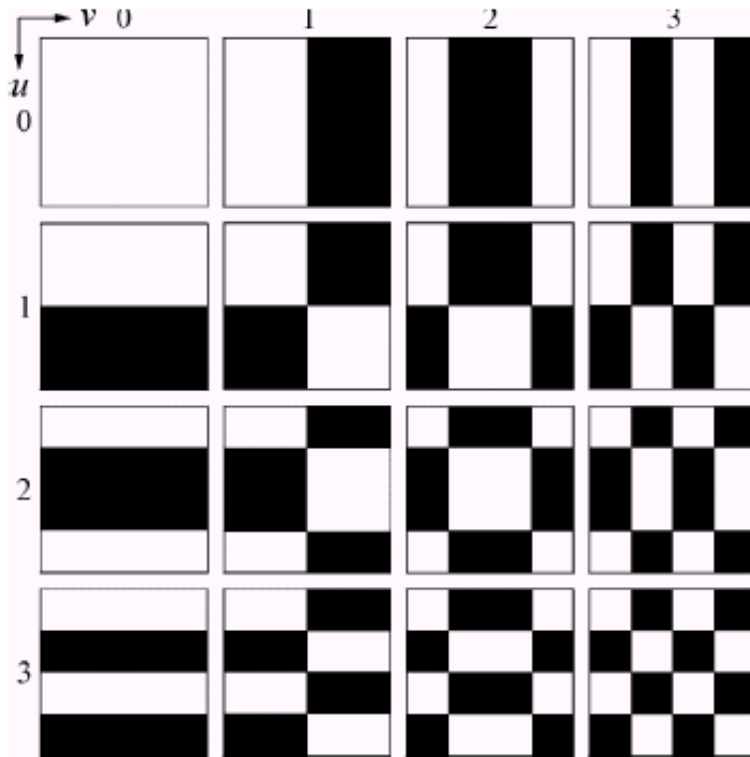
$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) h(x, y, u, v)$$

where  $h(x, y, u, v)$  = inverse transformation kernel or  
inverse basis function

# Transform Example: Walsh-Hadamard Basis Functions

48

$$g(x, y, u, v) = h(u, v, x, y) = \frac{1}{N} (-1)^{\sum_{i=0}^{m-1} [b_i(x)p_i(u) + b_i(y)p_i(v)]}$$



$N = 4$

$$N = 2^m$$

$b_k(z)$  = the  $k^{\text{th}}$  bit of  $z$

$$p_0(u) = b_{m-1}(u)$$

$$p_1(u) = b_{m-1}(u) + b_{m-2}(u)$$

$$p_2(u) = b_{m-2}(u) + b_{m-3}(u)$$

...

$$p_{m-1}(u) = b_1(u) + b_0(u)$$

Advantage: simple, easy to implement

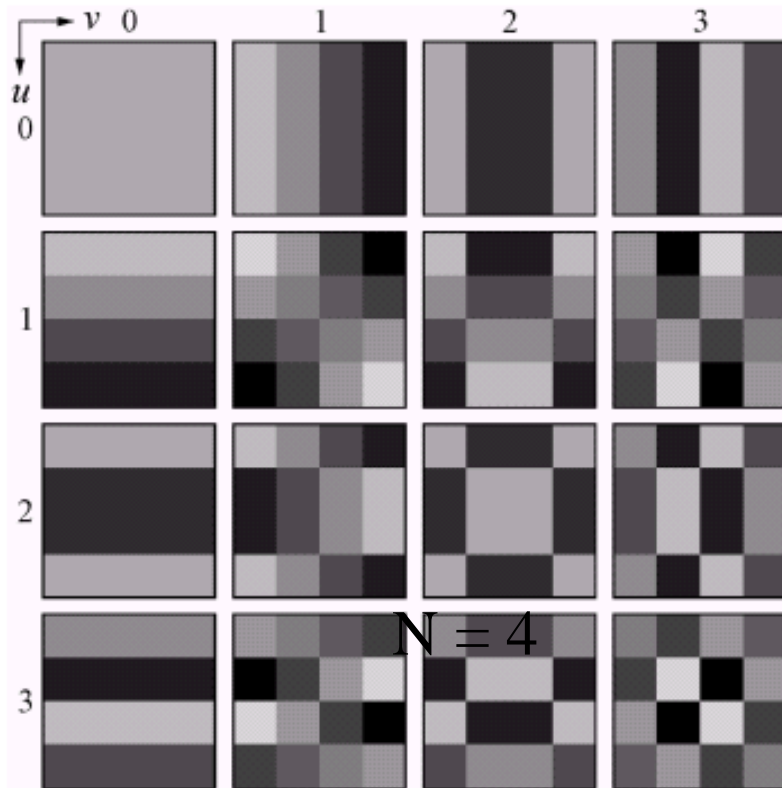
Disadvantage: not good packing ability



# Transform Example: Discrete Cosine Basis Functions

49

$$g(x, y, u, v) = h(u, v, x, y) = \alpha(u)\alpha(v) \cos\left[\frac{(2x+1)u\pi}{2N}\right] \cos\left[\frac{(2y+1)v\pi}{2N}\right]$$



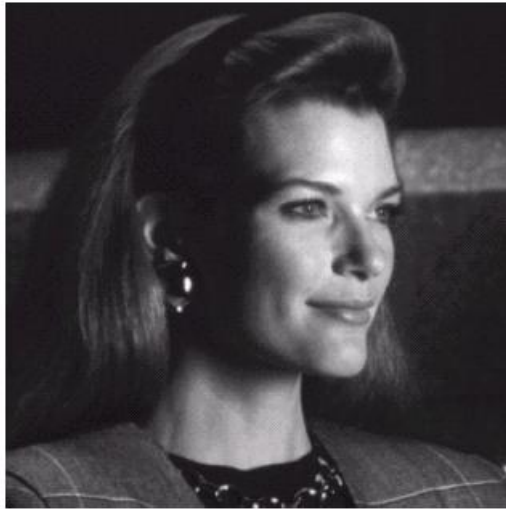
$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u = 1, \dots, N-1 \end{cases}$$

DCT is one of the most frequently used transform for image compression. For example, DCT is used in JPG files.

Advantage: good packing ability, modulate computational complexity

# Transform Coding Examples

50



Original image  
512x512 pixels

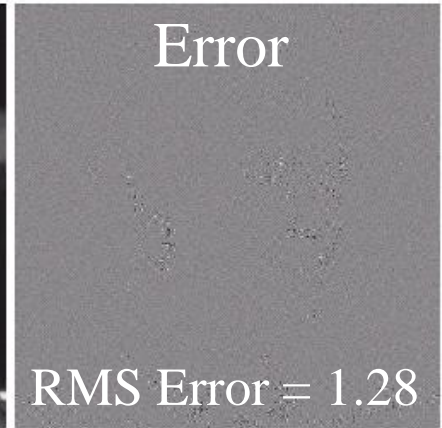
Subimage size:  
8x8 pixels = 64 pixels

Quatization by truncating  
50% of coefficients (only  
32 max coefficients are kept.)

STUDENTS-HUB.com

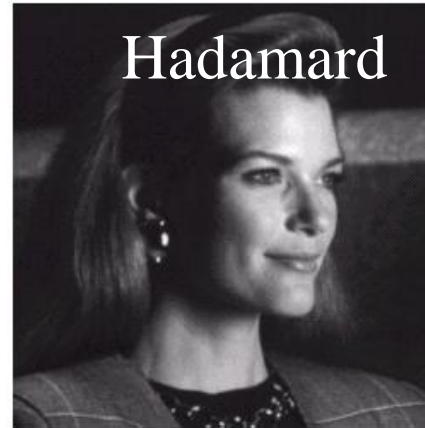


Fourier

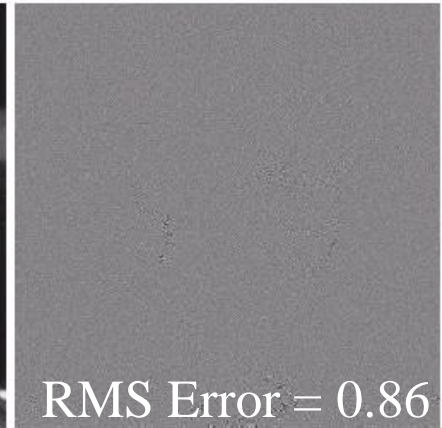


Error

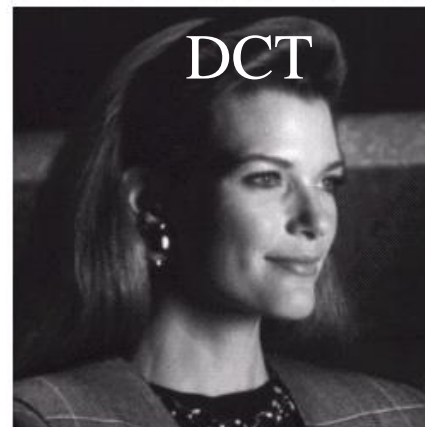
RMS Error = 1.28



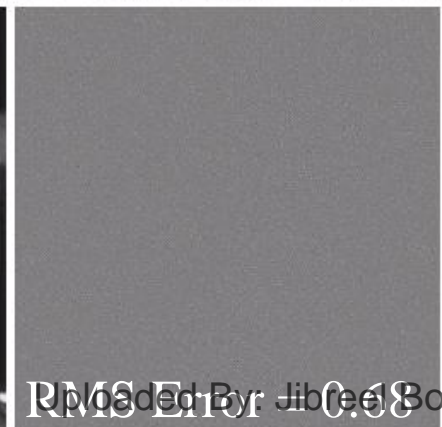
Hadamard



RMS Error = 0.86



DCT

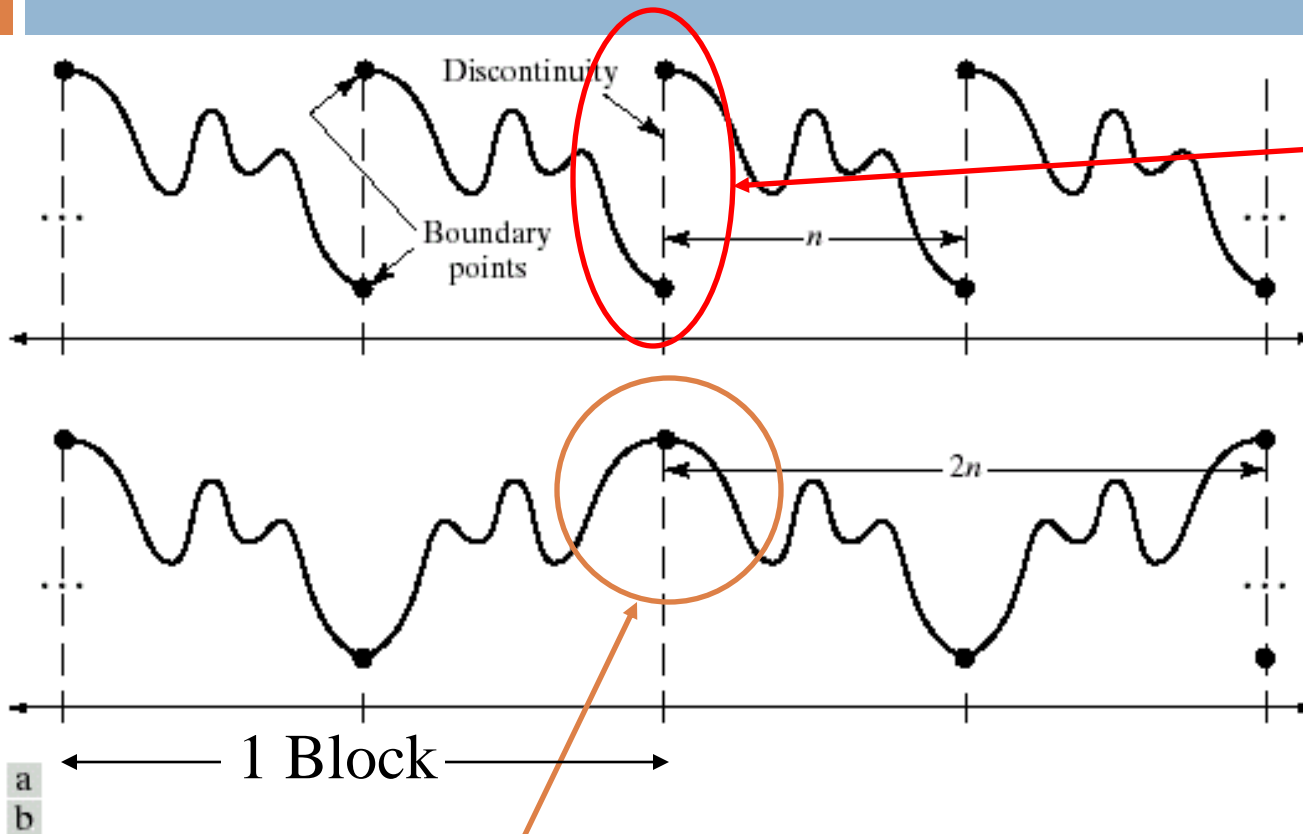


RMS Error = 0.68

Uploaded By: Jibreel Bornat

# DCT vs DFT Coding

51



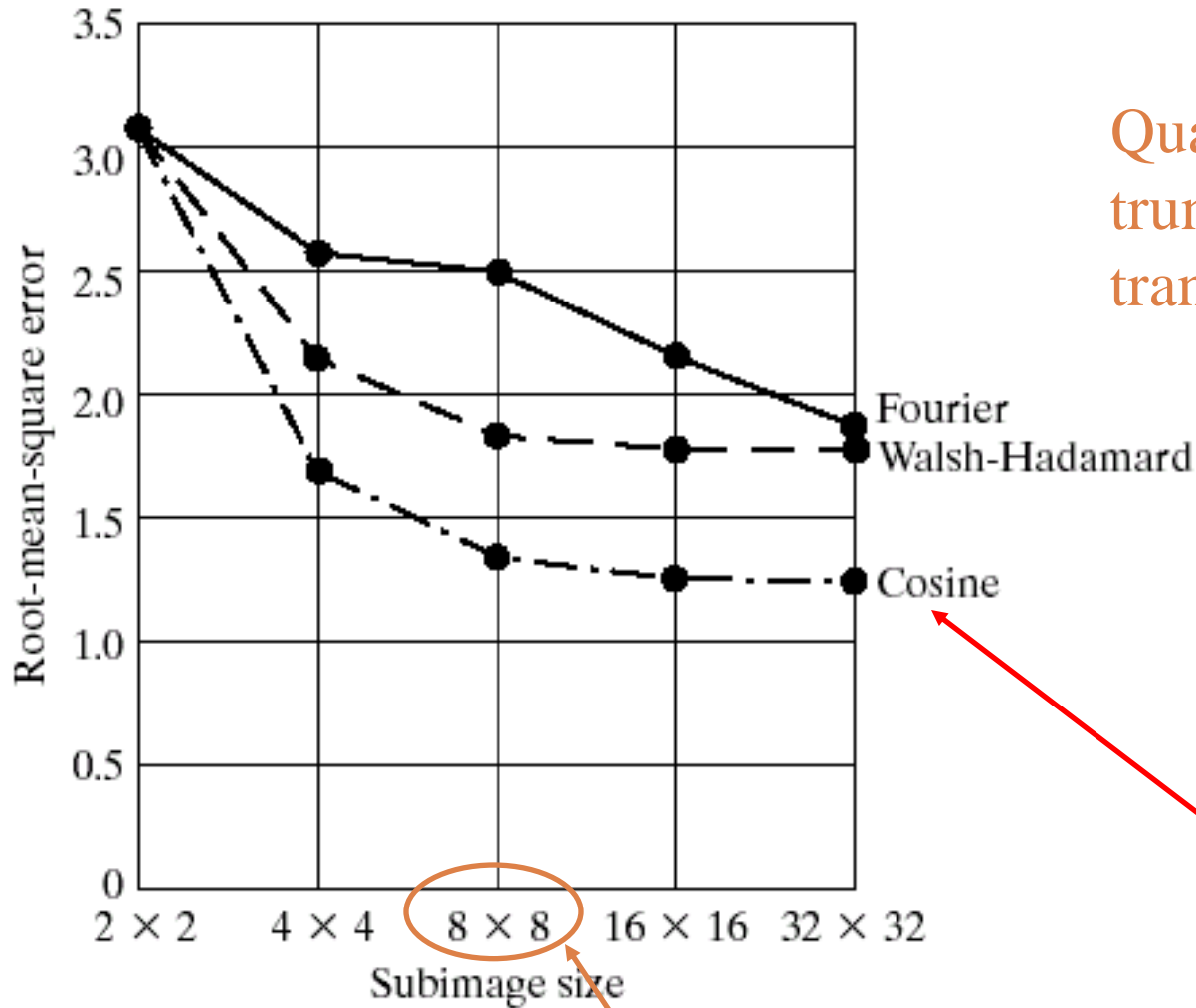
DFT coefficients have abrupt changes at boundaries of blocks

**FIGURE 8.32** The periodicity implicit in the 1-D (a) DFT and (b) DCT.

Advantage of DCT over DFT is that the DCT coefficients are more continuous at boundaries of blocks.

# Subimage Size and Transform Coding Performance

52



This experiment:  
Quatization is made by  
truncating 75% of  
transform coefficients

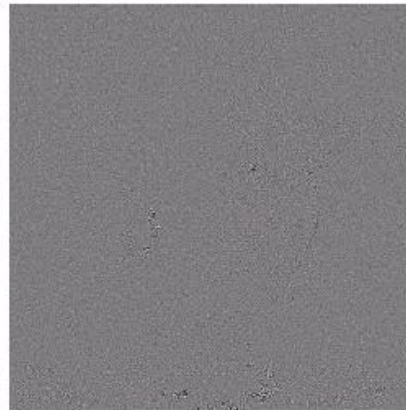
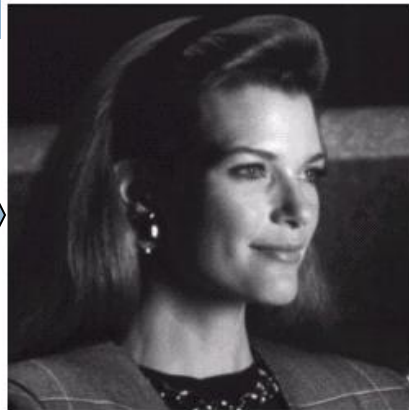
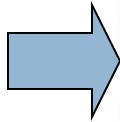
DCT is the best

Size 8x8 is enough

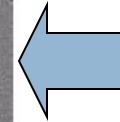
# Subimage Size and Transform Coding Performance

53

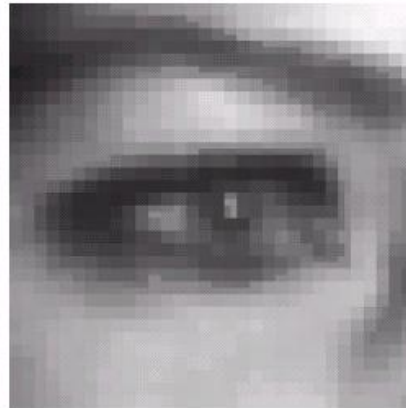
Reconstructed  
by using 25%  
of coefficients  
( $C_R = 4:1$ )  
with 8x8 sub-  
images



DCT Coefficients



Zoomed detail  
Original



Zoomed detail  
Subimage size:  
2x2 pixels

Zoomed detail  
Subimage size:  
4x4 pixels



Zoomed detail  
Subimage size:  
8x8 pixels

# Quantization Process: Bit Allocation

54

To assign different numbers of bits to represent transform coefficients based on importance of each coefficient:

- **More importance coefficients** → assign a **large number of bits**
- **Less importance coefficients** → assign a **small number of bits**  
or not assign at all

## 2 Popular bit allocation methods

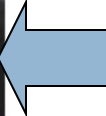
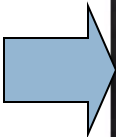
1. Zonal coding : allocate bits based on the basis of maximum variance, using fixed mask for all subimages
2. Threshold coding : allocate bits based on maximum magnitudes of coefficients



# Example: Results with Different Bit Allocation Methods

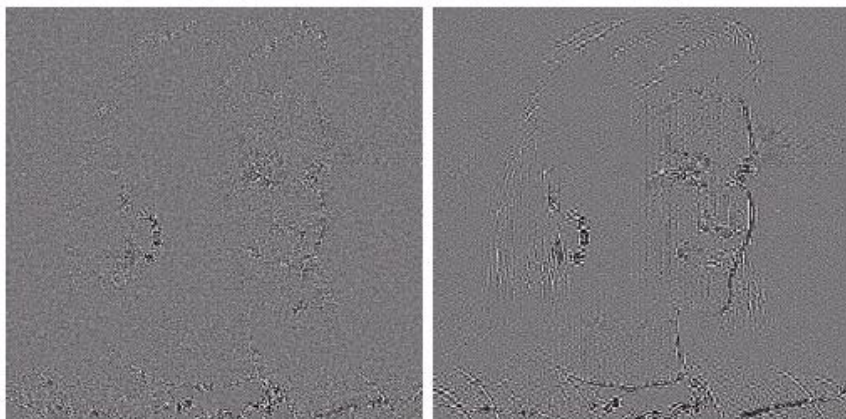
55

Reconstructed  
by using 12.5%  
of coefficients  
(8 coefficients  
with largest  
magnitude are  
used)



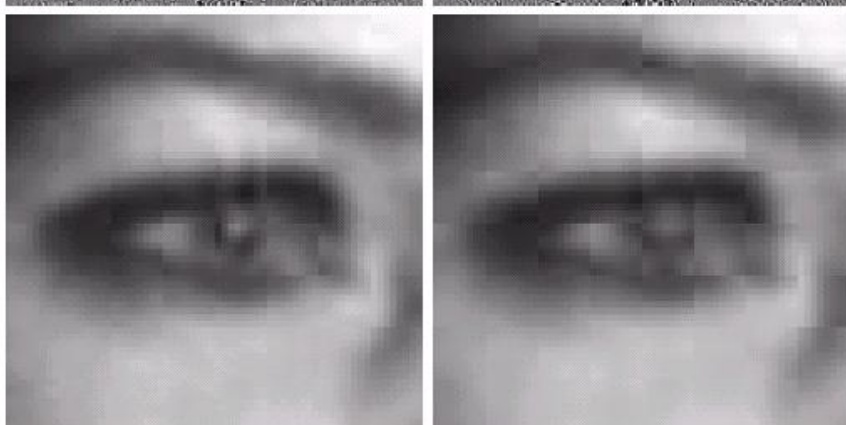
Reconstructed  
by using 12.5%  
of coefficients  
(8 coefficients  
with largest  
variance are  
used)

Threshold coding  
Error



Zonal coding  
Error

Zoom details



# Zonal Coding Example

1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	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

Zonal mask

8	7	6	4	3	2	1	0
7	6	5	4	3	2	1	0
6	5	4	3	3	1	1	0
4	4	3	3	2	1	0	0
3	3	3	2	1	1	0	0
2	2	1	1	1	0	0	0
1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0

Zonal bit allocation



# Threshold Coding Example

57

1	1	0	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	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

Threshold mask

0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

Thresholded coefficient  
ordering

# Thresholding Coding Quantization

## 3 Popular Thresholding Methods

**Method 1:** Global thresholding : Use a single global threshold value for all subimages

**Method 2:** N-largest coding: Keep only N largest coefficients

**Method 3:** Normalized thresholding: each subimage is normalized by a normalization matrix before rounding

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

Bit allocation

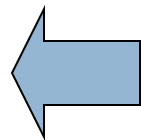


$$\hat{T}(u, v) = \text{round} \left[ \frac{T(u, v)}{Z(u, v)} \right]$$

Restoration before decompressing



$$\tilde{T}(u, v) = \hat{T}(u, v) Z(u, v)$$



Example of  
Normalization  
Matrix  $Z(u, v)$

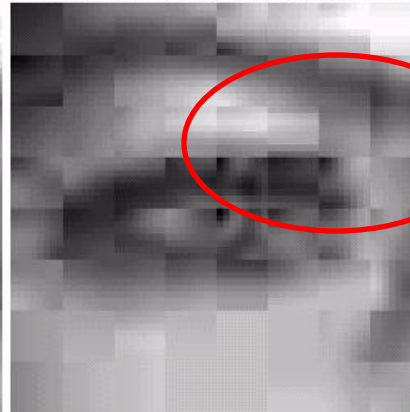
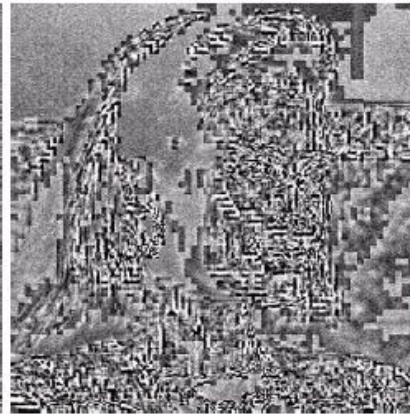
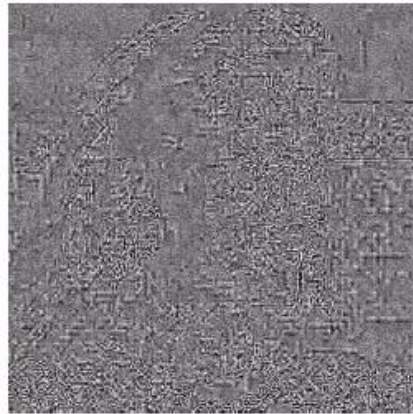
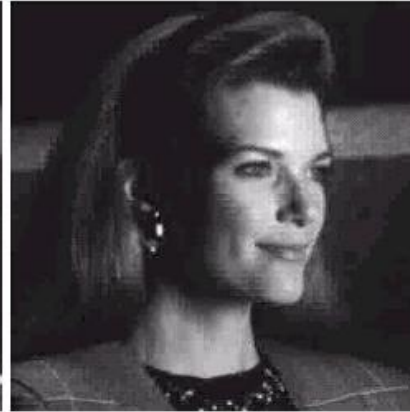
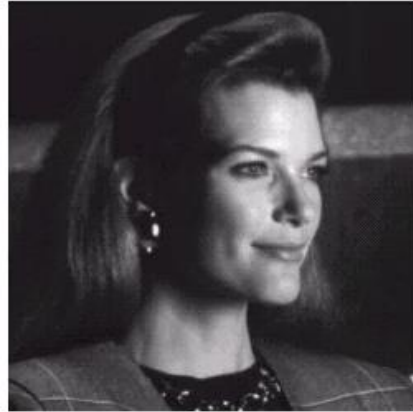
# DCT Coding Example

59

$(C_R = 38:1)$

Error image  
RMS Error = 3.42

Zoom details



$(C_R = 67:1)$

Method:

- Normalized Thresholding,
- Subimage size: 8x8 pixels

Blocking  
Artifact at  
Subimage  
boundaries