Image Processing and Pattern Recognition (IPPR)

Chapter5:Image Compression

Basanta Joshi, PhD

Asst. Prof., Depart of Electronics and Computer Engineering

Program Coordinator, MSc in Information and Communication Engineering

Member, Laboratory for ICT Research and Development (LICT)

Member, Research Management Cell (RMC)

Institute of Engineering

basanta@ioe.edu.np

http://www.basantajoshi.com.np

https://scholar.google.com/citations?user=iocLiGcAAAAJ
https://www.researchgate.net/profile/Basanta_Joshi2

Contents

- Need of Compression
- Coding Redundancy: Huffman Coding
- Interpixel Redundancy: run length coding
- Psychovisual Redundancy: bit plane
- Image Compression Models
- Lossy & Lossless compression: Transform coding and predictive coding.

Why we need image compression?

- One 90 minutes color movie, each second plays 24 frames. When we digitize it, each frame has 512×512 pixels, each pixel has three components R, G, B each one occupies 8 bits respectively, the total byte number is:

  $90 \times 60 \times 24 \times 3 \times 512 \times 512 = 97{,}200\text{MB}$

- A CD may save 600 megabytes data, the movie needs 160 CDs to save.

Image Compression

Objective: To reduce the amount of data required to represent an image.

Important in data storage and transmission

- Progressive transmission of images (internet, www)
- Video coding (HDTV, Teleconferencing)
- Digital libraries and image databases
  - Medical imaging
  - Satellite images

Lossy Vs. Lossless Compression

Compression techniques

Information preserving

(loss-less)

Images can be compressed and restored without any loss of information.
Application: Medical images, GIS

Lossy

Perfect recovery is not possible but provides a large data compression.
Example : TV signals, teleconferencing

Image Compression

•*Data compression* refers to the process of reducing the amount of data required to represent a given quantity of information

- *data* are the means where *information* is conveyed
- relative data redundancy R_D:

$$R_D = 1 - 1/C_R$$

where C_R is the compression ratio:

$$C_R = n_1 / n_2,$$

$$C_R = \frac{No\ of\ bits\ used\ to\ represent\ image\ before\ compression}{No\ of\ bits\ used\ to\ represent\ image\ after\ compression} \qquad C_R >= 1$$

Image Compression

- Three basic data redundancy can be identified and exploited
  - *coding redundancy*
    - to reduce the amount of data representing the same information
  - *interpixel redundancy*
    - including spatial redundancy, geometric redundancy and interframe redundancy
  - *psychovisual redundancy*
    - certain information has less relative importance in normal visual processing and can be eliminated

Coding Redundancy

• Assume a discrete random variable r_k in the interval [0,1] represents the grey levels of an image and that each r_k occurs with probability $p_r(r_k)$:

$$p_r(r_k) = \frac{n_k}{n} \quad k = 0,1,\ldots, L-1$$

and

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

is the average number of bits used to represent each pixel, where $l(r_k)$ is the number of bits required for each value r_k

Coding Redundancy

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

TABLE 8.1 Example of variable-length coding.

$$L_{avg} = \sum_{k=0}^{7} l_2(r_k)\, p_r(r_k)$$

$$= 2(0.19) + 2(0.25) + 2(0.21) + 3(0.16) + 4(0.08)$$

$$+ 5(0.06) + 6(0.03) + 6(0.02)$$

$$= 2.7 \text{ bits},$$

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

Coding Redundancy

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

L_{avg} 3 bits/symbol

L_{avg} 2.7 bits/symbol

Coding Redundancy

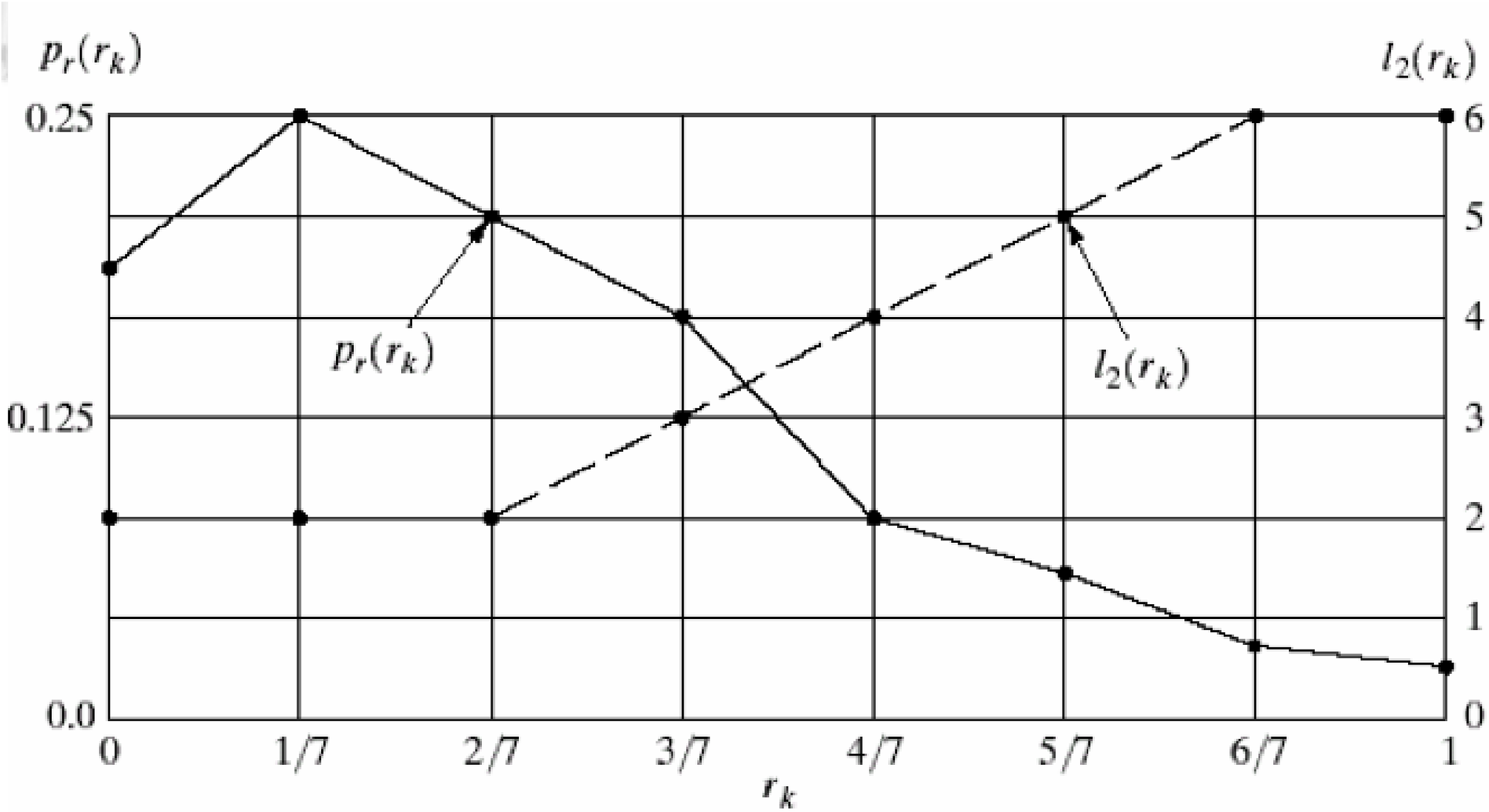


FIGURE 8.1 Graphic representation of the fundamental basis of data compression through variable-length coding.

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

Huffman coding

Huffman code: Consider a 6 symbol source

	a_1	a_2	a_3	a_4	a_5	a_6
$p(a_i)$	0.1	0.4	0.06	0.1	0.04	0.3

Huffman coding

- Huffman coding: give the smallest possible number of code symbols per source symbols.

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	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

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	0.4	1	0.6	0
a_6	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
a_1	0.1	011	0.1	011	0.2	010	0.3	01		
a_4	0.1	0100	0.1	0100	0.1	011				
a_3	0.06	01010	0.1	0101						
a_5	0.04	01011								

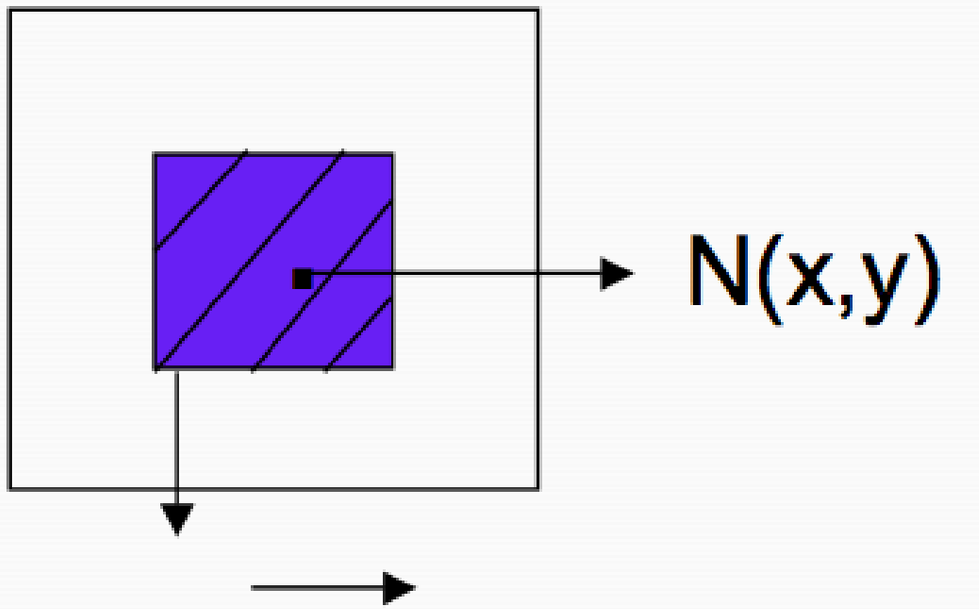
The code is instantaneous uniquely decodable without referencing succeeding symbols.

Average length:

(0.4) (1) + 0.3 (2) + 0.1 X 3 + 0.1 X 4 + (0.06 + 0.04) 5 = 2.2 bits/symbol

Interpixel / Interframe Redundancy

spatial

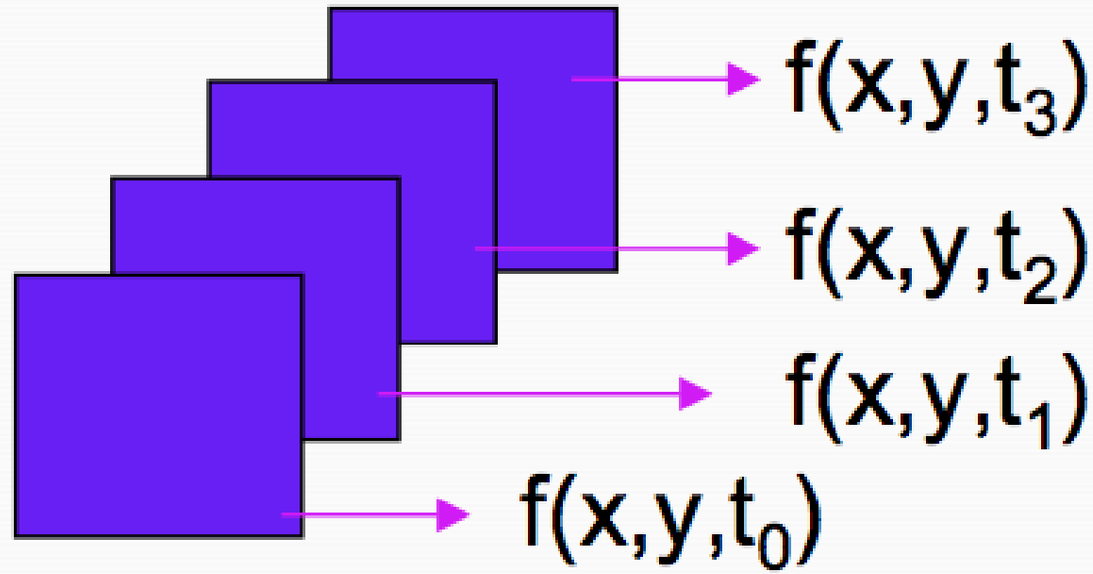


f(x,y) Depends on

f (x', y') , (x', y') ε N_{xy}

N_{xy} : Neighbourhood of pixels around (x,y)

Interframe



f(x, y, t_i) i=1, 2, 3, . . .

are related to each other.

This can be exploited for video compression.

Interpixel / Interframe Redundancy

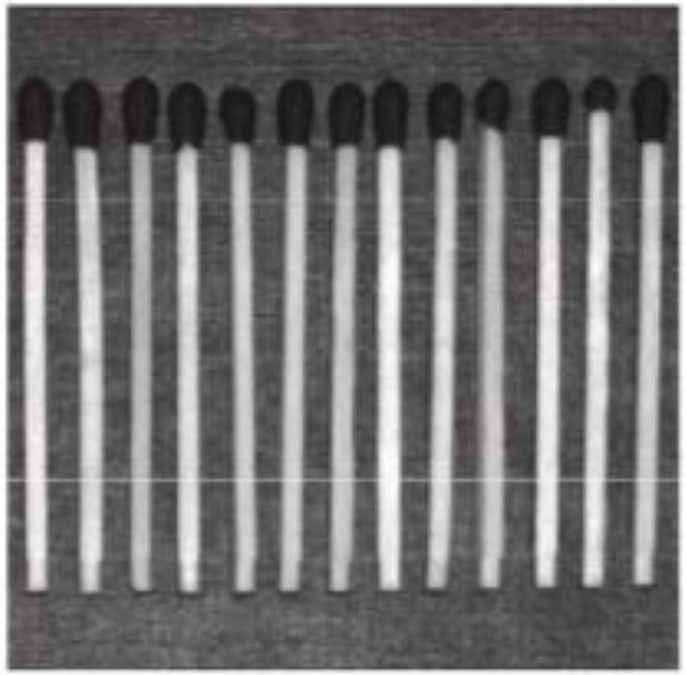
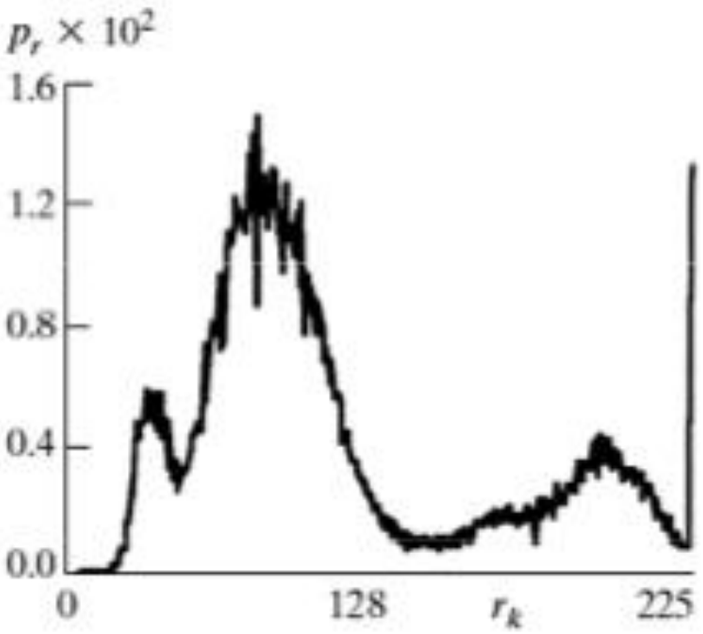
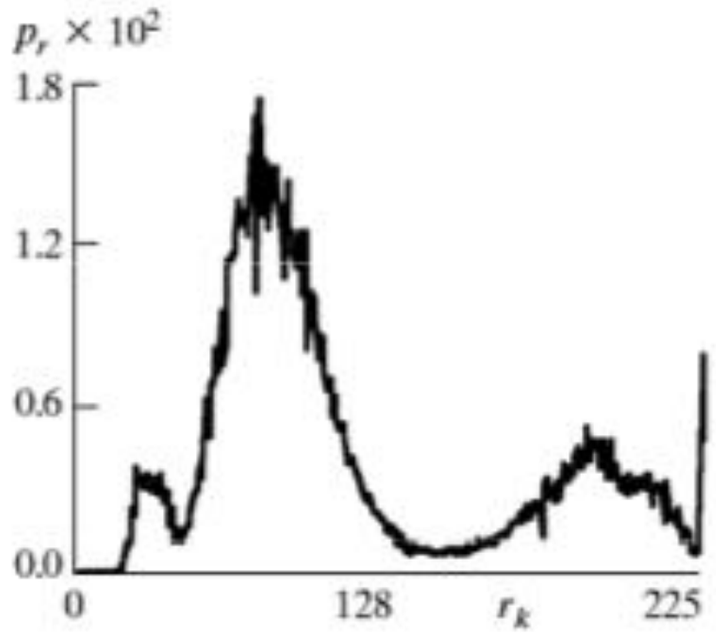


Figure 8.2 Two images and their grey-level pixel histograms and normalized autocorrelation coefficients along one line.





a b
c d

Interpixel / Interframe Redundancy

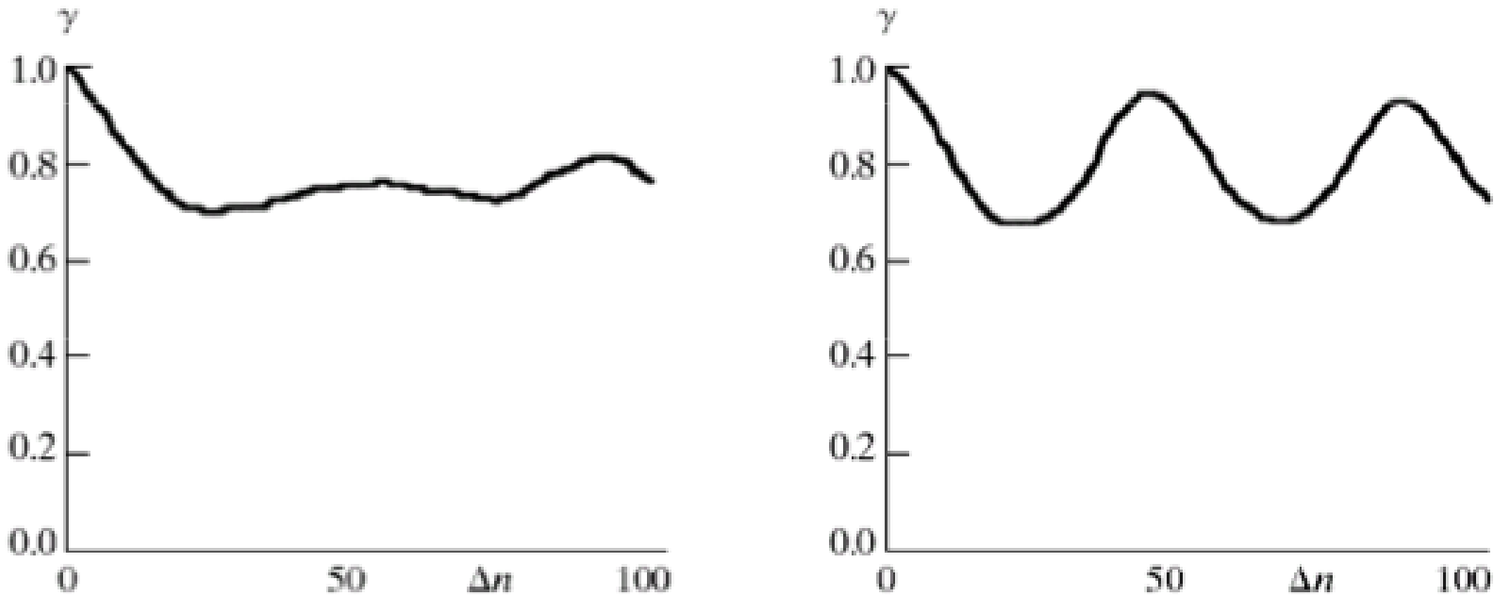


Figure 8.2 Two images and their grey-level pixel histograms and normalized autocorrelation coefficients along one line.

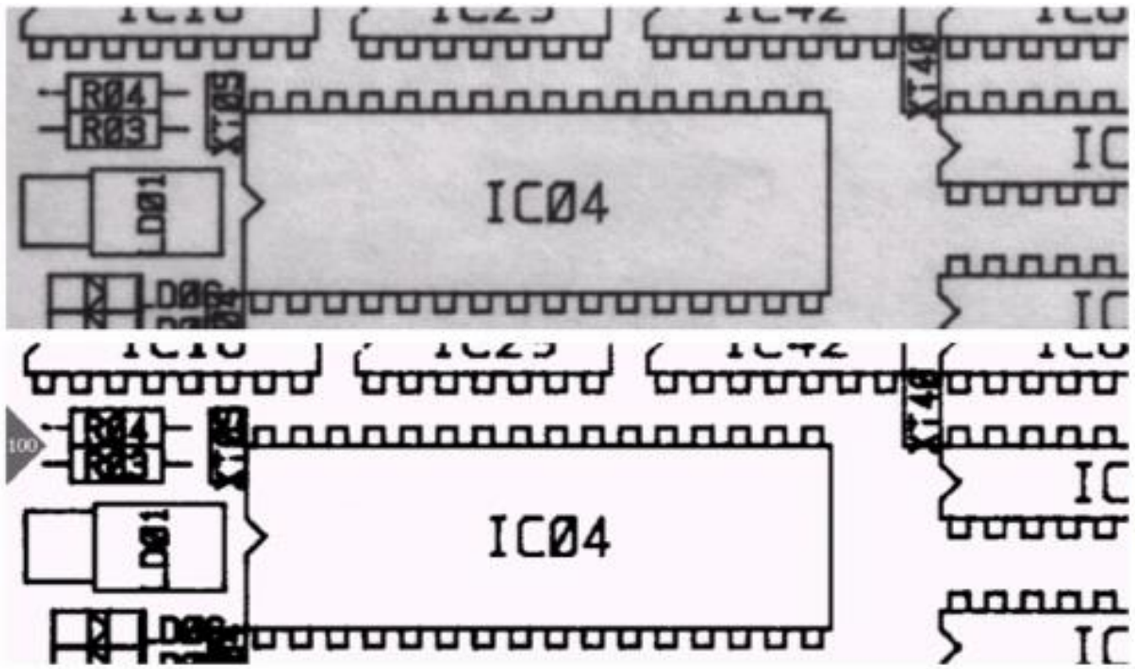
$$r(\Delta n) = \frac{A(\Delta n)}{A(0)},$$

$$where \quad A(\Delta n) = \frac{1}{N - \Delta n} \sum_{y=0}^{N-1-\Delta n} f(x, y) f(x, y + \Delta n)$$

Interpixel redundancy: Parts of an image are highly correlated.

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

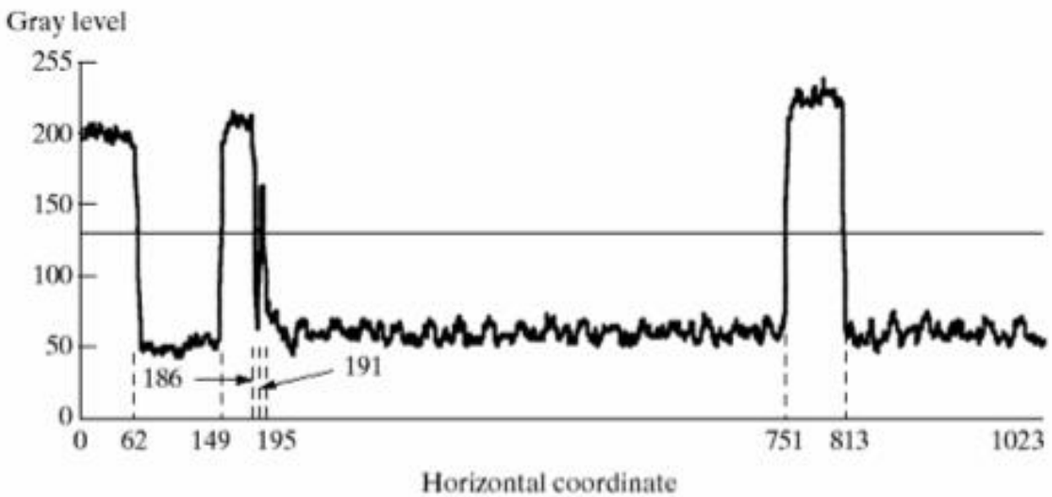
Run Length Coding



The gray scale image
of size 343x1024 pixels

Binary image
= 343x1024x1 = 351232 bits

$$C_R = \frac{(1024)(343)(1)}{(12166)(11)} = 2.63$$



Line No. 100

$$R_D = 1 - \frac{1}{2.63} = 0.62$$

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 → Total = 133826 Bits

Psychovisual Redundancy

- Certain information simply has less relative importance than other information in normal visual processing
- This information is said to be psychovisually redundant
  - can be eliminated without significantly impairing the quality of image perception

Psychovisual Redundancy

8-bit gray scale image

4-bit gray scale image

4-bit IGS image

False contours

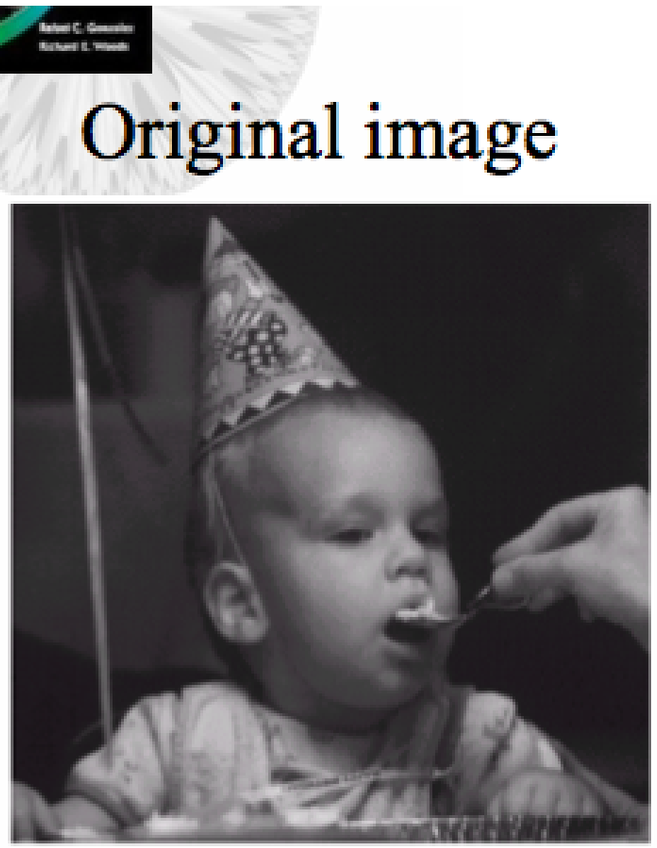
Psychovisual Redundancy

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

TABLE 8.2
IGS quantization procedure.

A sum-initially to zero-is first formed from the current 8-bit grey-level value and the four least significant bits of a previously generated sum. If the 4 most significant bits of the current value are 1111_2, however, 0000_2 is added instead. The 4 most significant bits of the resulting sum are used as the coded pixel value.

Bit Plane Coding



Example of binary image compression: Run length coding

Bit Plane Coding

Original gray scale image

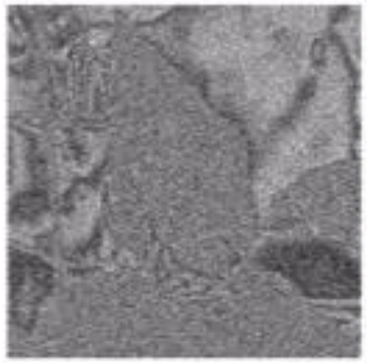
Bit 7

Bit 6

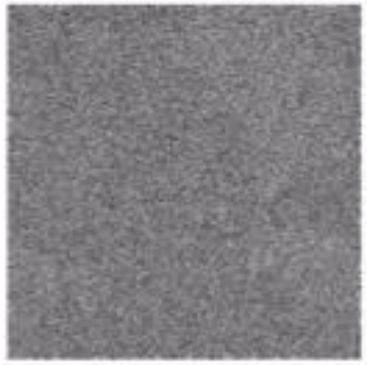
Bit 5

Bit 4

Bit 3



Bit 2



Bit 1

Bit 0

mages from Rafael C. Gonzalez and Richard E. Wood, Digital Image rocessing, 2nd Edition.

Gray Coded Bit Planes

Original bit planes

Gray code:

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

for $0 \le i \le 6$

and

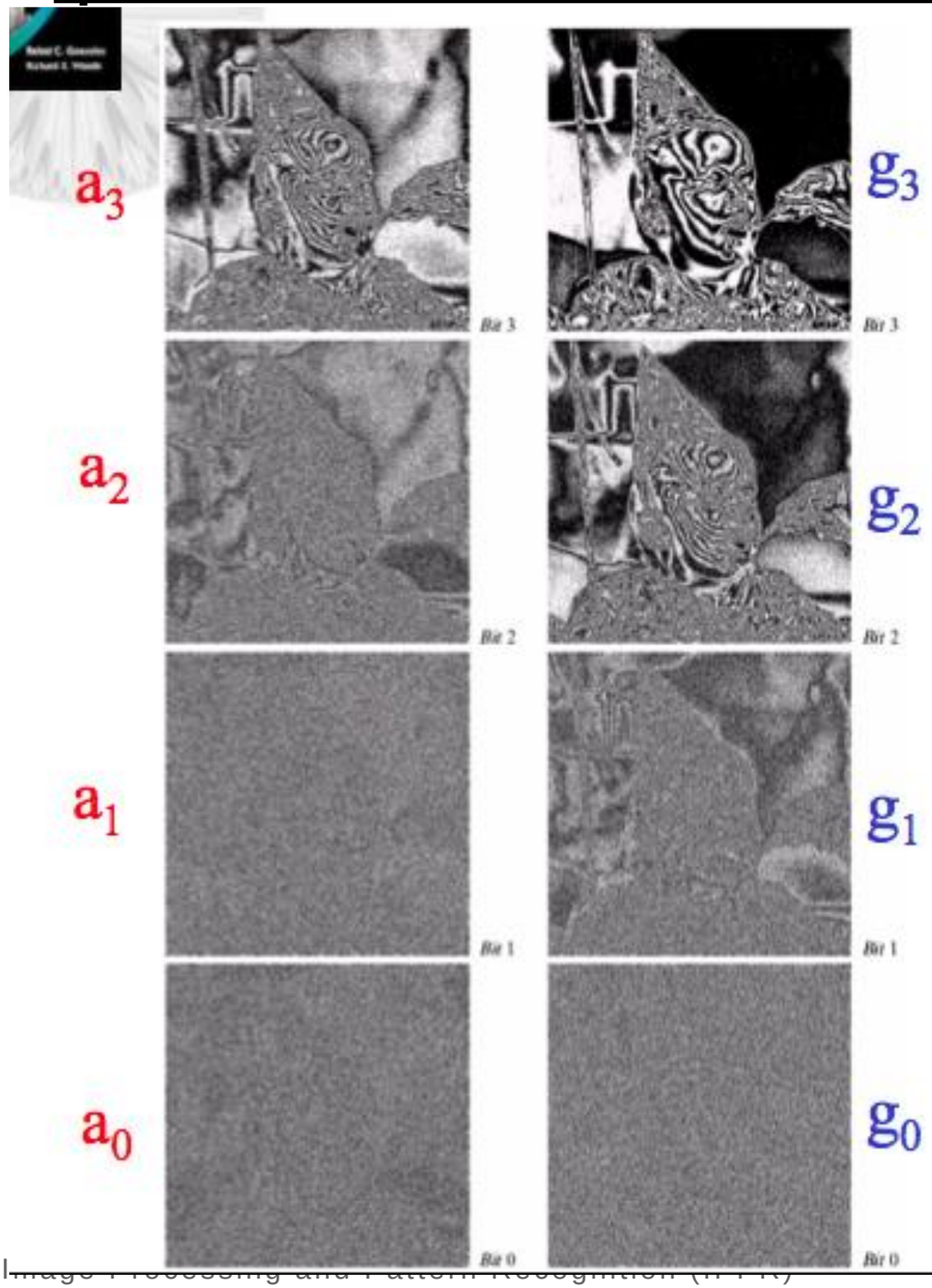
$$g_7 = a_7$$

a_i= Original bit planes

$\otimes$ = XOR

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

Gray Coded Bit Planes

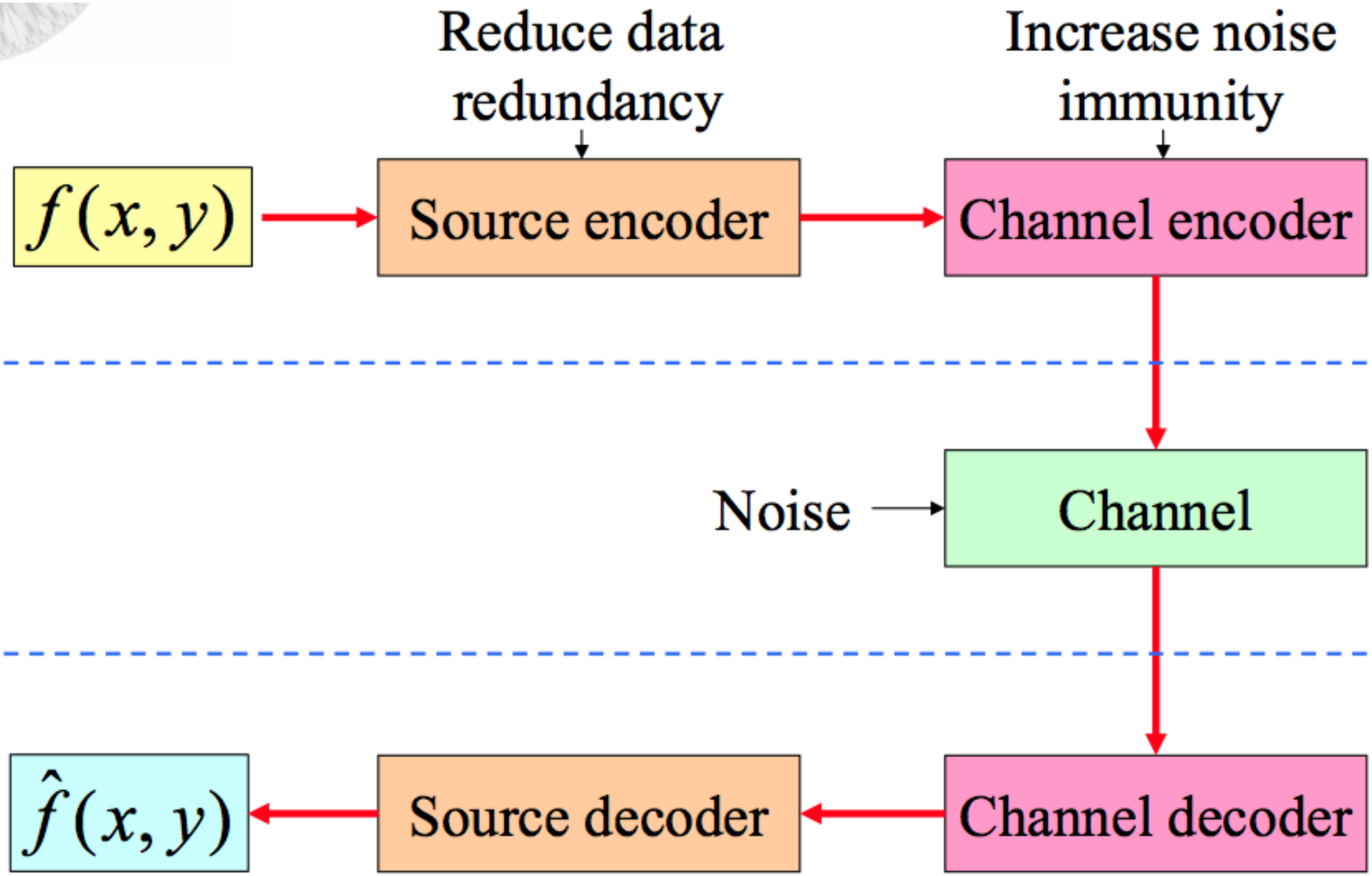


There are less 0-1 and 1-0 transitions in grayed code bit planes.

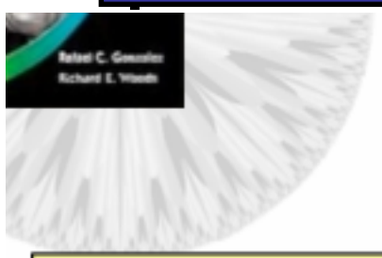
Hence gray coded bit planes are more efficient for coding.

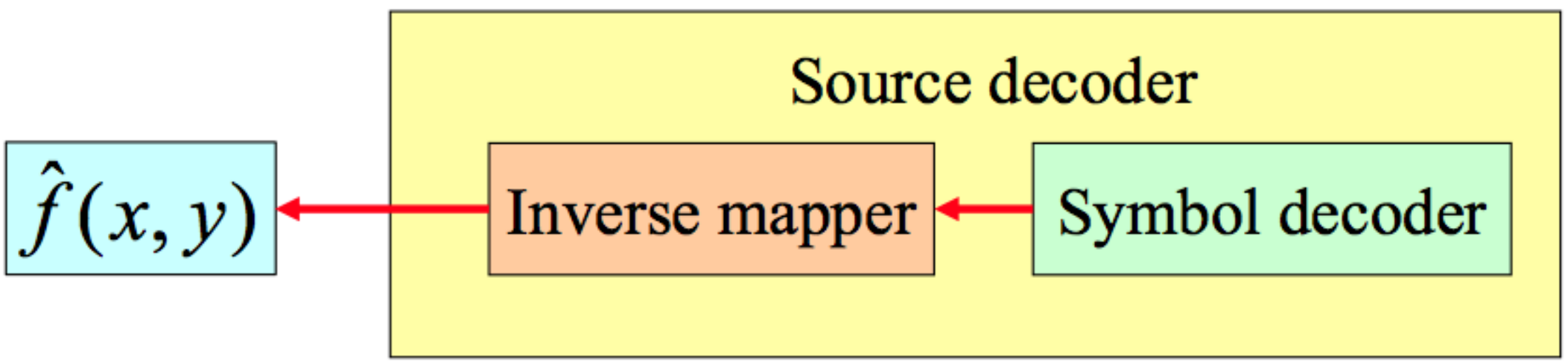
(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.

Image Compression Models



Source Encoder and Decoder Models





Error-Free Compression (Lossless)

- Error-free compression means lossless compression
- In numerous applications error-free compression is the only acceptable means of data reduction.
- They normally provide compression ratios of 2 to 10.

Lossless Predictive Coding Model

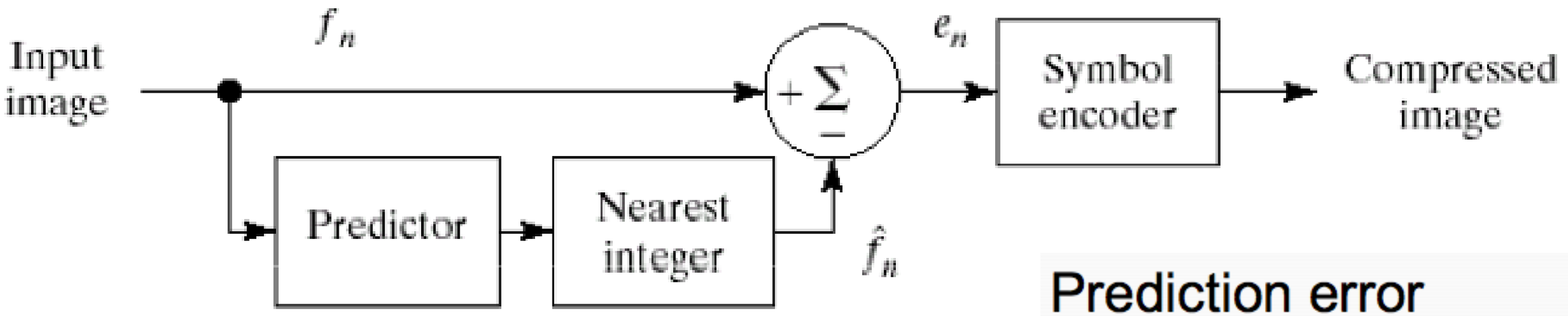


FIGURE 8.19 A lossless predictive coding model: (a) encoder; (b) decoder.

Prediction error

$$e_n = f_n - \hat{f}_n$$

Prediction error:

$e_n = f_n - \hat{f}_n$

e_n is coded using a variable length code

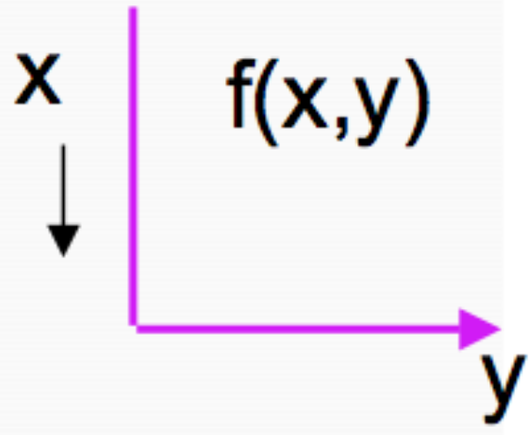
$f_n = e_n + \hat{f}_n$

Lossless Predictive Coding Model

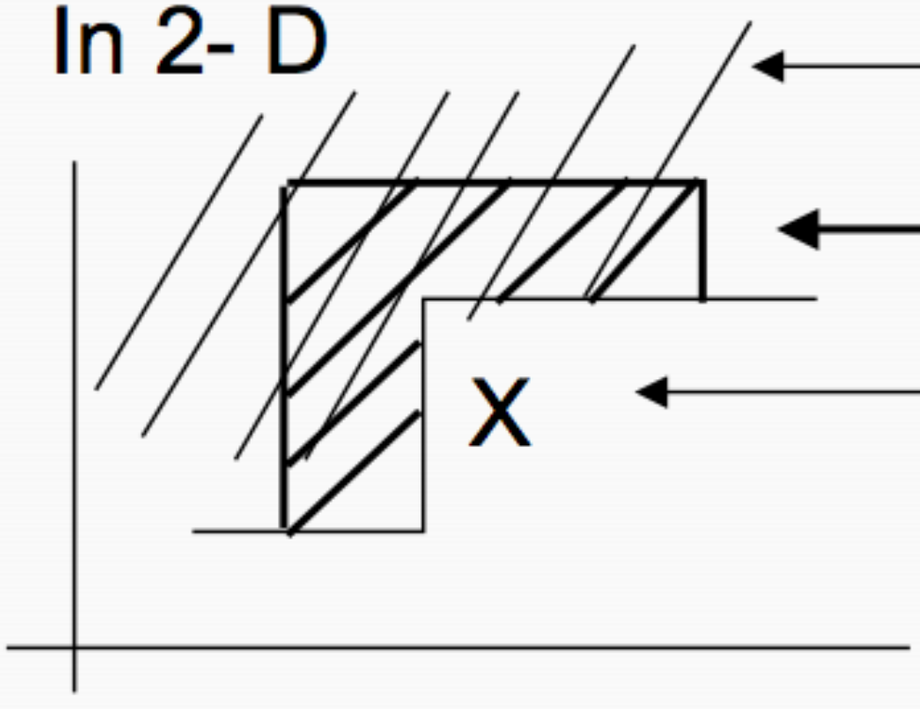
Example 1: $\hat{f}_n = \text{Int}\left(\sum_{i=1}^{m} \alpha_i \, f_{n-i}\right)$

→ Linear predictor ; m = order of predictor

Example 2: $\hat{f}_n(x,y) = \text{Int}\left(\sum_{i=1}^{m} \alpha_i \, f(x, y-i)\right)$



In 2- D



Past information

W_{xy}

Current pixel f (x,y)

$= \sum \alpha (x', y') f (x', y)$

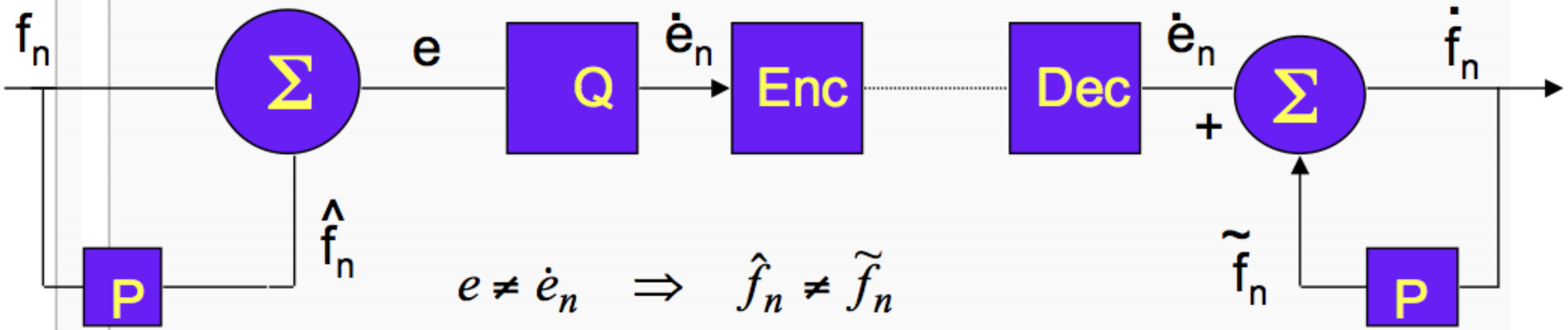
$(x', y') \varepsilon W_{xy}$

Lossy Compression

- Unlike the error-free approaches, lossy encoding is based on the concept of compromising the accuracy of the reconstructed image in exchange for increased compression.

Lossy Compression

Lossy compression: uses a quantizer to compress further the number of bits required to encode the 'error'.
First consider this:



Notice that, unlike in the case of loss-less prediction, in lossy prediction the predictors P "see" different inputs at the encoder and decoder

Lossy Compression

This results in a gradual buildup of error which is due to the quantization error at the encoder site.
In order to minimize this buildup of error due to quantization we should ensure that 'Ps' have the same input in both the cases.

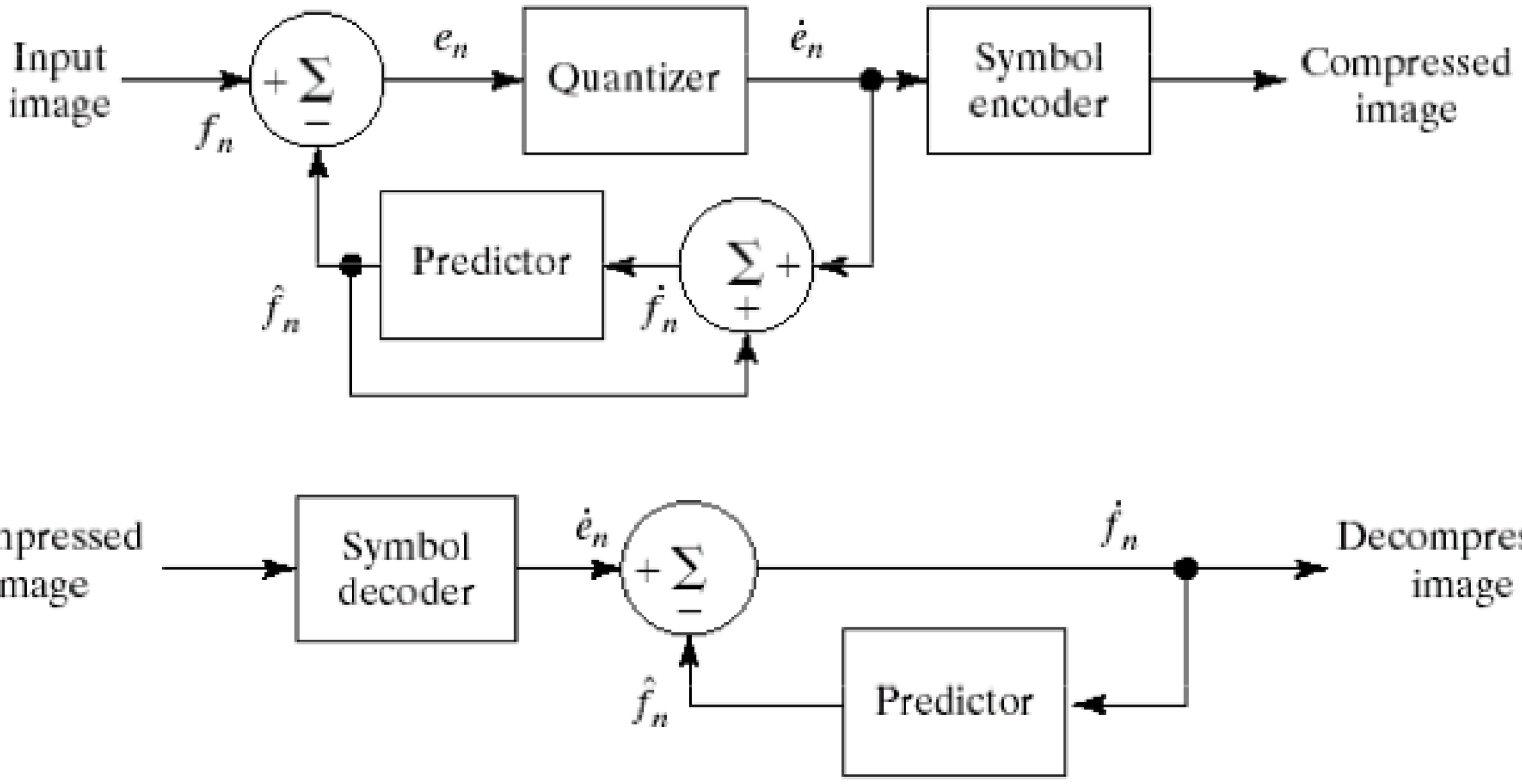
f_n	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$\hat{f}_n = f_{n-1}$															
$e_n =$	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
$\dot{e}_n =$	0	2	2	2	2	.	.	.	.	.					
$\dot{f}_n =$	0	2	4	6	8	10	.	.	.	.	.	.			

Lossy Compression

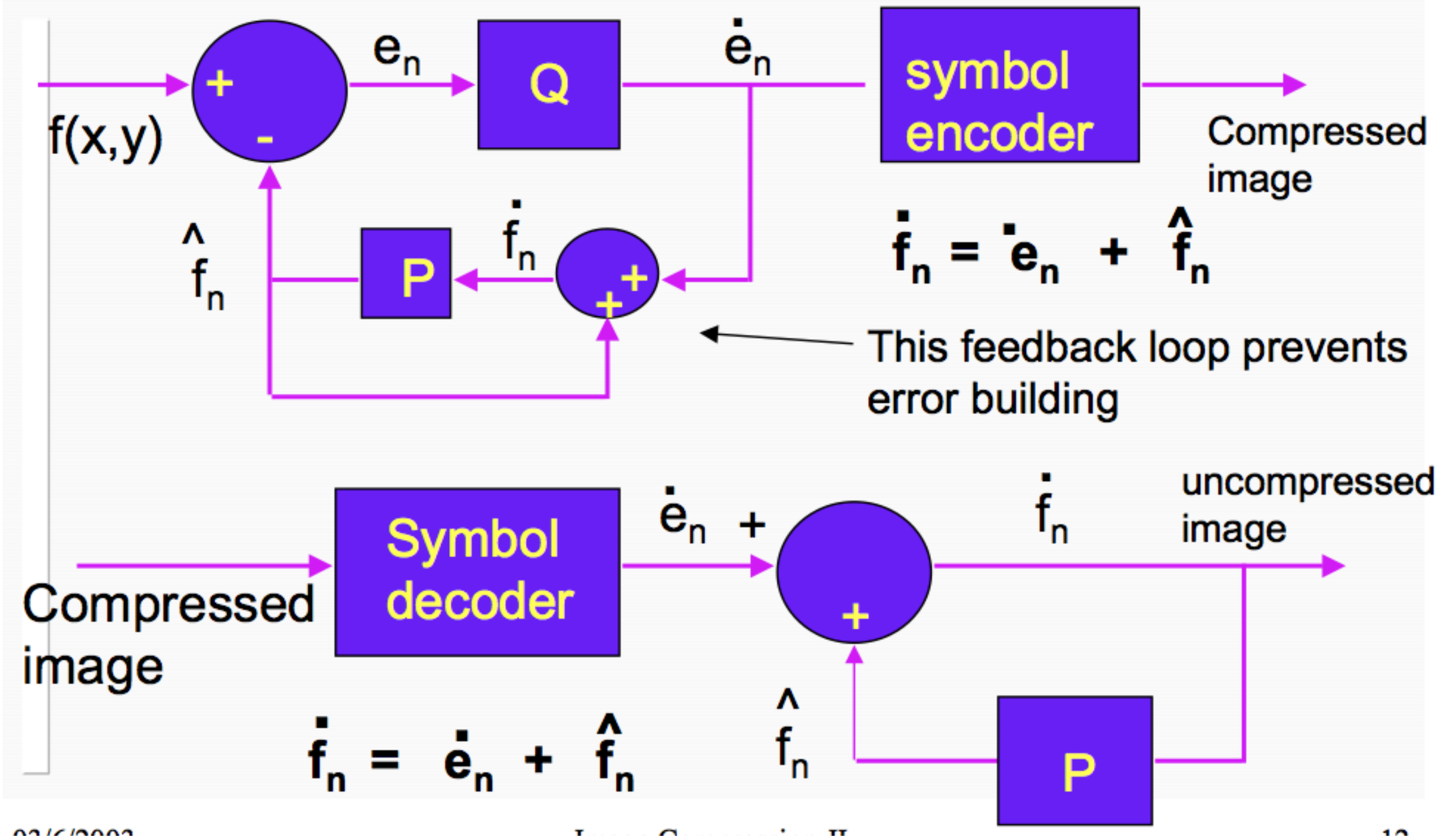
n	f_n	$\hat{f}_n$	e_n	$\dot{e}_n$	$\dot{f}_n$
0	0	-	-	-	0
1	1	0	1	2	2
2	2	1	1	2	4
3	3	2	1	2	6
4	4	3	1	2	8
5	5	4	1	2	10
6	6	5	1	2	12

$$\dot{f}_n = \dot{e}_n + \tilde{f} = \dot{e}_n + \dot{f}_{n-1}$$

Lossy Compression



Lossy Compression



Lossy Compression

Example:

$$\hat{f}_n = \alpha \, \dot{f}_{n-1}$$

$$\text{and} \quad \dot{e}_n = \begin{cases} +\xi & e_n > 0 \\ -\xi & e_n < 0 \end{cases}$$

$0 < \alpha < 1$
prediction coefficient

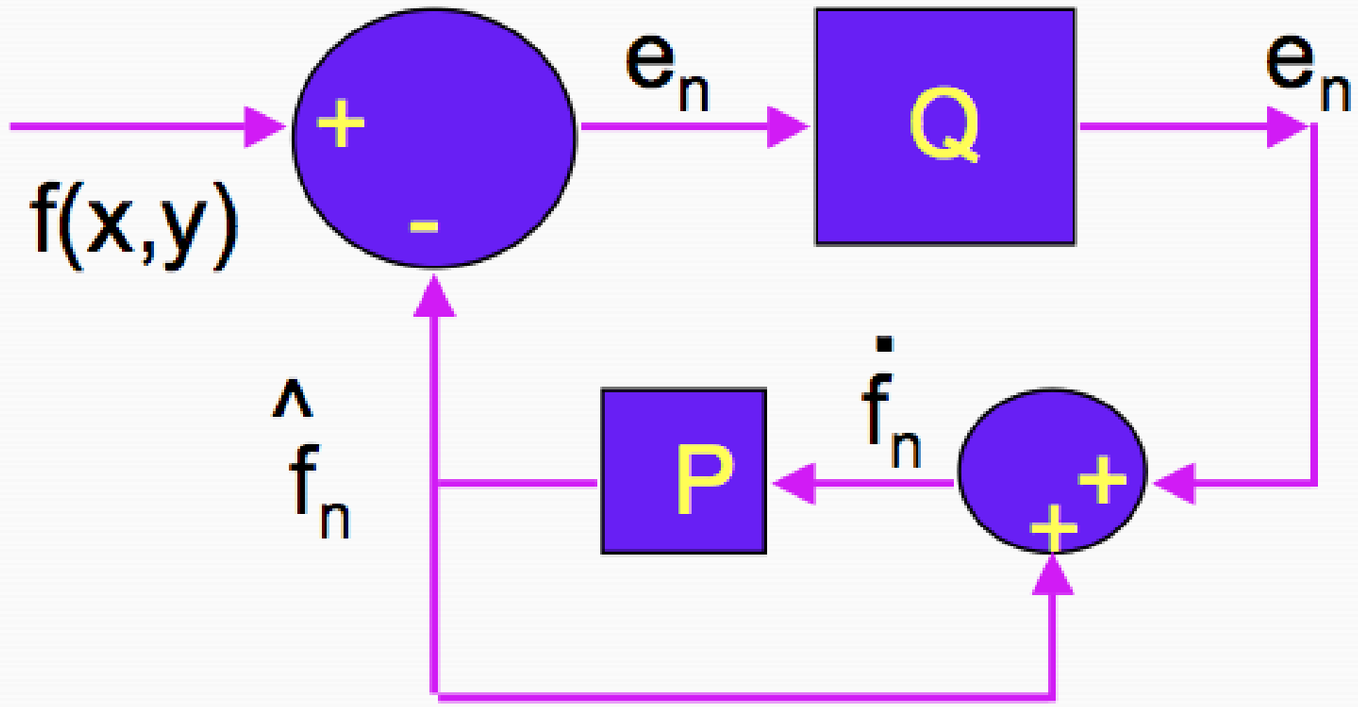
$$\dot{f}_n = \dot{e}_n + \hat{f}_n$$

$$= \dot{e}_n + \alpha \, \dot{f}_{n-1}$$

Lossy Compression

with feedback

f_n =	0	1	2	3	4
e_n =	_	1	2	1	2
$\dot{e}_n$ =	_	0	2	0	2
$\dot{f}_n$ =	0	0	2	2	4
$\hat{f}_n$ =	_	0	0	2	2



Note: The quantizer used here is-- floor $(e_n/2)*2$. This is different from the one used in the earlier example. Note that this would result in a worse response if used without Feedback (output will be flat at "0").

Lossy Compression

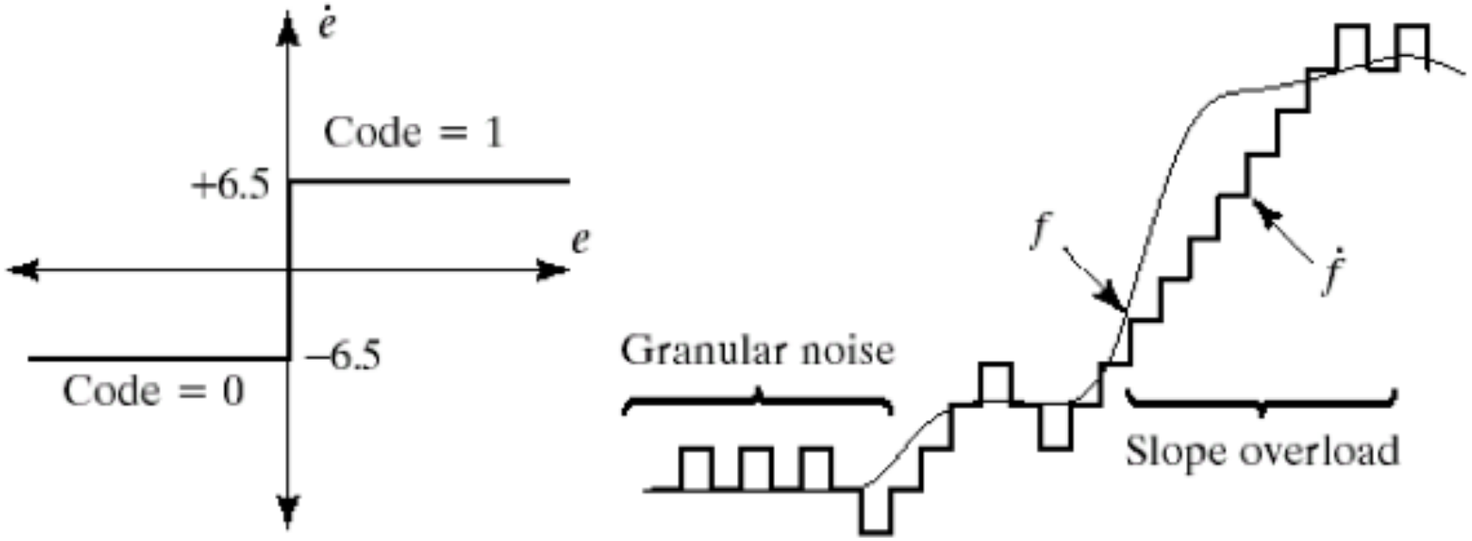
n	f_n	$\hat{f}_n$	e_n	$\dot{e}_n$	$\dot{f}_n$
0	0	-	-	-	0
1	1	0	1	0	0
2	2	0	2	2	2
3	3	2	1	0	2
4	4	2	2	2	4

$$\dot{f}_n = \dot{e}_n + \hat{f}_n$$

$\hat{f}_n = \dot{f}_{n-1}$ is used here for computation

Lossy Compression

{14, 15, 14, 15, 13, 15, 15, 14, 20, 26, 27, 28, 27, 27, 29, 37, 37, 62, 75, 77, 78, 79, 80, 81, 81, 82, 82}

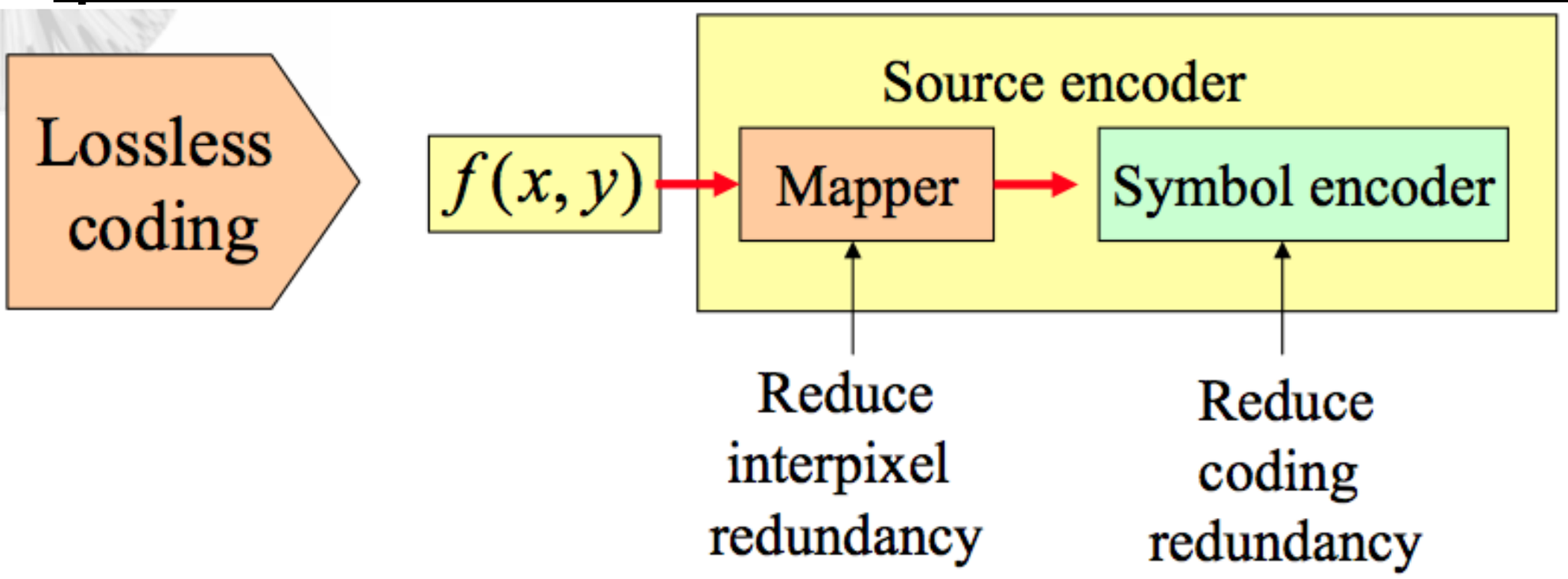


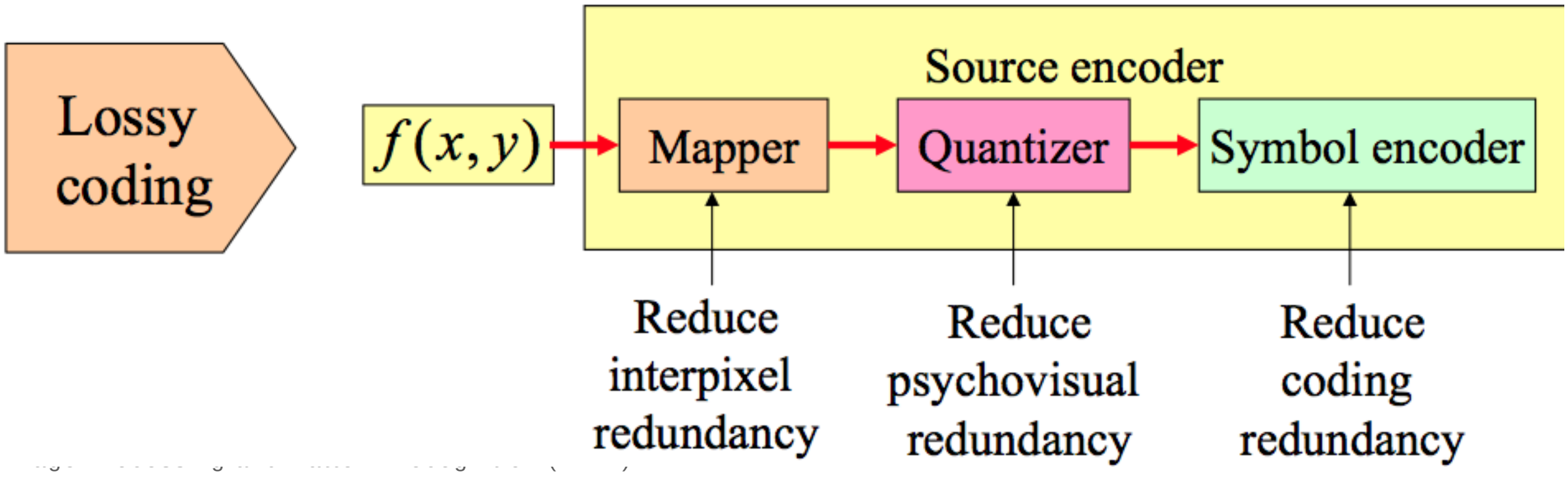
a b
c

FIGURE 8.22 An example of delta modulation.

Input		Encoder				Decoder		Error
n	f	$\hat{f}$	e	$\dot{e}$	$\dot{f}$	$\hat{f}$	$\dot{f}$	$[f - \dot{f}]$
0	14	—	—	—	14.0	—	14.0	0.0
1	15	14.0	1.0	6.5	20.5	14.0	20.5	–5.5
2	14	20.5	–6.5	–6.5	14.0	20.5	14.0	0.0
3	15	14.0	1.0	6.5	20.5	14.0	20.5	–5.5
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
14	29	20.5	8.5	6.5	27.0	20.5	27.0	2.0
15	37	27.0	10.0	6.5	33.5	27.0	33.5	3.5
16	47	33.5	13.5	6.5	40.0	33.5	40.0	7.0
17	62	40.0	22.0	6.5	46.5	40.0	46.5	15.5
18	75	46.5	28.5	6.5	53.0	46.5	53.0	22.0
19	77	53.0	24.0	6.5	59.6	53.0	59.6	17.5
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.

Lossless Vs. Lossy Coding





Lossless Vs. Lossy Coding

FACTORS	DATA COMPRESSION	
	LOSSLESS COMPRESSION	LOSSY COMPRESSION
Definition	Lossless compression is a class of data compression algorithms that allow the original data to be perfectly reconstructed from the compressed data[7].	Lossy compression is the class of data encoding methods that uses inexact approximations to represent the content. These techniques are used to reduce the data size for storage, handling, and transmitting content[8].
Algorithm	RLW, LZW, Arithmetic encoding, Huffman coding, Shannon-Fano coding	Transform coding, DCT,DWT, Fractal compression, Rectangle Segmentation and Sparse Matrix Storage (RSSMS).
Uses	Text or programs, images and sound	Images, audio and video.
Images	RAW, BMP, and PNG are all Lossless formats.	JPEG and GUI are lossy image formats.
Audio	WAV, FLAC, and ALAC are all Lossless formats.	MP3, MP4, and OGG are lossy audio formats.
Video	Few lossless video formats are in common consumer use, they would result in video files taking up a huge amount of space.	Common formats like H-264, MKV, and WMV are all lossy. H-264 can provides smaller files with higher qualities than previous generation of video codec because it has a "smaller" algorithm that's better at choosing the data to throw out.
Advantages	It maintains quality. Conversion in any other format possible without loss of audio information.	It can make a multimedia file much smaller than its original size.It can reduce file sizes much more than lossless compression.
Disadvantages	It doesn't reduce the file size as much as lossy compression. Lossless encoding technique cannot achieve high levels of compression.	Conversion to another format only with loss of audio information.It cannot be used in all types of files because it works by removing data. Text and data cannot be compressed because they do not have redundant information.

Transform Coding

Input image (NxN)

Encoder

Construct nxn subimages

Forward transform

Quantizer

Symbol encoder

Quantization process causes
The transform coding “lossy”

Compressed image

Decoder

Symbol decoder

Inverse transform

Construct nxn subimages

Decompressed image

Examples of transformations used for image compression: DFT and DCT

Transform Coding

- Parameters that effect transform coding performance:

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

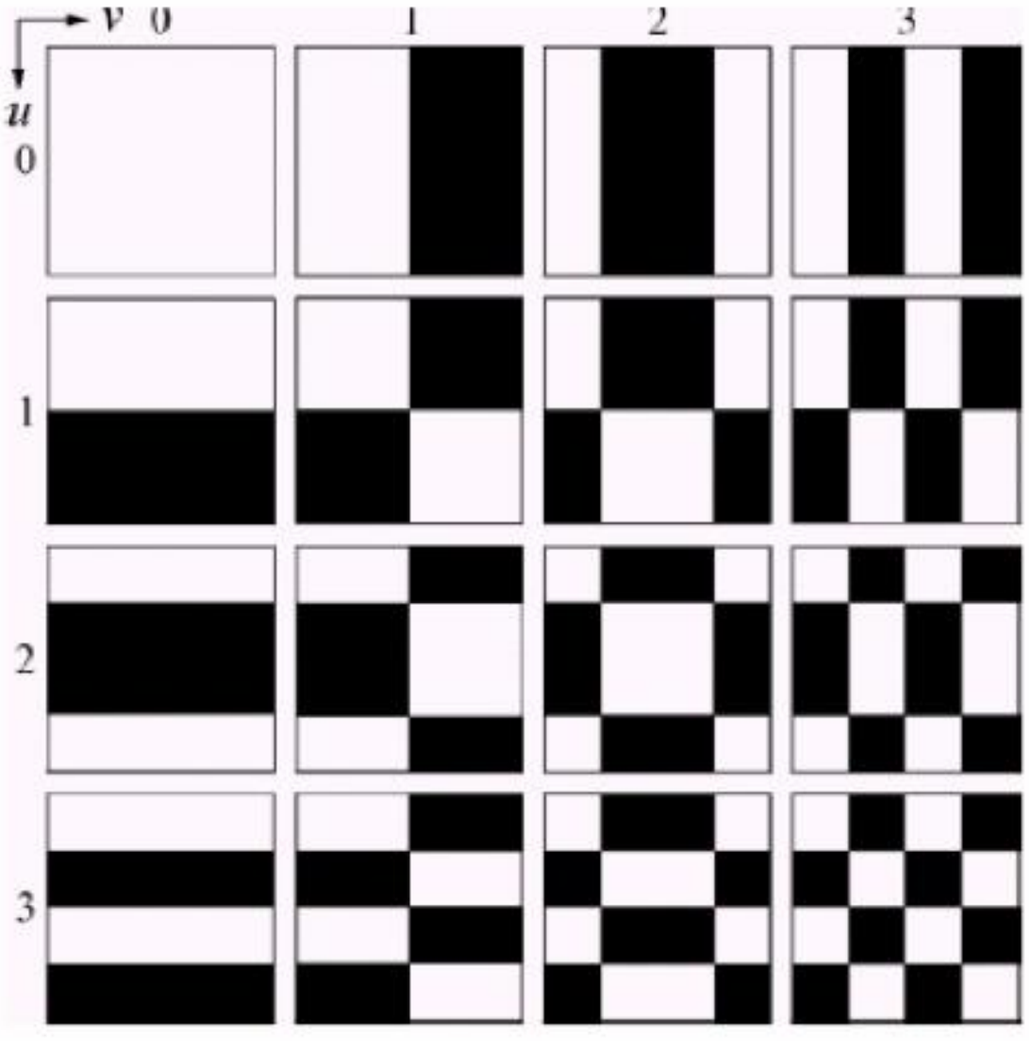
Hadamard Transform

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

$$N = 2^m$$

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

$$p_0(u) = b_{m-1}(u)$$
$$p_1(u) = b_{m-1}(u) + b_{m-2}(u)$$
$$p_2(u) = b_{m-2}(u) + b_{m-3}(u)$$
$$\cdots$$
$$p_{m-1}(u) = b_1(u) + b_0(u)$$

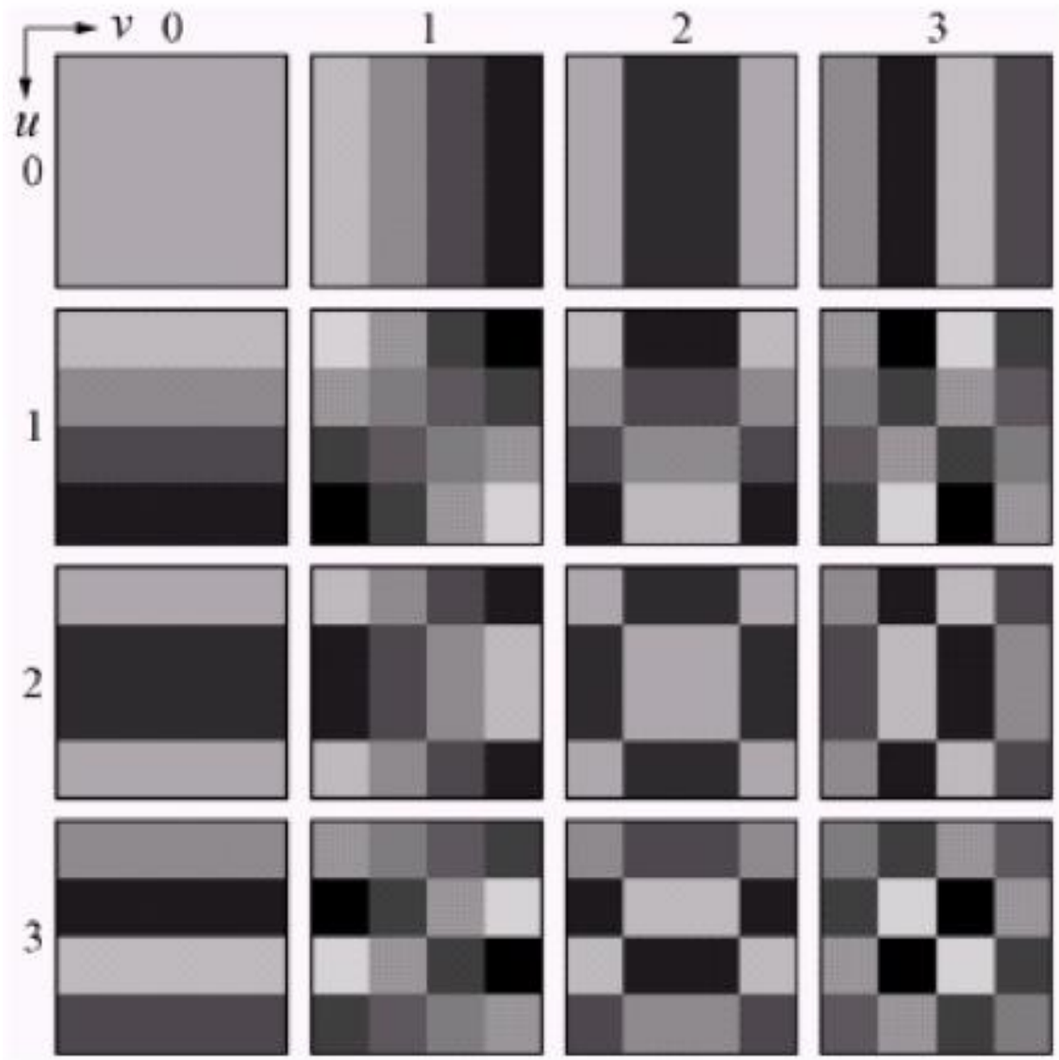


N = 4

Advantage: simple, easy to implement
Disadvantage: not good packing ability

Discrete Cosine Transform

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



N = 4

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

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

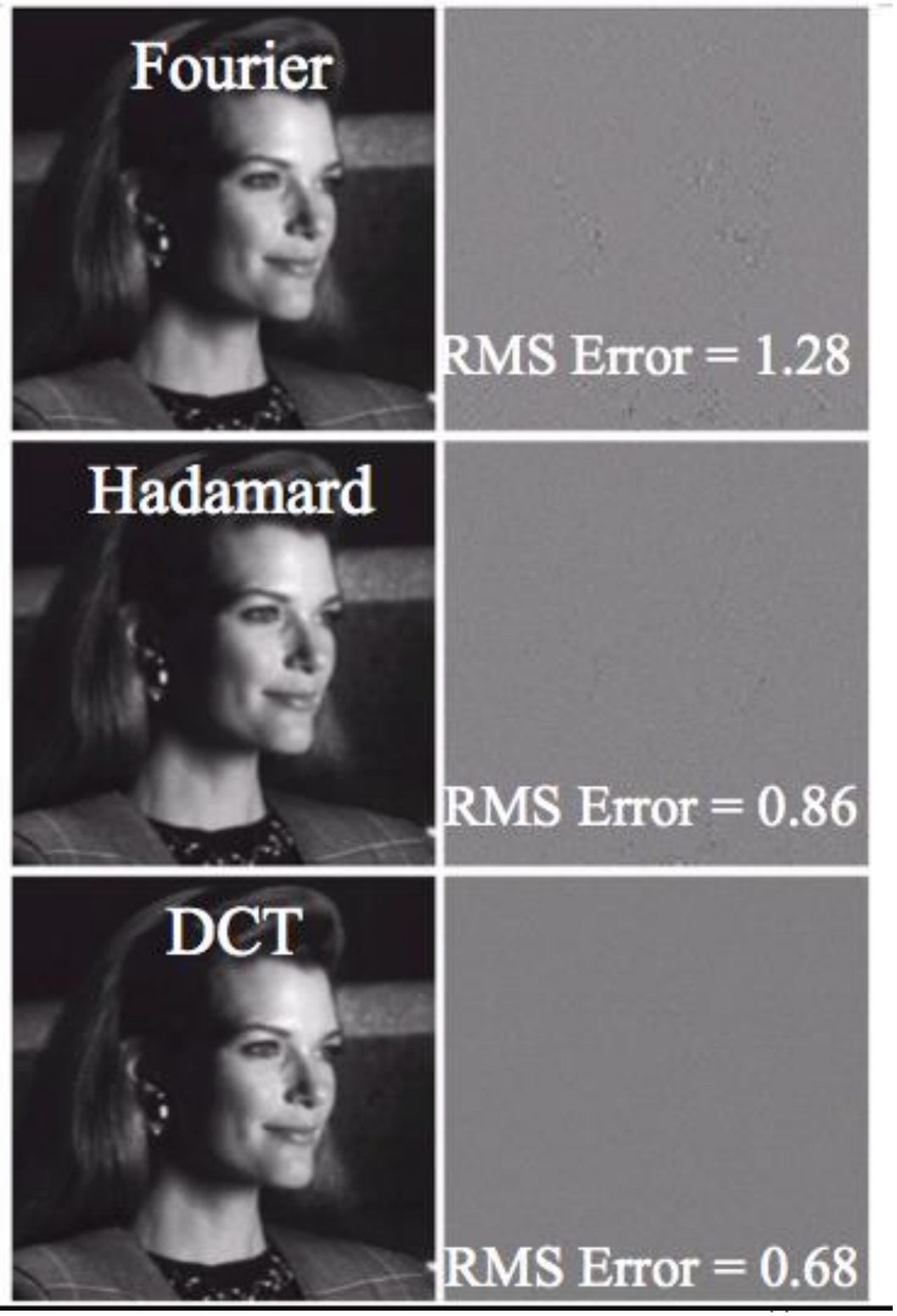
Transform Coding Example

Original image
512x512 pixels

Subimage size:
8x8 pixels = 64 pixels

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

(Images from Rafael C. Gonzalez and Richard E. Wood, Digital Image Processing, 2nd Edition.



Transform Coding Example

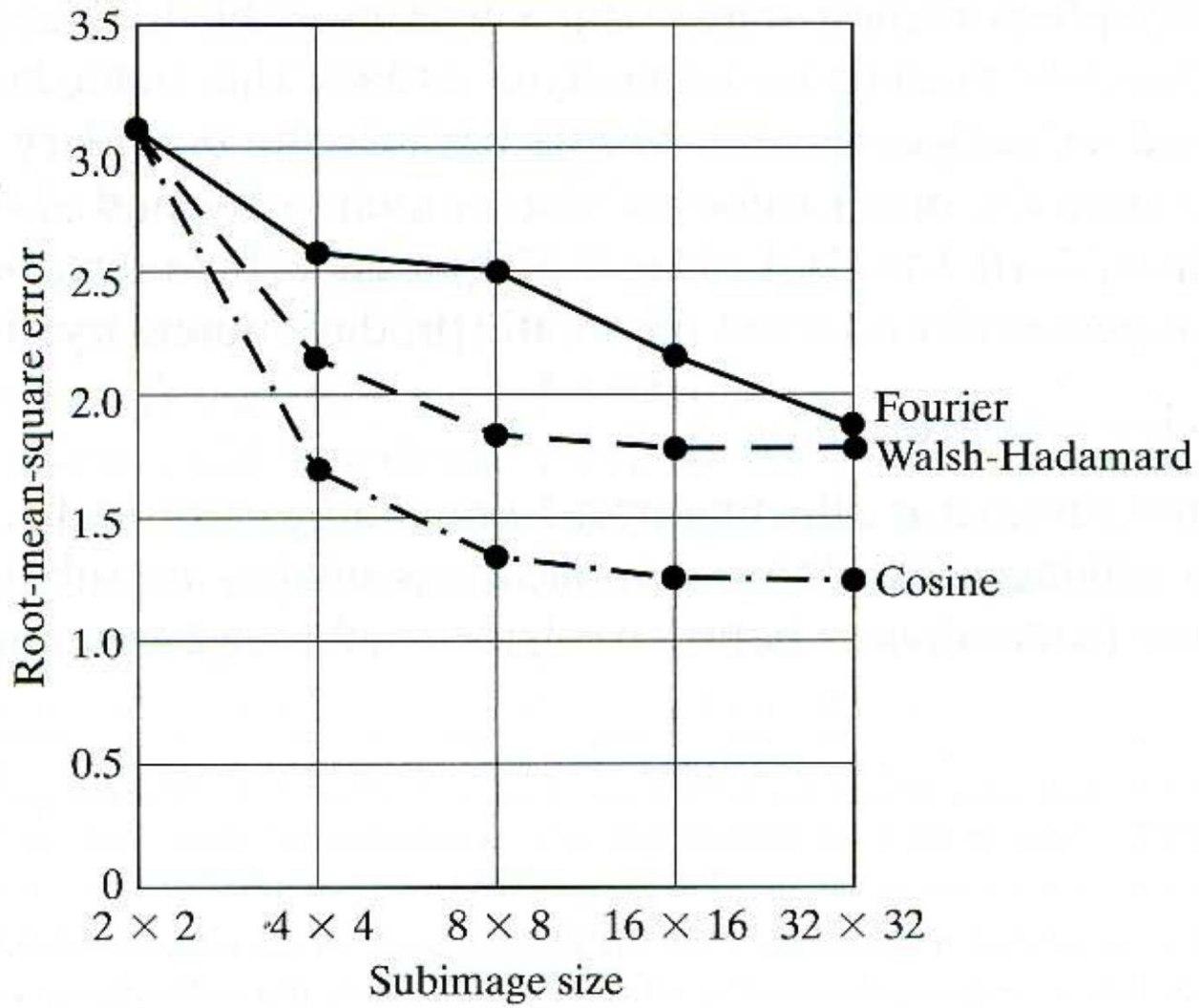


Figure 8.33 illustrates graphically the impact of subimage size on transform coding reconstruction error. The data plotted were obtained by dividing the monochrome image of Fig. 8.23 into subimages of size $n \times n$, for $n = 2, 4, 8, 16,$ and 32, computing the transform of each subimage, truncating 75% of the resulting coefficients, and taking the inverse transform of the truncated arrays.

Thank you !!!