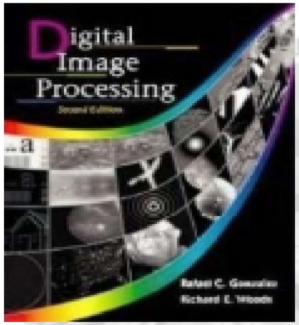
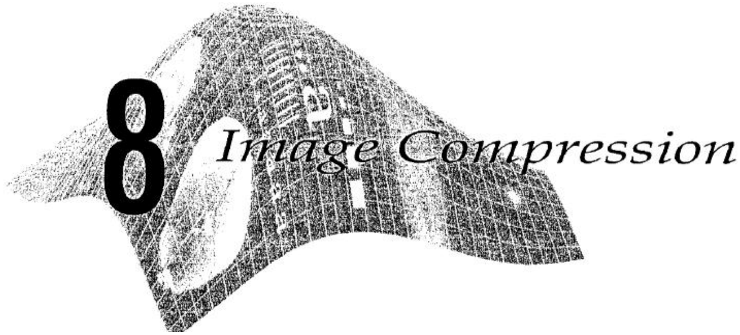


Chapter 8
Image Compression: Part2

Prepared by:
Dr. Ali J. Abboud



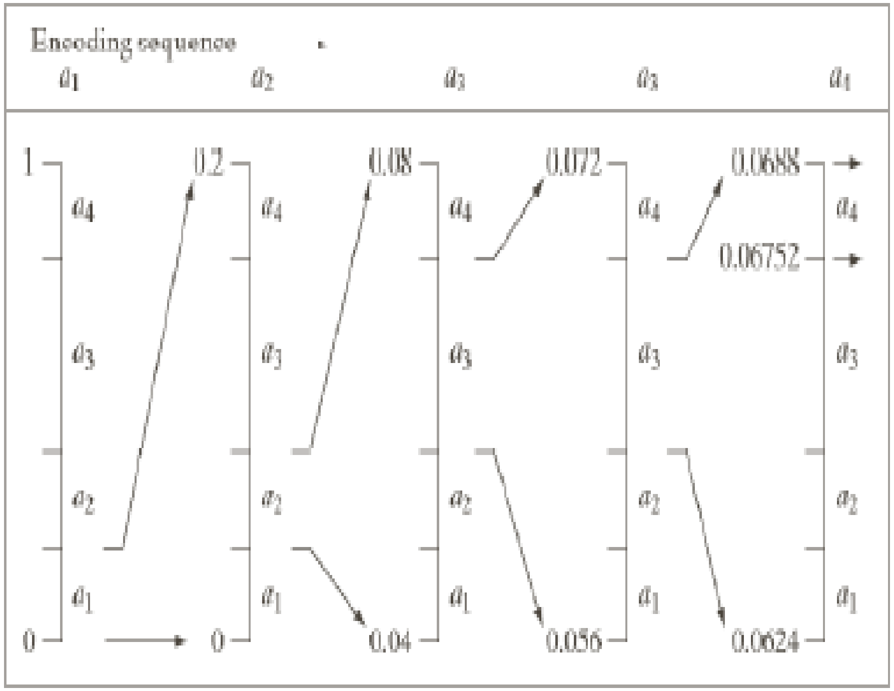
Key Features of Chapter 8

- **Compression Methods**
    1. **Huffman Coding.**
    2. **Arithmetic Coding.**
    3. **LZW Coding.**
    4. **Bit-plane Coding.**
    5. **Run Length Coding.**
    6. **Symbol-based Coding.**
- **Tutorials**

2. Arithmetic Coding

Arithmetic coding is a form of variable-length entropy encoding. A string is converted to arithmetic encoding, usually characters are stored with fewer bits

Arithmetic coding encodes the entire message into a single number, a fraction *n* where $(0.0 \leq n < 1.0)$.

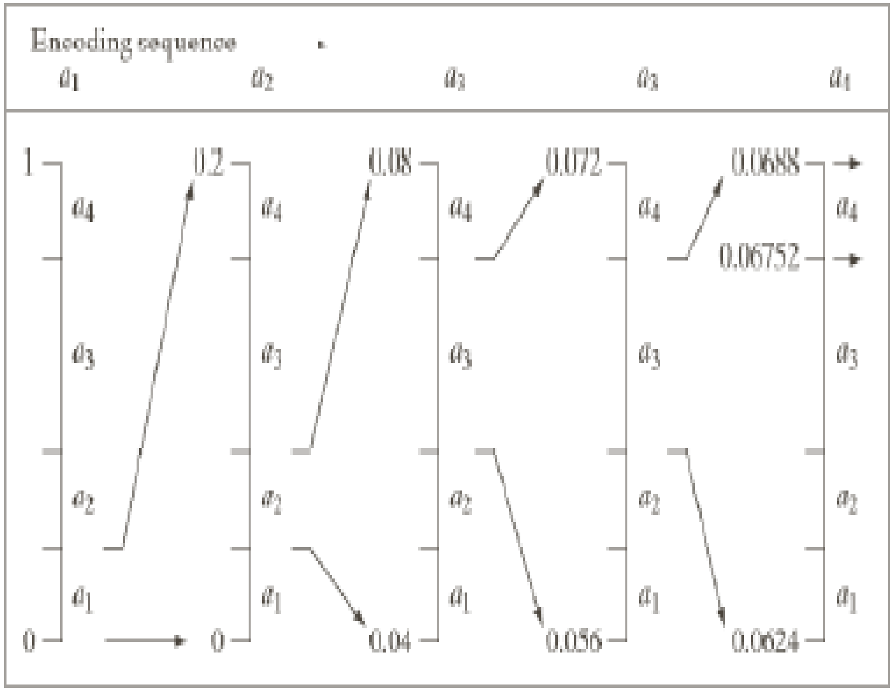


Source Symbol	Probability	Initial Subinterval
a_1	0.2	[0.0, 0.2)
a_2	0.2	[0.2, 0.4)
a_3	0.4	[0.4, 0.8)
a_4	0.2	[0.8, 1.0)

2. Arithmetic Coding

Arithmetic coding is a form of variable-length entropy encoding. A string is converted to arithmetic encoding, usually characters are stored with fewer bits

Arithmetic coding encodes the entire message into a single number, a fraction n where ($0.0 \leq n < 1.0$).



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)

3. LZW Coding

- Lempel-Ziv-Welch (LZW) coding assigns fixed-length code words to variable length sequences of source symbols but requires no a priori knowledge of the probability of occurrence of the symbols to be encoded.
- LZW compression has been integrated into a variety of mainstream imaging file formats, including the *graphic interchange format* (GIF), *tagged image file format* (TIFF), and the *portable document format* (PDF)
- At the onset of the coding process, a codebook or "dictionary" containing the source symbols to be coded is cocnstructed.
- For 8-bit monochrome images, the first 256 words of the dictionary are assigned to the gray value 0,1,2,…,255.

3. LZW Coding

- As the encoder sequentially examines the image's pixels, gray-level sequences that are not in the dictionary are placed in algorithmically determined (e.g., the next unused) locations.
- If the first two pixels of the image are white, for instance, sequence "255-255" might be assigned to location 256, the address following the locations reserved for gray levels 0 through 255.
- The next time that two consecutive white pixels are encountered, code word 256, the address of the location containing sequence 255-255, is used to represent them.
- If a 9-bit, 512-word dictionary is employed in the coding process, the original (8+8) bits that were used to represent the two pixels are replaced by a single 9-bit code word.

3. LZW Coding

- The size of the dictionary is an important system parameter.
- If it is too small, the detection of matching gray-level sequences will be less likely.
- If it is too large, the size of the code words will adversely affect compression performance.
- If a 9-bit, 512-word dictionary is employed in the coding process, the original (8+8) bits that were used to represent the two pixels are replaced by a single 9-bit code word.

Ex. A 9-bit, 512-word dictionary is employed in the coding process. Consider 4x4, 8-bit image of a vertical edge.

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

3. LZW Coding

The starting content of 512-word dictionary is:

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

Locations 256 through 511 are initially unused.

3. LZW Coding

TABLE 8.7 LZW coding example.

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

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		

- The image is encoded by processing its pixels in a left-to-right, top-to-bottom manner.
- Nine additional code words are defined.

- At the conclusion of coding, the dictionary contains 265 code words and the LZW algorithm has successfully identified several repeating gray-level sequences – leveraging them to reduce the original 128-bit image to 90 bits (i.e., 10 9-bit codes).

3. LZW Coding

- The resulting compression ration is 1.42:1
- Most practical applications require a strategy for handling "dictionary overflow".
  - simple solution is to flush or *reinitialize* the dictionary when it becomes full and continue coding with a new initialized dictionary.
  - a more complex option is to monitor compression performance and *flush* the dictionary *when it becomes poor or unacceptable*.
  - alternately, the *least used dictionary entries* can be tracked and *replaced* when necessary.

4. Bit-Plane Coding

Another effective technique for reducing an image's *interpixel redundancies* is to process the image's bit planes individually.

Bit-plane decomposition

The gray levels of an m-bit gray-scale image can be represented in the form of the base 2 polynomial

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \cdots + a_1 2^1 + a_0 2^0$$

m 1-bit *bit planes*.

The inherent disadvantage of this approach is that small changes in gray level can have a significant impact on the complexity of the bit planes.

Ex. 127 (01111111) and 128 (10000000)

→ every bit plane contain a corresponding 0 to 1 (or 1 to 0) transition

4. Bit-Plane Coding

An alternative decomposition approach (which reduces the effect of small gray-level variations) is to first represent the image by an *m*-bit *Gray code*.

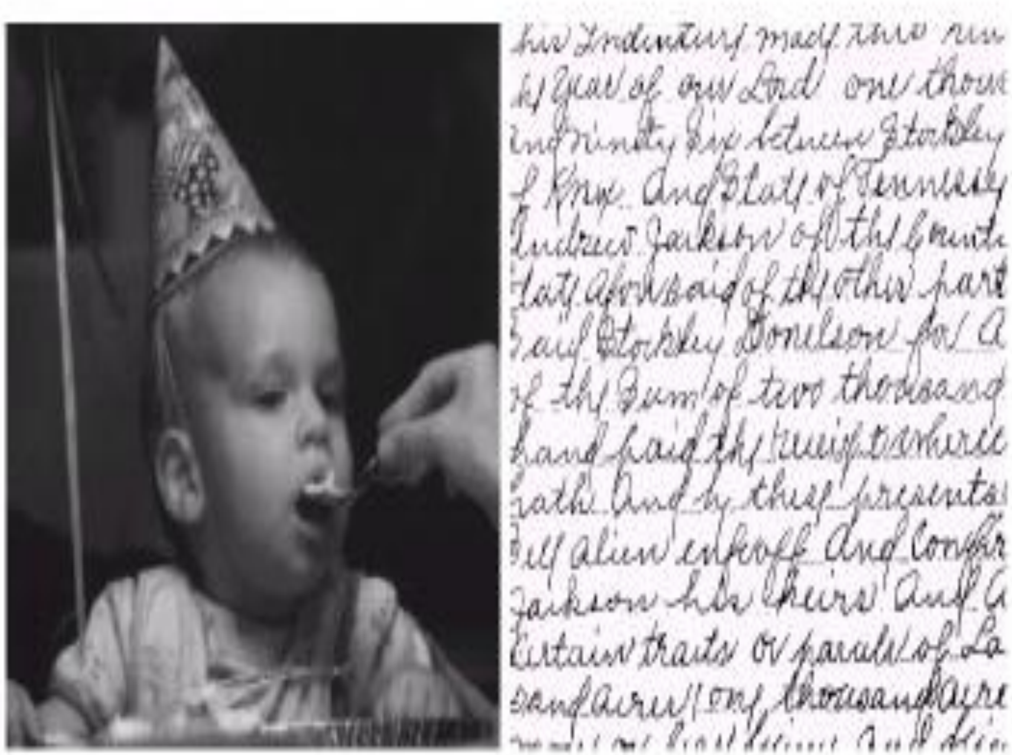
The *m*-bit Gray code $g_{m-1} \ldots g_2 g_1 g_0$

$$g_i = a_i \oplus a_{i+1} \qquad 0 \le i \le m-2$$

$$g_{m-1} = a_{m-1}$$

Gray codes that correspond to 127 and 128 are 01000000 and 11000000, respectively.

4. Bit-Plane Coding



a b

FIGURE 8.14 A 1024 × 1024 (a) 8-bit monochrome image and (b) binary image.

These two images are used to illustrate the compression techniques.

4. Bit-Plane Coding

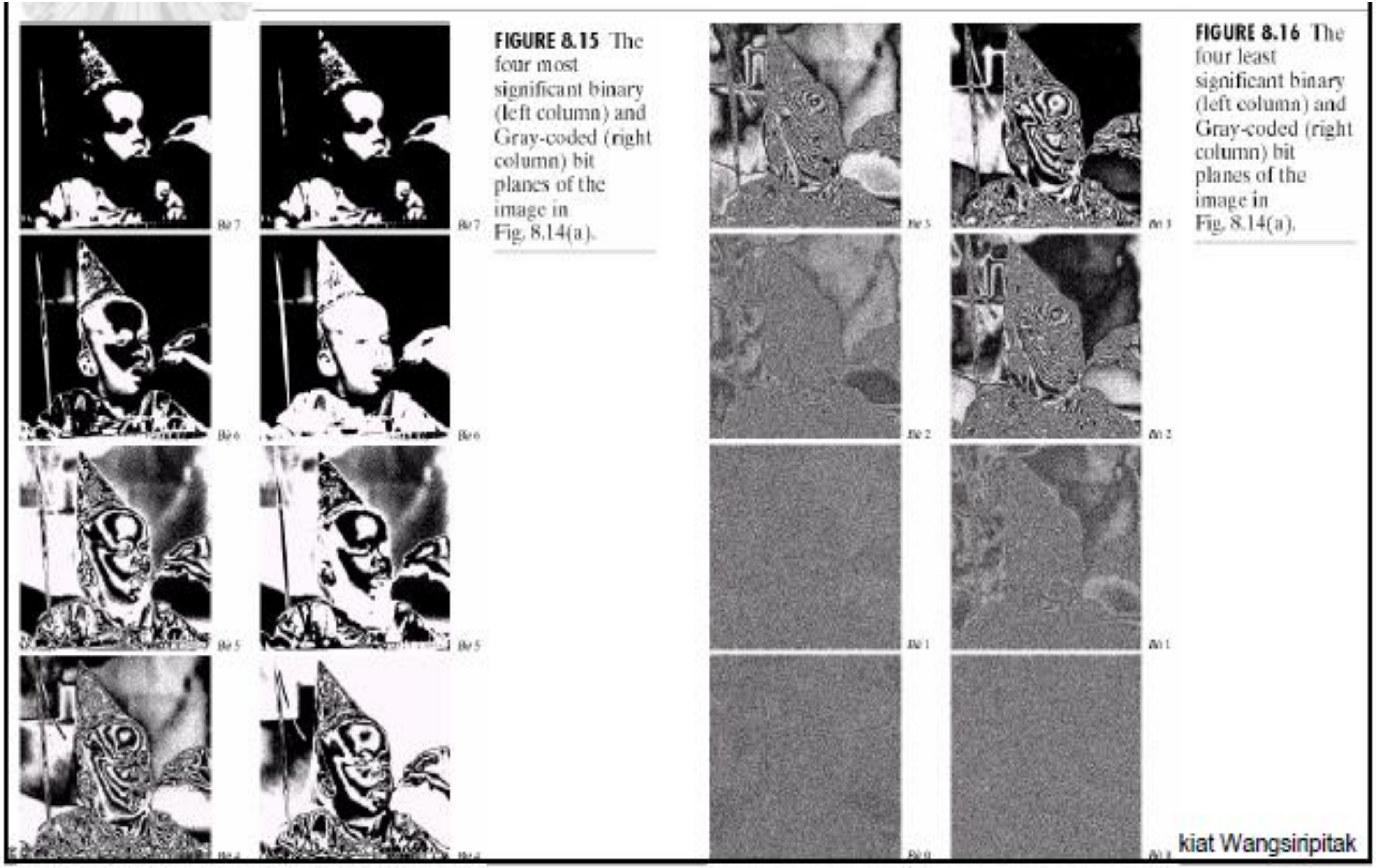


FIGURE 8.15 The four most significant binary (left column) and Gray-coded (right column) bit planes of the image in Fig. 8.14(a).

FIGURE 8.16 The four least significant binary (left column) and Gray-coded (right column) bit planes of the image in Fig. 8.14(a).

kiat Wangsiripitak

5. Run Length Coding

- 1-D run-length coding
  - RLC+VLC according to run-lengths statistics
- 2-D run-length coding
  - used for FAX image compression
  - Relative address coding (RAC)
    - based on the principle of tracking the binary transitions that begin and end each black and white run
    - combined with VLC

5. Run Length Coding

One-dimensional run-length coding

represent each row of an image or bit plane by a sequence of lengths that describe successive runs of black and white pixels.

→ *run-length coding*

The basic concept is to code each contiguous group of 0's or 1's encountered in a left to right scan of a row by its length and to establish *a convention for determining the value of the run*.

(1) to specify the value of the first run of each row, or

(2) to assume that each row begins with a white run,
whose run length may in fact be zero

5. Run Length Coding

- The black and white run lengths may be coded separately using variable-length codes that are specifically tailored to their own statistics.

H_0 : an estimate of the entropy of the black run-length source
H_1 : an estimate of the entropy of the white run-length source
L_0 : the average value of the black run lengths
L_1 : the average value of the white run-lengths

The approximate *run-length entropy* of the image is

$$H_{RL} = \frac{H_0 + H_1}{L_0 + L_1} \qquad (8.4\text{-}4)$$

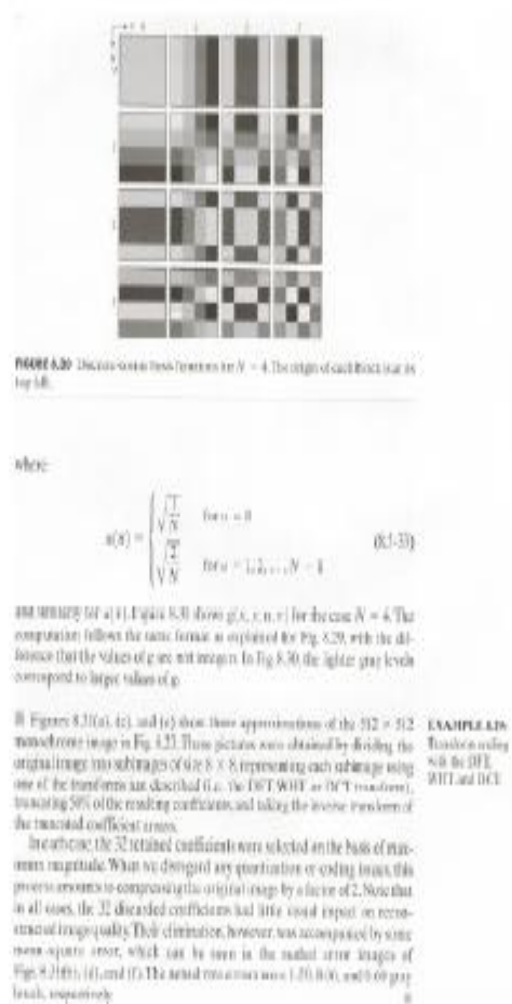
Eq. (8.4-4) provides an estimate of the average number of bits per pixel required to code the run lengths in a binary image using a variable-length code.

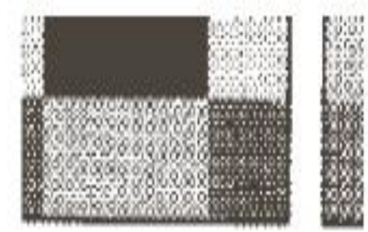
6. Symbol-based Coding

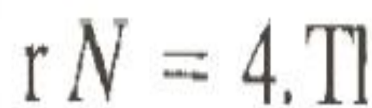
Symbol compression

This approaches determine a set of symbols that constitute the image, and take advantage of their multiple appearance. It convert each symbol into token, generate a token table and represent the compressed image as a list of tokens.

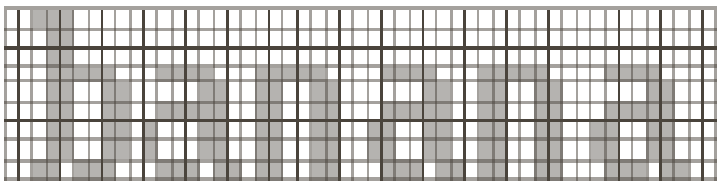
This approach is good for document images.

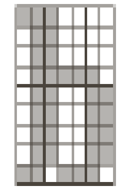
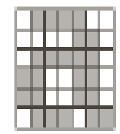
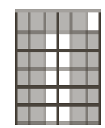






6. Symbol-based Coding



Token	Symbol
0	
1	
2	

Triplet
(0, 2, 0)
(3, 10, 1)
(3, 18, 2)
(3, 26, 1)
(3, 34, 2)
(3, 42, 1)

a b c

FIGURE 8.17
(a) A bi-level document, (b) symbol dictionary, and (c) the triplets used to locate the symbols in the document.

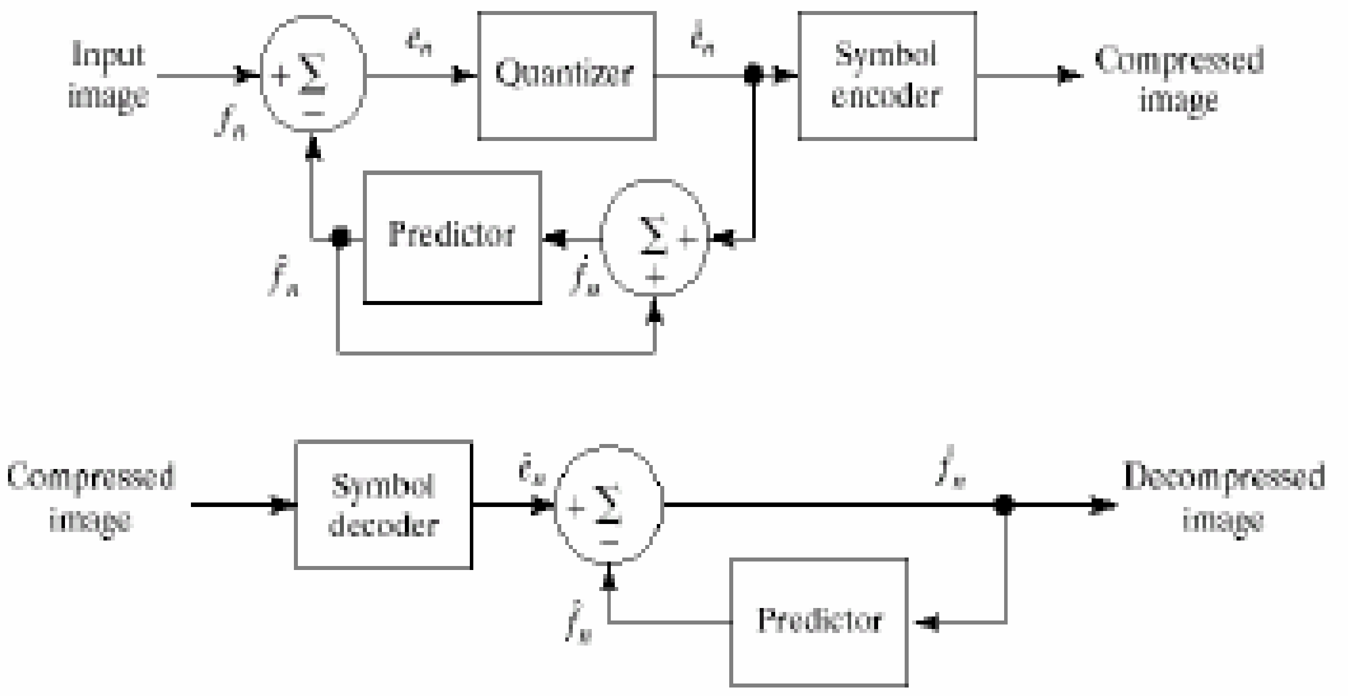
7. Lossy Compression

- Lossy encoding is based on the concept of *compromising the accuracy* of the reconstructed image in exchange for increased compression.

- If the resulting distortion (which may or may not be visually apparent) can be tolerated, the increase in compression can be significant.

  10:1 to 50:1 → more than 100:1

8. Lossy Predictive Coding



a
b

FIGURE 8.21 A lossy predictive coding model: (a) encoder and (b) decoder.

9. Lossy Transform Coding

- The predictive coding techniques operate directly on the pixels of an image and thus are spatial domain methods.
- In this section, we consider compression techniques that are based on *modifying the transform of an image*.
- In *transform coding*, a reversible, linear transform (such as Fourier transform) is used to map the image into a set of transform coefficients, which are then quantized and coded.
- For most natural images, a significant number of the coefficients have small magnitudes and can be coarsely quantized (or discarded entirely) with little image distortion.

9. Lossy Transform Coding

- The goal of the transformation process is to decorrelate the pixels of each subimage, or to pack as much information as possible into the smallest number of transform coefficients.

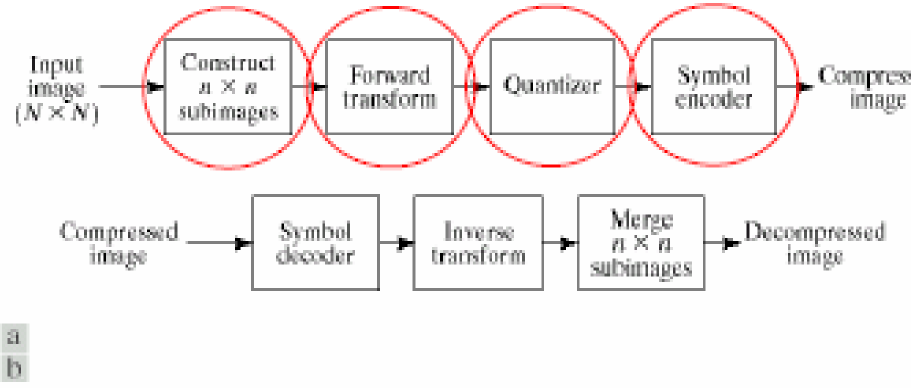


FIGURE 8.28 A transform coding system: (a) encoder; (b) decoder.

- The quantization stage then selectively eliminates or more coarsely quantizes the coefficients that carry the least information.

- The encoding process terminates by coding (normally using a variable length code) the quantized coefficients.

- Any or all of the transform encoding steps can be adapted to local image content, called *adaptive transform coding*, or fixed for all subimages, called *nonadaptive transform coding*.

9. Lossy Transform Coding

Transform selection

Walsh-Hadamard transform **(WHT)**

Discrete cosine transform **(DCT)**

One of the most frequently used transformation for image compression.

Wavelet Selection

- The most widely used expansion functions for wavelet-based compression are the Daubechies wavelets and biorthogonal wavelets.

9. Lossy Transform Coding

Three approximations of the 512 x 512 monochrome image in Fig.8.23.

These pictures were obtained by

1. Dividing the original image into subimages of size 8 x 8,
2. Transforms *rms* error
   - DFT 1.28
   - WHT 0.86
   - DCT 0.68
3. truncating 50% of the resulting coefficients (minimum magnitude).
4. inverse transform

a b
c d
e f

FIGURE 8.31 Approximations of Fig. 8.23 using the (a) Fourier, (c) Hadamard, and (e) cosine transforms, together with the corresponding scaled error images.

9. Lossy Transform Coding

Compression ratio

34 : 1 67 : 1

(the average compression ratio obtained by using all the error-free methods was only 2.62 : 1)

rms error

3.42 6.33 gray levels

a b
c d
e f

FIGURE 8.38 Left column: Approximations of Fig. 8.23 using the DCT and normalization array of Fig. 8.37(b). Right column: Similar results for 4Z.

END of Chapter 8 : Part2