An Introduction to Image Compression

Wei-Yi Wei
E-mail: s9361121@nchu.edu.tw
Graduate Institute of Communication Engineering
National Taiwan University, Taipei, Taiwan, ROC

Abstract

In recent years, the development and demand of multimedia product grows increasingly fast, contributing to insufficient bandwidth of network and storage of memory device. Therefore, the theory of data compression becomes more and more significant for reducing the data redundancy to save more hardware space and transmission bandwidth. In computer science and information theory, data compression or source coding is the process of encoding information using fewer bits or other information-bearing units than an unencoded representation. Compression is useful because it helps reduce the consumption of expensive resources such as hard disk space or transmission bandwidth. In this paper, we briefly introduce the fundamental theory of image compression in chapter 1, two typical standards - JPEG and JPEG 2000 will be described in chapter 2. Finally, the newly proposed image compression algorithm – Shape Adaptive image Compression will be introduced in chapter 3.

1. Introduction to Image Compression Fundamentals

Image compression is an application of data compression that encodes the original image with few bits. The objective of image compression is to reduce the redundancy of the image and to store or transmit data in an efficient form. Fig 1.1 shows the block diagram of the general image storage system. The main goal of such system is to reduce the storage quantity as much as possible, and the decoded image displayed in the monitor can be similar to the original image as much as can be. The essence of each block will be introduced in the following sections.

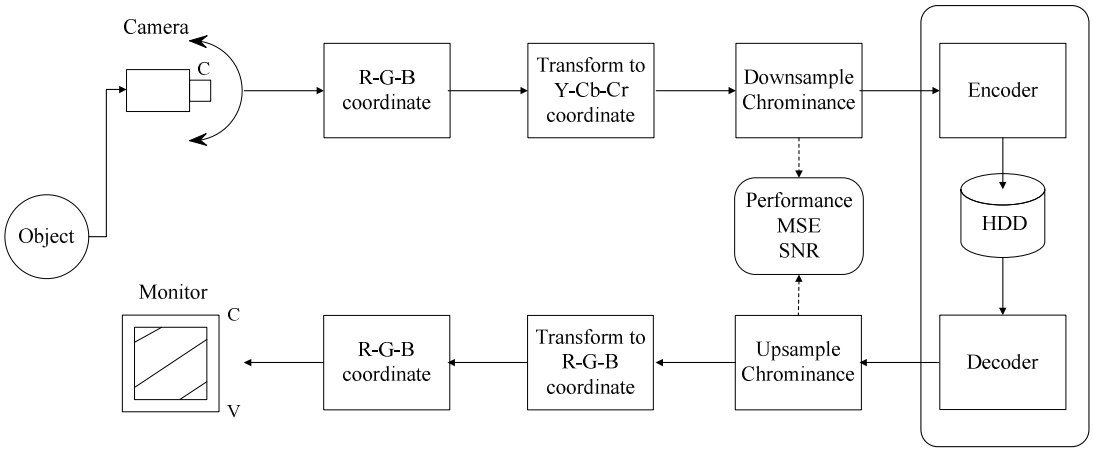

Fig. 1.1 General Image Storage System

1.1 Color Specification

The Y, Cb, and Cr components of one color image are defined in YUV color coordinate, where Y is commonly called the luminance and Cb, Cr are commonly called the chrominance. The meaning of luminance and chrominance is described as follows

- **Luminance:** received brightness of the light, which is proportional to the total energy in the visible band.
- **Chrominance:** describe the perceived color tone of a light, which depends on the wavelength composition of light chrominance is in turn characterized by two attributes – hue and saturation.
  1. **hue:** Specify the color tone, which depends on the peak wavelength of the light
  2. **saturation:** Describe how pure the color is, which depends on the spread or bandwidth of the light spectrum

The RGB primary commonly used for color display mixes the luminance and chrominance attributes of a light. In many applications, it is desirable to describe a color in terms of its luminance and chrominance content separately, to enable more efficient processing and transmission of color signals. Towards this goal, various three-component color coordinates have been developed, in which one component reflects the luminance and the other two collectively characterize hue and saturation. One such coordinate is the YUV color space. The $[Y\ Cb\ Cr]^T$ values in the YUV coordinate are related to the $[R\ G\ B]^T$ values in the RGB coordinate by

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.334 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix} \quad \textbf{(1.1)}$$

Similarly, if we would like to transform the YUV coordinate back to RGB coordinate, the inverse matrix can be calculated from (1.1), and the inverse transform is taken to obtain the corresponding RGB components.

1.2 Spatial Sampling of Color Component

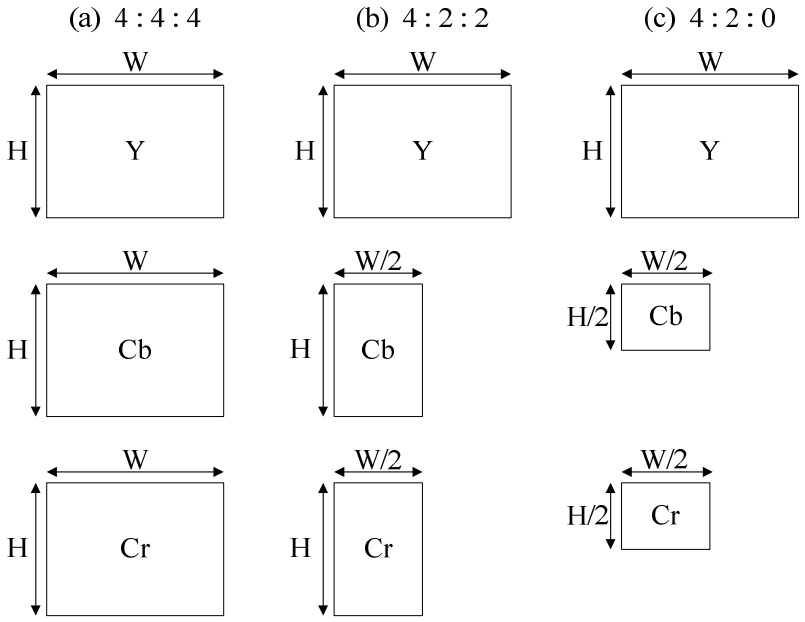

Fig. 1.2 The three different chrominance downsampling format

Because the eyes of human are more sensitive to the luminance than the chrominance, the sampling rate of chrominance components is half that of the luminance component. This will result in good performance in image compression with almost no loss of characteristics in visual perception of the new upsampled image. There are three color formats in the baseline system:

- **4:4:4 format**: The sampling rate of the luminance component is the same as those of the chrominance.
- **4:2:2 format**: There are 2 Cb samples and 2 Cr samples for every 4 Y samples. This leads to half number of pixels in each line, but the same number of lines per frame.
- **4:2:0 format**: Sample the Cb and Cr components by half in both the horizontal and vertical directions. In this format, there are also 1 Cb sample and 1 Cr sample for every 4 Y samples.

At the decoder, the downsampled chrominance components of 4:2:2 and 4:2:0 formats should be upsampled back to 4:4:4 format.

1.3 The Flow of Image Compression Coding

What is the so-called image compression coding? *Image compression coding is to store the image into bit-stream as compact as possible and to display the decoded image in the monitor as exact as possible*. Now consider an encoder and a decoder as shown in Fig. 1.3. When the encoder receives the original image file, the image file will be converted into a series of binary data, which is called the bit-stream. The decoder then receives the encoded bit-stream and decodes it to form the decoded image. If the total data quantity of the bit-stream is less than the total data quantity of the original image, then this is called image compression. The full compression flow is as shown in Fig. 1.3.

Fig. 1.3 The basic flow of image compression coding

The compression ratio is defined as follows:

$$Cr = \frac{n1}{n2}, \tag{1.2}$$

where *n1* is the data rate of original image and *n2* is that of the encoded bit-stream.

In order to evaluate the performance of the image compression coding, it is necessary to define a measurement that can estimate the difference between the original image and the decoded image. Two common used measurements are the **Mean Square Error (MSE)** and the **Peak Signal to Noise Ratio (PSNR)**, which are defined in (1.3) and (1.4), respectively. f(x,y) is the pixel value of the original image, and f'(x,y)is the pixel value of the decoded image. Most image compression systems are designed to minimize the MSE and maximize the PSNR.

$$MSE = \sqrt{\frac{\sum_{x=0}^{W-1}\sum_{y=0}^{H-1}\left[f(x,y) - f'(x,y)\right]^2}{WH}} \tag{1.3}$$

$$PSNR = 20\log_{10}\frac{255}{MSE} \tag{1.4}$$

The general encoding architecture of image compression system is shown is Fig. 1.4. The fundamental theory and concept of each functional block will be introduced in the following sections.

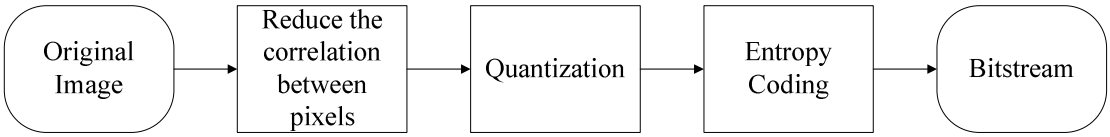

Fig. 1.4 The general encoding flow of image compression

1.3.1 Reduce the Correlation between Pixels

Why an image can be compressed? The reason is that *the correlation between one pixel and its neighbor pixels is very high*, or we can say that the values of one pixel and its adjacent pixels are very similar. Once the correlation between the pixels is reduced, we can take advantage of the statistical characteristics and the variable length coding theory to reduce the storage quantity. This is the most important part of the image compression algorithm; there are a lot of relevant processing methods being proposed. The best-known methods are as follows:

- **Predictive Coding**: Predictive Coding such as DPCM (Differential Pulse Code Modulation) is a lossless coding method, which means that the decoded image and the original image have the same value for every corresponding element.
- **Orthogonal Transform**: Karhunen-Loeve Transform (KLT) and Discrete Cosine Transform (DCT) are the two most well-known orthogonal transforms. The DCT-based image compression standard such as JPEG is a lossy coding method that will result in some loss of details and unrecoverable distortion.
- **Subband Coding**: Subband Coding such as Discrete Wavelet Transform (DWT) is also a lossy coding method. The objective of subband coding is to divide the spectrum of one image into the lowpass and the highpass components. JPEG 2000 is a 2-dimension DWT based image compression standard.

The details of these transform coding methods will be described in the chapter 2.

1.3.2 Quantization

The objective of quantization is to reduce the precision and to achieve higher compression ratio. For instance, the original image uses 8 bits to store one element for every pixel; if we use less bits such as 6 bits to save the information of the image, then the storage quantity will be reduced, and the image can be compressed. The shortcoming of quantization is that it is a lossy operation, which will result into loss of precision and unrecoverable distortion. The image compression standards such as

JPEG and JPEG 2000 have their own quantization methods, and the details of relevant theory will be introduced in the chapter 2.

1.3.3 Entropy Coding

The main objective of entropy coding is to achieve less average length of the image. Entropy coding assigns codewords to the corresponding symbols according to the probability of the symbols. In general, the entropy encoders are used to compress the data by replacing symbols represented by equal-length codes with the codewords whose length is inverse proportional to corresponding probability. The entropy encoder of JPEG and JPEG 2000 will also be introduced in the chapter 2.

2 An Overview of Image Compression Standard

In this chapter, we will introduce the fundamental theory of two well-known image compression standards –JPEG and JPEG 2000.

2.1 JPEG – Joint Picture Expert Group

Fig. 2.1 and 2.2 shows the Encoder and Decoder model of JPEG. We will introduce the operation and fundamental theory of each block in the following sections.

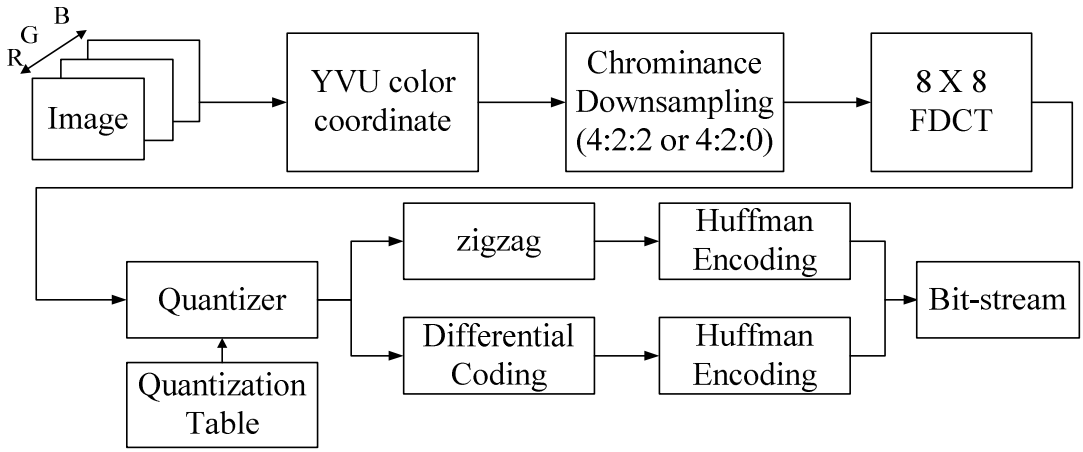

Fig. 2.1 The Encoder model of JPEG compression standard

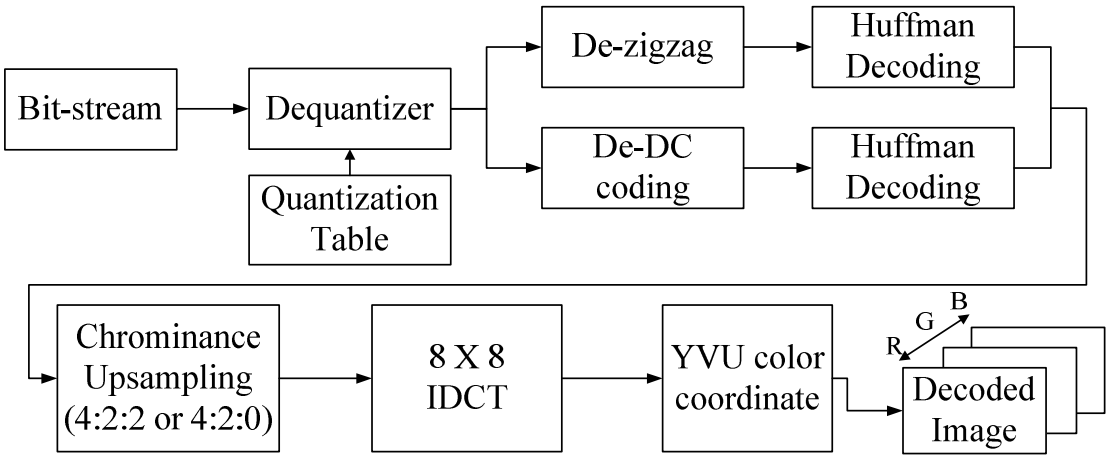

Fig. 2.2 The Decoder model of JPEG compression standard

2.1.1 Discrete Cosine Transform

The next step after color coordinate conversion is to divide the three color components of the image into many 8×8 blocks. The mathematical definition of the Forward DCT and the Inverse DCT are as follows:

Forward DCT

$$F(u,v) = \frac{2}{N} C(u)C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2N}\right]$$

for $u = 0,...,N-1$ and $v = 0,...,N-1$ **(2.1)**

where $N = 8$ and $C(k) = \begin{cases} 1/\sqrt{2} & \text{for } k = 0 \\ 1 & \text{otherwise} \end{cases}$

Inverse DCT

$$f(x,y) = \frac{2}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u)C(v)F(u,v) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \cos\left[\frac{\pi(2y+1)v}{2N}\right] \quad \textbf{(2.2)}$$

for $x = 0,...,N-1$ and $y = 0,...,N-1$ where $N = 8$

The f(x,y) is the value of each pixel in the selected 8×8 block, and the F(u,v) is the DCT coefficient after transformation. The transformation of the 8×8 block is also a 8×8 block composed of F(u,v).

The DCT is closely related to the DFT. Both of them taking a set of points from the spatial domain and transform them into an equivalent representation in the frequency domain. However, why DCT is more appropriate for image compression than DFT? The two main reasons are:

1. The DCT can concentrate the energy of the transformed signal in low frequency, whereas the DFT can not. According to Parseval's theorem, the

energy is the same in the spatial domain and in the frequency domain. Because the human eyes are less sensitive to the low frequency component, we can focus on the low frequency component and reduce the contribution of the high frequency component after taking DCT.

2. For image compression, the DCT can reduce the blocking effect than the DFT.

After transformation, the element in the upper most left corresponding to zero frequency in both directions is the "**DC coefficient**" and the rest are called "**AC coefficients**."

2.1.2 Quantization in JPEG

Quantization is the step where we actually throw away data. The DCT is a lossless procedure. The data can be precisely recovered through the IDCT (this isn't entirely true because in reality no physical implementation can compute with perfect accuracy). During Quantization every coefficients in the 8×8 DCT matrix is divided by a corresponding quantization value. The quantized coefficient is defined in (2.3), and the reverse the process can be achieved by the (2.4).

$$F(u,v)_{Quantization} = round\left(\frac{F(u,v)}{Q(u,v)}\right) \quad \textbf{(2.3)}$$

$$F(u,v)_{deQ} = F(u,v)_{Quantization} \times Q(u,v) \quad \textbf{(2.4)}$$

The goal of quantization is to reduce most of the less important high frequency DCT coefficients to zero, the more zeros we generate the better the image will compress. The matrix Q generally has lower numbers in the upper left direction and large numbers in the lower right direction. Though the high-frequency components are removed, the IDCT still can obtain an approximate matrix which is close to the original 8×8 block matrix. The JPEG committee has recommended certain Q matrix that work well and the performance is close to the optimal condition, the Q matrix for luminance and chrominance components is defined in (2.5) and (2.6).

$$Q_Y = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix} \quad \textbf{(2.5)}$$

$$Q_C = \begin{pmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{pmatrix} \quad \textbf{(2.6)}$$

2.1.3 Zigzag Scan

After quantization, the DC coefficient is treated separately from the 63 AC coefficients. The DC coefficient is a measure of the average value of the original 64 image samples. Because there is usually strong correlation between the DC coefficients of adjacent 8×8 blocks, the quantized DC coefficient is encoded as the difference from the DC term of the previous block. This special treatment is worthwhile, as DC coefficients frequently contain a significant fraction of the total image energy. The other 63 entries are the AC components. They are treated separately from the DC coefficients in the entropy coding process.

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

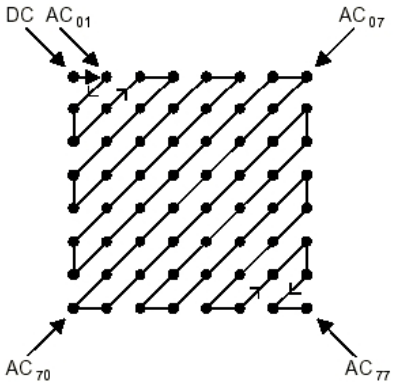

Fig. 2.3 The zigzag scan order

2.1.4 Entropy Coding in JPEG

2.1.4.1 Differential Coding

The mathematical representation of the differential coding is:

$$\text{Diff}_i = \text{DC}_i - \text{DC}_{i\text{-}1} \quad \textbf{(2.7)}$$

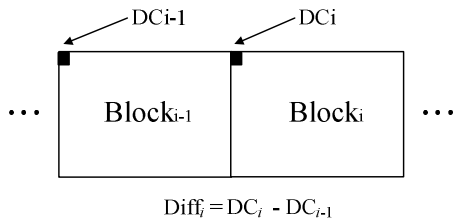

Fig. 2.4 Differential Coding

We set $DC_0 = 0$. DC of the current block DC_i will be equal to $DC_{i-1} + Diff_i$. Therefore, in the JPEG file, the first coefficient is actually the difference of DCs. Then the difference is encoded with Huffman coding algorithm together with the encoding of AC coefficients.

2.1.4.2 Zero-Run-Length Coding

After quantization and zigzag scanning, we obtain the one-dimensional vectors with a lot of consecutive zeroes. We can make use of this property and apply zero-run -length coding, which is variable length coding. Let us consider the 63 AC coefficients in the original 64 quantized vectors first. For instance, we have:

57, 45, 0, 0, 0, 0, 23, 0, -30, -16, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0 **(2.8)**

We encode then encode the vector (2.8) into vector (2.9).

(0,57) ; (0,45) ; (4,23) ; (1,-30) ; (0,-16) ; (2,1) ; EOB **(2.9)**

The notation (L,F) means that there are L zeros in front of F, and EOB (End of Block) is a special coded value means that the rest elements are all zeros. If the last element of the vector is not zero, then the EOB marker will not be added. On the other hand, EOC is equivalent to (0,0), so we can express (2.9) as (2.10).

(0,57) ; (0,45) ; (4,23) ; (1,-30) ; (0,-16) ; (2,1) ; (0,0) **(2.10)**

We give a special case that L is larger than 15. (2.11) is another example.

57, eighteen zeroes, 3, 0, 0, 0, 0, 2, thirty-three zeroes, 895, EOB **(2.11)**

The JPEG Huffman coding restricts the number of previous zero(s) within 15 because the length of the encoded data is 4 bits. Hence, we can encode (2.11) into (2.12).

(0,57) ; (15,0) ; (2,3) ; (4,2) ; (15,0) ; (15,0) ; (1,895) ; (0,0) **(2.12)**

(15,0) is a special coded value which indicates that there are 16 consecutive zeroes. Therefore, the both of the values 18 and 33 are converted to 15.

Now back to the example (2.8). The right value of the bracket is called category, and the left value of the bracket is called run. We can encode the category by looking up table 2.1 which is specified in the JPEG standard. For example, 57 is in the category 6 and the bits for the value is 111001, so we encode it as 6,111001. The full encoded sequence of (2.8) is as (2.13).

(0,6,111001);(0,6,101101);(4,5,10111);(1,5,00001);(0,4,0111);(2,1,1);(0,0) **(2.13)**

Table 2.1 Table of the category and bit-coded values

Category	Values	Bits for the value
1	-1,1	0,1
2	-3,-2,2,3	00,01,10,11
3	-7,-6,-5,-4,4,5,6,7	000,001,010,011,100,101,110,111
4	-15,...,-8,8,...,15	0000,...,0111,1000,...,1111
5	-31,...,-16,16,...31	00000,...,01111,10000,...,11111
6	-63,...,-32,32,...63	000000,...,011111,100000,...,111111
7	-127,...,-64,64,...,127	0000000,...,0111111,1000000,...,1111111
8	-255,..,-128,128,..,255	...
9	-511,..,-256,256,..,511	...
10	-1023,..,-512,512,..,1023	...
11	-2047,...,-1024,1024,...,2047	...

The first two values in bracket parenthesis can be represented on a byte because of the fact that each of the two values is 0,1,2,...,15. In this byte, the higher 4-bit (run) represents the number of previous zeros, and the lower 4-bit (category) represents the the value which is not 0.

The JPEG standard specifies the Huffman table for encoding the AC coefficients which is listed in table 2.2. The final step is to encode (2.13) by using the Huffman table defined in table 2.2.

Table 2.2 Huffman table of AC coefficients

run/category	code length	code word
0/0 (EOB)	4	1010
15/0 (ZRL)	11	11111111001
0/1	2	00
...		
0/6	7	1111000
...		
0/10	16	1111111110000011
1/1	4	1100
1/2	5	11011
...		
1/10	16	1111111110001000
2/1	5	11100
...		
4/5	16	1111111110011000
...		
15/10	16	1111111111111110

The encoding method is quite easy and straight forward. For instance, the code (0,6) is encoded as 1111000, and the code (4,5) is encoded as 1111111110001000. The final bit-stream stored in the JPEG file for example (2.8) is as (2.14).

1111000 1111001 , 111000 101101 , 1111111110011000 10111 ,
11111110110 00001 , 1011 0111 , 11100 1 , 1010 **(2.14)**

2.2 JPEG 2000

The JPEG standard has been used for many years and provided satisfactory quality for the users, but the quality of JPEG can not fulfill the advanced requirement for image coding today. JPEG 2000 outperforms JPEG in many aspects, the major features of the JPEG 2000 standard is outlined as follows:

- High compression efficiency
- Lossless color transformation
- Region-of-Interest Coding
- Lossless and lossy compression
- Random code stream access and processing
- Error resilience

Fig. 2.5 and 2.6 show the Encoder and Decoder architecture of JPEG 2000. We will introduce the operation and theory of each block in the following sections.

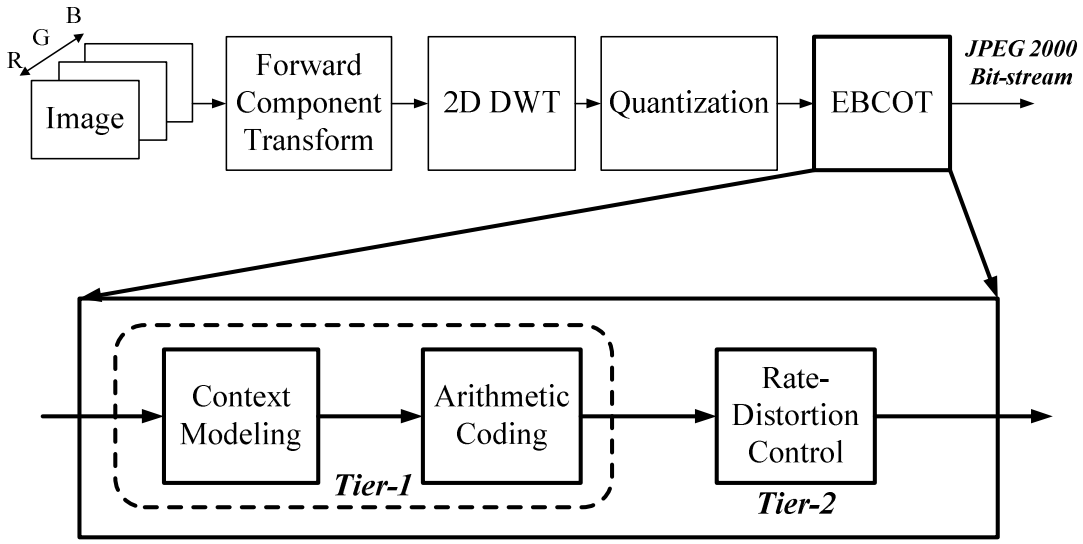

Fig 2.5 The encoder architecture of JPEG 2000

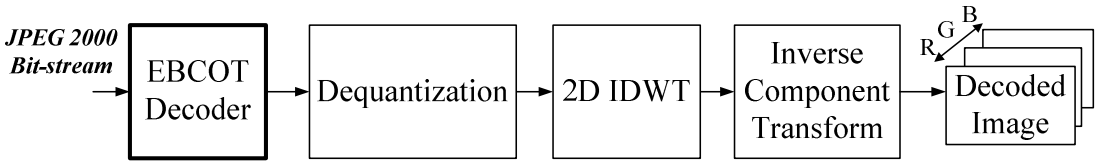

Fig 2.6 The decoder architecture of JPEG 2000

2.2.1 Color Space Conversion

Instead of using the Irreversible Color Transform, JPEG 2000 adopts the Reversible Color Transform (RCT), which is a modified YUV color transform that does not contribute to quantization errors, so it is fully reversible. Proper implementation of the RCT requires that numbers are rounded as specified that cannot be expressed exactly in matrix form. The transformation is:

$$Yr = \left\lfloor \frac{R + 2G + B}{4} \right\rfloor \;\; ; \;\; Ur = B - G \;\; ; \;\; Vr = R - G \qquad \textbf{(2.15)}$$

The Inverse transformation can be obtained by (2.16).

$$G = Yr - \left(\frac{Ur + Vr}{4} \right) \;\; ; \;\; R = Ur + G \;\; ; \;\; B = Vr + G \qquad \textbf{(2.16)}$$

2.2.2 Discrete Wavelet Transform

Although the DCT-based image compression algorithms such as JPEG have

provided satisfactory quality, it still leaves much to be desired. Thus, the new DWT-based image compression algorithms such as JPEG 2000 became increasingly popular. DWT (Discrete Wavelet Transform) is an application of subband coding; thus, before introducing DWT, we briefly describe the theory of subband coding.

Subband Coding

In subband coding, the spectrum of the input is decomposed into a set of bandlimitted components, which is called subbands. Ideally, the subbands can be assembled back to reconstruct the original spectrum without any error. Fig. 2.7 shows the block diagram of two-band filter bank and the decomposed spectrum. At first, the input signal will be filtered into lowpass and highpass components through analysis filters. After filtering, the data amount of the lowpass and highpass components will become twice that of the original signal; therefore, the lowpass and highpass components must be downsampled to reduce the data quantity. At the receiver, the received data must be upsampled to approximate the original signal. Finally, the upsampled signal passes the synthesis filters and is added to form the reconstructed approximation signal.

After subband coding, the amount of data does not reduce in reality. However, the human perception system has different sensitivity to different frequency band. For example, the human eyes are less sensitive to high frequency-band color components, while the human ears is less sensitive to the low-frequency band less than 0.01 Hz and high-frequency band larger than 20 KHz. We can take advantage of such characteristics to reduce the amount of data. Once the less sensitive components are reduced, we can achieve the objective of data compression.

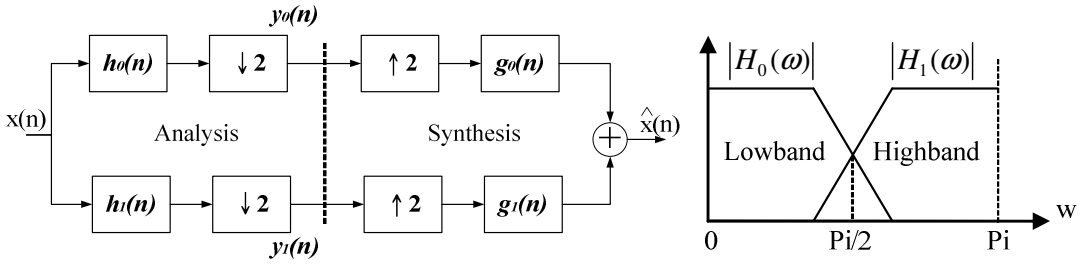

Fig 2.7 Two-band filter bank for one-dimension subband coding and decoding

Now back to the discussion on the DWT. In two dimensional wavelet transform, a two-dimensional scaling function, $\phi(x,y)$, and three two-dimensional wavelet function $\psi^H(x,y)$, $\psi^V(x,y)$ and $\psi^D(x,y)$, are required. Each is the product of a one-dimensional scaling function $\phi(x)$ and corresponding wavelet function $\psi(x)$.

$$\phi(x,y)=\phi(x)\phi(y) \qquad \psi^H(x,y)=\psi(x)\phi(y)$$
$$\psi^V(x,y)=\phi(y)\psi(x) \qquad \psi^D(x,y)=\psi(x)\psi(y) \qquad \textbf{(2.17)}$$

where ψ^H measures variations along columns (like horizontal edges), ψ^V responds to variations along rows (like vertical edges), and ψ^D corresponds to variations along diagonals.

Similar to the one-dimensional discrete wavelet transform, the two-dimensional DWT can be implemented using digital filters and samplers. With separable two-dimensional scaling and wavelet functions, we simply take the one-dimensional DWT of the rows of $f(x, y)$, followed by the one-dimensional DWT of the resulting columns. Fig. 2.8 shows the block diagram of two-dimensional DWT

.

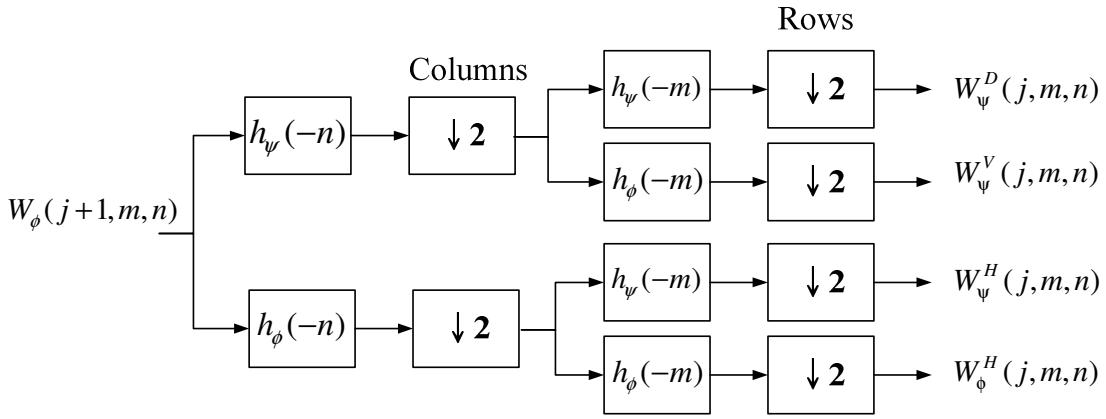

Fig. 2.8 The analysis filter bank of the two-dimensional DWT

As in the one-dimensional case, image $f(x, y)$ is used as the first scale input, and output four quarter-size sub-images — W_ϕ, W_ψ^H, W_ψ^V, and W_ψ^D as shown in the middle of Fig. 2.9. The approximation output $W_\phi^H(j,m,n)$ of the filter banks in Fig. 2.8 can be tied to other input analysis filter bank to obtain more subimages, producing the two-scale decomposition as shown in the left of Fig. 2.9. Fig. 2.10 shows the synthesis filter bank that reverses the process described above.

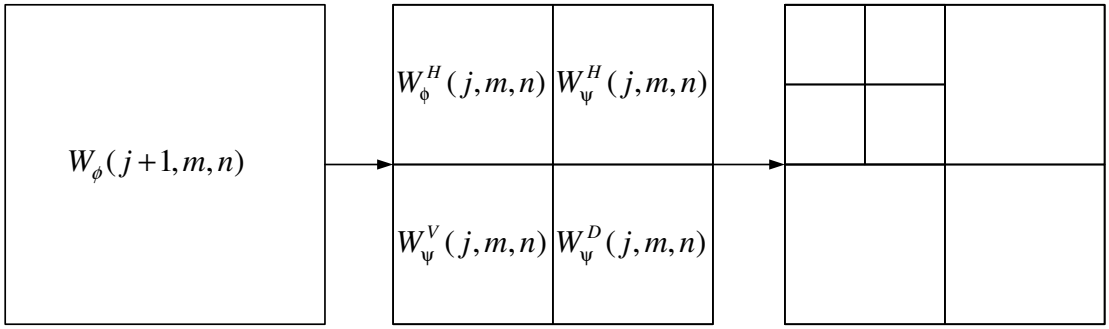

Fig. 2.9 Two-scale of two-dimensional decomposition

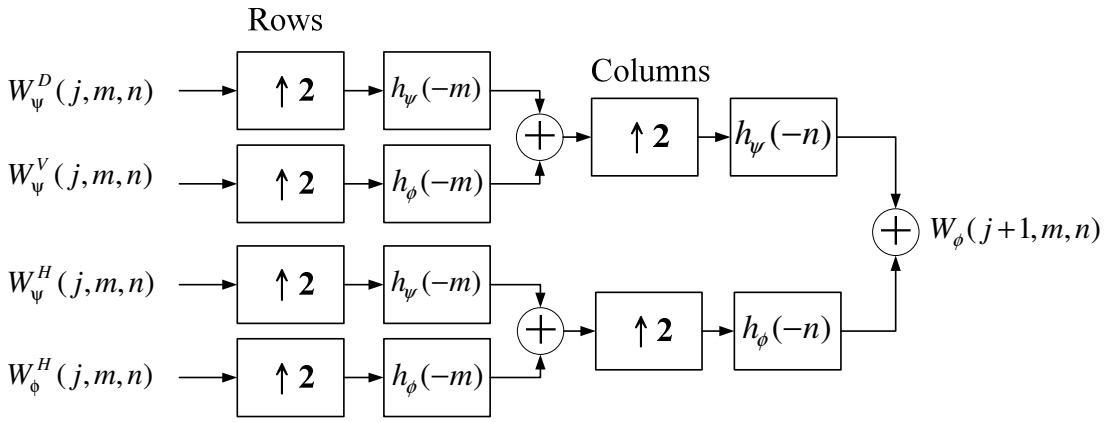

Fig. 2.10 The synthesis filter bank of the two-dimensional DWT

2.2.3 Quantization in JPEG 2000

To reduce the number of bits needed to represent the transform coefficients, the coefficient $a_b(u,v)$ of subband b is quantized to value $q_b(u,v)$ using (2.18)

$$q_b(u,v) = sign\left[a_u(u,v)\right] \bullet floor\left[\frac{|a_b(u,v)|}{\Delta_b}\right] \quad \textbf{(2.18)}$$

where the quantization step size is $\Delta_b = 2^{R_b-\varepsilon_b}\left(1+\frac{\mu_b}{2^{11}}\right)$ **(2.19)**

Rb is the nominal dynamic range of subband b, and ε_b and μ_b are the number of bits allotted to the exponent and mantissa of the subband's coefficients, respectively.

The nominal dynamic range of subband b is the sum of the number of bits used to represent the original image and the analysis gain bits for subband b.

Quantization operation is defined by the step size Δb, the selection of the step size is quite flexible, but there are a few restrictions imposed by the JPEG 2000 standard.

1. **Reversible wavelets**: when reversible wavelets are utilized in JPEG 2000, uniform deadzone scalar quantization with a step size of $\Delta b = 1$ must be used.
2. **Irreversible wavelets**: when irreversible wavelets are utilized in JPEG 2000, the step size selection is restricted only by the signaling syntax itself. The step size is specified in terms of an exponent ε_b, $0 \leqq \varepsilon_b < 2^5$, and a mantissa μ_b, $0 \leqq \mu_b < 2^{11}$.

2.2.4 EBCOT in JPEG 2000

Embedded Block Coding with Optimized Truncation (EBCOT) is adopted for the

entropy coding of JPEG 2000. The EBCOT can be divided into two steps: Tier-1 and Tier-2 as shown in Fig. 2.11. The Tier-1 part is composed of the context formation and Arithmetic Coding. The Tier-1 encoder divides the input DWT coefficients into separate code blocks and encodes each block into block-based bit-stream. After Tier-1 coding operation, the Tier-2 truncates the embedded bit-stream to minimize the embedded bit-stream. The theory of context formation, arithmetic coding and rate-distortion control will be described in the next sections.

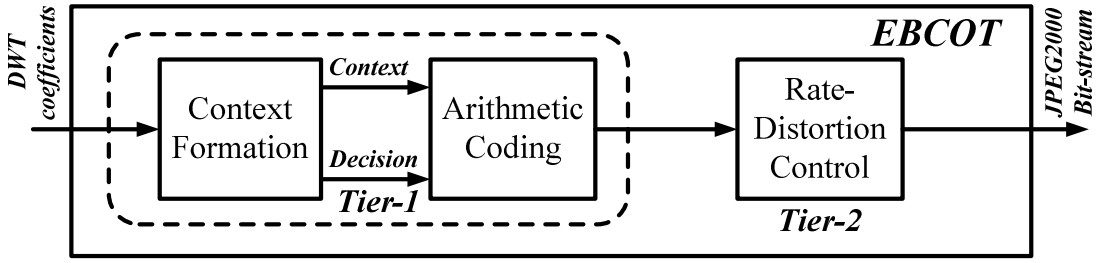

Fig. 2.11 The block diagram of EBCOT

2.2.4.1 Context Formation

The objective of context formation is to generate the context decision pairs for the arithmetic coder. We introduce some relevant concept in advance.

- **Bit-plan scanning**: The decimal DWT coefficients can be converted into signed binary format, so the DWT coefficients are decomposed into many 1-bit planes. Take one DWT coefficient for example, a bit is called **significant** after the first bit '1' is met from MSB to LSB, and the bits '0' before this bit '1' are **insignificant**, as shown in Fig. 2.12.

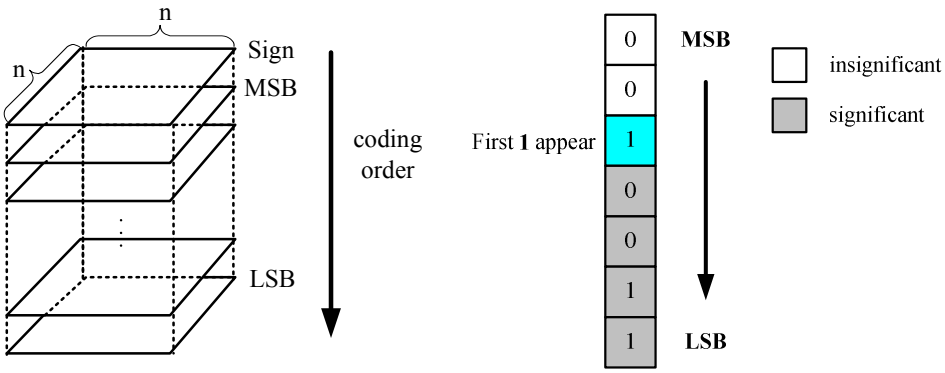

Fig. 2.12 Bit-plan scanning and the significant and insignificant bits

The scanning order of the bit-plane is shown as Fig. 2.13. Each element of the bit-plane is called a **sample**, and four vertical samples can form one **column stripe**. The 16 column stripes in the horizontal direction can form a **full stripe**.

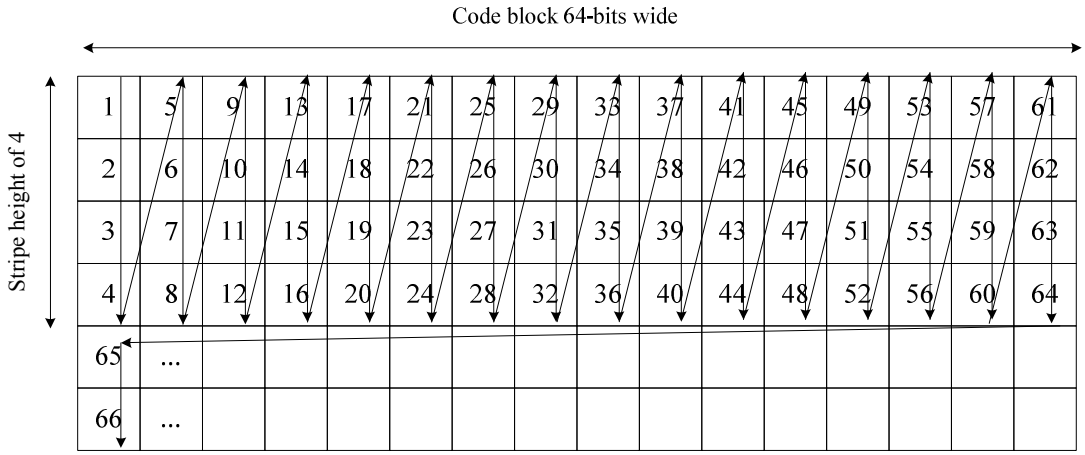

Fig. 2.13 Context formation scanning sequence for one bit-plane

- **The context window**: The context window of JPEG 2000 is shown in Fig. 2.14, the "curr" is the sample which is to be coded, and the other 8 samples are its neighbor samples. The samples *1d*, *3d*, *6d*, and *8d* are the diagonal ones. The samples *2v* and *7v* are the vertical ones. The samples *4h* and *5h* are the horizontal ones.

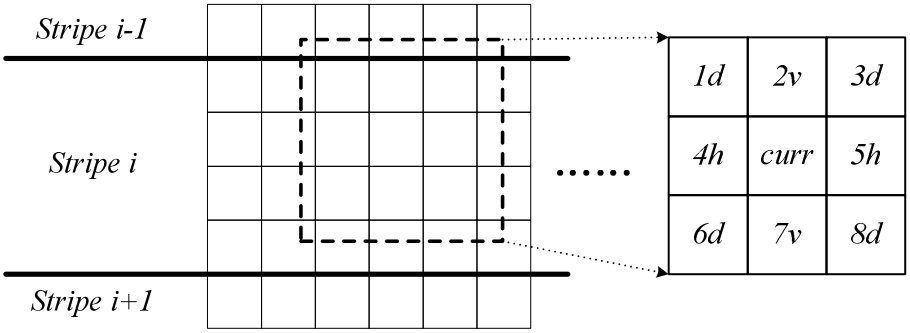

Fig. 2.14 The context window of JPEG 2000

- Three coding passes: refer to Fig 2.14

2. **Significance Propagation Pass (pass1)**: Scanning all insignificant samples which have at least one of the neighbors become significant to determine whether it will become significant at current bit plane.
3. **Magnitude Refinement Pass (pass2)**: The coefficients-bits have become significant in previous bit-plane will be coded in pass2.
4. **Cleanup Pass (pass3)**: The remained coefficients rejected by pass1 and pass2 are coded in pass3.

- Four Types of coding operations for Arithmetic coding

1. **Zero Coding (ZC)**: This coding method is used to code the new significance. The context is determined according to the significance of the neighbors. The

context assignment table for zero coding is defined in Table 2.3.

Table 2.3 Context assignment table for zero coding

LL and LH subbands			HL subband			HH subband		
ΣH	ΣV	ΣD	ΣH	ΣV	ΣD	Σ(H+V)	ΣD	Context Label
2	x	x	x	2	x	x	≥3	8
1	≥1	x	≥1	1	x	≥1	2	7
1	0	≥1	0	1	≥1	0	2	6
1	0	0	0	1	0	≥2	1	5
0	2	x	2	0	x	1	1	4
0	1	x	1	0	x	0	1	3
0	0	≥2	0	0	≥2	≥2	0	2
0	0	1	0	0	1	1	0	1
0	0	0	0	0	0	0	0	0

2. **Sign Coding (SC):** This coding method is used to code the sign samples right after the corresponding coefficient is identified significant. The context is determined by the sign of the four neighbors in the vertical and horizontal directions. There are two relevant tables defined in Table 2.4 and 2.5, respectively: (a) Sign contribution of the H and V neighbors for sign coding. (b) Context assignment table for sign coding. The value of the decision can be obtained form $D = \text{sign bit} \oplus \text{XOR bit}$

Table 2.4 Sign contribution of the H and V neighbors for sign coding

Sign Distribution	Significant (+)	Significant (-)	Insignificant
Significant (+)	1	0	1
Significant (-)	0	-1	-1
Insignificant	1	-1	0

Table 2.5 Context assignment table for sign coding

Horizontal Contribution	Vertical Contribution	Context Label	XOR bit
1	1	13	0
1	0	12	0
1	-1	11	0
0	1	10	0
0	0	9	0
0	-1	10	1
-1	1	11	1
-1	0	12	1
-1	-1	13	1

3. **Magnitude Refinement (MR)**: The context depends on the significance of its neighbors and whether it is the first time for refinement. The context is determined by the summation of the significance state of the horizontal, vertical, and diagonal neighbors.

Table 2.6 Context assignment table for magnitude refinement

ΣH + ΣV + ΣD	1st refinement for this coefficient	Context label
X	False	16
≥1	True	15
0	True	14

4. **Run-Length Coding (RLC)**: This coding method is used to reduce the average number of symbols needed to be coded. RLC must satisfy the following two condition:
   1. Four consecutive coefficients in the same stripe must be insignificant.
   2. All consecutive neighbors for the four coefficients must be insignificant.

Only one context is needed when all the four samples are insignificant. If any one of the four samples becomes significant, more than one context is needed to indicate the location of the significant.

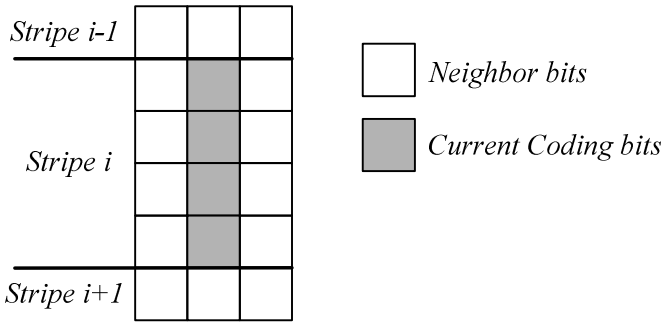

Fig. 2.15 Current coding bits and their neighbors for RLC operation

2.2.4.2 Arithmetic Coding

The decision and context data generated from context formation is coded in the arithmetic encoder (AE). The arithmetic encoder used by JPEG 2000 standard is a binary adaptive MQ coder. The basis of the binary arithmetic coding is a recursive probability interval subdivision process. Since it is a binary AE, there are only two sub-intervals. With each decision, the current probability interval is subdivided into two sub-intervals. If the value of decision is 1 then it means that it is More Possible Symbol (MPS). Otherwise, the value of decision is 0 and it means that it is Less Possible Symbol. The data distribution is shown in Fig. 2.16. The probability of MPS and LPS is represented by the gray interval and the white interval, respectively. The basic operation of the arithmetic encoder is calculating the new MPS and LPS according to the context and the decision form context formation. Because the AE of EBCOT is an adaptive encoder, the intervals of MPS and LPS will dynamically change with the value of decision. For example, if the context and the decision are equal to the value of MPS, then code MPS; otherwise, code LPS. The process is shown in Fig. 2.17 .The operation stops until all context and decision are coded.

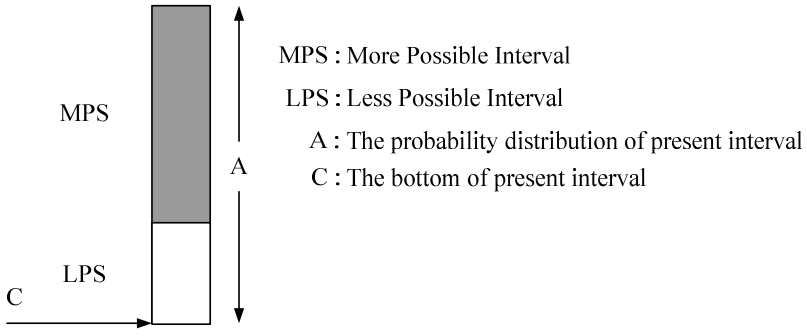

Fig. 2.16 The probability distribution of the MPS and LPS

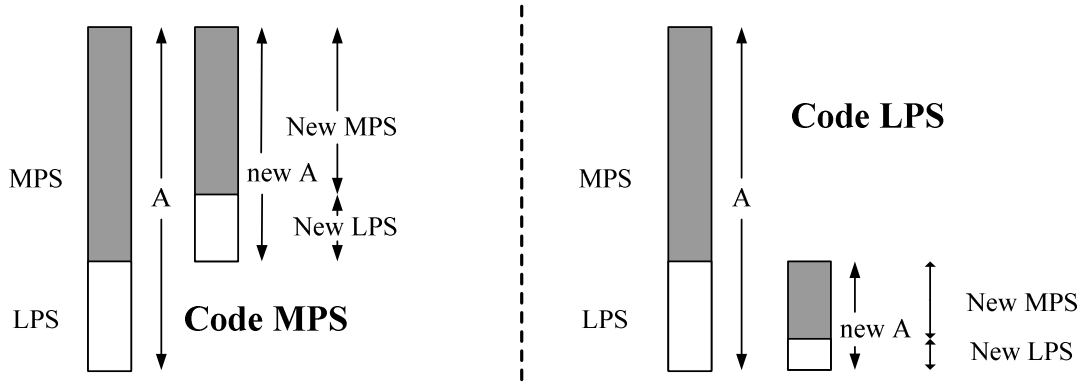

Fig. 2.17 The MPS and LPS of the arithmetic encoder

2.2.4.3 Rate Distortion Optimization

Each coding from tier-1 is a candidate truncation point of a code-block and the coding pass information is packaged into data units called packets in tier-2 coding. For meeting a target bit-rate or transmission time, the packaging process imposes a particular organization of coding pass data in the output code-stream. The rate-control assures that the desired number of bytes used by the code-stream while assuring the highest image quality possible.

3 Shape-Adaptive Image Compression

Both the JPEG and JPEG 2000 image compression standard can achieve great compression ratio, however, both of them do not take advantage of the local characteristics of the given image effectively. Here is one new image compression algorithm proposed by Huang [4], it is called Shape Adaptive Image Compression, which is abbreviated as SAIC. Instead of taking the whole image as an object and utilizing transform coding, quantization, and entropy coding to encode this object, the SAIC algorithm segments the whole image into several objects, and each object has its own local characteristic and color. Because of the high correlation of the color values in each image segment, the SAIC can achieve better compression ratio and quality than conventional image compression algorithm.

The architecture of the shape-adaptive image compression is shown in Fig. 3.1. There are three parts of operation. First, the input image is segmented into boundary part and internal texture part. The boundary part contains the boundary information of the object, while the internal texture part contains the internal contents of the object such as color value. The two parts of the object is transformed and encoded, respectively. Finally, the two separate encoded data will be combined into the full

bit-stream, and the processing of shape-adaptive image compression is finished. The theory and operation of each part will be introduced in the following sections.

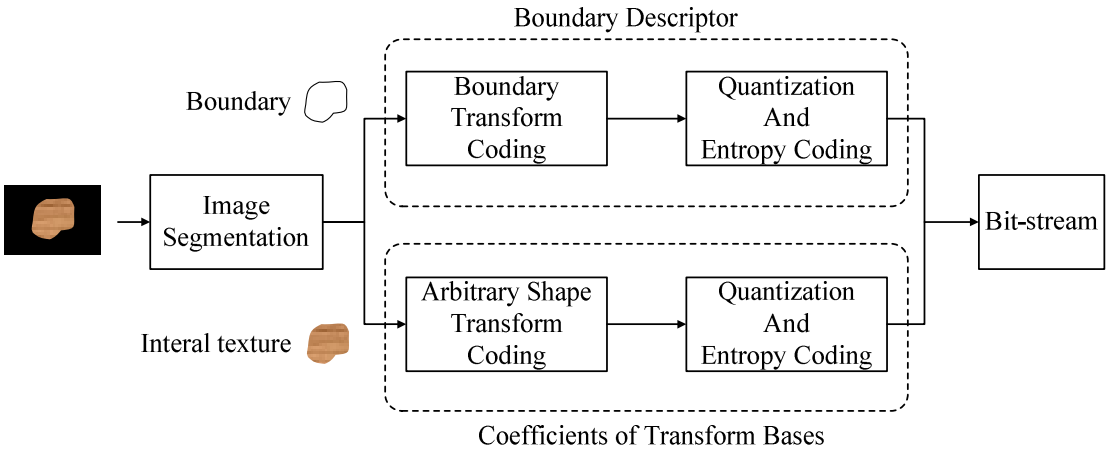

Fig. 3.1 The block diagram of shape-adaptive image compression

3.1 Boundary Description and Compression

First we obtain the boundary of the each object from the given image by means of image segmentation algorithm. Secondly, the boundary must be divided into two parts, and each part forms a non-closed boundary. We can make use of a second order polynomial to approximate the non-closed boundary. The equation of the second order polynomial is represented in (3.1) and (3.2). The original and the approximate curve are shown in Fig. 3.1.

$$y_k = Ax_k^2 - Ax_{K-1}x_k, \quad \text{for } k = 0,1,...,K-1. \tag{3.1}$$

$$A = -\frac{\sum_{k=0}^{K-1}\left[s_3(k)\left(-x_k^2 + x_{K-1}x_k\right)\right]}{\sum_{k=0}^{K-1}\left(-x_k^2 + x_{K-1}x_k\right)^2}. \tag{3.2}$$

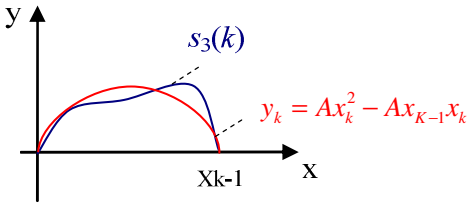

Fig. 3.1 The original boundary $s_3(k)$ and the approximate 2nd order curve y_k.

After the approximation of the boundary obtained, we express the coordinate of each points in the approximation as a complex number so the two dimensional data

can be transformed into one dimensional data. The mathematical expression is in (3.3).

$$s(k) = x(k) + jy(k) \quad \text{for } k = 0, 1, 2, ..., K-1 \tag{3.3}$$

s(k) can be transformed into frequency domain by means of Fourier descriptor. The frequency components are expressed by *a(u)*, and the forward transform is

$$a(u) = \sum_{k=0}^{K-1} s(k)\, e^{-j2\pi uk/K} \quad \text{for } u = 0, 1, 2, ..., K-1 \tag{3.4}$$

In order to reduce the amount of data required to save the boundary, we truncate the less necessary high-frequency components. However, this may result in distortion, and the recovery boundary will be closed as shown in Fig. 3.2.

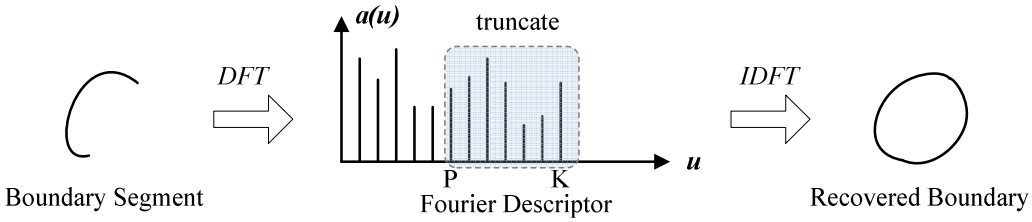

Fig. 3.2 Distortion due to truncation of high frequency data

In Ref. [4], the author proposes an effective method to solve the distortion problem. The steps of the moehod is shown in Fig. 3.3 and listed as follows:

1. Record the coordinate of two end points of the non-closed boundary.
2. Shift the boundary points linearly according to the distance on the curve between the two end points.
3. Add a boundary segment which is odd-symmetric to the original one, and transform the two-dimensional data into one-dimensional complex number.
4. Transform the one-dimensional complex data from spatial domain into frequency domain.

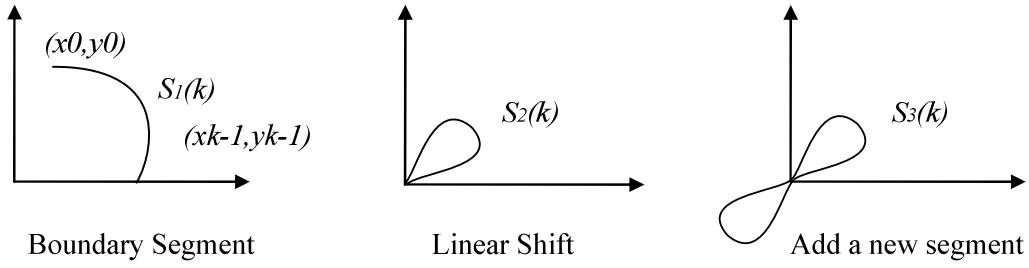

Fig. 3.3 The steps to solve the non-closed problem

Odd-symmetry property of DFT:

$$-s(-k) \xrightarrow{DFT} -a(-u)\,. \tag{3.5}$$

Therefore, if the signal $s(k)$ is odd-symmetric, its DFT $a(u)$ is also odd-symmetric.

$$s(k) = -s(-k) \xrightarrow{DFT} a(u) = -a(-u)\,. \tag{3.6}$$

Since the expanded data is odd symmetric to the original data, we can obtain the symmetric part from the original boundary by (3.6). Therefore, the odd symmetric part is useless and can be truncated, and the boundary can achieve compression of the boundary segments successfully. The frequency data can be encoded with entropy coding algorithm afterward.

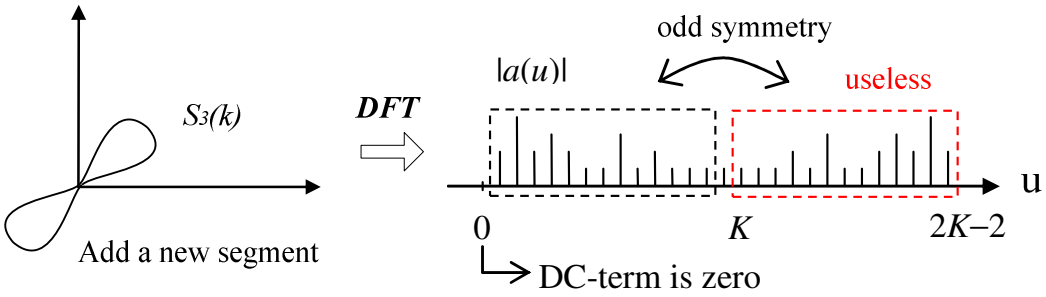

Fig. 3.4 The new boundary segment and its Fourier descriptor

3.2 Shape-Adaptive Transform Coding

After the internal texture of one image segment is obtained, we can take DCT of the data of the internal texture. The theory of conventional N-by-N DCT is described in section 2.1.1. However, the height and width of one image segment are usually not equal. Hence, we redefine the DCT as Eq. (3.7) and (3.8):

Forward Shape-Adaptive DCT

$$F(k) = \sum_{x=0}^{W-1}\sum_{y=0}^{H-1} f(x,y)\omega'_{x,y}(k), \quad \text{for } k = 1,2,...,M$$
$$C(k) = \begin{cases} 1/\sqrt{2} & \text{for } k = 0 \\ 1 & \text{otherwise} \end{cases} \tag{3.7}$$

Inverse Shape-Adaptive DCT

$$f(x,y) = \frac{2}{\sqrt{H*W}} \sum_{k=1}^{M} F(k)\omega'_{x,y}(k), \quad \text{for } x = 0,...,W-1 \text{ and } y = 0,...,H-1$$
$$C(k) = \begin{cases} 1/\sqrt{2} & \text{for } k = 0 \\ 1 & \text{otherwise} \end{cases} \tag{3.8}$$

where W and H are the width and height of the image segment, respectively. M is the length of the DCT coefficients.

The $F(u,v)$ is called the Shape-Adaptive DCT coefficient, and the DCT basis is

$$\omega_{x,y}'(u,v) = \frac{2C(u)C(v)}{\sqrt{H*W}} \cos\left[\frac{\pi(2x+1)u}{2W}\right] \cos\left[\frac{\pi(2y+1)v}{2H}\right] \quad \textbf{(3.9)}$$

Because the number of points M is less than or equal to $H \times W$, we can know that the $H \times W$ bases may not be orthogonal. We can obtain the M orthogonal bases by means of Gram-Schmidt process. After obtaining the orthogonal bases, we can take Shape-Adaptive DCT afterward.

3.3 Shape-Adaptive Quantization

After the transform coefficients are obtained, we quantize the coefficients to compress the data. Because the length of the coefficients of the arbitrary shape is not fixed, the quantization table for JPEG in (2.4) and (2.5) must be modified. Huang [4] propose the unfixed quantization array as follows:

$$Q(k) = Q_a k + Q_c, \quad \text{for } k = 1,2,...,M \quad \textbf{(3.10)}$$

where the two parameters Q_a and Q_c are the slope and the intercept of the line, respectively. And M is the length of the DCT coefficients.

Each DCT coefficient $F(k)$ is divided by the corresponding quantization array $Q(k)$ and rounded to the nearest integer as (3.11):

$$F_q(k) = Round\left(\frac{F(k)}{Q(k)}\right), \quad \text{where } k = 1,2,...,M \quad \textbf{(3.11)}$$

3.4 Shape-Adaptive Entropy Coding

Before introducing the theory of Shape-Adaptive entropy coding, here is one problem must be solved. Because the variation of color around the boundary is large, if we truncate these high frequency components, the original image may be corrupted by some unnecessary noise.

In Ref. [4], the author proposes a method to solve this problem. Since most of the variation is around the boundary region, we can divide the internal texture segments into internal part and boundary region part.

The way to divide the image segment is making use of morphological erosion. For two binary image set A and B, the erosion of A by B, denote $A \ominus B$, is defined as

$$A \ominus B = \{z \mid (B)_z \subseteq A\} \quad \textbf{(3.12)}$$

where the translation of set B by point $z = (z_1, z_2)$, denoted $(B)_z$, is defined as

$$(B)_z = \{c \mid c = b + z, \quad \text{for } b \in B\} \quad \textbf{(3.13)}$$

That is, the erosion of A by B is the set of all points z such that B which is translated by z is contained in A. Base on the definition described above, we erode the

shapes of image segments by a 5×5 disk-shape image and we get the shape of the internal region. Then we subtract it from the original shape and we get the shape of boundary region. The process is illustrated in Fig. 3.5.

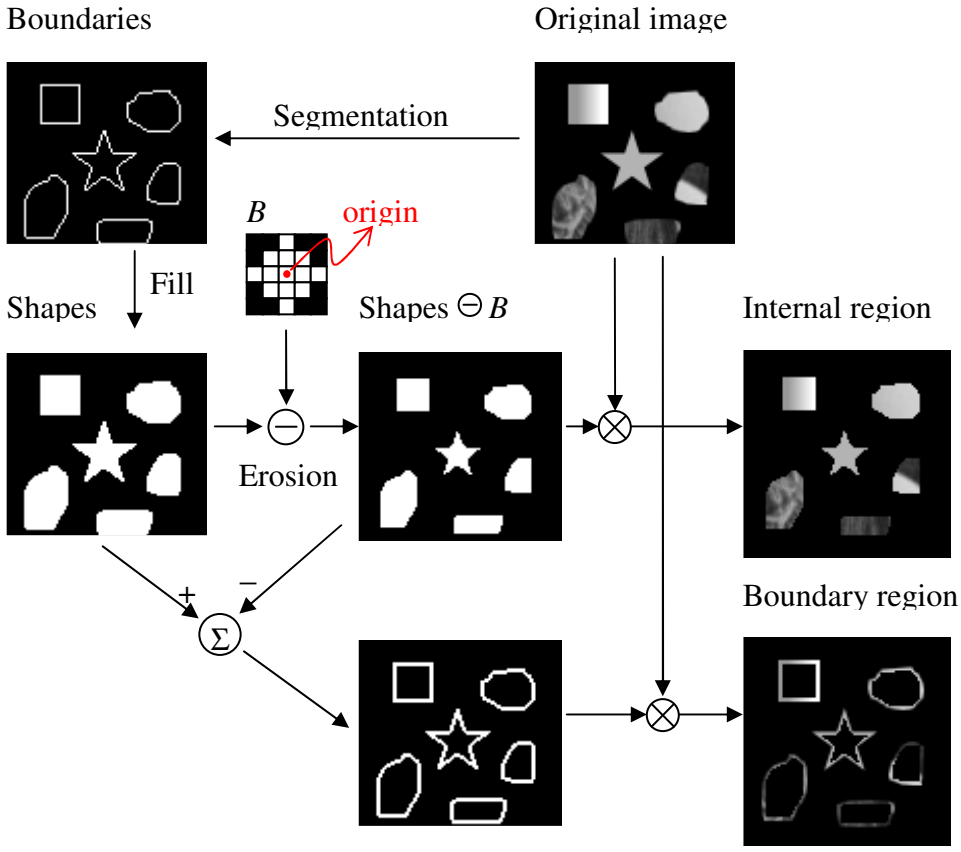

Fig. 3.5 Divide the image segment by morphological erosion

After transforming and quantizing the internal texture, we can encode the quantized coefficients and combine these encoded internal texture coefficients with the encoded boundary bit-stream. The full encoding process is shown in Fig. 3.6.

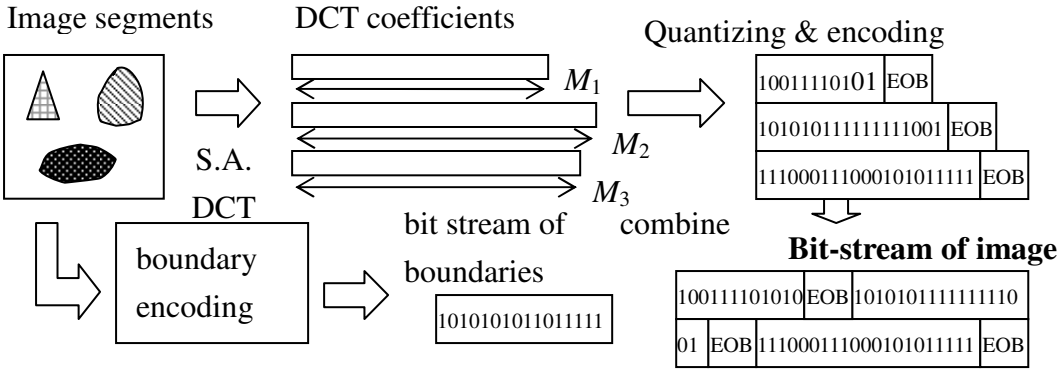

Fig. 3.6 Process of Shape-Adaptive Coding Method

4 Conclusion and Future Work

The DCT-based image compression such as JPEG performs very well at moderate bit rates; however, at higher compression ratio, the quality of the image degrades because of the artifacts resulting from the block-based DCT scheme.

Wavelet-based coding such as JPEG 2000 on the other hand provides substantial improvement in picture quality at low bit rates because of overlapping basis functions and better energy compaction property of wavelet transforms. Because of the inherent multi-resolution nature, wavelet-based coders facilitate progressive transmission of images thereby allowing variable bit rates.

We also briefly introduce the technique that utilizes the statistical characteristics for image compression. The new image compression algorithm called Shape-Adaptive Image Compression, which is proposed by Huang [5], takes advantage of the local characteristics for image compaction. The SAIC compensates for the shortcoming of JPEG that regards the whole image as a single object and do not take advantage of the characteristics of image segments.

However, the current data compression methods might be far away from the ultimate limits. Interesting issues like obtaining accurate models of images, optimal representations of such models, and rapidly computing such optimal representations are the grand challenges facing the data compression community. Image coding based on models of human perception, scalability, robustness, error resilience, and complexity are a few of the many challenges in image coding to be fully resolved and may affect image data compression performance in the years to come.

5 References

[1] R. C. Gonzalea and R. E. Woods, "*Digital Image Processing*", 2_{nd} Ed., Prentice Hall, 2004.

[2] Liu Chien-Chih, Hang Hsueh-Ming, "Acceleration and Implementation of JPEG 2000 Encoder on TI DSP platform" *Image Processing, 2007.* ICIP 2007. IEEE International Conference on, Vo1. 3, pp. III-329-339, 2005.

[3] ISO/IEC 15444-1:2000(E), "Information technology-JPEG 2000 image coding system-Part 1: Core coding system", 2000.

[4] Jian-Jiun Ding and Jiun-De Huang, "Image Compression by Segmentation and Boundary Description", Master's Thesis, National Taiwan University, Taipei, 2007.

[5] Jian-Jiun Ding and Tzu-Heng Lee, "Shape-Adaptive Image Compression",

Master's Thesis, National Taiwan University, Taipei, 2008.

[6] G. K. Wallace, "The JPEG Still Picture Compression Standard", *Communications of the ACM*, Vol. 34, Issue 4, pp.30-44, 1991.

[7] 張得浩，"*新一代 JPEG 2000 之核心編碼 — 演算法及其架構(上)*"，IC 設計月刊　2003.8 月號.

[8] 酒井善則、吉田俊之 共著，白執善 編譯，"*影像壓縮技術*"，全華，2004.

[9] Subhasis Saha, "Image Compression - from DCT to Wavelets : A Review", available in http://www.acm.org/crossroads/xrds6-3/sahaimgcoding.html