CHAPTER 2

THEORY AND LITERATURE REVIEW

2.1 Theoretical Considerations

RGB and Indexed Color System, fundamentals of image compression, and image compression techniques, lossless and lossy, including the Run-Length and Huffman Coding compression algorithms, are reviewed as follows:

2.1.1 RGB and Indexed Color System

RGB Color System defines colors in a unit cube by the additive color-mixing model. Red, green, and blue are additive primaries represented by the three axes of the cube, all other colors in the cube can be represented as a triplet (r,g,b) where values for r, g, and b are assigned in the range from 0 to 1[1]. Most of digital images are practically saved in this color system but one problem occurs that it uses a large amount of memory for storage if the image has very large size.

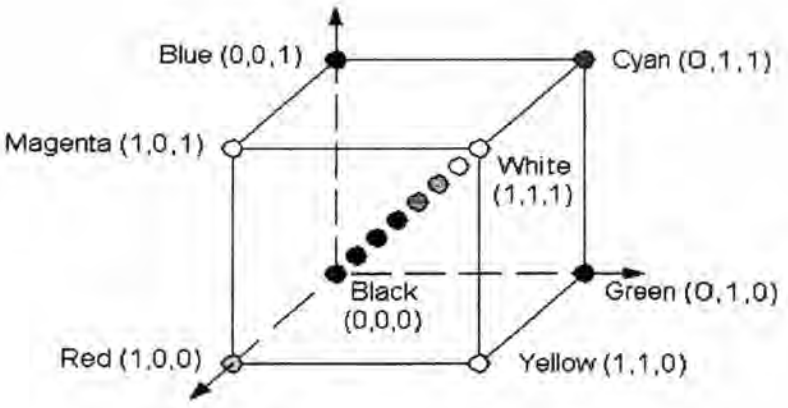

Figure 2-1 : RGB color cube

Indexed Color System is one popular system in computer graphic because it is a more economical way of storing color image. In this system, colors will be changed to the integers 0 to the last number in the color table. Each entry in the color table gives the red, green, and blue components for that particular index. There are two processes involved. The first is creating a color table, and the second is giving a red, green, and blue values determined in the index that is the closet representation in the table[2]. This color system can be assumed as one of compression technique because it reduces data in an image. It is lossless compression when the color table contains colors which allows the image to be represented exactly but lossy if there are not enough colors in the table.

2.1.2 Image Compression

Image compression is one branch of digital image processing which refers to the process of decreasing the amount of data required to represent the image. If the

amount of data necessary to represent an image can be reduced, then the amount of time to transport it is also reduced. Likewise, the amount of storage space required to store the data is reduced[3].

2.1.2.1 Image Compression Fundamental

The goal of image compression is to reduce the amount of data required to represent the information presented within an image. Given two data of same image, original image data and the compressed one, requiring d_1 and d_2 units of data. The compression ratio is defined as[4]:

$$C = \frac{d_1}{d_2} \qquad (2\text{-}1)$$

Image compression can be separated into two stages, the encoding and decoding[5], as shown in Figure 2-2. The encoding is taking an original image data and compress it. The goal of encoding is to remove redundant information in the original image. The encoding can be divided into three processes, the mapping function, quantizing, and data encoding. These three processes, both mapping function and data encoding are completely reversible because they do not remove any information from the image. On the other hand, the quantizing degrades the image quality because it removes information from the image.

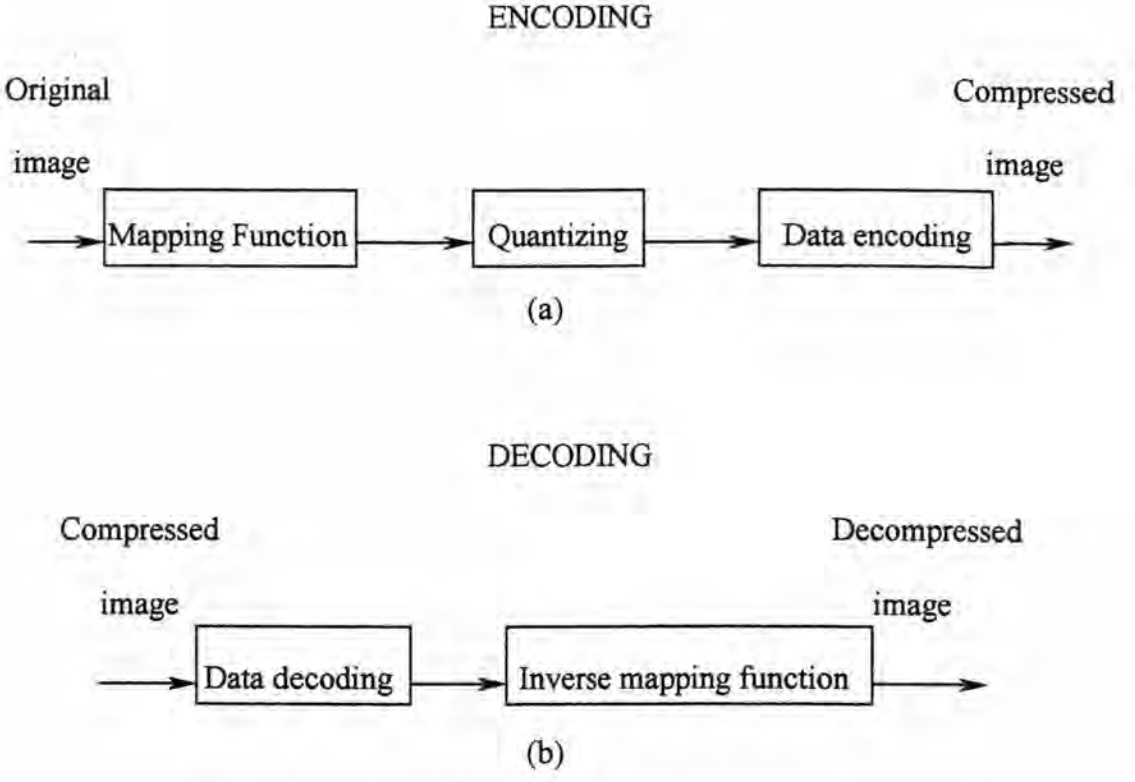

Figure 2-2 : A block diagram illustrating image compression:

(a) the encoding and (b) the decoding.

The decoding stage takes a compressed image data and decompress it to produce a new image that represents the original image. It contains only two processes, the data decoding and inverse mapping function. The output image from lossless compression is exact replica of the original image. But in lossy compression, the output image is a degraded version of the original image.

2.1.2.2 Image Data Redundancy

Image compression is based upon the removal of redundant data, which is defined as data that is not needed to represent the information in an image. The goal of image compression is to characterize these redundancies and to code them to a new form that requires less data than the original[6]. Given two data of same image, original

image data and the compressed one, requiring d_1 and d_2 units of data and with $d_1>d_2$, the relative data redundancy in percentage is[7]

$$R = \frac{d_1 \quad d_2}{d_1} \times 100\% \qquad (2\text{-}2)$$

or in terms of compression ratio,

$$R = \frac{C \quad 1}{C} \times 100\ \% \qquad (2\text{-}3)$$

In general, there are three types of data redundancies that can be eliminated, coding, spatial, and visual redundancies.

(a) Coding Redundancy

The simplest approach to lossless compression is to reduce only coding redundancy[8]. Most lossless compression, the methods are based upon removing the coding redundancy that are presented within the image by the replacement of fixed length code with variable length code. The concept of variable length code is to reduce the number of bits required to represent the information within an image.

A measurement of the coding redundancy is the average bit length defined as:

$$B_{avg} = \sum_{i=0}^{max} B_i h_i \qquad (2\text{-}4)$$

where B_i is the number of bits used to represent the i^{th} information and the histogram (h_i) of an NxM image is

$$h_i = \frac{n_i}{NM} \quad , \quad \text{for} \quad 0 \le i < max \tag{2-5}$$

where n_i is the amount of pixels at information i, NM is the total number of pixels in the image.

Based upon the information theory developed by Shannon in 1940s[9], the lowest average bits length required to represent the information within an image is called entropy:

$$ENTROPY = -\sum_{i=0}^{max} h_i \log_2(h_i) \tag{2-6}$$

Graylevels	Number of pixels	Histogram	Fixed length	Variable length
0	1116	0.0681	000	00
1	4513	0.2754	001	01
2	5420	0.3308	010	10
3	2149	0.1312	011	1100
4	1389	0.0848	100	1101
5	917	0.0560	101	1110
6	654	0.0399	110	111100
7	226	0.0138	111	111101
n_t	16384			

Table 2-1 : An example of coding redundancy

(b) Spatial Redundancy

The goal of image compression based upon spatial redundancy is to eliminate neighboring pixels that are the same in information. The amount of spatial redundancy present in an image depends on the spatial details contained within an image[10]. Variety names of spatial redundancy are interpixel, geometric, and interframe redundancy. Removal of spatial redundancy is in the mapping function of encoding stage.

Figure 2-3 shows the 6x4-pixel image that some pixels have same value. By grouping pixels that have same values to a new code set, it can eliminate spatial redundancy and can reduce a memory required for storage.

1	2	1	1	1	1
1	3	4	4	4	4
1	1	3	3	3	5
1	1	1	1	3	3

Figure 2-3 : 6x4 pixel sample image.

The rearrange of 24 pixel values in row-like manner are shown as:

1 2 1 1 1 1 1 3 4 4 4 4 1 1 3 3 3 5 1 1 1 1 3 3

A new code set, (value, run-length), that can reduce number of characters from 24 to 20 while maintaining all information is:

(1,1) (2,1) (1,5) (3,1) (4,4) (1,2) (3,3) (5,1) (1,4) (3,2)

(c) Visual Redundancy

(c) Visual Redundancy

Visual or psychovisual redundancy is based upon the limitations of human visual system. Unlike coding and spatial redundancies, which produce a new image that is the same compared with the original image, compression process using visual redundancy gives a new image that is a degraded version of the original image. The visual information is nonreversible because it is lost during the compress process. It is commonly referred to as quantizing of encoding process[11].

2.1.2.3 Image Compression Techniques

There are many techniques for image compression, which may be categorized into two fundamental groups as: lossless and lossy[12].

(a) Lossless Compression

In lossless compression technique (known as bit-preserving, reversible or error-free compression), the encoding process contains only mapping function and data encoding. The quantizing is rejected because it removes visual redundancy from an image. The decompressed image is to be the same compared with the original image because no information is lost. However, low compression ratio is possible compared with lossy. There are many algorithms used in lossless compression technique such as Huffman, Run-Length, Arithmetic, Bit-Plane, and Lossless Predictive Coding. We will focus on the Run-Length Coding that is applies to an Optimized Run-Length Coding and Huffman Coding algorithms later.

(b) Lossy Compression

For lossy compression (known as irreversable compression), the decompressed image is a degraded version of the original image. It also removes visual redundancy that can not be noticeable from the image because of the visual limitation of human eye. So much higher compression ratio can be obtained. Lossy compression is based upon two methods, Lossy Predictive and Transform coding. JPEG standard, the most one popular compressed image file format, uses Discrete Cosine Transform (DCT) to achieve compression.

2.1.2.4 Run-Length Coding

Run-Length Coding is a standard of the CCITT (Consultative Committee of the International Telephone and Telegraph). It is used in the transmission of binary image over phone lines. Image compression using this technique can eliminate spatial redundancy in an image by grouping the adjacent pixels that have same color into single code. It starts by looking at the first pixel and following neighbor pixels across the first line. It determines the color value of the first pixel and counts the adjacent pixels that have the same value. Then they are all assigned by a new code using the pair (color value, run-length). This process continues by moving to the next pixel in the line with new color value, counting the adjacent pixels that have the same value, and assigning a new code again. When the end of line is reached, the process starts again at the first pixel of the next line. Run-Length Coding is completed when all pixels have been coded.

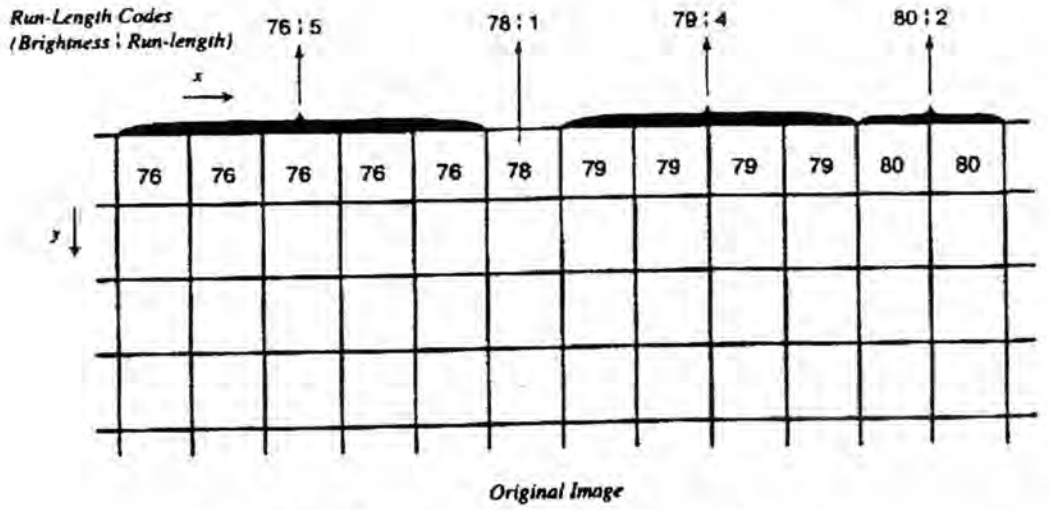

Figure 2-4 : The Run-Length Coding operation.

Run-Length Coding is effective when using with images that contain large regions of same color values. If it is based on row-like manner, spatial redundancy in the horizontal direction is rejected. While Run-Length Coding using column eliminates the spatial redundancy in the vertical direction.

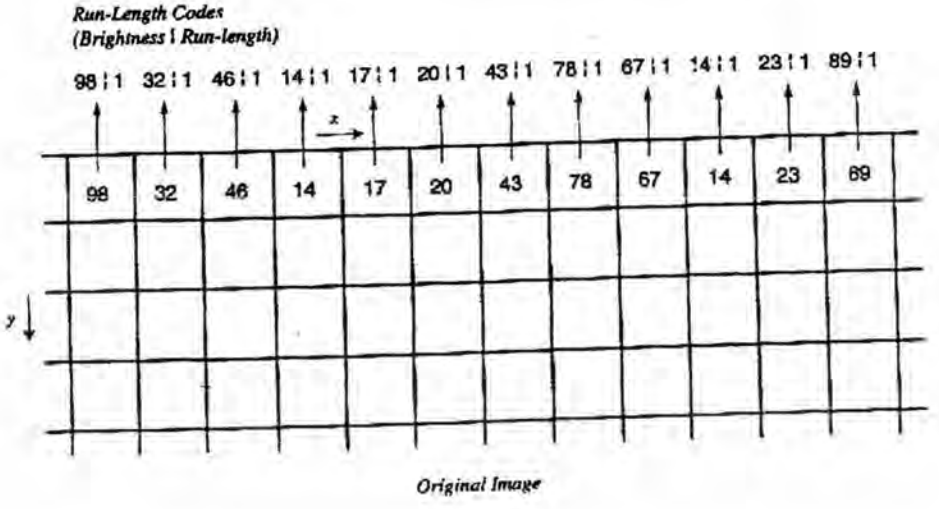

Figure 2-5 : Run-Length Coding can be prone to data explosion under certain conditions of rapid color value changes in an image.

Data channel error can cause problems when image is compressed by Run-Length Coding and then transferred over communication channel to another location, as shown in Figure 2-6. By embedding the marker at the start or the end of each new line, decompress operation can synchronize and resume correct decompression in next line. This way, only a single line of the decompressed image will be corrupted by the data error[13].

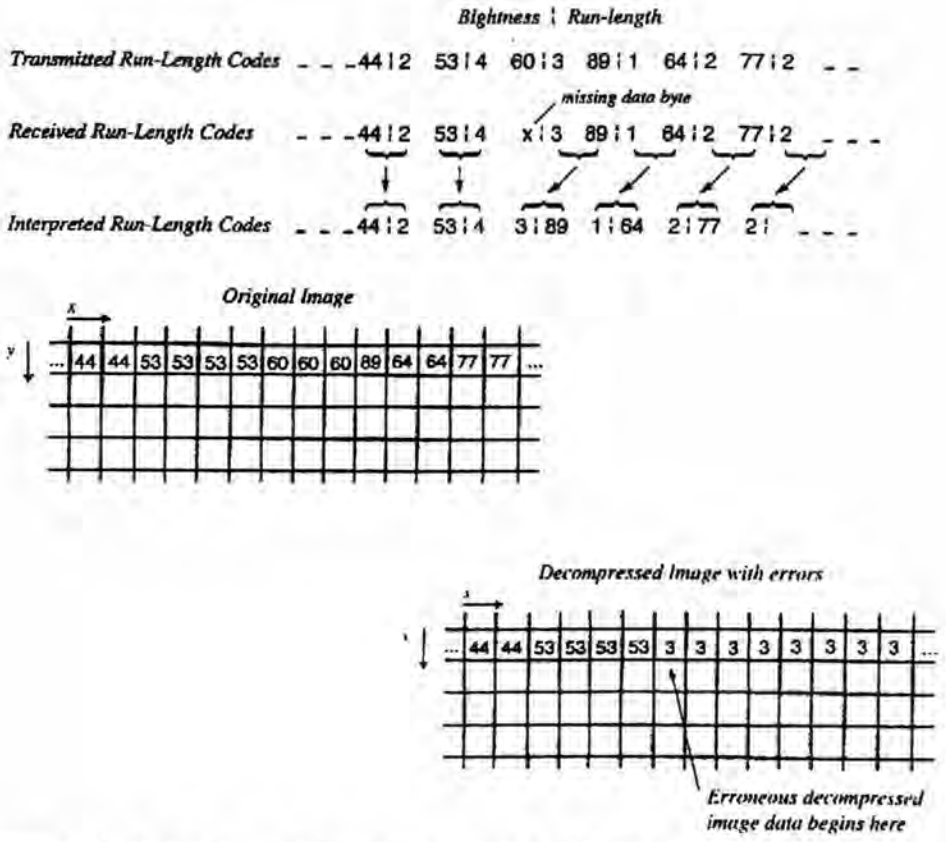

Figure 2-6 : Data–channel error can cause significant decompression problems of the corrupted Run-Length Coded image.

2.1.2.5 Huffman Coding

The most common code used to represent digital images is that of the standard binary code. For an N x M image containing G different graylevels, the number of bits B required per pixel is

$$B = \log_2(G) \tag{2-7}$$

Thus, the total number of bits required for this image is

$$B_{total} = N \cdot M \cdot B \tag{2-8}$$

Coding an image using the fixed binary code produces coding redundancy. A better method of coding an image is to use a variable length code that assigns the shortest code to the most frequently occurring graylevels within an image and the longest codes to the least frequently occurring graylevels. There are many types of variable length codes, but the most popular technique for removing coding redundancy is Huffman's[14]. This code gives an average bits length that is slightly greater than the entropy contained within the image. So this technique is referred to as entropy coding[15].

In particular, the average bits length, B_{avg}, required to represent a pixel will lie between

$$\text{entropy} < B_{avg} < \text{entropy} + 1 \tag{2-9}$$

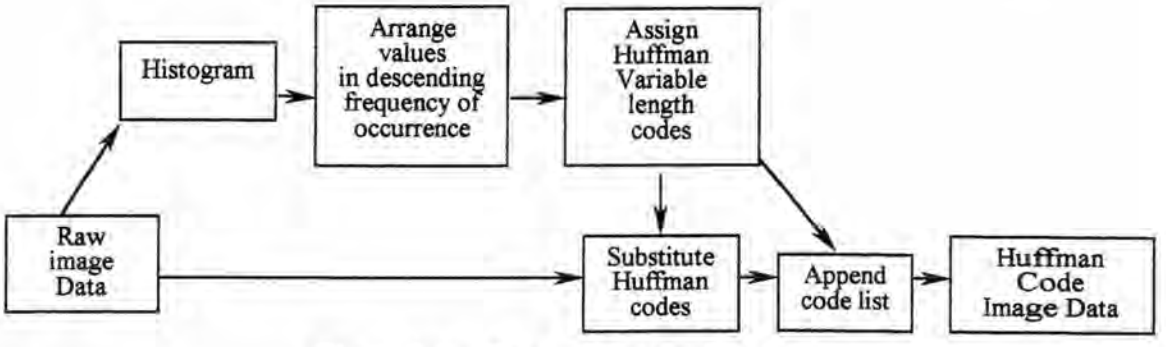

Figure 2-7 : The flow of Huffman Coding.

Huffman coding is done firstly by calculating the histogram of the image using Equation (2-5). Secondly, the probabilities of occurrence are arranged by descending order. The next step is to reduce the number of probability by combining the lowest two probabilities together to form one new probability. The other unmodified probabilities and the newly formed probability are rearranged by descending order and the lowest two probabilities are again combined to form one new probability. This process is then repeated until it remains only last two probabilities.

Graylevels	n_i	h_i
0	3441	0.21
1	4423	0.27
2	3932	0.24
3	1802	0.11
4	1311	0.08
5	819	0.05
6	492	0.03
7	164	0.01
N_t	16384	

Table 2-2 : Distribution of graylevels in an 8 graylevel image and its corresponding histogram.

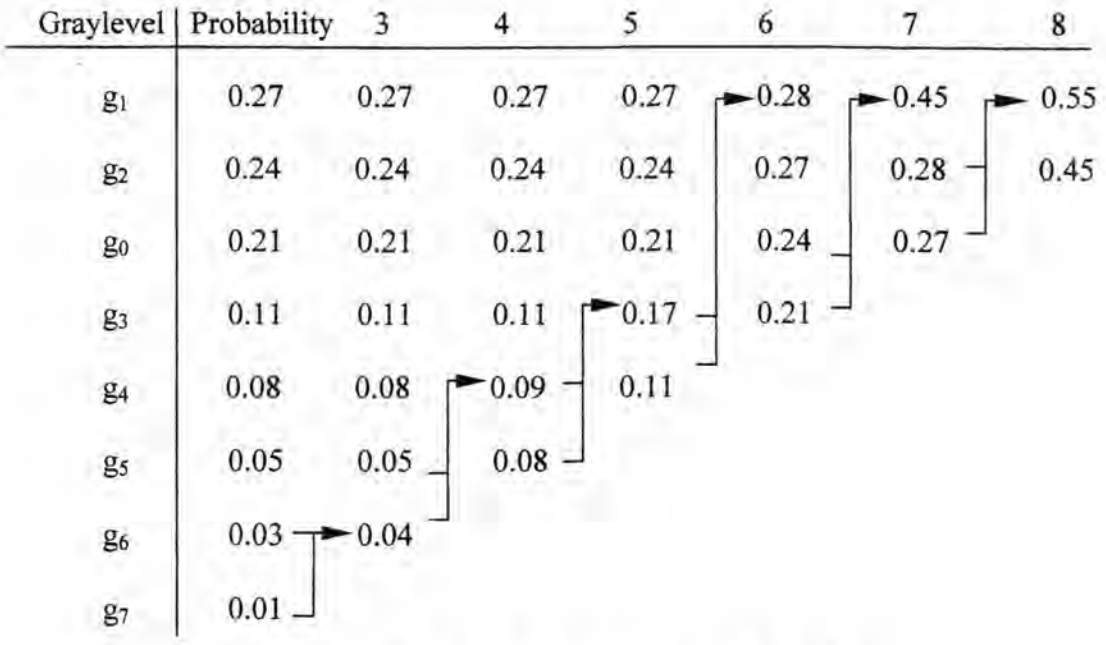

Figure 2-8 : Phase one of Huffman Coding,

reducing the number of probabilities.

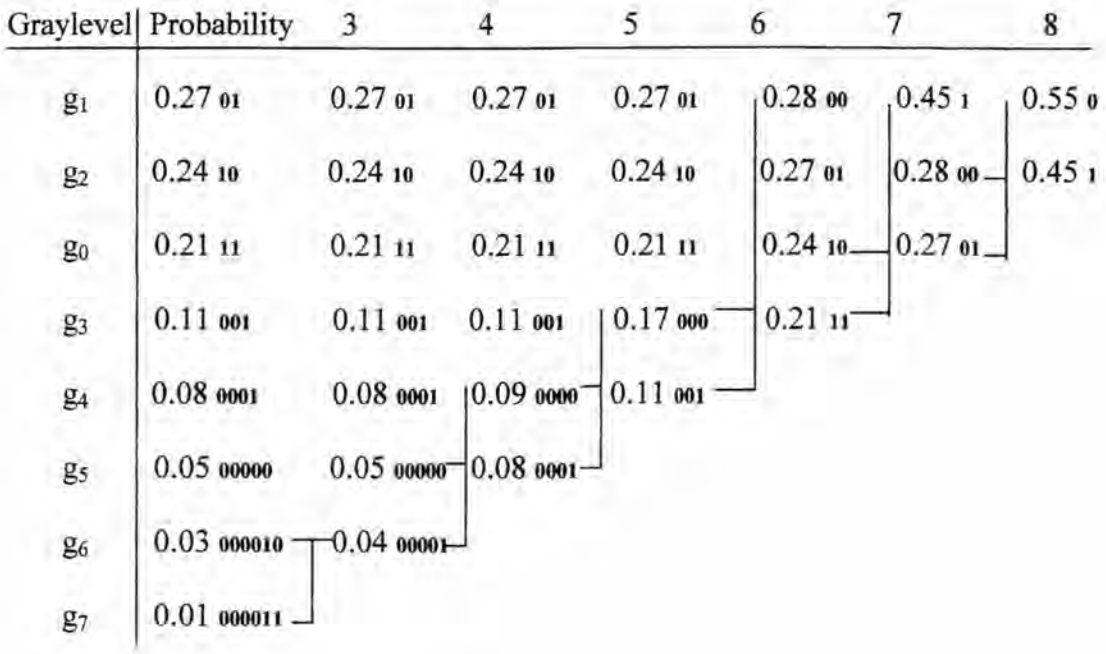

Figure 2-9 : Phase two of Huffman Coding,

assigning the codes to the graylevels.

The second step of Huffman Coding is to assign a set of codes contained with 0 and 1 to each graylevels. By reversing back from the last two probabilities in the last column in Figure 2-9, 0 and 1 are assigned to the probabilities of 0.55 and 0.45. Next, an additional 0 and 1 code are assigned to two probabilities in the next reverse column that form encountered probability. This process continues until all probabilities have its variable length codes.

In this 8-graylevel sample image, the entropy computed from Equation (2-6) gives the value of 2.55. By using Huffman coding, the average bits length computed from Equation (2-4) is the value of 2.59. It is few greater than the entropy of the image.

2.2 Literature review

Gormish[16] presented a method of lossless compression of palletized or indexed images. Indexed image was divided into several binary images or bit planes using one binary image for each color. Thus, each binary image was the set of pixels of a certain color. It was no need to compress all images because all remain pixels in the last color plane must be the last color in the image. Hence, the last image needed not be coded at all. It is especially effective on images with a small set of colors.

Jaisimha et al.[17] presented a comparative evaluation of various compression techniques as applied to color maps, Run-Length, Contour, and Quadtree coding. The results from these algorithms were coded again with Lempel-Ziv coding and were compared in terms of compression ratio and computation speed. The Quadtree coding gave a higher compression ratio than Run-Length coding. But the comparison of

computation times showed that Run-Length coding was significant faster than Quadtree coding and required less memory for the decoding algorithm. Accordingly, Huffman coding of the runs would result in more efficient bit assignment and hence greater compression ratios. The improvement in overall compression ratio when Lempel-Ziv coding was applied to the output of the Run-Length coder further indicated that Huffman coding of the runs would improve the compression ratio.

Liu et al.[18] described a method for segmentation of color geographic map images based on color opponency. A color map image was transformed from RGB color space to color-opponent representation (WB, RG, and YB), and each component was then processed separately. For chromatic channels, opponent contrast enhancement was used to separate the different regions, and foreground regions were extracted based on the local deviation threshold and luminance map of the original image. In achromatic opponent, black regions, was exempted from opponent contrast enhancement in order to reduce noise. Finally, the foreground outputs from each channel were combined to give the overall foreground regions.

Ou et al.[19] proposed a way of binary map image compression using segmentation of the map image into two images: long-line image and the short-line image. For long-line image, it was first sampled some points to represent the lines and used Arithmetic coding to code it. For short-line image, it was coded by Arithmetic coding. In the decompress process, the cubic B-spline interpolation was used to finish the curve fitting after Arithmetic decoding for the compressed long-line image. And the short-line image was then decoded with Arithmetic decoding and combined with the decompressed long-line image to reconstruct the whole image.

Overloop et al.[20] described an extension of an existing segmented image coding method for monochrome images to images containing a limited number of colors. This technique segmented the color image into two parts: The luminance image and color information. It coded the luminance image, while the color information was compactly and independently stored in a bit map using JBIG coding.