

การบีบอัดข้อมูลภาพสีโดยการแปลงเวฟเลตแบบคอเบซีส์



นางสาว ขนิษฐา โพลีทธีวิญญู

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต

สาขาวิชาวิทยาการคอมพิวเตอร์ ภาควิชาคณิตศาสตร์

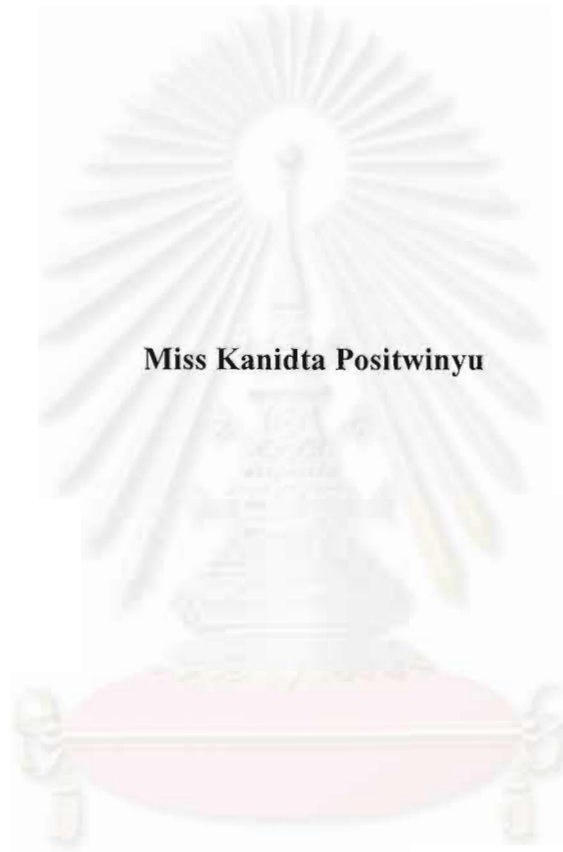
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

ปีการศึกษา 2542

ISBN 974-333-809-8

ลิขสิทธิ์ของ จุฬาลงกรณ์มหาวิทยาลัย

COLOR IMAGE DATA COMPRESSION
USING DAUBECHIES WAVELET TRANSFORM



**A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Science in Computational Science**

Department of Mathematics

Faculty of Science

Chulalongkorn University

Academic Year 1999

ISBN 974-333-809-8

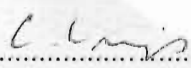
Thesis Title Color Image Data Compression Using Daubechies Wavelet Transform
By Miss Kanidta Positwinyu
Department Mathematics
Thesis Advisor Associate Professor Suchada Siripant

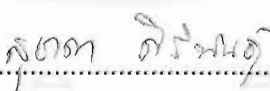



Accepted by the Faculty of Science, Chulalongkorn University in Partial Fulfillment of the Requirement for the Master's Degree


..... Dean of Faculty of Science
(Associate Professor Wanchai Phothiphichitr, Ph.D.)

Thesis Committee


..... Chairman
(Professor Chidchanok Lursinsap, Ph.D.)


..... Thesis Advisor
(Associate Professor Suchada Siripant)


..... Member
(Assistant Professor Peraphon Sophatsathit, Ph.D.)

ขนิษฐา โพลีทรีวิญญู : การบีบอัดข้อมูลภาพสีโดยการแปลงเวฟเลตแบบคอบเบชีส์ (COLOR IMAGE DATA COMPRESSION USING DAUBECHIES WAVELET TRANSFORM)

อ. ที่ปรึกษา : ร.ศ. สุชาดา ศิริพันธุ์ ; 64 หน้า. ISBN 974-333-809-8

ปัจจุบันได้มีการนำรูปภาพมาใช้ในงานคอมพิวเตอร์จำนวนมากในงาน คอมพิวเตอร์กราฟฟิก รวมทั้งกราฟฟิกในอินเทอร์เน็ต ทำให้เกิดปัญหาในการเก็บรูปภาพซึ่งจำเป็นต้องใช้พื้นที่ในหน่วยความจำมาก เป็นผลให้การเคลื่อนย้ายภาพต้องใช้เวลาาน การบีบอัดรูปภาพ (Image Compression) จึงมีประโยชน์ในการแก้ปัญหาเหล่านี้ได้ ดังนั้นจึงมีผู้ สนใจทำงานวิจัยทางด้านนี้มาก วิธีการบีบอัดรูปภาพจึงมีการพัฒนาอย่างต่อเนื่อง งานวิจัยนี้ได้เสนอแนวทางหนึ่งในการทำการบีบอัดข้อมูลภาพสี โดยการนำการแปลงเวฟเลตมาใช้ในการบีบอัดข้อมูลรูปภาพ โดยหลักการที่ว่า สายตาของคนไม่สามารถสังเกตเห็นภาพส่วนที่มีความถี่สูงในรูปภาพได้ เราจึงสามารถตัดข้อมูลส่วนนั้นออกไปได้โดยที่คุณภาพของภาพยังเป็นที่ยอมรับได้ การแปลงเวฟเลตมีคุณสมบัติในการแปลงข้อมูลไปเป็นข้อมูลอีกรูปแบบหนึ่ง ซึ่งข้อมูลที่มีความถี่ต่างกันจะถูกแยกออกมาเป็นระดับ ทำให้เราสามารถเลือกเก็บข้อมูลเฉพาะส่วนที่มีความสำคัญต่อการมองเห็นของคน ขั้นตอนในการบีบย่อภาพในงานวิจัยนี้ประกอบด้วย ขั้นตอนใหญ่ ๆ คือ การแปลงข้อมูลโดยการแปลงเวฟเลต การควอนไทซ์เวกเตอร์ (vector quantization) โดยใช้เทคนิคของ แอลบีจี (LBG algorithm) และ การเข้ารหัส โดยใช้เทคนิคการเข้ารหัสเลขคณิต (Arithmetic coding)

ผลของงานวิจัยนี้ ได้ทำการทดลองบีบอัดข้อมูลรูป Lena ซึ่งเป็นภาพสีขนาด 256x256 อัตราส่วนการบีบอัด (compression ratio) 50-60 ให้ PSNR (peak signal to noise ratio) 34-35 สำหรับอัตราส่วนการบีบอัด 80-100 ให้ PSNR 30-31 ส่วนผลการบีบอัดข้อมูลรูป Peppers อัตราส่วนการบีบอัด 50-60 ให้ PSNR 34 สำหรับอัตราส่วนการบีบอัด 80-100 ให้ PSNR 30

สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

ภาควิชาคณิตศาสตร์.....
สาขาวิชาวิทยาการคอมพิวเตอร์.....
ปีการศึกษา ๒๕๖๒

ลายมือชื่อนิสิต ขนิษฐา โพลีทรีวิญญู
ลายมือชื่ออาจารย์ที่ปรึกษา ร.ศ. สุชาดา ศิริพันธุ์
ลายมือชื่ออาจารย์ที่ปรึกษาร่วม -

4072220723 : MAJOR COMPUTATIONAL SCIENCE

KEY WORD: WAVELET / DAUBECHIES WAVELET / VECTOR QUANTIZATION /
LBG VECTOR QUANTIZATION / ARITHMETIC CODING.

KANIDTA POSITWINYU: COLOR IMAGE DATA COMPRESSION USING
DAUBCHIES WAVELET TRANSFORM. THESIS ADVISOR: ASSOC. PROF.
SUCHADA SIRIPANT 64 pp. ISBN 974-333-809-8

Image processing requires an enormous storage, processing, and communication. To reduce this amount of data, image compression is necessary. We applied wavelet transform, which is an up coming technique and very powerful tools for signal processing, to color image compression. In this thesis, we use Daubechies 20 wavelet transform to the image compression procedure. There are five steps in the procedure. First, we extract color image into RGB components. Second, we transform RGB components to wavelet coefficients by using wavelet transform and pyramid algorithm. Third, we cut off some coefficients and scaled the results to eliminate less important data. Forth, we quantize wavelet coefficients by LBG vector quantization according to the significant of wavelet coefficients. Finally, we apply arithmetic coding, which is a lossless coding process, to wavelet coefficients.

We test our algorithm with the images of Lena and Peppers. Lena is compressed with compression ratio 50-60 yielding PSNR around 34-35 and compression ratio 80-100 yielding PSNR around 30-31. Peppers is compressed with compression ratio 55-58 yielding PSNR around 34 and compression ratio 80-100 yielding PSNR around 30. Comparing with JPEG, this technique gives high PSNR than JPEG, at high compression ratio or compression ratio over 90, but less PSNR at low compression ratio or compression ratio below 85.

ภาควิชา ลายมือชื่อนิติศ
สาขาวิชา ลายมือชื่ออาจารย์ที่ปรึกษา
ปีการศึกษา ลายมือชื่ออาจารย์ที่ปรึกษาร่วม

ACKNOWLEDGMENTS

The author wished to express her sincere appreciation and gratitude to her advisor, Assoc. Prof. Suchada Siripant for her valuable advise, guidance and encouragement given through out the course of the investigation.

The author also wishes to express her special thanks to the thesis committee, Prof. Dr. Chidchanok Lursinsap and Assist. Prof. Dr. Peraphon Sophatsathit for their valuable advise, reading and criticizing the manuscript.

Finally, she would like to thank her family for their love and encouragement throughout my life and Mr. Paween Mahasuweerachai for his assistance in typing some parts of this thesis.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

TABLE OF CONTENTS

	PAGE
Thai Abstract	iv
English Abstract	v
Acknowledgments	vi
Table of Contents	vii
List of Tables	x
List of Figures	xi
CHAPTER	
I INTRODUCTION	1
1.1 Problem Identification	1
1.2 Objectives and Usefulness of This Work	2
1.3 Scope of Work	2
II LITERATURE REVIEW	3
2.1 Review of Literatures Related of Color Image Compression	4
2.2 Review of Literature Related of Wavelet Transform and Image Compression.....	4
2.3 Review of Literatures Related of Vector Quantization	5

TABLE OF CONTENTS (CONT.)

III	WAVELET TRANSFORM	7
	3.1 Introduction to Wavelet	7
	3.2 Continuous Wavelet Transform	11
	3.3 Discrete Wavelet Transform	12
	3.4 A Discrete Wavelet and Multiresolution Analysis	13
	3.4.1 The Scaling Function	14
	3.4.2 The Wavelet Functions	17
	3.5 Filter Banks and The Discrete Wavelet Transform	19
	3.5.1 Down-Sampling	20
	3.5.2 Up-Sampling	21
	3.6 Daubechies Wavelet Filter Coefficients	22
	3.7 Discrete Wavelet Transform and Pyramid Algorithm	23
IV	VECTOR QUANTIZATION	25
	4.1 Definition of Vector Quantization	25
	4.2 Quantization Process	26
	4.3 Codebook Generation	27
	4.4 Linde–Buzo–Gray (LBG) Algorithm	28

TABLE OF CONTENTS (CONT.)

4.5	Codebook Initialization	30
4.5.1	Random Codes	30
4.5.2	Splitting	31
4.5.3	Pairwise Nearest Neighbor (PNN) Clustering	31
4.6	The Empty Cell Problem	31
V	ARITHMETIC CODING	32
VI	COMPRESSION PROCEDURE DESIGN AND RESULTS	38
6.1	Procedure of Image Compression	40
6.2	Results and Discussion	48
VII	CONCLUSION	60
	References	61
	Curriculum Vitae	64

สถาบันวิทยบริการ
 จุฬาลงกรณ์มหาวิทยาลัย

LIST OF TABLES

	PAGE
Table 5.1 Probability of occurrence of alphabet symbols	34
Table 5.2 Assigning subinterval ranges	35
Table 6.1 Show percent of reduced wavelet coefficient and PSNR when some level and shape are cut off	44
Table 6.2 The Result PSNR and Compression Ratio of Lena and Peppers When Cut Off Some of Wavelet Coefficients	46
Table 6.3 Comparison PSNR and Cr of Lena when use different scaling parameter	48
Table 6.4 Comparison PSNR and Cr of Peppers when scaling parameter	49
Table 6.5 Comparison PSNR and Cr of Lena when change level of codebook and cut off shape.....	50
Table 6.6 Comparison PSNR and Cr of Peppers when change level of codebook and cut off shape.....	51

LIST OF FIGURES

	PAGE
Figure 3.1 The Window Function	7
Figure 3.2 Windowed Fourier Analysis versus Wavelets	8
Figure 3.3 Wavelet Functions	9
Figure 3.4 A Wave and a Wavelet	10
Figure 3.5 Nested Vector Spaces Spanned by the Scaling Function	16
Figure 3.6 Scaling Function and Wavelet Vector Spaces	18
Figure 3.7 Two-Stage Two-Band Analysis Tree	21
Figure 3.8 Two-Stage Two-Band Synthesis Tree	22
Figure 3.9 Daubechies 4's Wavelet Filter Matrix	22
Figure 3.10 Wavelet Decomposition Using Pyramid Algorithm	24
Figure 4.1 The Vector Quantization Procedure	27
Figure 4.2 The Flow Chart of LBG Algorithm	30
Figure 5.1 Show the Procedure of Creating Subinterval	36
Figure 6.1 Compression Procedure	39
Figure 6.2 2-Dimension Wavelet Transform	41
Figure 6.3 Pyramidal Algorithm	42
Figure 6.4 Wavelet coefficient in final step	43

LIST OF FIGURES (CONT.)

	PAGE
Figure 6.5 Show Graph of PSNR and Cr of Lena Compressed by Wavelet versus JPEG	50
Figure 6.6 Show Graph of PSNR and Cr of Lena Compressed by Wavelet versus JPEG	51
Figure 6.7 Results form Table 4.2 : cut off some shape of level of Wavelet Coefficient.....	53
Figure 6.8 Compare Lena compressed by Wavelet Versus JPEG at Cr109	54
Figure 6.9 Compare Lena compressed by Wavelet Versus JPEG at Cr96	55
Figure 6.10 Compare Lena compressed by Wavelet Versus JPEG at Cr71	56
Figure 6.11 Compare Peppers compressed by Wavelet Versus JPEG at Cr100	57
Figure 6.12 Compare Peppers compressed by Wavelet Versus JPEG at Cr80	58
Figure 6.13 Compare Peppers compressed by Wavelet Versus JPEG at Cr73	59

CHAPTER I



INTRODUCTION

1.1 Problem Identification

In the technology of computer development, transmission and storage of information is essential. The problem from a data storage and transmit perspective is growing, used of image in computer applications is important but it is not economical. A screen of text, consisting of 80 columns by 25 rows, can contain a maximum of 2000 characters. In comparison, consider a VGA 640 by 480 pixel 16-color image displayed on a personal computer screen. The image would require 640x480 bits of storage without considering the fact that four bits are required to represent the color of each pixel. Thus, the image would require 153,600 bytes of storage, or approximately 77 times the storage required to represent a full screen of text [1]. Image compression can solve these problems. Many images compression have been presented by researchers.

In this thesis, we present one technique for 24-bits color image compression. We apply wavelet transform to image compression. The concept that wavelet transform can decompose data in a set of subimages with different frequency bands, we can cut off some higher frequency which is less visible to the eye. The wavelet coefficients which

are very small or close to zero will be ignored, and the less amount will be kept to encode the information[2].

1.2 Objectives and Usefulness of This Work

The purpose of this thesis is to study wavelet transform and color image compression and to apply wavelet transform to color image compression.

The usefulness of this work is to obtain a technique for color image compression using wavelet transform.

1.3 Scope of Work

In this thesis, we developed a procedure of color image compression using wavelet transform, vector quantization and arithmetic coding.

- Analyzing the process consist of significant of wavelet coefficients for reconstruction in each shape and each level.
- Analyzing the vector quantization that appropriate for wavelet coefficients.
- Applying arithmetic coding to wavelet coefficients to be a binary file.

This thesis is organized into seven chapters. Chapter 2 reviews the literatures. Chapter 3-5 are about theoretical background. We discusse about the wavelet transform in Chapter 3, the vector quantization in Chapter 4 and arithmetic coding in Chapter 5. Chapter 6 provides the design of out color image compression procedure, experimental results and discussion. Chapter 7 is conclusion.

CHAPTER II

LITERATURE REVIEW

Image compression addresses the problem of reducing the amount of data required to represent a digital image. Image compression techniques are divided into two main techniques: transforms and non-transforms. The non-transform techniques such as PCM (Pulse Code Modulation) and DPCM (Differential Pulse Code Modulation), based on the signal model, compress data without transforming it. The transform techniques such as DCT (Discrete Cosine Transform), FFT (Fast Fourier Transform), and wavelet transform compress data by transforming the data to coefficients. This is the same principle employed by many efficient transform techniques which represent data by very small coefficients or close to zero. However, the transform process cannot reduce the amount of the data to be compressed. One approach to reduce the number of bits required for representing image is to quantize the coefficients by means of vector quantization.

In many applications such as computer graphics and graphic on internet, they are necessary to deal with color images. Typically, a color image is represented by three color planes corresponding to red, green, and blue. Compression of a color image can be done by encoding each plane independently using the above transformation and

vector quantization techniques. We shall review the underlying principles relating to wavelet transform and color image compression in the sections below.

2.1 Review of Literatures Related of Color Image Compression

Lori A. Overturf, Mary L. Comer, and Edward J. Delp [7] presented an algorithm that utilizes mathematical morphology for pyramidal coding of color images. This method is lossy color image compression by using block truncation coding at the pyramid levels to attain reduced data rates. The pyramid approach is attractive due to low computational complexity, simple parallel implementation, and the ability to produce acceptable color images at moderate data rates. The structure of this morphological pyramid allows quantization errors to pass from one level of the pyramid to the next, similar to the structure used by Ho and Gersho [8]. This research compressed color image by transformed RGB components [9] to the YIQ components [9] used in NTSC television transmission [10], which Y component is the luminance image, and the I and Q components are the inphase and quadrature chrominance images.

2.2 Review of Literature Related of Wavelet Transform and Image Compression

Mar Antonini, Michel Barlaud, Pierre Mathieu and Ingrid Daubechies [3] proposed an image compression scheme by using biorthogonal wavelet transform and vector quantization using a multiresolution codebook. To encode the wavelet coefficients, the author proposed a noise shaping bit allocation procedure.

Furthermore, this research presented a progressive transmission scheme, which showed that the wavelet transform is particularly adapted to progressive transmission.

Amir Arirbuch, Danny Lazer and Moshe Israeli [4] presented a scheme for image compression for black and white images based on the wavelet transform. After applying wavelet transform, the wavelet coefficients are quantized by vector quantization (VQ) using Linde-Buzo-Gray (LBG) algorithm [5,6]. Otherwise, the author represented an error correction method, which approximates the reconstructed coefficients quantization error. By this method, distortion is minimized for a given compression rate at low computational cost. This research achieves a compression ratio of 60-65 and PSNR of 30-33 for 512x512 black-and-white images.

2.3 Review of Literatures Related of Vector Quantization

Yoseph Linde, Andrus Buzo and Robert M. Gray [5] presented the LBG vector quantization algorithm based either on a known probabilistic model on a long training sequence of data. This algorithm for vector quantization is the most popular which is used for speech, signal and image compression. This algorithm is also used in this thesis.

In this thesis, we make use of color image compression strategy proposed by Overturf *et al.* which extracts color image to YIQ, consisting of three color components and compresses each color independently. On the other hand, the algorithm in this thesis extracts color image to RGB components. Overturf *et al.*'s

method is a non-transform compression method, but our approach uses transform compression method. The transformation used in this thesis is wavelet transform introduced by Antonini *et al.* and Arirbuch *el at.*, which apply wavelet transform on black-and-white image. Our approach differs slightly in that we apply wavelet transform on color image. We further reduce data used to represent the image by applying vector quantization in our algorithm using LBG vector quantization algorithm presented by Linde *et al.*



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย

CHAPTER III

WAVELET TRANSFORM

3.1 Introduction to Wavelet

A signal is usually defined as an oscillation function of time and space. Fourier analysis, a signal analysis tool, expanded the signal in terms of sinusoids which is valuable for periodic, time-invariant, or stationary signal. But in the transient, non-stationary, or time-varying signal, the signal may change suddenly in time that Fourier analysis is very computationally hard to detect it.

In 1940, Gabor developed windowed Fourier transform [11], which is known as the sliding-window Fourier transform which shown in Figure 3.1. The idea is to study the frequencies of a signal segment by segment. The size of the segment is defined by window, which is fixed size.

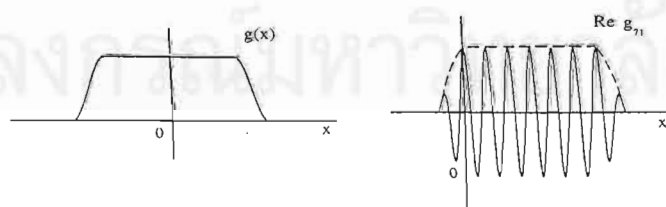


Figure 3.1 The Window Function

Fourier transform compares a signal in infinite sines and cosines of different frequency to see how much of each frequency the signal contains. Windowed Fourier transform compares a segment of the signal to bits of oscillating curves, first of one frequency, then of another, and so on. However, this method has some drawbacks. A small window locates the high signal frequencies, but misses the low ones since lower signal frequencies do not fit into the smaller window. In contrast, the wider window shows more of the low signal frequencies, but misses the higher ones. To solve this problem, wavelet transform is introduced. Wavelet is a function that is localized in both time and frequency domains. Wavelets can present a large class of signal with simple elements. Wavelet transform provides multiresolution analysis [12] with dilated windows. The high frequency analysis is done using narrow windows and the low frequency analysis is done using wide windows. Figure 3.2 shows windowed Fourier analysis versus wavelets, which above is windowed Fourier analysis and below is wavelets.

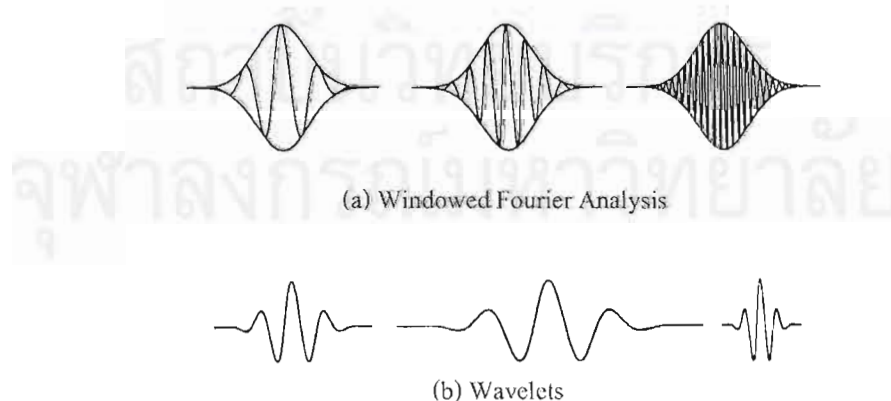


Figure 3.2 Windowed Fourier Analysis versus Wavelets

According to time-frequency localization of wavelet, wavelet transform is widespread in many works, such as denoising, signal processing, image processing, communication system, radar, etc.

The basis function of the wavelet transform is generated from a basic wavelet function $\psi(x)$ together with translations (b) and dilations (a). The scaled and translated wavelet can be written as :

$$\psi_{a,b}(x) = \frac{1}{\sqrt{a}} \psi\left(\frac{x-b}{a}\right) \quad (3.1)$$

where $b \in \mathbb{R}^+, a \in \mathbb{R}$. $f(x)$ is translated by $f(x-b)$. In the other words, $f(x)$ is translated by b units. $f(x)$ is dilated by a scaling factor (a) that stretches or compresses the wavelet. Note that the scaled wavelets include an energy normalization term, $\frac{1}{\sqrt{a}}$, that keep the energy of the scaled wavelet the same as the energy in the original mother wavelet. Mother wavelet is wavelet function which is not translated or dilated. Figure 3.3 shown wavelet functions which difference scaling factor (a) and translation factor (b).

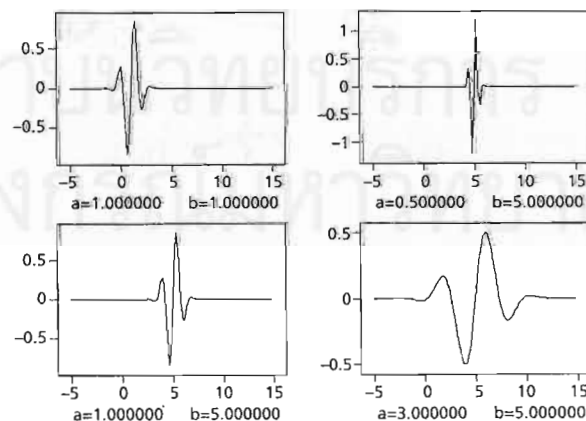
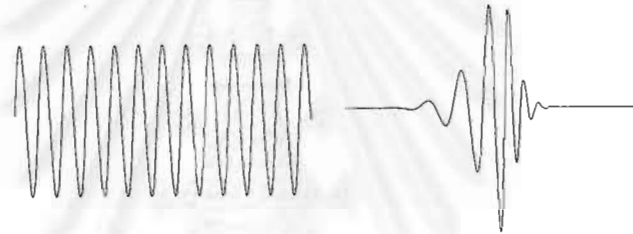


Figure 3.3 Wavelet Functions

The basis function of wavelet transform must satisfy these properties.

1. The original signal can be reconstructed by the inverse wavelet transform.
2. The wavelet must oscillate to have its mean value equal to zero.
3. The wavelet coefficients decrease rapidly with a decreasing of the scale in scaling function.
4. The wavelets have exponential decay so that their first low-order moments are equal to zero. Like their names, wavelets are like wave that oscillate and vanish.



(a) A Sine wave (b) Daubechies' wavelet ψ_{D20}

Figure 3.4 A Wave and a Wavelet

The signal function $f(x)$ can be described by a linear combination of wavelet function $\psi_{j,k}(x)$ by

$$f(x) = \sum_{j=0}^a \sum_{k=0}^b c_{jk} \psi_{jk}(x) \quad (3.2)$$

which j and k are integer indices and c_{jk} are wavelet coefficients.

3.2 Continuous Wavelet Transform

The continuous wavelet transform decomposes a signal $f(x)$ into a set of basis functions $W_f(a, b)$, which is called wavelets:

$$W_f(a, b) = \int f(x)\psi_{a,b}(x)d(x) \quad (3.3)$$

The wavelets are generated from a mother wavelet $\psi(x)$ by scaling and translation:

$$\psi_{a,b} = \frac{1}{\sqrt{a}}\psi\left(\frac{x-b}{a}\right) \quad (3.4)$$

which $a \in \mathbb{R}^+, b \in \mathbb{R}$

The constant $\frac{1}{\sqrt{a}}$ is for energy normalization. The wavelets are normalized as

$$\int |\psi_{a,b}(x)|^2 = \int |\psi(x)|^2 dx = 1 \quad (3.5)$$

So that all the wavelets which is scaled by different scaling function have the same energy.

The signal can be reconstructed by

$$f(x) = \iint W_f(a,b)\psi_{a,b}(x)d(x) \quad (3.6)$$

This transformation is useful when we want to recognize or extract features. Scaling or translating of $f(x)$ leads to shift of the $W_f(a,b)$, so that whole analysis can be made to be scaled and translated invariant a desirable property in some applications.

3.3 Discrete Wavelet Transform (DWT)

Continuous wavelet transform output is the square of a time-bandwidth product of a signal. For signal processing, we attempt to represent the signal efficiently with fewer information. The DWT can reduce the time bandwidth product of the wavelet transform output. In addition to compute the wavelet transform in digital computer, the input and filter functions are all discrete. Like continuous wavelet transform, the DWT decomposes the signal into a set of basis function. But the scaling functions and translation functions in DWT are discrete.

Let $L^2(\mathfrak{R})$ denotes the space of square integrable functions. The DWT can represent any signal in $L^2(\mathfrak{R})$ by

$$f(x) = \sum_{j=0}^a \sum_{k=0}^b c_{jk} \psi_{jk}(x) \quad (3.7)$$

where $\psi_{jk}(x)$ is wavelet basis and c_{jk} is wavelet coefficient. If the wavelet basis is orthogonal. This means that

$$\langle \psi_k(x), \psi_l(x) \rangle = \int \psi_k(x) \psi_l(x) dx = 0, \quad k \neq l \quad (3.8)$$

where ψ_k and ψ_l are wavelet base which called expansion set, then c_{jk} . wavelet coefficients can be calculated by the inner product

$$c_{jk} = \langle f(x), \psi_{jk}(x) \rangle = \int f(x) \psi_{jk}(x) dx \quad (3.9)$$

The first goal of most expansions of a signal is to have the coefficients of the expansion c_{jk} which give more useful information about the signal than directly obvious from the signal itself. The second goal is to have most of the coefficients to be zero or very small which can encode with low bit per pixel.

The discrete wavelet transform maps a function of a continuous signal into a sequence coefficients which are zero or very small.

3.4 A Discrete Wavelet and Multiresolution Analysis

A wavelet can describe a signal by decomposing the signal into finer and finer detail by using scaling function and wavelet function which derived from concept of multiresolution. To decompose the signal, a wavelet spans the set of signal into

various subspaces by the various scales of the scaling function, and describes the different between the spaces spanned by different set of wavelet functions. In order to understand wavelet, we will use multiresolution concept to define the scaling function and wavelet function.

3.4.1 The Scaling Function

A set of scaling function is defined in term of integer translates of the basic scaling function by

$$\varphi_k(x) = \varphi(x - k), \quad k \in \mathbb{Z}, \quad \varphi \in L^2(\mathbb{R}) \quad (3.10)$$

which \mathbb{Z} is integer, \mathbb{R} is real number and $L^2(\mathbb{R})$ is a space of integral of square of a modulus of a function. The subspace of $L^2(\mathbb{R})$ spanned by this function is defined as

$$\nu_o = \overline{\text{span}_k \{ \varphi_k(x) \}} \quad (3.11)$$

for all integers k . The over-bar denotes closure. This means that

$$f(x) = \sum_k c_k \varphi_k(x) \quad (3.12)$$

for any $f(x) \in \nu_o$

The size of the subspace spanned can increase by changing the time scale of the scaling functions. A family of functions is generated from the basic scaling function by scaling and translation by

$$\varphi_{j,k}(x) = 2^{j/2} \varphi(2^j x - k) \quad (3.13)$$

whose span over k is

$$v_j = \overline{\text{span}_k \{ \varphi_k(2^j x) \}} = \overline{\text{span}_k \{ 2^{j/2} \varphi_{j,k}(x) \}} \quad (3.14)$$

for all integer $k \in \mathbb{Z}$. This means that if $f(x) \in v_j$, then it can be expressed as

$$f(x) = \sum_k a_k \varphi_k(2^j x + k) \quad (3.15)$$

For $j > 0$, the span $\varphi_{j,k}(x)$ can be larger since $\varphi_{j,k}(x)$ is narrower and is translated in smaller steps. It can represent a finer detail of the information.

For $j < 0$, $\varphi_{j,k}(x)$ is wider and $\varphi_{j,k}(x)$ is translated in larger step. The wide scaling function can present only coarse information, and the span is smaller.

We formulate the basic requirement of multiresolution analysis by requiring a nesting of the spanned spaces as

$$\dots \subset \nu_{-2} \subset \nu_{-1} \subset \nu_0 \subset \nu_1 \subset \nu_2 \subset \dots \subset L^2 \quad (3.16)$$

with $\nu_{-\infty} = \{0\}, \nu_{\infty} = L^2$

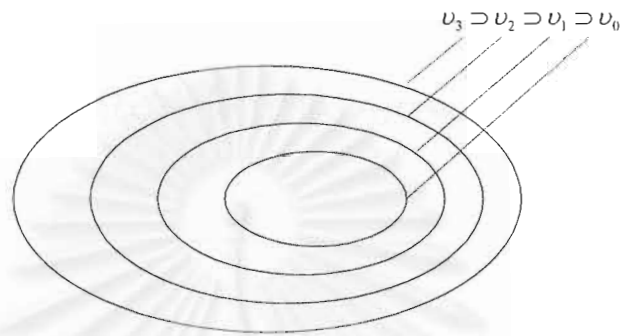


Figure 3.5 Nested Vector Spaces Spanned by the Scaling Function [13]

The space that contains high resolution signals will also contain those of lower resolution.

The $\varphi(x)$ can be expressed in terms of a weighted sum of shifted $\varphi(2x)$ as

$$\varphi(x) = \sum_n h(n) \sqrt{2} \varphi(2x - n), \quad n \in \mathbb{Z} \quad (3.17)$$

where $h(n)$ are scaling function coefficients of filters and the $\sqrt{2}$ maintains the norm of the scaling function with the scale of two.

3.4.2 The Wavelet Functions

The orthogonal complement of ν_j in ν_{j+1} is define as w_j . This means that the members of ν_j are orthogonal to all members of w_j . We require

$$\langle \varphi_{j,k}(x), \psi_{j,l}(x) \rangle = \int \varphi_{j,k}(x) \psi_{j,l}(x) dx = 0, \quad k \neq l \quad (3.18)$$

for all appropriate $j, k, l \in \mathbb{Z}$

We now define the wavelet spanned subspace w_0 such that

$$\nu_1 = \nu_0 \oplus w_0 \quad (3.19)$$

which \oplus is add subspace operation. In general this gives

$$L^2 = \nu_0 \oplus w_0 \oplus w_1 \oplus \dots \quad (3.20)$$

when ν_0 is the initial space spanned by the scaling function $\varphi(x-k)$. Figure 3.6 shows the nesting of the scaling function spaces ν_j for different scales j and how the wavelet spaces are the disjoint differences (except for the zero element) or the orthogonal complements.

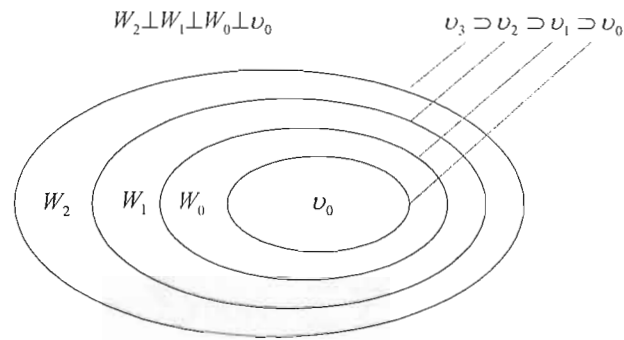


Figure 3.6 Scaling Function and Wavelet Vector Spaces [13]

Since these wavelets reside in the space spanned by the next narrower scaling function, $w_0 \subset v_1$, they can be represented by a weight sum of shifted scaling function $\varphi(2x)$ by

$$\psi(x) = \sum_n h_1(n) \sqrt{2} \varphi(2x - n), \quad n \in \mathbb{Z} \quad (3.21)$$

for some set of coefficients $h_1(n)$ which can view as a filter.

The function $\psi(x)$ is mother wavelet which can expand functions by

$$\psi_{j,k}(x) = 2^{j/2} \psi(2^j x - k) \quad (3.22)$$

where 2^j is the scaling of x (j is the \log_2 of the scale), $2^{-j}k$ is the translation in x , and $2^{j/2}$ maintains the (perhaps unity) L^2 norm of the wavelet at different scales.

A set of functions $\varphi_k(x)$ and $\psi_{j,k}$ that we have constructed can be span any function $f(x) \in L^2(\mathfrak{R})$ as

$$f(x) = \sum_{k=-\infty}^{\infty} c(k)\varphi_k + \sum_{j=0}^{\infty} \sum_{k=-\infty}^{\infty} d(j,k)\psi_{j,k}(t) \quad (3.23)$$

which $c(k)$ and $d(i,k)$ are wavelet coefficients. If wavelets and scaling functions are from orthogonal basis, the wavelet coefficients can be calculated by inner products as

$$c(k) = \langle g(x), \varphi_k(x) \rangle = \int g(x)\varphi_k(x) dx \quad (3.24)$$

and

$$d(j,k) = \langle g(x), \psi_{j,k}(x) \rangle = \int g(x)\psi_{j,k}(x)dx \quad (3.25)$$

3.5 Filter Banks and The Discrete Wavelet Transform

In digital processing, the filtering is to convolve a sequence of numbers or filter coefficients with the input signal. For an input sequence $x(n)$ and filter coefficient $h(k)$, the output sequence $y(n)$ is given by

$$y(n) = \sum_{k=0}^{N-1} h(k)x(n-k) \quad (3.26)$$

Wavelet filter is designed for separating signals. We can separate and convolve the signal by two basic operations called down-sampler and up-sampler.

3.5.1 Down-Sampling

The down-sampler takes a signal $x(n)$ as an input and produces an output of $y(n) = x(2n)$, which is used in forward wavelet transform. The wavelet transform coefficients at a lower scale level can be expanded in terms of higher scale as shown in these equations

$$c_j(k) = \sum_m h(m - 2k)c_{j+1}(m) \quad (3.27)$$

and

$$d_j(k) = \sum_m h_1(m - 2k)c_{j+1}(m) \quad (3.28)$$

These equations show that the scaling and wavelet coefficients at different levels of scale can be obtained by convolving the expansion coefficients at scale j by the time-reversed recursion coefficients $h(-n)$ and $h_1(-n)$ then down-sampling to give the expansion coefficients at the next level of $j-1$. In the other word, the scale $-j$ coefficients are filtered by two filters with coefficient $h(-n)$ (low pass filter) and $h_1(-n)$ (high pass filter). After down-sampling gives the next coarser scaling (d_j) and wavelet coefficient (c_j). This filtering and decimation can be repeated on the scaling coefficients to give the two-scale structure in Figure 3.7.

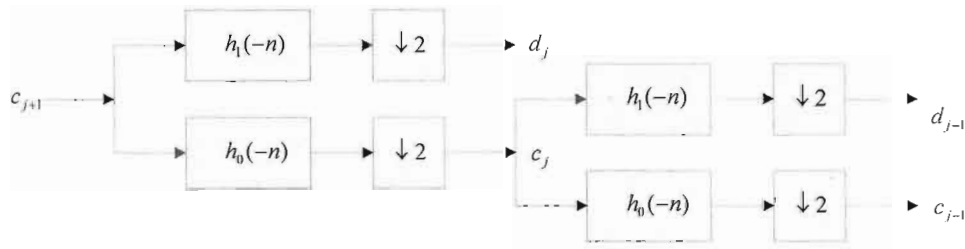


Figure 3.7 Two-Stage Two-Band Analysis Tree

3.5.2 Up-Sampling

The Up-Sampling means the input signal $x(n)$ is structured to twice its original length, $x(n) = y(2n)$, which is used in backward wavelet transform. To reconstruct a finer wavelet coefficient, the finer coefficient can be made from a combination of the a scaling function and wavelet coefficients at a coarse resolution as shown in this equation.

$$c_{j+1}(k) = \sum_m c_j(m)h(k-2m) + \sum_m d_j(m)h_1(k-2m) \quad (3.29)$$

This equation is evaluated by up-sampling the j scale coefficient sequence $c_j(k)$, which means double its length by inserting zeros between each term, then convolving it with the scaling coefficients $h(n)$. This combining process can be continued to any level by combing the appropriate scale wavelet coefficient. The resulting a two-scale tree is shown in Figure 3.8.

Here blank entries signify zeroes. Each row generates one component of the data. The data rows are convolved with the filter coefficients c_0, \dots, c_3 being a smoothing filter, called H, something like a moving average of four points. In the same way, even rows are convolved with filter coefficient $c_3, -c_2, c_1, -c_0$, being a non-smoothing filter called G. Actually the c 's are chosen so as to make G yield, insofar as possible, a zero response to a sufficiently smooth data vector.

The results in the output of H, decimated by half, accurately representing the data's "smooth" information, and out put of G, also decimated, is referred to as the data's "detail" information.

Requiring matrix in Figure 3.9 to be orthogonal, its inverse is its transpose matrix that is used in the reconstruction. The reconstruction original data vector of length N is performed by using its N/2 smooth or s-components and its N/2 detail or d-components.

3.7 Discrete Wavelet Transform and Pyramid Algorithm

The wavelet transform is made by calculated the filter banks being an on-going string of coefficients at each of the scales. If the input of the filter banks has a certain, the output at the next lower scale will be two sequences, one of scaling function coefficient $c_{j-1,k-1}$ and one of wavelet coefficients $d_{j-1,k-1}$, each, after down-sampling, being at half the rate of the input. At the next lower scale, the same process is done on

the scaling coefficients to give a total output of three strings, one at half rate and two at quarter rate. The diagram of Forward Wavelet Transform is shown as Figure 3.10.

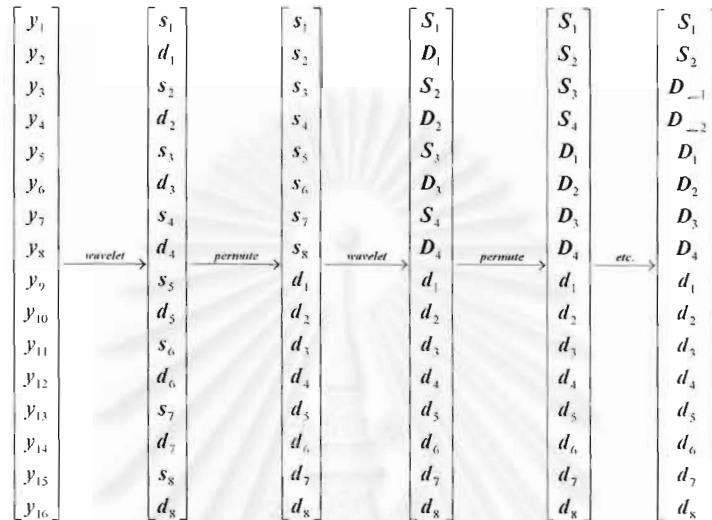


Figure 3.10 Wavelet Decomposition Using Pyramid Algorithm

From Figure 3.10, we have seen that the wavelet transform consists of applying a wavelet coefficient matrix like hierarchically, first to the original data vector of length N , then to the "smooth" vector of length $N/2$ and then to the "smooth-smooth" vector of length $N/4$, and so on until only a trivial number of "smooth-...-smooth" components (usually 2) remain. The output of the wavelet transform consists of these remaining components and all the "detail" components that were accumulated along the way.

We called the final values c_j , "Mother-Function Coefficients" or "Mother Wavelet".

CHAPTER IV

VECTOR QUANTIZATION

Vector quantization is a method of grouping inputs together and encoding them as a single vector. Vector quantization is used in lossy compression to reduce the transmission bit rate or the storage of image and speech compression.

4.1 Definition of Vector Quantization

A vector quantization is defined as a mapping Q of K - dimensional Euclidean space R^K into a finite subset Y of R^K . Thus,

$$Q: R^K \rightarrow Y \quad (4.1)$$

where $Y = (\hat{x}_j ; j = 1, 2, \dots , N)$ is a set of reproduction vectors, N is the number of vectors in Y . It can also be seen as a combination of two functions, an encoder and a decoder. The encoder views the input vector x and generates an index of the reproduction vector specified by $Q(x)$, and the decoder uses this index to generate the reproduction vector \hat{x} . If a distortion measure $d(x, \hat{x})$ which represents the cost

associated with reproduction vector x by \hat{x} is defined, then the best mapping Q is the minimization of $d(x, \hat{x})$.

The simple distortion measure used in this thesis is a square error distortion given by

$$d(x, \hat{x}) = \|x - \hat{x}\|^2 = \sum_{j=0}^{K-1} (x_j - \hat{x}_j)^2 \quad (4.2)$$

In generally, there are other distortion measures[6], but they are not a good practical computation and implementation.

4.2 Quantization Process

In vector quantization process, we generate $n = l \times m$ blocks or vectors to represent the source data, which are grouped into $n = l \times m$ blocks or vectors. For example, we group source data to be blocks or vectors which will be the input vectors of vector quantizer. Vector quantizer is composed of encoder and decoder. The encoder and decoder have a set of n -dimensional vector called codebook. The vector in a codebook is called code-vectors which are selected to be represent the input vectors from original image. Each code-vectors have index to indicate the address of that code-vectors. In the encoder, the vectors from source data is compared with the codevectors in codebook to find the closest vectors, and then the vectors are formed to reconstruct the data, which have loss some information.

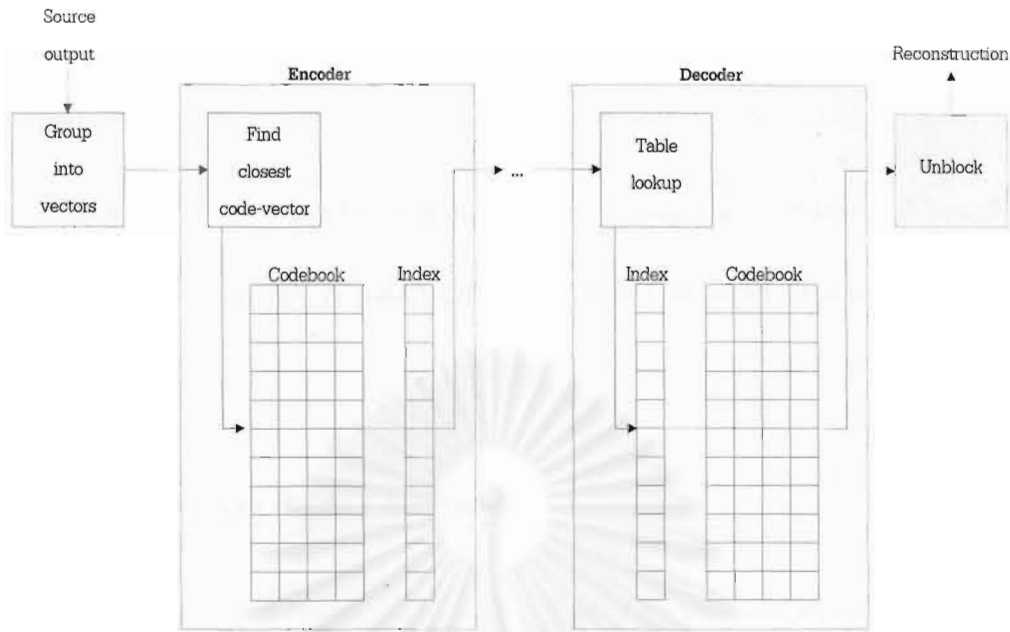


Figure 4.1 The Vector Quantization Procedure [16]

Then we turn to the important procedure in the design of vector quantizer, the generation of the codebook.

4.3 Codebook Generation

Codebook is usually generated by using a training set of images that are representative of the images to be encoded. To encode the image, the optimal codebook would be generated by using the image itself to be training set. Codebook generated by this procedure is called local codebook which usually results in good performance for moderate codebook sizes. This is because most of the images features

such as adges, lines, etc. that are representing image are adequately represented by code-vectors.

There are many ways to generate the vector quantizer codebook, although most of them are based on the popular approach known as Linde–Buzo–Gray (LBG) algorithm.

4.4 Linde–Buzo–Gray (LBG) Algorithm [5]

Let the expected distortion be approximated by the time averaged square error distortion given by the expression

$$D(x, q(x)) = \frac{1}{N} \sum_{j=0}^{N-1} d(x_j, \hat{x}_j) \quad (4.3)$$

The step in LBG algorithm are :

1. Let N = number of levels; distortion threshold $\varepsilon \geq 0$. Assume an initial N level reproduction alphabet \hat{A}_0 , and a training sequence $(x_j; j = 0, 1, 2, \dots, n-1)$, and m = number of iterations, set to zero.
2. Given $\hat{A}_m = (y_i; i = 1, 2, \dots, N)$, find the minimum distortion partition $P(\hat{A}_m) = (S_i; i = 1, 2, \dots, N)$ of the training sequence ; $x_j \in S_i$ if $d(x_j, y_i) \leq d(x_j, y_l)$, for all l . Compute the average distortion

$$D_m = D[(\hat{A}_m, P(\hat{A}_m))] = n^{-1} \sum_{j=0}^{n-1} \min_{y \in \hat{A}_m} d(x_j, y) \quad (4.4)$$

3. If $(D_{m-1} - D_m) / D_m \leq \varepsilon$, stop the iteration with \hat{A}_m as the final reproduction alphabet; otherwise continue.
4. Find the optimal reproduction alphabet $\hat{x}(P(\hat{A}_m)) = (\hat{x}(S_i); i = 1, 2, \dots, N)$ for $P(\hat{A}_m)$ where

$$\hat{x}(S_i) = \frac{1}{\|S_i\|} \sum_{j: x_j \in S_i} x_j \quad (4.5)$$

5. Set $\hat{A}_{m+1} = \hat{x}(S_i)$, increment m to $m+1$, and go to (2).

The flow chart of this iteration is shown in Figure 4.2. In the algorithm in Figure 4.2 an initial reproduction alphabet A was assumed in order to start the algorithm. There are a number of techniques to obtain the initial codebook as shown in the next topic.

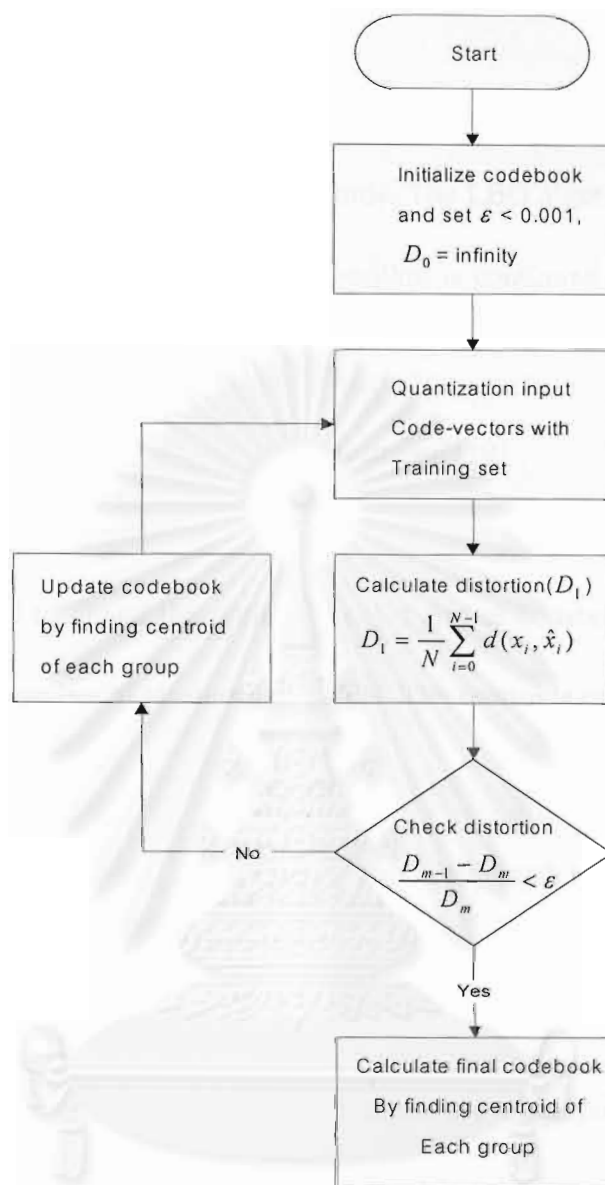


Figure 4.2 The Flow Chart of LBG Algorithm

4.5 Codebook Initialization [17]

4.5.1 Random Codes [18]

This approach is to use a random initial codebook where the first N vectors of the training sets are chosen as the initial codebook.

4.5.2 Splitting [5]

In this approach, we start with the centroid for the entire training set, and this single codeword is split to form two codewords. The LBG algorithm is applied to get an optimal codebook of size 2. Then the algorithm is continued until the codebook is split to size N .

4.5.3 Pairwise Nearest Neighbor (PNN) Clustering [19]

Starting with N clusters which contain the training vectors, the two closest vectors are merged to create the optimal $(n-1)$ cluster codebook. This algorithm is repeated until the number of clusters equal to N . The centroids of these cluster are then used as the initial codebook.

In this thesis, we used the PNN initialization approach to initial the codebook.

4.6 The Empty Cell Problem [16]

In generating codebook process, if there are no input training point gets assigned to the initial output point, this is a problem because we will end up with output point in codebook that never used. A approach that can avoid the empty cell problem is to remove an output point that has no inputs associated with it, and replace it with a point from the quantization region. This can be done by selecting a point at random from the region with the highest population of training vectors.

CHAPTER V

ARITHMETIC CODING

Arithmetic coding is especially useful when dealing with sources with small alphabets, such as binary sources, and alphabets with highly skewed probabilities. In applications where the alphabet size is large and the probability of the most frequently occurring symbol (P_{\max}) is generally quite small. However, in cases where the alphabet is small and the probability of occurrence of the different letters is skewed, the values of P_{\max} can be quite large and the Huffman code can become rather inefficient when compared to the entropy. One way to avoid this problem is to block more than one symbol together and generate code, which is the concept of arithmetic coding.

Arithmetic coding is operation by determining the probability or frequency of each symbol. Next, we list the symbols in a fixed order, usually by ascending or descending probability or by the order of the symbols in the symbol set. Any order can be used as long as it is the same order for both encoding and decoding. The sum of the individual probabilities is always unity. The next step is to subdivide unity into ranges, giving each symbol its own range and assigning the length of the range so that it equals the probability

of the symbol. Once this is accomplished, we are ready to construct a number, which represents a sequence of symbols. The process is performed the following steps:

1. Select the first symbol in the sequence and locate the ranges associated with the symbol.
2. Select the next symbol. Multiply the range associated with the previous symbol by the low and high ends of the range of the current symbols. Add the results to the low range of the prior symbol to obtain a new low range and new high range for the current symbol.
3. Continue step 2 until all of the symbols to be encoded are processed.

To illustrate the operation of arithmetic coding, let's first assume alphabet restricted to the set of eight symbols [d, e, f, g, n, r, u and space]. To simplify illustration of the operation of arithmetic coding, alphabet was also restricted to lowercase symbols. Since arithmetic coding uses the probability of occurrence of each symbol in the alphabet to construct appropriate coding intervals, the probability of occurrence to each symbol is assigned in the Table 5.1.

Once the probability of the occurrence of each symbol is known, the range of each symbol is assigned upon its probability. This range assignment subdivides the interval from zero to one $[0,1)$ into subintervals that correspond to the probabilities of each symbol.

The table 5.2 illustrates the assignment of subinterval ranges to each symbol in out alphabet. Symbols can be assigned to subintervals in any order, as long as the compressor and expander use the same assignment. However, subinterval range assignments are normally based upon the sequence of symbols in the alphabet. In examining Table 5.2, note that since the symbol probabilities sum to one, the subinterval ranges fill the interval between zero and one.

Symbol	Probability
D	0.1
E	0.3
F	0.1
G	0.05
N	0.1
R	0.2
U	0.15
Space	0.1

Table 5.1 Probability of occurrence of alphabet symbols

In examining the range intervals in Table 5.2, note that each symbol owns the entire range interval up to but not including the high number. Thus, d has the range 0.0 to 0.0999 ... and so on. As symbols that form a message are processed, each new symbol reduces the range of the floating-point number used to represent the message.

Symbol	Probability	Range
d	0.1	[0.0,0.1]
e	0.3	[0.1,0.4]
f	0.1	[0.4,0.5]
g	0.05	[0.5,0.55]
n	0.1	[0.55,0.65]
r	0.2	[0.65,0.85]
u	0.05	[0.85,0.90]
space	0.1	[0.90,1.0]

Table 5.2 Assigning subinterval ranges

To illustrate the operation of the encoder, assume the name 'fredunger', to include the space between the first and last name, is to be compressed. The first symbol, 'f', results in the encoder selecting the range [0.4,0.5] to allocate to the symbol. The next symbol, 'r', owns the range [0.65,0.85], which narrows the new range to 65 to 85% of the prior range. That is, the old range started at 0.4 and was 0.1 units long. Multiplying 0.65 by 0.1 and adding to 0.4 gives a new low range starting point of 0.465. Similarly, multiplying 0.85 by 0.1 and adding the result to 0.4 gives a new high range of 0.485. Thus, the new interval becomes [0.465,0.485].

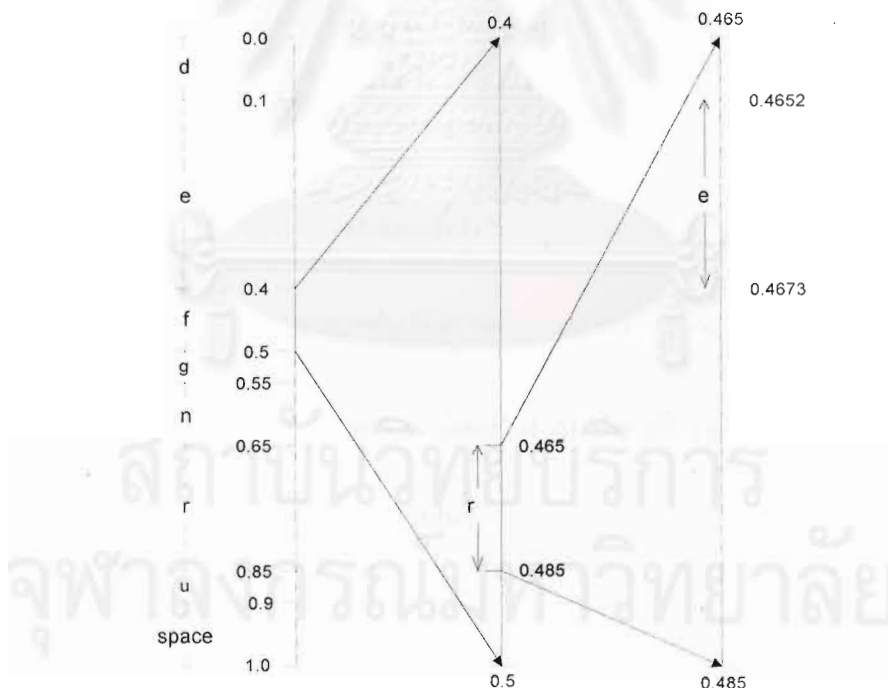


Figure 5.1 Show the Procedure of Creating Subinterval

The third symbol in the message is 'e', which has the interval [0.1,0.4]. Since the previously computed range commenced at 0.465 and was 0.02 units in length, we multiply the length of 0.02 by the starting interval of 0.1 and add the result to 0.465 to obtain the new low range starting point of 0.4652. Similarly, we multiply the prior range length of 0.02 units by 0.4 and add it to 0.465 to obtain a new high range terminating point of 0.473. Thus, encoding the third symbol results in the new interval [0.4652, 0.473]. This procedure is shown in Figure 5.1. By now the encoding pattern should be observable. That pattern can be summarized as follows:

start with the range [0,1] where low (0) becomes 0 and high (0) becomes 1

Then for $i > 1$

$$low(i) = low(i-1) + [high(i-1) - low(i-1)] * low(i)$$

$$high(i) = low(i) + [high(i-1) - low(i-1)] * high(i)$$

Continue iterat3 until all symbols in sequence are encoded.

where $low(i)$ represents the low interval of the i^{th} symbol encoded and $high(i)$ represents the high interval of the i^{th} symbol.

CHAPTER VI

COMPRESSION PROCEDURE DESIGN AND RESULTS

In this thesis, we proposed a procedure for color image compression by using wavelet transform. This compression is lossy image compression. The concept of this thesis is that wavelet transform can decompose data to be a set of subimages with different frequency bands, so that we can cutoff some higher frequency which less visible to the eye and reconstruct data with some loss. Further wavelet coefficients only indicate change, areas with no change or very small change give small or zero coefficients, which can be ignored, reducing the number of coefficients that have to be kept to encode the information. In addition, only wavelet transform cannot give high compression ratio. An important procedure to increase compression ratio, we added vector quantization and arithmetic coding in this compression algorithm. Our compression procedure is showed in Figure 6.1.

We used lena and peppers to be tested images. Our tested images are 256x256 pixel 24-bits color images in Tiff format.

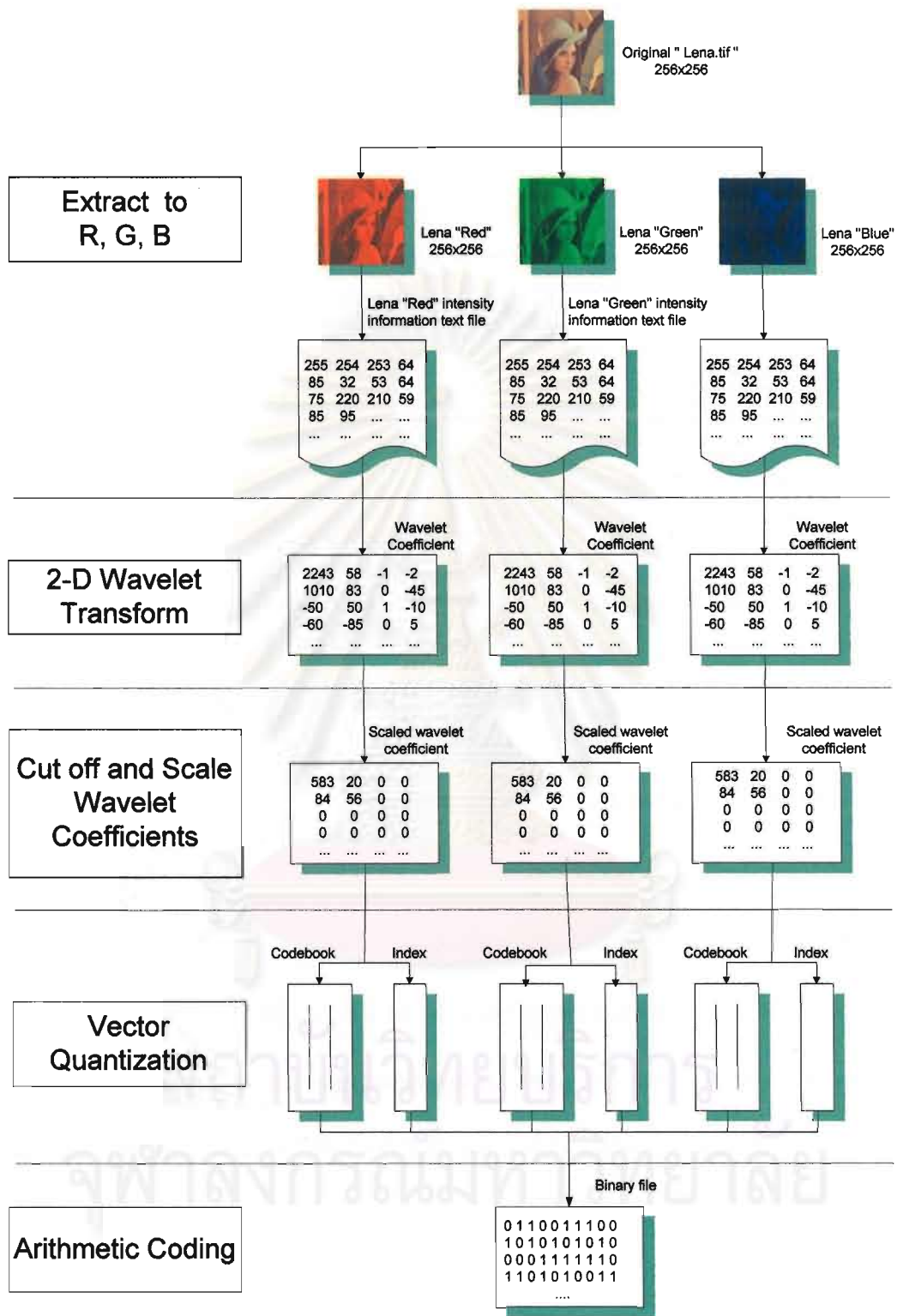


Figure 6.1 Compression Procedure

6.1 Procedure of Image Compression

Step 1 Extraction color image into RGB plane.

The original Tiff format image is a binary file. To get the data that presents the image, we have to extract the image into three text files: red, blue and green. Each file has the same format which represents integer numbers of the color intensity in each pixel. So, if the image have $n \times n$ pixels, the amount of the intensity in each file is $n \times n$ components. The intensity of each color has range between 0-255 like gray scale. We called the intensity of the color as data or signal, which will be compressed in other steps.

Step 2 Apply 2 dimension wavelet transform to the data.

The data in red, blue and green text files are transformed separately by a 2-dimension wavelet transform. In this thesis, we used Daubechies 20 wavelet filter which is discussed in Chapter 3. Using pyramid algorithm, the original data are transformed step by step which is showed in Figure 6.2 to be wavelet coefficients. First, the wavelet filter coefficients are applied to the data in horizontal direction. The outputs of transformation are wavelet coefficients with $n \times n$ components and composed of two type of data smooth and detail each has $(n \times n)/2$ numbers. Second, all of the wavelet coefficients from first step are passed the wavelet filter coefficients in vertical direction. The outputs are

separated into four parts. One part is the wavelet coefficients called smooth data, and the other three parts called shape 1, shape 2 and shape 3 are the details.

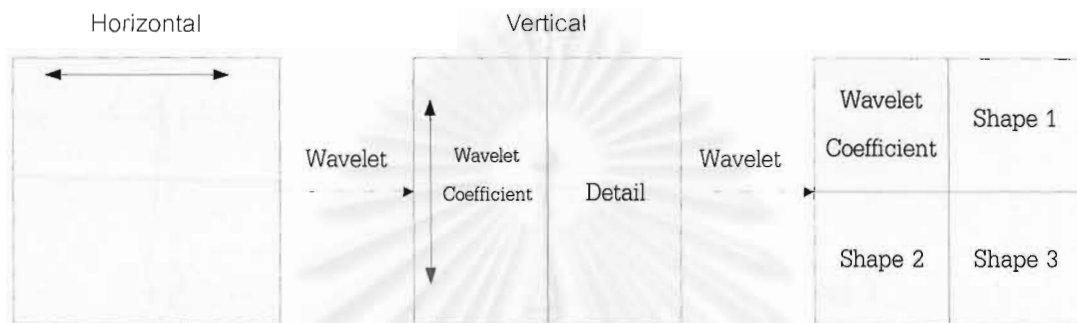


Figure 6.2 2-Dimension Wavelet Transform

The smooth data in first level are transformed again by wavelet transform. The smooth data in level 2 are separated into four parts which one of them is smooth data and the others are three shapes of the details. The process is going on until the last level has four wavelet coefficients. We called the wavelet coefficients in final level “mother wavelet” which is the most important in a reconstruction. This wave transform procedure is called pyramidal algorithm which is showed in Figure 6.3. All wavelet coefficients have eight levels and each level have three shapes which we called shape 1 (horizontal direction), shape 2 (vertical direction) and shape 3 (diagonal direction) as shown in Figure 6.4.

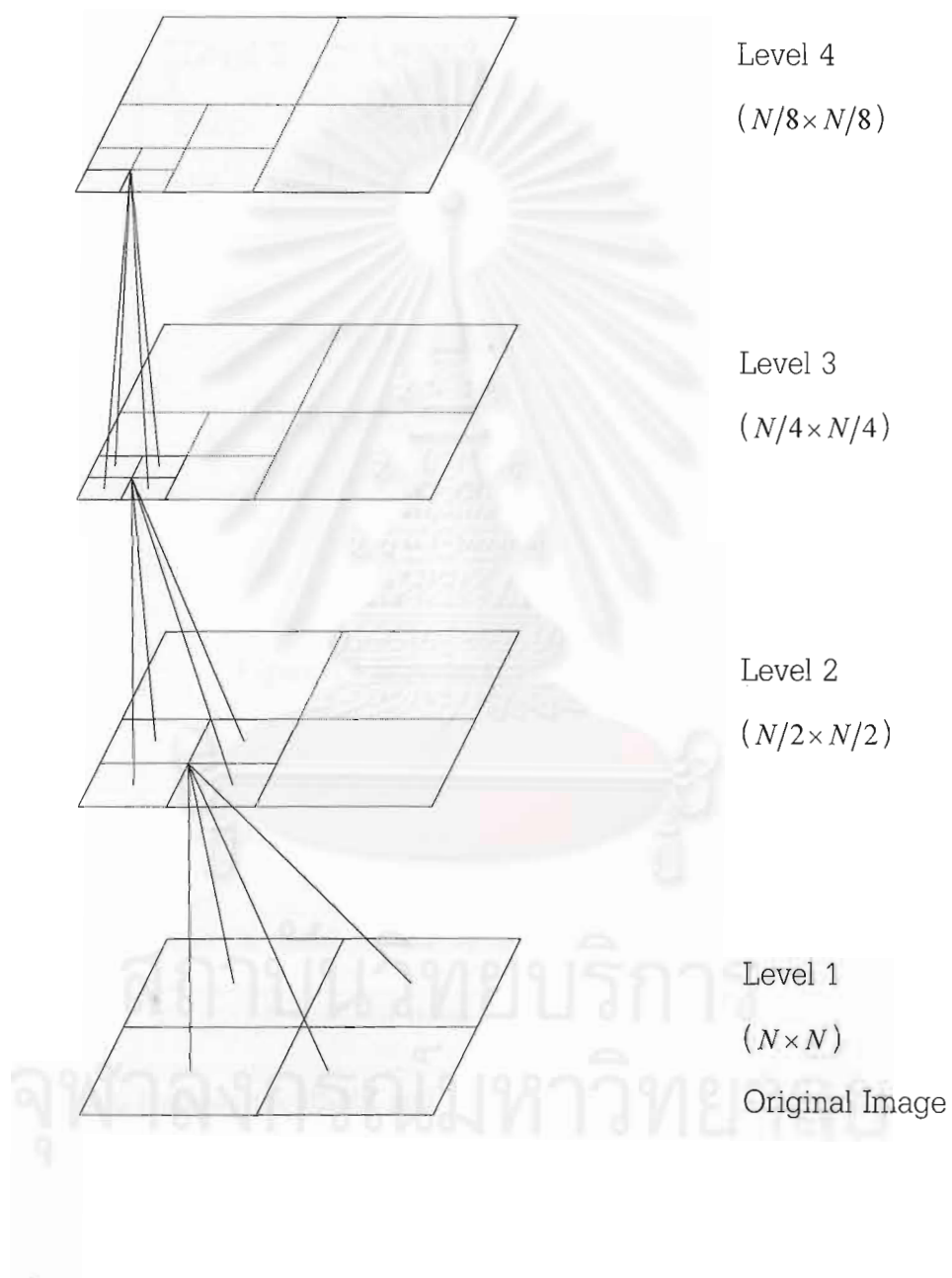


Figure 6.3 Pyramidal Algorithm

		Level 3	Level 2	Level 1
		Shape 1		
Level 3	Level 3	Shape 1		
Shape 2	Shape 3			
Level 2	Level 2	Level 2	Shape 1	
Shape 2	Shape 3	Shape 3		
Level 1		Level 1		
Shape 2		Shape 3		

Figure 6.4 Wavelet Coefficients in Final Step

Analyze the significant of difference shape and level in wavelet coefficients

Some shapes in some levels of wavelet coefficients have less important for reconstruction. We cut off some of them and set them to be zero to test the significant of them. Table 6.1 shows the PSNR and percent of the zero in wavelet coefficients of Lena and Peppers images when some shapes in some levels are cut off. From the Table 6.1, we find that the shape 3 level 1 is less important because when we cut all of coefficients in this level the PSNR is 41.17 for Lena and 44.56 for Peppers which much more than the

PSNR when cut other levels. From the Table 6.1, we also find the importance of the shape, which is arranged by shape 1, shape 2, and shape 3 respectively.

Cut off level	% Reduced coefficient	PSNR	
		Lena	Peppers
Level 1 shape 1, 2, 3	75%	31.55	31.38
Level 2 shape 1, 2, 3	18.75%	28.03	26.97
Level 3 shape 1, 2, 3	4.69%	25.44	24.13
Level 4 shape 1, 2, 3	1.17%	22.91	21.01
Level 5 shape 1, 2, 3	0.29%	21.22	20.06
Level 1 shape 1	25%	33.44	34.44
Level 1 shape 2	25%	37.61	34.74
Level 1 shape 3	25%	41.17	44.56
Level 2 shape 1	6.25%	30.15	30.10
Level 2 shape 2	6.25%	34.73	30.61
Level 2 shape 3	6.25%	35.63	37.74
Level 3 shape 1	1.56%	27.12	26.87
Level 3 shape 2	1.56%	33.27	28.75
Level 3 shape 3	1.56%	33.47	33.24

Table 6.1 Show percent of reduced wavelet coefficients and PSNR when some levels and shape are cut off.

The results from Table 6.1 show that when applied the wavelet transform to the images it gives the wavelet coefficients in same properties:

1. Each shape have different significant, the most important shape is shape 1.

For shape 2 and shape 3 the important were decreased respectively.

2. Each level had different significant, the most important level is level 8 which is called mother wavelet, and in the lower level the importance had decrease respectively.

The wavelet coefficient in level 1 is less important than other level. If we cut all wavelet coefficients in level 1 of some images such as Lena and Peppers, the wavelet coefficients decrease 75% when the PSNR is around 31 as shown in Table 6.1, that still gives the good quality of image as shown in Figure 6.7.

Step 3 Cut off and scale wavelet coefficients

This step in our algorithm, we will cut off the wavelet coefficients started with less important shape and level to more important shape and level. Every time we cut we will check that PSNR. If PSNR less than PSNR threshold we desired, we will stop cutting. The PSNR threshold we desired depends on quality of image that we suppose to get. We will not code the level that we set to be zero in the level 1 when coding step, so at least 25% of

data is reduce in this step. The result after cut off some level of Lena and Peppers image is showed in Table 6.2.

Color Image	Cutoff Coefficients	PSNR	Cr
Lena	Level1 Shape1	41.17	17.52
Lena	Level1 Shape1,2	36.04	21.24
Lena	Level1 Shape1,2,3	31.55	36.36
Peppers	Level1 Shape1	44.56	15.52
Peppers	Level1 Shape1,2	34.05	21.43
Peppers	Level1 Shape1,2,3	31.38	36.36

Table 6.2 The Result PSNR and Compression Ratio of Lena and Peppers
When Cut Off Some of Wavelet Coefficients

Although the cutting off level still gives a good quality of image, but it gives a low compression ratio. We can increase the compression ratio using concept that wavelet coefficients only indicate changes, areas with no change or very small change give small or zero coefficients, which can be ignored, reducing the number of coefficients that have to be kept to encode the information. We scale the wavelet coefficients by dividing it with scaling number. Scaling wavelet coefficients make some small numbers of data that less important for reconstruction became zero, and the large numbers of data are scaled to

small numbers which results in increasing compression ratio. However, higher levels of wavelet coefficients are important especially in level 6-8 which we will not scale them.

Step 4 Quantizing the wavelet coefficients.

Quantization is an important step to increase the compression ratio which has the detail in Chapter 4. Using vector quantization method, we can reduce some of data before coding. We use LGB vector quantization algorithm. After quantized the wavelet coefficients, we have got codebook and codeword that represent index of wavelet coefficients in the codebook. In creating codebook, we divide the wavelet coefficients to 16-blocks vectors. We select the initial codebook from the wavelet coefficients according to step3 and use wavelet coefficients to be training set except level 6-8. We fix level 6-8 of wavelet coefficients in the codebook, but do not put them on the training because these levels have influence for error in reconstruction.

Step 5 Arithmetic coding

The codebook and codeword from step 4 are coded by arithmetic coding which has the detail in Chapter 5.

The reconstruction process is inversion of compression process.

6.2 Results and Discussion

The results of PSNR and compression ratio of tested image after passed from step 1 to step 5 and reconstruction are showed in Table 6.3-6.6. The result images are in Figure 6.8-6.9.

Lena				
Cut off Wavelet Coefficients	Scaling Parameter	Level of Codebook	PSNR	Cr
Shape 1,2,3 of Level 1	1	640	31.17	46.15
Shape 1,2,3 of Level 1	5	640	31.14	64.86
Shape 1,2,3 of Level 1	10	640	31.02	77.42
Shape 1,2,3 of Level 1	15	640	30.86	85.71
Shape 1,2,3 of Level 1	20	640	30.67	96.00
Shape 1,2,3 of Level 1	25	640	30.45	100

Table 6.3 Comparison PSNR and Cr of Lena after compression when use different scaling parameter

The Table 6.3 shows the results of PSNR and Cr when we cut off the wavelet coefficients in level 1 shape 1, 2, 3, and scaling the wavelet coefficients with parameter 1-25, then using codebook 640 level. We will see that if we increase the scaling parameter 1 to 25, the Cr increases from 46.15 to 100 while the PSNR decreases slightly. The cause of increasing of Cr is after scaling, many of wavelet coefficients became the same number that will code in the same code in coding step giving the short length of codeword. The

slightly decrease of PSNR, although the wavelet coefficients have reduced about 84%, which 75% is from cut off all of wavelet coefficients in level 1 and about 9% is from vector quantization, presents the powerful of wavelet transform. However, the scaling wavelet coefficients appropriates with low quality needed image.

Peppers				
Cut off Wavelet Coefficients	Scaling Parameter	Level of Codebook	PSNR	Cr
Shape 1 of Level 1	1	1536	38.46	24.00
Shape 1 of Level 1	5	1536	37.85	30.77
Shape 1 of Level 1	10	1536	36.81	36.92
Shape 1 of Level 1	15	1536	35.75	42.11
Shape 1 of Level 1	20	1536	34.74	47.06
Shape 1 of Level 1	25	1536	33.82	57.14

Table 6.4 Comparison PSNR and Cr of Peppers when use different scaling parameter

Table 6.4 shows the results of Peppers image. We cut off shape 1 and use codebook level 1536. When scaling parameter decreases, PSNR decrease more than testing with cut off all of level 1, because the shapes 2 and 3 have many zero numbers. The Cr still increases.

Lena				
Cut off Wavelet Coefficients	Scaling Parameter	Level of Codebook	PSNR	Cr
-	20	1536	34.85	48.99
Shape 3 of Level 1	20	1408	34.44	52.17
Shape 3 of Level 1	20	1208	34.21	55.81
Shape 3 of Level 1	20	1152	33.98	58.54
Shape 3 of Level 1	20	1024	33.78	63.61
Shape 2,3 of Level 1	20	896	32.89	70.89
Shape 2,3 of Level 1	20	768	32.41	77.42
Shape 1,2,3 of Level 1	20	768	30.88	85.71
Shape 1,2,3 of Level 1	20	640	30.67	96
Shape 1,2,3 of Level 1	20	512	30.03	109.09

Table 6.5 Comparison PSNR and Cr of Lena when change level of codebook and cut off

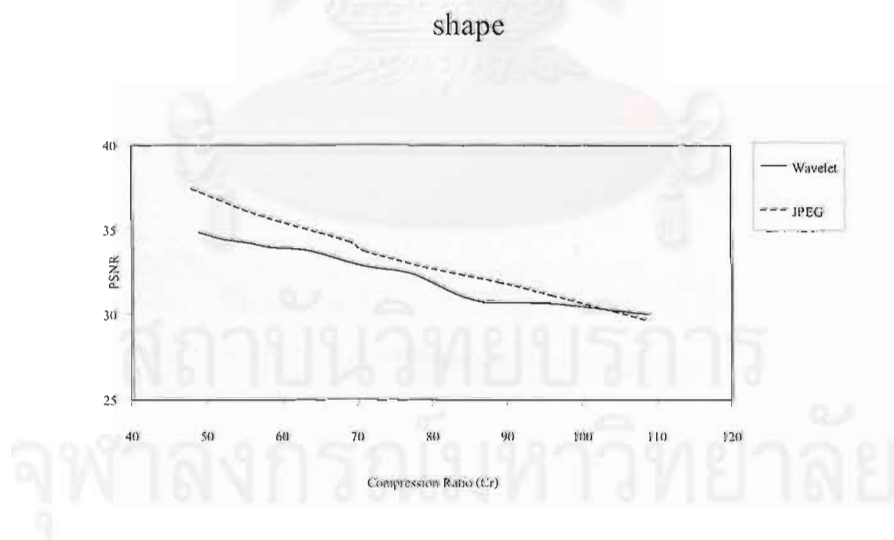


Figure 6.5 Show Graph of PSNR and Cr of Lena Compressed by Wavelet versus JPEG

Peppers				
Cut off Wavelet Coefficients	Scaling Parameter	Level of Codebook	PSNR	Cr
-	20	1536	38.46	24.00
Shape 3 of Level 1	20	1408	34.59	54.81
Shape 3 of Level 1	20	1208	34.24	55.33
Shape 3 of Level 1	20	1152	34.43	55.81
Shape 3 of Level 1	20	1024	34.09	58.54
Shape 2,3 of Level 1	20	896	31.90	66.67
Shape 2,3 of Level 1	20	768	31.74	72.73
Shape 1,2,3 of Level 1	20	768	30.73	80.00
Shape 1,2,3 of Level 1	20	640	30.33	100.00
Shape 1,2,3 of Level 1	20	512	29.72	104.35

Table 6.6 Comparison PSNR and Cr of Peppers when change level of codebook and cut off shape

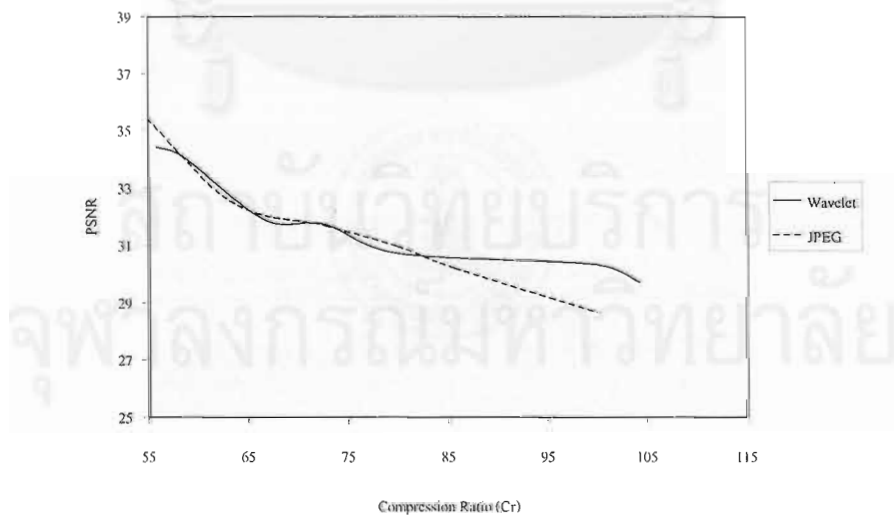
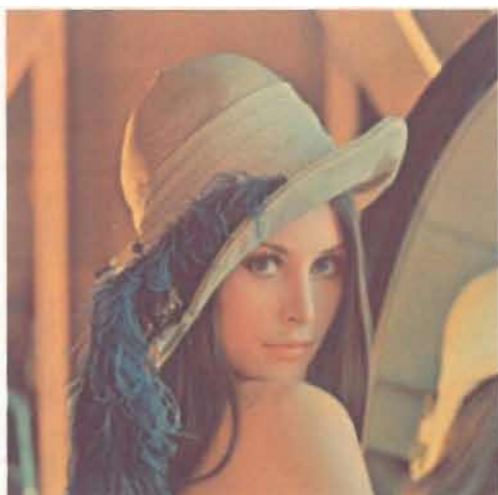


Figure 6.6 Show Graph of PSNR and Cr of Peppers Compressed by Wavelet versus JPEG

Table 6.5–6.6 show the results of PSNR and Cr of Lena and Peppers image when we fix scaling parameter at 20 and decrease size of codebook. The results show that Lena is compressed with Cr 50-60 yielding PSNR around 34-35 and Cr 80-100 yielding PSNR around 30-31, and Peppers is compressed with Cr 55-58 yielding PSNR around 34 and Cr 80-100 yielding PSNR around 30. Comparing results with JPEG we plot graphs of PSNR and Cr of these results compared with JPEG which are showed in Figure 6.5-6.6. From graphs, they present that this compression procedure gives the results nearly JPEG that depend on image and Cr. At high Cr, this method gives better PSNR than JPEG, but in low Cr it gives worse or equal results.





(a) Lena cut off shape1 of level 1

PSNR 41.17 Cr 17.52



(d) Peppers cut off shape 1 of level 1

PSNR 44.56 Cr 15.52



(b) Lena cut off shape 1, 2 of level 1

PSNR 36.04 Cr 21.24



(e) Peppers cut off shape 1,2 of level 1

PSNR 34.05 Cr 21.43



(c) Lena cut off shape 1,2,3 of level 1

PSNR 34.05 Cr 21.43



(f) Peppers cut off shape 1,2,3 of level1

PSNR 31.38 Cr 36.36

Figure 6.7 Results from Table 4.2: cut off some shape of level of Wavelet Coefficients



Original Lena



Lena Compressed by JPEG Cr 109 PSNR 29.66



Lena Compressed by Wavelet Cr 109 PSNR 30.03

Figure 6.8 Compare Lena Compressed by Wavelet Versus JPEG at Cr 109



Original Lena



Lena Compressed by JPEG Cr 96 PSNR 31.15



Lena Compressed by Wavelet Cr 96 PSNR 30.67

Figure 6.9 Compare Lena Compressed by Wavelet Versus JPEG at Cr 96



Original Lena

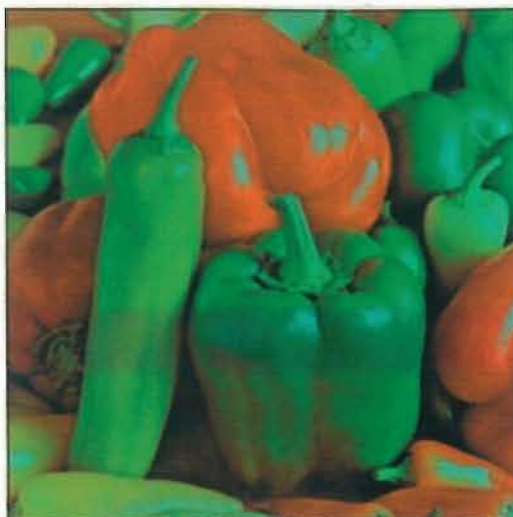


Lena Compressed by JPEG Cr 71 PSNR 33.76

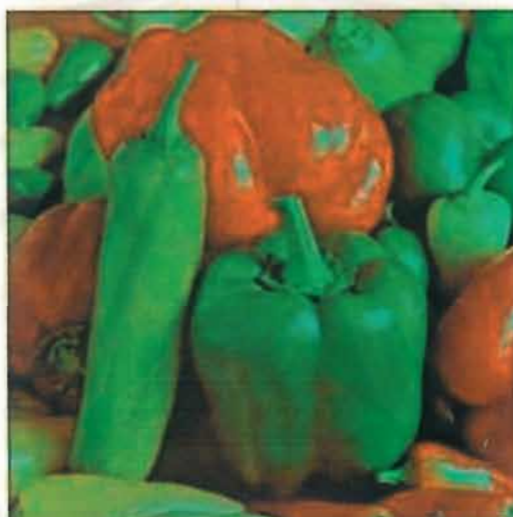


Lena Compressed by Wavelet Cr 71 PSNR 32.89

Figure 6.10 Compare Lena Compressed by Wavelet Versus JPEG at Cr 71



Original Peppers



Peppers Compressed by JPEG Cr 100 PSNR 28.67

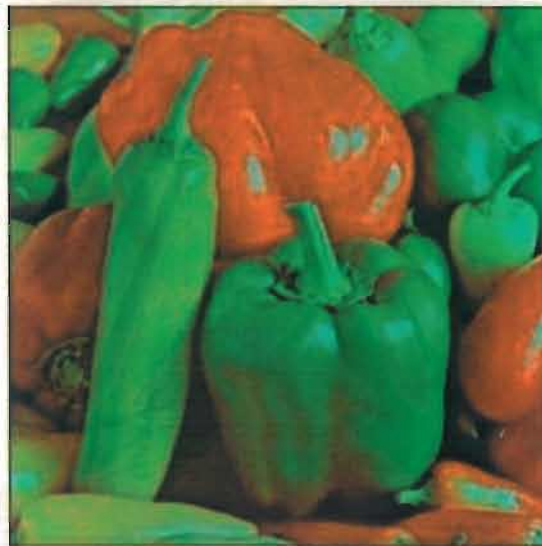


Peppers Compressed by Wavelet Cr 100 PSNR 30.33

Figure 6.11 Compare Peppers Compressed by Wavelet Versus JPEG at Cr100



Peppers Original



Peppers Compressed by JPEG Cr 80 PSNR 30.97



Peppers Compressed by Wavelet Cr 80 PSNR 31.74

Figure 6.12 Compare Peppers Compressed by Wavelet Versus JPEG at Cr80



Original Peppers



Peppers Compressed by JPEG Cr 73 PSNR 31.70



Peppers Compressed by Wavelet Cr 73 PSNR 31.74

Figure 6.13 Compare Peppers Compressed by Wavelet Versus JPEG at Cr73

CHAPTER VII



CONCLUSION

This thesis develops a color image compression procedure using wavelet transform technique, LGB vector quantization and arithmetic coding. Compression is done separately on the RGB plane. Before quantization procedure, we remove the unnecessary wavelet coefficients for reconstruction by cutting off wavelet coefficients, set wavelet coefficients to zero and scaled wavelet coefficients to reduce a amount of information for encoding.

We test our algorithm with the images of Lena and Peppers. The results are as follows: Lena is compressed with compression ratio 50-60 yielding PSNR around 34-35 while compression ratio 80-100 yielding PSNR around 30-31; Peppers is compressed with compression ratio 55-58 yielding PSNR around 34 while compression ratio 80-100 yielding PSNR around 30. This technique gives high PSNR of image than JPEG at high compression ratio over 90, but low PSNR than JPEG at compression ratio below 85.

The benefit of this work is a new wavelet transform procedure for color image compression which gives higher quality of image than JPEG at high compression ratio.

REFERENCES

1. Held, G., and Marchall, T.R. Data and Image Compression Tools and Techniques.
4th ed. England: John Wiley & Sons Ltd., 1996.
2. Hubbard, B.B., The World According to Wavelets. MA: Wellesley, 1995.
3. Antonini, M., Barlaud, M., Mathieu, P., and others. Image Coding Using Wavelet Transform. IEEE Transactions on Image Processing vol. 1, No. 2 (April 1992):
205-219.
4. Averbuch, A., Lazer, D., and Israeli, M. Image Compression Using Wavelet Transform and Multiresolution Decomposition. IEEE Transactions on Image Processing vol. 5, No 1 (January 1996): 4-15.
5. Linde, Y., Buzo, A., and Gray, R.M. An Algorithm for Vector Quantizer Design IEEE Transactions on Communications vol. Com-28, No. 1 (January 1980):
84-95.
6. Nasrabadi, N.M., and King, R.A. Image Coding Using Vector Quantization: A Review. IEEE Transactions on Communications vol. 4, No. 2 (February 1995):
177-185.
7. Overturf, L.A., Comer, M.L., and Delp, E.J. Color Image Coding Using Morphological Pyramid Decomposition. IEEE Transactions on Image

- Processing vol. 4, No. 2 (February 1995): 177-185.
8. Ho, Y., and Gersho, A. A Pyramidal Image Coder with Contour-based Interpolative Vector Quantization. In Proc. SPIE Visual Commun., Image Processing Conf. IV vol. 1199 (November 1989): 733-740.
 9. Gonzalez, R.C., and Woods, R.E. Digital Image Processing. USA: Addison Wesley Publishing company, 1993.
 10. Hunt, R.W.G. The Reproduction of Colour in Photography, Printing, and Television. England: Fountain, 1987.
 11. Garbor, D. Theory of Communication. J. Inst. Electr. Engineering London, vol. 93 (III) (1946): 429-457
 12. Mallat, S. A Theory for Multiresolution Signal Decomposition: The Wavelet Representation. IEEE Trans. Pattern Anal. Machine Intell. PAMI-31 (1989): 674-693.
 13. Burrus, C.S., Gopinath, R.A., Guo, H. Introduction to Wavelets and Wavelet Transforms A Primer. New Jersey: Prentice-Hall, Inc., 1998.
 14. Daubechies I. Ten lectures on wavelets. Philadelphia, PA : SIAM,1992.
 15. William H.P., Sual A.T., William T.V. and others. Numerical

CURRICULUM VITAE

Miss Kanidta Positwinyu was born on September 5, 1975 in Bangkok. She received her B.Sc. degree in Polymer Science from the Department of Material Science, Faculty of Science, Chulalongkorn University in 1996.



สถาบันวิทยบริการ
จุฬาลงกรณ์มหาวิทยาลัย