National Library of Canada
Acquisitions and Bibliographic Services Branch
395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada
Direction des acquisitions et des services bibliographiques
395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Canada

Algorithms and Architectures for Progressive Image Transmission

Giridharan Iyengar, B.Tech

A THESIS

submitted to the School of Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the Degree of

MASTER OF APPLIED SCIENCE

in

Electrical Engineering

Ottawa-Carleton Institute of Electrical Engineering

Department of Electrical Engineering

Faculty of Engineering

University of Ottawa

OTTAWA, ONTARIO, K1N 6N5

ISBN 0-315-89610-8

Canada

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

Dedicated to my beloved parents and to my teachers

Acknowledgments

First, I would like to thank both my supervisors Dr.M.Goldberg and Dr.S.Panchanathan for introducing me to the exciting field of Image/video coding and for their support, encouragement during my thesis work. I enjoyed the benefits of being a member of the Multimedia Research Group at University of Ottawa.

I would also like to thank all the past and current students of Room B-409, Colonel By of University of Ottawa, for making my stay at this university a memorable one.

My special thanks are also due to all the support staff members of Electrical Engineering for their help, specially to Michèle Roy, Suzanne St-Michel and Lucette Lepage.

I am truly grateful to my beloved fianceé Sri and my family for their consistent support, without which this work would not have been possible.

I am also thankful to the Canadian Institute for Telecommunications Research for their financial support.

Abstract

Transmission of images over low speed, low bandwidth channels has several applications. Examples include transmission of remote sensing images, scanning of vast global databases and facsimile transmission of printed material. In progressive image transmission, a low resolution image is first transmitted with as few bits as possible. The resolution of the transmitted image is then improved interactively upon viewer's request. Eventually, an exact copy of the original image is reconstructed.

In this thesis, image coding techniques and parallel architectures are first reviewed followed by a review of progressive image transmission techniques. A software toolkit for progressive image transmission is then presented. This tool is user-friendly and permits rapid development of new algorithms and fast evaluations of different approaches for progressive image transmission. Unlike conventional techniques, it offers a convenient platform for performing both objective and subjective evaluations with ease. Two algorithms for progressive image transmission are then detailed. The first algorithm follows the coefficient scanning methodology. The second algorithm uses an iterative residual error feedback technique. An architecture which implements these algorithms in real-time is then presented.

The software tool is used as the platform with which the comparative analysis between different algorithms are performed. Simulation results are reported for the proposed algorithms in terms of Peak Signal to Noise Ratio (PSNR). Simulation results indicate that the proposed algorithms perform better than JPEG-DCT in terms of subjective and objective quality. However, Gabor decomposition is computationally expensive and hence requires special purpose architectures for real-time implementation. The proposed architecture achieves a high efficiency of parallelism (99 %) and implements the two algorithms for progressive image transmission in real-time. The architecture is simple and modular and hence can be easily implemented in VLSI as a codec.

List of Figures

2.1 Gabor and DCT basis functions of order 0,0. The mesh plot on top is the Gabor basis function and the plot on bottom is the DCT basis function 29
2.2 Gabor and DCT basis functions of order 2,0 30
2.3 Gabor and DCT basis functions of order 0,2 30
2.4 Neural network for Gabor decomposition 31
2.5 Associative string processor substring 32

3.1 A Bit plane representation of an 8 bit image 42
3.2 Example of quadtree pyramid structures 42
3.3 The zig-zag scan path of coefficients used in JPEG-DCT 43
3.4 Bit maps (left) and incremental bit maps (right) for a 4×4 block and an incremental rate of 1 bits/pixel 44

4.1 An example of the Khoros workspace 53
4.2 The PIT tool workspace 54
4.3 Rate distortion curve (PSNR vs bit rate) for JPEG-DCT encoded Lenna image 55
4.4 An example of the interactive zooming operation 56
4.5 Difference image of JPEG-DCT encoded Lenna at 0.28 bpp 57

5.1 Image compression system for Gabor decomposition 72

5.2 Splitting of the image into 13 bands and subsequent quantization tuned to human visual system characteristics 73
5.3 Rate distortion curve (PSNR vs bit rate) for the Lenna image encoded using JPEG-DCT and Gabor algorithms 74
5.4 Flowchart of the CST algorithm 75
5.5 Flowchart of the PGT algorithm 76
5.6 Rate distortion curve (PSNR vs bit rate) for the Lenna image encoded using JPEG-DCT, CST and PGT algorithms 77
5.7 Rate distortion curve (PSNR vs bit rate) for the Mandril image encoded using the JPEG-DCT, CST and PGT algorithms 78
5.8 The original Lenna image 79
5.9 The original Mandril image 80
5.10 JPEG-DCT encoded Lenna image reconstructed at 0.08 bits per pixel 81
5.11 JPEG-DCT encoded Lenna image reconstructed at 0.32 bits per pixel 82
5.12 CST encoded Lenna image reconstructed at 0.078 bits per pixel . . . 83
5.13 CST encoded Lenna image reconstructed at 0.3 bits per pixel 84
5.14 PGT encoded Lenna image reconstructed at 0.06 bits per pixel 85
5.15 PGT encoded Lenna image reconstructed at 0.31 bits per pixel 86
5.16 PGT encoded Lenna image reconstructed at 0.022 bits per pixel . . . 87
5.17 JPEG-DCT encoded Mandril image reconstructed at a bit rate of 0.08 bits per pixel 88
5.18 JPEG-DCT encoded Mandril image reconstructed at 0.31 bits per pixel 89
5.19 CST encoded Mandril image reconstructed at 0.08 bits per pixel . . . 90
5.20 CST encoded Mandril image reconstructed at 0.3 bits per pixel 91
5.21 PGT encoded Mandril image reconstructed at 0.07 bits per pixel . . . 92
5.22 PGT encoded Mandril image reconstructed at 0.3 bits per pixel . . . 93

6.1 The principle of systolic arrays 108

6.2 Systolic array for banded matrix multiplication 109
6.3 Overview of the proposed SCAM module 109
6.4 Example of the inner product operation 110
6.5 Layout of the basic CAM cell 110
6.6 SCAM module for real-time Gabor decomposition 111
6.7 Snapshots of the SCAM module at time 1 and time 2 112
6.8 Cell occupancy diagram for the SCAM module 113
6.9 Speedup S_p versus Gabor kernel bandwidth p for 512 PEs 114
6.10 Codec for PIT by the CST algorithm using the SCAM architecture for Gabor decomposition 115
6.11 Codec for PIT by the PGT algorithm using SCAM architecture for Gabor decomposition 116

List of Tables

2.1 Arithmetic coding: allocation of range to source symbols 29

4.1 Simulations results for the Lenna image using JPEG-DCT algorithm 53

5.1 Simulation results for Lenna image using JPEG-DCT algorithm . . . 69
5.2 Simulation results for Lenna image using the CST algorithm 70
5.3 Simulation results for Lenna image using the PGT algorithm 70
5.4 Simulation results for Mandril image using the JPEG-DCT algorithm 71
5.5 Simulation results for Mandril image using the CST algorithm 71
5.6 Simulation results for Mandril image using the PGT algorithm 72

Contents

1 Introduction 1

1.1 Progressive Image Transmission 1

1.2 Definition of the Problem 2

1.3 Investigated approach 2

1.4 Thesis organization 3

1.5 Main contributions 4

2 Image coding 5

2.1 Source coding theory 5

2.1.1 Entropy 6

2.1.2 Rate Distortion Function 7

2.1.3 Distortion Measure 9

2.2 Lossless coding 10

2.2.1 Arithmetic coding 11

2.3 Lossy Coding 12

2.3.1 Transform coding 13

2.3.2 Vector quantization 19

2.4 Gabor Decomposition 20

2.4.1 1-D and 2-D Gabor functions 21

2.4.2 Pyramidal Gabor functions 22

2.4.3 Discrete Gabor functions 22
2.4.4 Inverse Gabor decomposition 23
2.5 Image Coding Architectures 24
2.5.1 Codecs for the image coding standards 25
2.5.2 Implementations of Gabor decomposition 25
2.6 Summary 28

3 Review of Progressive Image Transmission **33**
3.1 Direct spatial techniques 34
3.2 Spatial pyramidal techniques 34
3.2.1 Quadtree 35
3.2.2 Nonuniform Quadtree 36
3.2.3 Filtered pyramids 37
3.3 Transform domain techniques 38
3.3.1 Coefficient Scanning 38
3.3.2 Bit Slicing 39
3.3.3 S-Transform 40
3.3.4 Multistage Vector Quantization 40
3.4 Summary 41

4 Evaluation environment for Progresive Image Transmission **45**
4.1 KHOROS Software environment 46
4.2 Software Tool for Progressive Image Transmission 47
4.2.1 Details of the Software 48
4.2.2 Development of new algorithms using the tool 49
4.3 Evaluation of PIT Schemes Using the Tool 49
4.3.1 Evaluation of PIT Schemes 50
4.3.2 Usage of the tool 50

4.4 Summary 51

5 Progressive Image Transmission using Gabor Decomposition 58

5.1 Image compression using Gabor decomposition 59

5.2 Algorithms for PIT using Gabor decomposition 60

5.2.1 Coefficient scanning algorithm (CST) 60

5.2.2 Pyramidal Gabor algorithm (PGT) 62

5.3 Simulation results and discussions 66

5.3.1 Comparisons with JPEG-DCT 67

5.4 Summary 68

6 Codec for PIT using Gabor decomposition 94

6.1 Systolic arrays and content-addressable memories 95

6.1.1 Systolic arrays 95

6.1.2 Content-addressable memory 96

6.2 Systolic array architecture for real-time Gabor decomposition 97

6.2.1 Systolic array for banded matrix multiplication 97

6.2.2 CAM based Systolic array for Gabor decomposition (SCAM) . 98

6.3 Performance analysis of the proposed systolic array 102

6.4 Codec for progressive image transmission using Gabor decomposition 104

6.4.1 Codec for PIT using the CST algorithm 104

6.4.2 Codec for PIT using the PGT algorithm 105

6.5 Summary 107

7 Conclusions and Future work 117

7.1 Conclusions 117

7.2 Future work 118

Chapter 1

Introduction

1.1 Progressive Image Transmission

Transmission of still images is an important field in image communications. Many important applications such as teleconferencing, still image broadcasting, remote surveillance, computer graphics communications, interactive visual database search, etc. are based on the technology of image communications. Interactive image communications over low bandwidth channels is an emerging area of study. Its applications include scanning of vast global image databases, digital video on demand, etc.

Interactive low-bandwidth image communications are best served by progressive image transmission (PIT). In PIT, a low resolution version of the image is first transmitted. Upon viewers' request, the resolution is further enhanced interactively until the desired fidelity is achieved.

1.2 Definition of the Problem

PIT applications such as interactive scanning of a large image database require that the user be able to decide and zoom in on the desired image . Hence the PIT scheme must not only be subjectively appealing but also be capable of transmitting as much gross structural information as possible in the first iterations.
The basic requirements of a PIT scheme can be summarized as follows:

1) The image to be transmitted must be reorganized in a form suitable for progressive image transmission.
2) The first approximations, transmitted at a very low bit rate, must provide enough gross structural information so that the image can be recognized.
3) The algorithm should provide a good coding performance.
4) If desired, the algorithm should provide a lossless reproduction of the original image.

1.3 Investigated approach

In this thesis the problem of PIT is approached from three different perspectives. In order to evaluate different techniques, a software tool is first introduced. Algorithms for PIT using Gabor decomposition are then presented. In order to implement these algorithms, a high performance systolic array architecture is then designed.
In order to compare different PIT techniques, a unified platform is desired which enables fast and efficient evaluations of different algorithms. In this thesis, an X-window based toolkit is developed which enables rapid prototyping of new algorithms and performs both objective and subjective comparisons with ease. This tool has an open architecture capability and hence permits addition of new algorithms with ease. It also enables simultaneous evaluations of two or more techniques, both in terms of objective and subjective measures.

From the algorithmic perspective, Gabor decomposition was chosen as the underlying technique. Gabor decomposition has been found to be very attractive for image compression. The main advantage of using Gabor elementary functions (GEF) is their ability to achieve the lowest bound on the joint entropy of data in the spatial-frequency domain. They are capable of exploiting the image data correlation and are also well suited for multiresolution pyramidal schemes. The main drawback of these GEFs is the non-orthogonality of the basis functions. Thus an analytical solution does not exist. Iterative solutions or approximations such as singular value decomposition have been applied in the past to tackle this problem. Singular value decomposition is a powerful technique which is computable with relative ease. In this thesis, the singular value decomposition approach is used.

Although the Gabor kernel has been calculated using techniques such as SVD or bi-orthogonal functions, the decomposition of an image is still compute intensive and therefore needs special purpose architectures for real-time implementations. It has been shown that Gabor decomposition can be reformulated as a matrix multiplication operation [60]. Systolic arrays are efficient for matrix multiplication problems [68] and content-addressable memories offer fast data access [67]. In this thesis, we propose a systolic array - content addressable memory architecture for real-time Gabor decomposition.

1.4 Thesis organization

The thesis is organized as follows: Chapter 2 reviews image coding and architectures for image coding. Chapter 3 presents different progressive image transmission techniques that have been proposed in the literature. Chapter 4 details the software toolkit which will be used for performing the necessary evaluations of different PIT schemes. Chapter 5 outlines the two algorithms for PIT using Gabor decomposition.

Chapter 6 describes the codec for real-time implementation of Gabor decomposition. This follows in Chapter 7 with the conclusions and suggestions for further research.

1.5 Main contributions

The main contributions of this thesis are summarized below:

- Development of a software toolkit which permits convenient evaluations of different algorithms and protyping of new algorithms for PIT.
- Algorithms for PIT using Gabor decomposition.
- Architecture for real-time Gabor decomposition.

Chapter 2

Image coding

In this chapter, the basic concepts of information theory, including distortion measures for image coding are first presented. A short review of lossless and lossy image coding follows. This continues with a discussion on Gabor decomposition. Finally, a short review of different image coding architectures is presented.

2.1 Source coding theory

In source coding theory, the two basic concepts are *entropy* and *rate distortion*. Rate distorion theory provides a lower bound on the average bit rate for a given distortion and hence an upper bound on the performance of different coders. The lowest bit rate required to represent an information source without any loss is called the entropy of the source. Entropy is a very useful measure since it provides the minimum average bit rate for perfect reconstruction. The concepts of entropy and rate distortion are detailed in the following sections.

2.1.1 Entropy

Consider an image of size $N \times N$ with each pixel quantized to K gray levels. A total of $K^{N \times N}$ possible image patterns can be generated by the image source. Let the probablity of a specific image pattern be given by $p(x)$ where

$$x = x_{ij}, i, j = 0, 1, \cdots, N-1, \tag{2.1}$$

where x_{ij} is the (i,j)th element of x. The average information of the source is specified by its entropy which is defined as [1]:

$$H(X) = -\frac{1}{N \times N} \sum_{all\ x} p(x) \log_2 p(x) \quad bits/pixel. \tag{2.2}$$

It has been shown that the source entropy is bounded by 0 and $\log_2 K$, that is [2],

$$0 \leq H(X) \leq \log_2 K. \tag{2.3}$$

The left side equality holds if all probabilities except one are zero, in which case the source is totally predictable. The right side equality holds when every source symbol has an equal probability. The redundancy of the source is the difference between $\log_2 K$ and the source entropy $H(X)$.

If every pixel in an image were statistically independent, then the source probability $p(x)$ can be expressed as:

$$p(x) = \prod_{i,j=0}^{N-1} p_{ij}(x_{ij}) \tag{2.4}$$

Here, p_{ij} represents the probability that the pixel X_{ij}, of the image source X, has a value equal to x_{ij}. In this case we get $H(X)$ as:

$$H(X) = -\frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sum_{all\ x_{ij}} p_{ij}(x_{ij}) \log_2 p_{ij}(x_{ij}). \tag{2.5}$$

In practice, the statistical information of an image, $p(x)$, is, however, not easily measured or modelled and therefore, the true entropy of the image is, in general, very

difficult to obtain. Hence a simpler measure, the first order entropy $H^1(X)$, which is defined on a pixel-by-pixel basis is often used. $H^1(X)$ can be defined as:

$$H^1(X) = -\sum_{k=0}^{N-1} P_k \log_2 P_k. \tag{2.6}$$

If the pixels of the image are identically and independently distributed (i.i.d), that is:

$$p_{ij}(x_{ij}) = p_{uv}(x_{uv}), \quad for\ x_{ij} = x_{uv} \tag{2.7}$$

then the entropy $H(X)$ of the image then equals the first order entropy, $H^1(X)$:

$$H(X) = H^1(X) = -\sum_{k=0}^{N-1} P_k \log_2 P_k. \tag{2.8}$$

In this case, P_k is simply the probability of the occurence of the kth grey level value.

2.1.2 Rate Distortion Function

The entropy of a source determines the lower bound on the average bit rate for lossless coding. For lossy coding, the rate distortion function (RDF) determines the minimum distortion that can be achieved for a given average bit rate. Hence the RDF provides an upper bound on the performance of practical coders [3]. The RDF is a measure of transmission of information from the source to the receiver. The average mutual information $I_{N\times N}(X, \hat{X})$ between the source and the receiver is defined as:

$$I_{N\times N} = \frac{1}{N \times N} \sum_{all\ x,\hat{x}} p(x)p(\hat{x}/x) \log \frac{p(\hat{x}/x)}{\sum_{all\ x} p(x)p(\hat{x}/x)}. \tag{2.9}$$

This is a measure of the statistical dependence between the source output X and the decoded output $\hat{X}$. Since $p(x)p(\hat{x}/x) = p(x, \hat{x})$, it follows that

$$\sum_{all\ x} p(x)p(\hat{x}/x) = \sum_{all\ x} p(x, \hat{x}) = p(\hat{x}) \tag{2.10}$$

and therefore,

$$I_{N\times N} = \frac{1}{N \times N} \sum_{all\ x,\hat{x}} p(x, \hat{x}) \log_2 \frac{p(\hat{x}/x)}{p(\hat{x})}. \tag{2.11}$$

We can show that

$$I_{N\times N} = H(X) - H(X/\hat{X}) \tag{2.12}$$

where $H(X/\hat{X})$ is the entropy of the source X given the decoder output $\hat{X}$. Hence the mutual information between X and $\hat{X}$ is the difference in the uncertainty at the source X and the uncertainty at the source X given the decoder output $\hat{X}$. If the encoding is lossless then the average mutual information is given by:

$$I_{N\times N} = H(X) - H(X/\hat{X}) = H(X) \tag{2.13}$$

This implies that for lossless coding, the least average bit rate is $H(X)$. On the other hand, if there is no information between the encoder and the decoder, the mutual information $I_{N\times N}$ is equal to 0. Therefore, the average mutual information is lower bounded by 0 and upper bounded by the source entropy, i.e.

$$0 \leq I_{N\times N} \leq H(X). \tag{2.14}$$

If we define a distance or distortion measure $d(x,\hat{x})$ between x and $\hat{x}$, the average distortion per pixel is therefore,

$$D = \frac{1}{N\times N} E[d(X,\hat{X})] = \frac{1}{N\times N} \sum_{all\ x,\hat{x}} d(x,\hat{x})p(x)p(\hat{x}/x). \tag{2.15}$$

The $N \times N$ block rate distortion function $R_{N\times N}(D)$ [4] is defined as the minimum average mutual information between X and $\hat{X}$ subject to the constraint of a fixed average distortion D:

$$R_{N\times N}(D) = \inf_{all\ p(\hat{x}/x)} I_{N\times N}(X,\hat{X}). \tag{2.16}$$

Note that since the source is known, the minimization can only be over the conditional probability $p(\hat{x}/x)$. The rate distortion function $R(D)$ [4] is simply the limiting value of the $N \times N$ - block rate distortion function $R_{N\times N}(D)$ as the block size $N \times N \rightarrow \infty$,

$$R(D) = \lim_{N\times N\rightarrow\infty} R_{N\times N}(D). \tag{2.17}$$

In other words, the distortion D and the mutual information $I_{N\times N}(X,\hat{X})$ depend on the type of source coding. However, there is a minimum $I_{N\times N}$ that is needed so that the average distortion of reconstruction at the destination does not exceed the specified upper limit D. This minimum value of $I_{N\times N}$ is $R(D)$. For perfect reconstruction ($D = 0$), the bit rate required is just equal to the source entropy or average self-information,

$$R(0) = H(X) \tag{2.18}$$

2.1.3 Distortion Measure

In order to evaluate a coding scheme, a distortion measure has to be used which is a cost function $d(x,\hat{x})$ for reproducing the input x by an output $\hat{x}$. With such a cost function, the performance of a coding system can be evaluated using the average distortion introduced by the coding system, $E[d(X,\hat{X})]$,

$$E[d(X,\hat{X})] = \sum_{all\ x,\hat{x}} d(x,\hat{x})p(x,\hat{x}). \tag{2.19}$$

To be of value for both design and comparison, a distortion measure must be:

- Computable, so that it can be efficiently evaluated in real time;
- Subjectively meaningful so that it correlates well with human visual observations;
- Tractable, so that it is mathematically analyzable.

One of the most popular distortion measures used in image coding is the Peak Signal to Noise Ratio (PSNR) [5], defined as:

$$Peak\ Signal\ to\ Noise\ Ratio(PSNR) = 10\log_{10}(\frac{255\times 255}{MSE}) \tag{2.20}$$

where

$$MSE = \frac{1}{MN}\times\sum_{i=1}^{M}\sum_{j=1}^{N}(u_{ij}-v_{ij})^2 \tag{2.21}$$

The other distortion measures that are used include, the Normalized mean square error (NMSE), Signal to Noise ratio (SNR) and Mean absolute error (MAE) defined as follows:

$$Normalized\ mean\ square\ error(NMSE) = \frac{\left[\sum_{i=1}^{M}\sum_{j=1}^{N}(u_{ij}-v_{ij})^2\right]}{\left[\sum_{i=1}^{M}\sum_{j=1}^{N}(u_{ij})\right]} \tag{2.22}$$

$$Signal\ to\ Noise\ Ratio(SNR) = 10\log_{10}(\frac{1}{NMSE}) \tag{2.23}$$

$$Maximum\ Absolute\ Error(MAE) = \max(|u_{ij}-v_{ij}|)\ \forall i = 1,\cdots,M\ and\ j = 1,\cdots,N \tag{2.24}$$

where u_{ij} is the original image pixel and v_{ij} is the reconstructed image pixel.
Neither the mean square error nor the mean absolute error correlate very well with the subjective ratings. For example, two different images with the same SNR values may receive very different subjective evaluations. There is a considerable effort in the research community to define newer measures which yield higher correlation with the subjective observations [6, 7].
In this thesis, the PSNR is used as the standard distortion measure for evaluating different schemes for progressive image transmission. This measure is analytically tractable and computable in real-time. The low and high PSNR ratings correspond to low and high subjective quality, respectively. In this thesis, in order to compare the different coding schemes, rate distortion curves (PSNR vs. bit rate) are used as evaluation measures.

2.2 Lossless coding

In natural images, there exists a high degree of correlation between the neighbouring pixels, implying statistical redundancy in the image. In lossless coding, this redundancy is exploited in such a way that the process in reversible; i.e the original image

is recovered exactly. There is a lot of interest in lossless techniques, especially in applications such as medical imaging. In this thesis, arithmetic coding is used for lossless encoding. The following section presents a brief review of arithmetic coding.

2.2.1 Arithmetic coding

Arithmetic coding is an optimal variable length coding scheme. It can achieve higher compression than Huffman coding. In Huffman coding, the minimum number of bits required to represent a source symbol is 1 bit. Arithmetic coding achieves fractional bit allocations for source symbols. Hence the performance of arithmetic coders is superior and approaches the source entropy. Huffman coding is optimal only for those sources whose symbol probabilities are negative powers of two.

In arithmetic coding scheme [13], a message is represented by an interval of real numbers between 0 and 1. As the message becomes longer, the interval needed to represent it becomes smaller and the number of bits required to specify the interval grows. Successive symbols reduce the interval in accordance with the symbol probabilities. The most likely symbols reduce the size of the interval less than the unlikely symbols and hence add fewer bits to the message. Before the message is transmitted, the range is $[0, 1)$. As each symbol is processed, the range is narrowed to that portion allocated to the symbol. For example, let the source alphabet be $\{a, e, i, o, u, !\}$ with a fixed probability table as shown in Table 2.6. Let the transmitted message be *eaii!*. Initially, both the encoder and the decoder know that the range is $[0, 1)$. After seeing the first symbol, e, the encoder narrows the range to $[0.2, 0.5)$, the range allocated to this symbol as in Table 2.1. The second symbol, a, will narrow this new range to first one-fifth of it, since a has been allocated to $[0, 0.2)$. This reduces the range to $[0.2, 0.26)$. Proceeding this way, the encoded message builds up as follows:

Initially $[0, 1)$

after seeing	e	[0.2, 0.5)
	a	[0.2, 0.26)
	i	[0.23, 0.236)
	i	[0.233, 0.2336)
	!	[0.23354, 0.2336)

It is not necessary for the decoder to know both the ends of the range. Any number within the range is sufficient for the decoder to decode the transmitted message, for example 0.23355. Since the decoder will not know when to stop as a single number 0.0 could mean *a*, *aa*, *aaa* and so on. To resolve this ambiguity, a special EOF (end of file) symbol is used. When the decoder sees this symbol, it stops decoding.

The decoder operates by simulating the encoder: given the number 0.23355, it decodes that the first symbol is *e* since it lies within the range [0.2, 0.5). The decoder proceeds similarly to decode the rest of the message.

Arithmetic coding scheme can also be implemented in an adaptive manner in which the symbol probabilities are upated as the message is transmitted symbol by symbol. In addition to this; in order to avoid situations such as numeric underflows which occur when the interval is finer than the finite precision of the computer, refresh schemes are built into the arithmetic coder and decoder [14].

Varible length coding schemes (also known as entropy coding schemes) are typically used to exploit any residual redundancy in encoded images using techniques as DCT and VQ. The use of these techniques are not limited to images but also applicable to text files, audio signals, etc. as well.

2.3 Lossy Coding

In lossy coding, the objective is to reduce the transmission bit rate subject to some image quality constraint. There are two classes of lossy techniques: predictive coding

and transform coding. Both techniques have their relative advantages and limitations. From an implementation point of view, the predictive techniques are simpler to implement and real-time compression is possible. Real-time or near real-time performance can be achieved using transform coding schemes with the advent of special purpose VLSI chip sets. Transform coding achieves higher performance in terms of compression ratio for a given distortion value. In transform coding, the errors arising from quantization are spread over the whole block as opposed to being limited to one or two pixels. Hence, in transform coding, the errors are less severe in terms of subjective quality.

The differential pulse-code modulation (DPCM) [2] is perhaps the basic scheme of predictive coding. In DPCM the value of the pixel to be coded is predicted from the previous pixels and then the predicted estimate is subtracted from the actual value. A special case of DPCM is delta modulation in which the difference is categorized into two classes: positive and not-positive. It is an extreme form of DPCM and is not very efficient for rapidly changing image data. Predictive coding is local in nature and is hence quite adaptive. Nevertheless in order to achieve improved performance, an adaptive design can be employed for a predictive coder. For example, in linear prediction, the correlations and the weighting coefficients are recomputed periodically.

2.3.1 Transform coding

In transform coding, the input image X of size $N \times N$ undergoes an *orthogonal transform*:

$$Q = AXA' \tag{2.25}$$

and the resulting decorrelated coefficients Q_{ij} of the transformed image are quantized and entropy coded. At the receiver, the quantized coefficients are inverse transformed to give the reconstructed image.

An optimum orthogonal transform completely decorrelates the transform coefficients and therefore achieves high compression. If the transform coefficients are decorrelated, the geometric mean of their variances is minimized which implies that the reconstruction error is minimized [2]. The Karhunen-Loeve transform is a well known optimum orthogonal transform. However the huge computational overhead associated with the KLT makes it difficult to implement. Moreover, the KLT transform kernel is image dependent. Hence it is not very practical to implement. Sub-optimal transforms such as Discrete Cosine Transform (DCT) are more frequently used in transform image coding.

Two useful properties of orthogonal transform coding are now highlighted [2]:

1) The average energy of the transformed image is equal to the average energy of the input image, that is,

$$\frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{ij}^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{x,ij}^2 = \sigma_x^2. \tag{2.26}$$

Here $\sigma_{ij}^2 = E(Q_{ij}^2)$, $i,j = 0,1,\cdots,N-1$ is the second moment of the (i,j)th transform coefficient and $\sigma_{x,ij}^2 = E(X_{ij}^2)$, $i,j = 0,1,\cdots,N-1$ is the second moment of the (i,j)th input element.

2) The average reconstruction error variance is equal to the average quantization error variance in the transform domain, that is,

$$\sigma_r^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{r,ij}^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{q,ij}^2 = \sigma_q^2. \tag{2.27}$$

Here $\sigma_{r,ij}^2 = E((X_{ij} - \hat{X}_{ij})^2)$, $i,j = 0,1,\cdots,N-1$, is the reconstruction error variance of the (i,j)th input element and $\sigma_{q,ij}^2 = E((Q_{ij} - \hat{Q}_{ij})^2)$, $i,j = 0,1,\cdots,N-1$, is the quantization error variance of the (i,j)th transform coefficient.

Equations 2.26 and 2.27 imply that the analysis in the spatial domain will be completely reflected in the transform domain in terms of the average value of the second moment.

Quantization

In terms of scalar quantization, the quantization error variance σ_q^2 can be related to the input signal variance σ_x^2 as follows [2],

$$\sigma_q^2 = \epsilon_q^2 \sigma_x^2 = \epsilon^2 2^{-2B} \sigma_x^2 \tag{2.28}$$

Here $\epsilon_q^2 = \epsilon^2 2^{-2B}$ is the quantizer performance factor which depends on the probability density function (PDF) of the input signal and on the quantizer characteristics, especially with regards to the number of quantization levels. The quantity ϵ^2 can be considered as a variable correction factor that takes into account the performance of a practical quantizer and B is the number of bits assigned to the quantizer.

Bit Allocation

The bit allocation map indicates the number of bits assigned to each coefficient in the transform domain. The advantage of using an optimal bit allocation in the transform domain is now evaluated. It is first shown that there is little benefit obtained by transforming the input image without optimizing the bit allocation; in other words, if equal bit allocation is used for the transform coefficients. We then calculate the gain that can be achieved using an optimal bit allocation subject to the minimization of the average coefficient error variance for a fixed bit rate.

For the sake of analysis, it is assumed that all elements in the spatial and transform domains have probability distribution functions of the same type but with different variances. An equal bit allocation is assumed; that is, all the elements $X_{ij}, i, j = 0, 1, \cdots, N-1$ in the spatial domain and $Q_{ij}, i, j = 0, 1, \cdots, N-1$ in the transform domain are assigned the same number of bits B (the average bit rate). From equation 2.28, the quantization error variance for the element X_{ij} of the input image X can be written as,

$$\sigma_{q,SD,ij}^2 = \epsilon_{SD,ij}^2 2^{-2B} \sigma_{x,ij}^2 = \epsilon_{SD}^2 2^{-2B} \sigma_{x,ij}^2. \tag{2.29}$$

Here, $\epsilon^2_{SD,ij} = \epsilon^2_{SD}$ based on the assumption that all the elements in the spatial domain have probability distribution functions of the same type but different variances and that an equal bit allocation is used. The average quantization error variance in the spatial domain is, therefore,

$$\begin{aligned}\sigma^2_{q,SD} &= \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma^2_{q,SD,ij} \\ &= \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \epsilon^2_{SD} 2^{-2B} \sigma^2_{x,ij} \\ &= \epsilon^2_{SD} 2^{-2B} \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma^2_{x,ij} \\ &= \epsilon^2_{SD} 2^{-2B} \sigma^2_x. \end{aligned} \tag{2.30}$$

In a similar fashion, the quantization error variance for the transform coefficient Q_{ij} can be written as,

$$\sigma^2_{q,TD,ij} = \epsilon^2_{TD,ij} 2^{-2B} \sigma^2_{ij} = \epsilon^2_{TD} 2^{-2B} \sigma^2_{ij}. \tag{2.31}$$

Once more $\epsilon^2_{TD,ij} = \epsilon^2_{TD}$ based on the same assumption as in the spatial case, and the average quantization error variance in the transform domain is, therefore similarly,

$$\sigma^2_{q,TD} = \epsilon^2_{TD} 2^{-2B} \sigma^2_x \tag{2.32}$$

For both cases, the average quantization (reconstruction) error variance is proportional to the average energy of the input image (or the arithmetic mean of the transform coefficient variance). If logarithmic quantization is assumed in both the spatial and transform domains,

$$\epsilon^2_{SD} = \epsilon^2_{TD}. \tag{2.33}$$

Note that logarithmic quantization is very insensitive to a mismatch relative to the PDF of the input sources [2]. Therefore, it is clear from equations 2.30 and 2.32 that,

$$\sigma^2_{q,TD} = \sigma^2_{q,SD}. \tag{2.34}$$

In other words, there is very little benefit obtained by transforming the input image and just using an equal bit allocation in transform domain.

We now calculate the gain that can be achieved by optimally distributing the bits among the transform coefficients. Let $B_{i,j}$ be the number of bits allocated to the (i,j)th transform coefficient, then, B, the average bit rate is given by,

$$B = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} B_{ij} \tag{2.35}$$

The average coefficient quantization error variance, σ_q^2, is

$$\sigma_q^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \sigma_{q,ij}^2 = \frac{1}{N \times N} \sum_{i,j=0}^{N-1} \epsilon_{ij}^2 2^{-2B_{ij}} \sigma_{ij}^2, \tag{2.36}$$

where

$$\sigma_{q,ij}^2 = \epsilon_{ij}^2 2^{-2B_{ij}} \sigma_{ij}^2. \tag{2.37}$$

The optimality is with regard to the minimization of σ_q^2 under the constraint of a fixed average bit rate B. The calculation of the gain follows [2], where the ϵ_{ij}^2 $i,j = 0,1,\cdots,N-1$, are no longer constant. In fact, ϵ_{ij}^2 $i,j = 0,1,\cdots,N-1$ depend on both the number of bits assigned to and the PDF of the coefficients [2].

If we calculate the gain using Lagrange multiplier method given in [2], we get an optimum bit allocation given by

$$\begin{aligned} B_{i,j} &= B + \frac{1}{2}\log_2 \epsilon_{ij}^2 \sigma_{ij}^2 - \frac{1}{N \times N} \sum_{k,l=0}^{N-1} \frac{1}{2} \log_2 \epsilon_{kl}^2 \sigma_{kl}^2 \\ &= B + \frac{1}{2}\log_2 \frac{\epsilon_{ij}^2 \sigma_{ij}^2}{\left[\prod_{k,l=0}^{N-1} \epsilon_{kl}^2 \sigma_{kl}^2\right]^{\frac{1}{N \times N}}}, \quad i,j = 0,1,\cdots,N-1. \end{aligned} \tag{2.38}$$

It should be observed that the number of quantizer levels $2^{B_{ij}}$ is proportional to both the coefficient variance, σ_{ij}^2, and the quantizer performance factor, ϵ_{ij}^2. The bit allocation formula (equation 2.38) also implies identical error variances in the coefficient quantization:

$$\sigma_{q,kl}^2 = \epsilon_{kl}^2 2^{-2B_{kl}} \sigma_{kl}^2$$

$$= 2^{-2B}\left(\prod_{k,l=0}^{N-1} \epsilon_{kl}^2 \sigma_{kl}^2\right)^{\frac{1}{N\times N}}$$

$$= \epsilon_0^2 2^{-2B}\left(\prod_{k,l=0}^{N-1} \sigma_{kl}^2\right)^{\frac{1}{N\times N}}, \quad i,j = 0,1,\cdots,N-1 \tag{2.39}$$

where

$$\epsilon_0^2 = \left(\prod_{k,l=0}^{N-1} \epsilon_{kl}^2\right)^{\frac{1}{N\times N}} \tag{2.40}$$

Therefore, the average reconstruction error variance is

$$\min(\sigma_r^2) = \min(\sigma_q^2) = \frac{1}{N\times N}\sum_{i,j=0}^{N-1} \sigma_{q,ij}^2 = \epsilon_0^2 2^{-2B}\left(\prod_{k,l=0}^{N-1} \sigma_{kl}^2\right)^{\frac{1}{N\times N}} \tag{2.41}$$

Note that the average reconstruction error variance is proportional to the geometric mean of the coefficient variances.

The ratio of distortion with the optimum bit allocation to that with equal bit allocation in the transform domain is, therefore,

$$\frac{\min(\sigma_q^2)}{\sigma_{q,TD}^2} = \frac{\epsilon_0^2 2^{-2B}\left(\prod_{ij=0}^{N-1} \sigma_{ij}^2\right)^{\frac{1}{N\times N}}}{\epsilon_{TD}^2 2^{-2B}\frac{1}{N\times N}\sum_{i,j=0}^{N-1} \sigma_{ij}^2} = \frac{\epsilon_0^2}{\epsilon_{TD}^2}\frac{\left(\prod_{i,j=0}^{N-1} \sigma_{ij}^2\right)^{\frac{1}{N\times N}}}{\frac{1}{N\times N}\sum_{i,j=0}^{N-1} \sigma_{ij}^2} = \alpha \tag{2.42}$$

if

$$\frac{\epsilon_0^2}{\epsilon_{TD}^2} 1 \tag{2.43}$$

which is a reasonable approximation in the case of logarithmic quantization [2], then,

$$\alpha \leq 1 \tag{2.44}$$

with equality if, and only if, all the variances of the coefficients are equal.

Transform Coding Gain

Based on the above analysis, it can be concluded that,

1) The optimal transform minimizes the geometric mean of the transform coefficient variances and keeps the arithmetic mean constant (equation 2.26).

2) The optimum bit allocation results in an average reconstruction error variance proportional to the geometric mean (equation 2.41), instead of the arithmetic mean, of the transform coefficient variances.
The gain due to transform coding [2] is just the reciprocal of α, i.e. the ratio of the arithmetic mean to the geometric mean of the transform coefficient variances.

Other transform coding techniques

A transform domain technique which is gaining importance is the wavelet approach. Wavelets are a technique for multiresolution representation of image data [15, 16]. At different resolutions, the details of an image generally characterize different physical structures of the scene. This hierarchical framework is very useful in understanding image information. Wavelets are very useful in pattern recognition. They have also been successfully applied in image compression with good results [15, 16].

2.3.2 Vector quantization

Rate distortion theory [4] states that better performance can be achieved by coding vectors instead of scalars. Vector quantization(VQ) design techniques and various VQ structures have been developed recently [19]. In VQ a block of input data is quantized together as opposed to scalar quantization where an unit of data is treated seperately. In VQ, the input vector V, drawn from an M-dimensional Euclidean space is mapped onto a finite set of reproduction vectors. In other words, the input vector V is represented by $\hat{V}$, the vector quantized version of V. This can be represented as:

$$q(V) = \hat{V} \tag{2.45}$$

where $q(.)$ represents the quantization operation. VQ techniques are of many types, varying in their complexity and performance [19]. The basic principle of all VQ

techniques remains the same; grouping a block of data into a finite number of vectors. Instead of transmitting the actual elements, it is sufficient to transmit a label pointing to the nearest codeword vector and the reconstruction process is a simple lookup table operation.

VQ is a generic technique which can be combined with other techniques such as transform coding to improve the coding performance [19].

2.4 Gabor Decomposition

As in transform coding, Gabor decomposition transforms the input image into a frequency domain representation. It is different from transform coding in the sense that it transforms the input image into a joint spatial-frequency domain. It has been shown that image representation in the visual cortex involves both spatial and frequency variables [58]. In addition, both the measured receptive fields and the frequency tuning curves of the visual system conform closely with the functional form of Gabor elementary signals. Hence Gabor decomposition is an attractive tool for pattern recognition, image analysis and image compression.

The Gabor elementary functions (GEF) are a set of Gaussian functions that are modulated by complex sinusoids. In Gabor decomposition, the signal is represented in terms of a series of modulated Gaussian windows. Figs. 2.1, 2.2 and 2.3 show some of the basis functions of Gabor decomposition and DCT. From these figures, we can see that compared to the basis functions of DCT, the Gabor functions are very smooth and taper down gradually. This results in a considerable reduction in the blocking effect and therefore a higher subjective quality of the reconstructed image.

The main advantage of using Gabor elementary functions (GEFs) is their ability in achieving the lowest bound on the joint entropy of data in the joint spatial-frequency domain. They are capable of exploiting the image data correlation and are also well

suited for multiresolution pyramidal schemes. The main drawback of these GEFs is the non-orthogonality of the basis functions. Thus an analytical solution does not exist. Daugman [59] has proposed a neural network architecture for calculating the coefficients of decomposition. This method suffers from being iterative in nature. Ebrahimi *et al.* [60] have proposed a method based on the least squares approximation for computing the coefficients of decomposition. In this approach, the operation of Gabor decomposition is reduced to a matrix multiplication operation.
We now present the formalism of one-dimensional and two-dimensional Gabor functions.

2.4.1 1-D and 2-D Gabor functions

In the 1-D case, the Gabor function can be expressed as:

$$g_{m,n}(t) = \hat{g}_D(t - mD)\ exp(jn\Omega t) \tag{2.46}$$

where

$$\hat{g}_D(t) = exp[-\pi(t/D)^2] \tag{2.47}$$

where $\Omega \leq \frac{2\pi}{D}$.
The parameters D and Ω determine the scale of this modulated Gaussian in the time and frequency domains, respectively. The indices m and n specify the spatial and frequency locations, respectively. The Fourier transform of a Gabor function can be expressed as:

$$G_{m,n}(\omega) = \frac{\sqrt{2\pi}}{\Omega}\hat{g}_\Omega(\omega - n\Omega)exp[-jmD(\omega - n\Omega)] \tag{2.48}$$

When the scaling parameters D and Ω are chosen such that $\Omega < \frac{2\pi}{D}$, the representation is noncritical. It contains some redundancy in the spatial-frequency information. When $\Omega = \frac{2\pi}{D}$, there is no redundancy in representation and is the critical case. This case is of interest from the perspective of image compression and hence will be

considered in this thesis. However, for applications such as image analysis, noncritical formulations may be of interest [17, 18].

The 2-D seperable Gabor function can be expressed in the spatial domain as:

$$g_{m1,n1,m2,n2}(k_1, k_2) = g_{m1,n1}(k_1) g_{m2,n2}(k_2), \tag{2.49}$$

where $g_{m,n}(k)$ is the 1-D Gabor function.

2.4.2 Pyramidal Gabor functions

We note that the central frequencies (m,n) have octave relationships [57, 59]. Hence, coding an image based on logarithmic scaling along the frequency axis (i.e., using octave relations instead of harmonic) results in a pyramidal Gabor scheme. This scheme is similar to the conventional pyramidal representation in the sense that the elementary functions have similar shapes but vary in their effective widths. The elementary functions of order (m,s) are given by:

$$h_{m,s}(t) = \begin{cases} \hat{g}_D & for\ s = 0 \\ \hat{g}_D[2^{|s|-1}(t - 2^{-|s|+1} mD] & for\ s \neq 0 \end{cases} \tag{2.50}$$

where

$$n(s) = \frac{3 * 2^{|s|-1}}{2} sgn(s)\ for\ s = -N, \cdots, -1, 0, 1, \cdots, N \tag{2.51}$$

and $\hat{g}_D(.)$ is given by equation 2.47. We note that these pyramidal functions are also optimally localized in the joint spatial-frequency domains.

2.4.3 Discrete Gabor functions

In the discrete case, the relationship between the 1-D sampled signal $\{f(k)\}$ and its discrete expansion into a weighted sum of elementary functions $\{g_l(k)\}$ can be expressed in matrix notation as:

$$\mathbf{f} = \mathbf{G} \cdot \mathbf{x}, \tag{2.52}$$

where

$$\mathbf{f} = \begin{bmatrix} f(0) \\ \vdots \\ f(M-1) \end{bmatrix}, \mathbf{G} = \begin{bmatrix} g_0(0) & \dots & g_{N-1}(0) \\ \vdots & \ddots & \vdots \\ g_0(M-1) & \dots & g_{N-1}(M-1) \end{bmatrix}, \mathbf{x} = \begin{bmatrix} x(0) \\ \vdots \\ x(N-1) \end{bmatrix} \quad (2.53)$$

The 2-D seperable Gabor expansion can be similarly expressed as:

$$\mathbf{F} = \mathbf{G}_1 \cdot \mathbf{X} \cdot \mathbf{G}_2^{\mathrm{T}}, \quad (2.54)$$

where $\mathbf{F}$ is the image data matrix and $\mathbf{X}$ is the array of Gabor coefficients.
We note that the transform matrices $\mathbf{G}_1$ and $\mathbf{G}_2$ are banded matrices. For example, a 4×4 transform matrix is given as:

$$\mathbf{G}_1 = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{bmatrix} \quad (2.55)$$

This matrix has a bandwidth $p = 3$ where the bandwidth is defined as the maximum number of non-zero elements in any row / column of the matrix.

2.4.4 Inverse Gabor decomposition

We note from section 2.4.3 that:

$$\mathbf{F} = \mathbf{G}_1 \cdot \mathbf{X} \cdot \mathbf{G}_2^{\mathrm{T}}, \quad (2.56)$$

The original image, F, can be computed from the Gabor coefficients, X if G_1 and G_2 are known. Since the GEFs are not orthogonal, an analytic solution for the decomposition is not always possible. This means that the matrices G_1 and G_2 are not invertible.

Several solutions have been proposed for the evaluation of the Gabor coefficients [59] - [64]. The use of biorthogonal functions [62] provides an analytical method of finding Gabor coefficients. In this approach, a biorthogonal function $\gamma(t)$ is first defined which is then used to compute the coefficients of decomposition. This method defines an irrational number, L_0 as a normalization constant. The problem then reduces to that of finding a suitable aproximation to L_0 [64]. This method suffers from the limitations imposed on the values of the scaling parameters D and Ω. Hence, it does not necessarily permit the choice of optimal scaling parameters. Zak-transform [63] can be used to compute Gabor coefficients. This technique is more efficient and achieves higher precision in the resulting coefficients. It is also a fast method for the computation of Gabor coefficients. However, Zak-transform suffers from the instability of coefficients as it can result in division by zero in some cases. Hence it is limited in its use. An iterative approach employed to compute the coefficients needs a special purpose architecture for execution which is detailed in section 2.5. A powerful technique, using least mean squares approximations is proposed in [60]. This technique uses singular value decomposition (SVD) for computing the approximate inverses of the matrices G_1 and G_2. It is computationally tractable and is very robust. This technique is employed in this thesis.

2.5 Image Coding Architectures

The non-availablity of economical real-time codecs is a limiting factor in the widespread use of interactive image communications. Recently, the design of low cost compression and decompression hardware is receiving considerable attention from both the academia and industry. Some powerful architectures have been proposed for efficient algorithms like vector quantization [27, 28].

In this section, we review some of the architectures that have been proposed. We will

review some VLSI implementations for the JPEG-DCT, H.261 and MPEG standard algorithms and some of the proposed implementations for Gabor decomposition.

2.5.1 Codecs for the image coding standards

Recently, chipsets and codec products were offered for the JPEG-DCT, H.261 and MPEG standard algorithms. Among the first chipsets offered for the JPEG-DCT and H.261 algorithms were those by LSI Logic Corp [23]. For H.261 and MPEG, LSI Logic has a seven-chip set, one chip for each functional block such as DCT, VLC, motion estimation estimation, inter/intra frame decision. Graphics Communications America has a twelve-chip set for the H.261 [21]. AT&T Bell Laboratories has recently proposed a chipset which executes the MPEG and H.261 algorithms along with JPEG-DCT [21]. The chipset comprises of 4 chips; two different encoder options, one decoder, and one system controller. Unlike the previous designs where each module of the algorithm were on a seperate chip, each chip in this set is more comprehensive in its function. Pictel offers a codec for H.261 and the JPEG-DCT algorithms.
All the chipsets can be used effectively to build systems that fit onto an expansion slot on a personal computer. These systems are powerful real-time implementations of the standard coding algorithms (JPEG-DCT. , MPEG, H.261, etc.). However, for the next generation of coding techniques, these implementations may not always provide real-time performance [21].

2.5.2 Implementations of Gabor decomposition

Recently two implementations of Gabor decomposition have been proposed in literature [59, 65]. One implementation is based on a three layered neural network and the other is an associative string processor (ASP).

Since Gabor decomposition does not have an analytical solution and usage of biorthogonal functions [58] may restrict the conjoint sampling rates, an iterative solution offers more flexibility in terms of the choice of the Gabor parameters and hence good compression. This issue is addressed in the neural network model [59]. This network consists of three layers with the upper and lower layers having fixed weights and the middle layer capable of adjusting its weights. This neural network is shown in Fig. 2.4. The image $I(x,y)$ is an $N \times N$ image whose Gabor coefficients are a_i and the basis functions are $G_i[x,y]$. Since $G_i[x,y]$ set of functions are not orthogonal, the solution for a_i is not simple. If, $H(x,y)$ is such that:

$$H(x,y) = \sum_{i=1}^{n} a_i G_i[x,y] \tag{2.57}$$

then we know that a_i's are an approximation of the Gabor coefficients of $I(x,y)$ if

$$E = \sum_{x,y} (I(x,y) - H(x,y))^2 \tag{2.58}$$

is minimized. This minimization is carried out by the neural network in an iterative manner. The technique is cumbersome and may take a long time to converge to the final values of the Gabor coefficients. In addition, it is difficult to implement the neural network architecture in VLSI.

The associative string processor implementation for Gabor decomposition is a massively parallel multiplier. Since Gabor decomposition can be computed approximately using Singular Value Decomposition, the task of finding Gabor coefficients can be looked upon as a matrix multiplication operation [60]. In order to compute Gabor coefficients using matrix multiplication in real-time, the ASP has one processor per pixel of image data. A section of the ASP is shown in Fig. 2.5. The ASP module is a low-MIMD/high-SIMD parallel processing structure of intercommunicating ASP substrings, each supported with a data buffer, a controller and a data communications network. An ASP substring (Fig. 2.5) is a fine grain SIMD massively parallel

processor, incorporating a string of identical associative processing elements (APEs). The LKL and LKR (link left and link right) ports of a substring can be used to link up with similar substrings to form a longer ASP module.

The ASP is based on the content-addressable paradigm. That is, each APE in the ASP is located by its content rather than by a physical location address. Two types of Inter-APE communications are supported: a bit-parallel single APE communication via the shared data bus and a bit-serial multiple APE communication via the inter-APE communications network. The inter-APE communications network permits dynamic reconfigurability to emulate standard network topologies such as array, tree, graph, shuffle-exchange and butterfly connections. An unlimited modular extension of this topology is achieved via the LKL and LKR ports. The vector data buffer optimizes the efficiency of the input-output data transfer, by supporting a dual port exchange of vector data (output sent and input loaded in one operation) with alternating primary and secondary data exchange as shown in Fig. 2.5. There are two forms of primary data exchange (PDX) between the vector data buffer and the APEs, bit-parallel via the data bus and bit-serial via the inter-APE communication network. The secondary data exchange (SDX) provides a bit-parallel vector data exchange between the vector data buffer and the ASP data buffer (ADB) at a lower data rate.

In this structure, Gabor decomposition is implemented as an outer product algorithm [26]. A matrix multiplication requires n^3 multiplications and $n^2(n-1)$ additions. Hence, maximum parallelism can be achieved by using n^3 processors. Such an implementation leads to a very large number of processors, even for small values of n. Moreover it is a very inefficient implementation of matrix multiplication, especially for sparse matrices such as Gabor kernels. The ASP structure uses n^2 processors and hence offers a reduction of an order of n processors. Each image pixel is stored in one APE of the ASP module. The Gabor kernel is then passed over the ASP module. The outer product algorithm exploits the inherent parallelism in matrix multiplication and

the ASP module performs Gabor decomposition in n iterations.

In spite of the reductions in complexity by one order of magnitude, this design is very expensive in terms of the number of APEs used. It requires n^2 APEs to process an $n \times n$ image. Hence, this design is difficult to implement. However, the ASP is a massively parallel matrix multiplier, and hence is a more general design compared to the neural network. It can be used for a variety of matrix multiplication problems including Gabor decomposition. The primary drawback with the ASP architecture is that cannot be implemented as a codec for image / video compression applications.

2.6 Summary

In this chapter some basic principles of information theory, namely the concepts of rate distortion theory and entropy were reviewed. Objective error measures which will be used in various parts of the thesis were defined. Next, a brief review of different image coding techniques was presented followed by a review of Gabor decomposition. Finally, some architectures for image coding and Gabor decomposition were reviewed.

Symbol	Probability	Range
a	0.2	$[0, 0.2)$
e	0.3	$[0.2, 0.5)$
i	0.1	$[0.5, 0.6)$
o	0.2	$[0.6, 0.8)$
u	0.1	$[0.8, 0.9)$
!	0.1	$[0.9, 1.0)$

Table 2.1: Arithmetic coding: allocation of range to source symbols

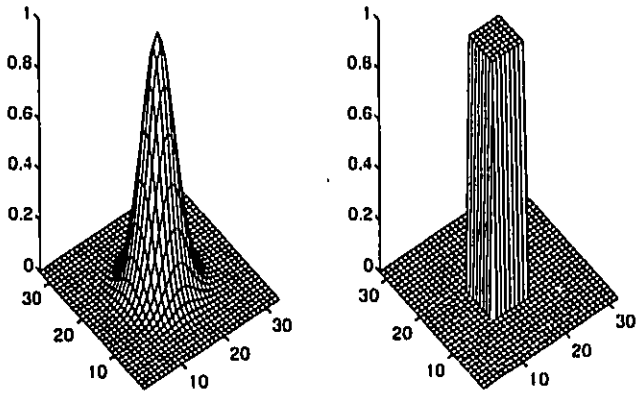


Figure 2.1: Gabor and DCT basis functions of order 0,0. The mesh plot on top is the Gabor basis function and the plot on bottom is the DCT basis function

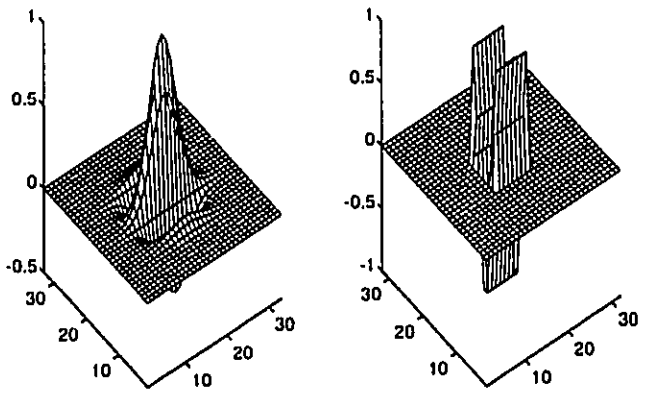


Figure 2.2: Gabor and DCT basis functions of order 2,0

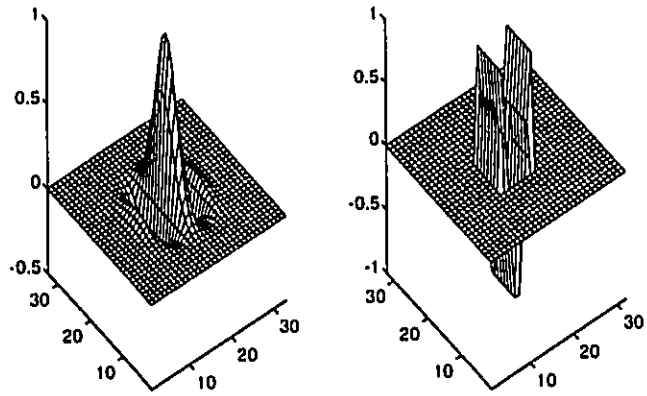


Figure 2.3: Gabor and DCT basis functions of order 0,2

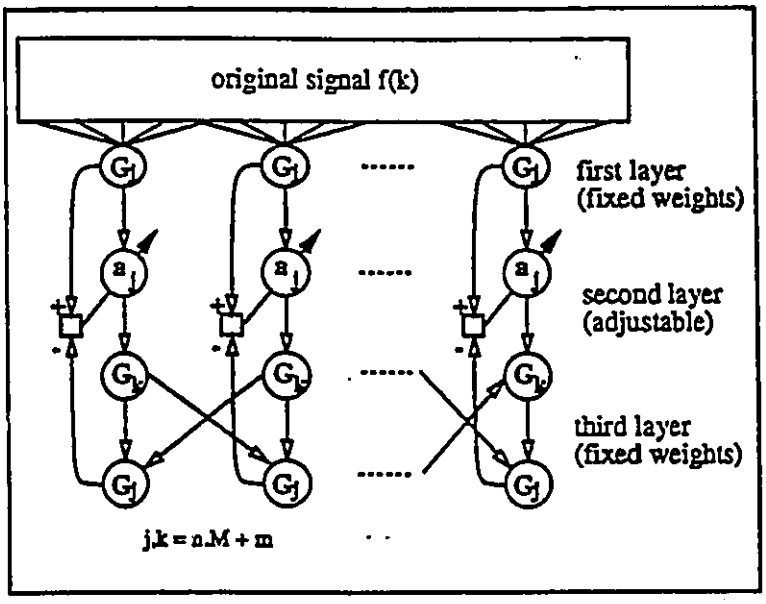
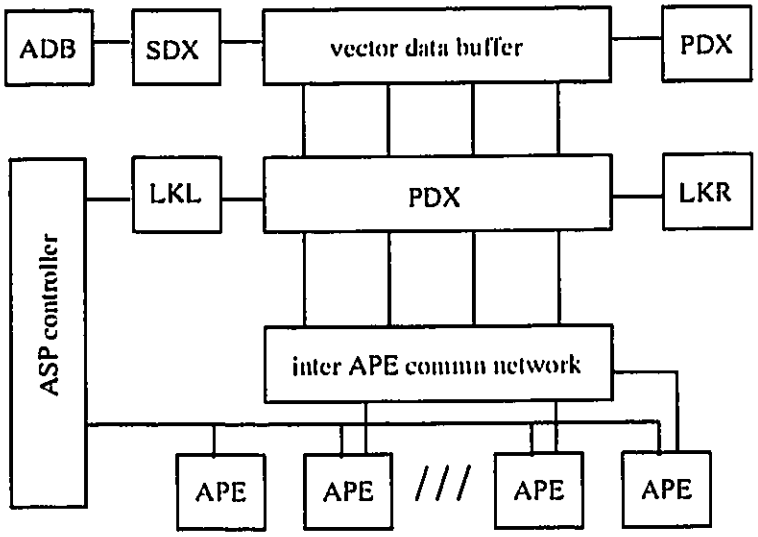


Figure 2.4: Neural network for Gabor decomposition



SDX: Secondary data exchange
ADB: ASP data buffer
PDX: Primary data exchange
LKL/LKR: Link Right and Left registers
APE: Associative Processing Element

Figure 2.5: Associative string processor substring

Chapter 3

Review of Progressive Image Transmission

In progressive image transmission a low resolution version of the image is first transmitted. Upon viewers' request, the resolution is further enhanced interactivey until the desired fidelity is achieved.

In PIT [29] - [50], the image is first reorganized and then transmitttted progressively, where the reorganization is either in the spatial or transform domain. In the spatial domain, progressive image transmission can take place either directly on the image pixels, such as in the bit plane and subsampling, or on the preprocessed data, such as in the pyramid techniques. The transform domain techniques can be classified into coefficient scanning, bit slicing and S-transform. The progressive image transmission techniques are reviewed below.

3.1 Direct spatial techniques

A digital image can be divided into a set of bitplanes as shown in Fig. 3.1 where the ith bit plane corresponds to the ith significant bit of the image pixels. Progressive image transmission is achieved by sending the bit planes successively starting with the most significant bit plane. This is a very simple method and requires no additional information to be transmitted. This technique achieves very little compression.

Another direct spatial technique is the progressive subsampling approach. In this method, the image pixels are subsampled and then progressively transmitted. At the receiving end, the missing pixels are interpolated using a variety of means such as bicubic splines, bilinear interpolation, cubic convolution etc. However, this technique does not yield good performance [38].

Residual error vector quantization is a high performance direct spatial technique [48]. In this technique, a residual error image is formed at each stage but subtracting the reconstructed image at the current stage with the original image. The residual error images are iteratively fedback and then requantized. This technique achieves a high performance compared to other direct spatial techniques because of its iterative feedback and vector quantization of the error image. The main drawback of this technique is the computational overhead. It is difficult to implement in real-time, and needs special purpose architectures.

3.2 Spatial pyramidal techniques

Pyramid data structures are a class of hierarchical representation of images. This data structure has found an important role in the fields of image processing, computer graphics, computer architectures, etc. For an image X of size $2^n \times 2^n$, a structure called pyramid can be defined on this image as a sequence of matrices $\{X_j\}$ such that

X_{j-1} is a lower resolution version of X_k. In the limiting sense, X_0 is a single pixel. Since each level of the pyramid represents different resolutions of the original image, progressive image transmission can be achieved by transmitting the different levels in each iteration. We now review different pyramidal structures for progressive image transmission. A quadtree pyramid is illustrated in Fig. 3.2.

3.2.1 Quadtree

A quadtree representation of an image can be formed as follows: Given an image of size $2^n \times 2^n$, a pyramid data structure can be formed by successively operating over 2×2 neighbouring pixels [30, 33, 40, 41, 43]. In other words, the value of a node at level j is a function of the values of 2×2 neighbouring nodes at level $j+1$. The function can be minimum, maximum, mean, median, mode, sum, etc. As each node at level $j-1$ is associated with only the four siblings at a lower level j, this data structure is called the quadtree. For example, we can form a quadtree pyramid with the function defined as a truncated mean i.e. the mean value does not require more precision compared to its four siblings. It can be seen that in such a structure there is a 33.3% additional overhead in representing the image data. The additional overhead is incurred because of redundancies in the representation of data using the quadtree structure. A method of reducing the redundancy of the quadtree is the reduced quadtree in which only 3 of the four siblings are transmitted. The fourth sibling can be calculated from the parent and 3 siblings. Further reductions can be achieved by exploiting the significant amount of correlation between the siblings. An efficient progressive image transmission scheme based on the quadtree structure is the vector quantization in quadtree form [30]. In this technique, a mean quadtree is first formed from the image. From this mean pyramid a difference quadtree is formed in which the difference between the parent node and the siblings are stored. The

difference quadtree has a lower first order entropy than the mean pyramid. Hence it can be coded more efficiently. Once this difference pyramid has been formed, vector quantization is applied seperately to each level of the pyramid. Thus an efficient coding performance with robustness to errors is achieved. The errors from one level of the pyramid are corrected at the next level and hence provide a good error delivery mechanism. This makes it a very elegant approach.

3.2.2 Nonuniform Quadtree

In general, the information contained in an image is not uniformly distributed over the entire image canvas; for example, natural images contain regions which are uniform and others which are textured. This can be exploited to obtain further reductions in the data to be transmitted. A nonuniform decomposition applied to the image yields a non-uniform set of quadtree pyramids [39]. A set of thresholds, $P_i : P_i > P_{i+1}$, called *transmission pass*, is first chosen. The top left pixel of the image is taken as the root of the quadtree. The difference between the maximal and minimal pixel values of the image is compared with the first transmission pass, P_1. If the difference is less than P_1, then no decomposition is done. If not, the image is divided into four subimages of equal size. For each subimage, the difference is once again computed with P_1 as the transmission pass. This process is repeated until subimages of 1 pixel are reached or no further decomposition can be done. This forms a non uniform quadtree and corresponds to an approximation of the image.

The same procedure is now repeated for other transmission pass values and a number of quadtrees are produced. Since $P_i > P_{i+1}$, the resolution of the transmitted image increases as the index, i, of the quadtree increases. Furthermore, the lower resolution quadtree is contained in the higher resolution quadtree. Progressive image transmission is achieved by transmitting quadtrees with increasing indices i.

The advantage with this approach is that the high detail areas appear first and thus make image identification easier. But there is a high computational and transmission overhead in terms of the node information to be sent to the receiver. In addition, the optimal transmission parameter value selection is an open problem.

3.2.3 Filtered pyramids

Burt and Adelson have proposed a more general method to form a pyramid [40]. In this approach, the original image $X = G_0$ is first convolved with a Gaussian-like weighting function, equivalent to low-pass filtering the image. The low-pass filtered image, G_1, is subtracted from G_0 resulting in the difference image L_0,

$$L_0 = G_0 - G_1. \tag{3.1}$$

Since L_0 is largely an uncorrelated image, fewer bits per pixel are required to represent L_0 compared to the original image G_0. Moreover, G_1 is a low-pass filtered version of the original image and hence can be represented using fewer spatial samples. Hence, an overall reduction in the data rate to be transmitted is achieved. By repeating this process a sequence of pyramids called the Gaussian and the Laplacian are formed. These pyramids are general cases of the sum and difference pyramids, respectively.
A variation of the Burt and Adelson's method is the filtered pyramid using two dimensional Quadrature Mirror Filters (QMF) [42]. To construct a 2-D QMF, a 1-D QMF is applied sucessively in the horizontal and vertical directions, respectively. The process of filtering with a 2-D QMF gives one Gaussian (low pass in both directions) and three Laplacian images. This recursive filtering results in a set of Gaussian and Laplacian images. Only the top Gaussian image and the rest of the Laplacian images are needed to be transmitted to reconstruct the original image. The wavelet decomposition is similar to the QMF filtering operation. In wavelet decomposition, the filter is a more general filter. Hence QMF filtering is a subset of wavelet decomposition.

To encode these Laplacian pyramids, techniques such as run-length coding, vector quantization, etc. are used [40, 42].

3.3 Transform domain techniques

In the transform domain techniques mentioned in section 2.3.1, the image is transformed by an orthogonal transform. The image transformation can be considered as a decomposition of the image data into a generalized two dimensional spectrum. Each coefficient in the transform domain corresponds to the amount of energy of the *spectral* function within the image. Typically, the first (DC) coefficient represents the average brightness of the image block. In most of the natural images the energy is concentrated in the lower order coefficients. Hence, a reasonable approximation of the image can be reconstructed using only a few lower order coefficients. This image can be further refined by the addition of higher order coefficients. The proposed transform domain techniques for progressive image transmission can be classfied as coefficient scanning, bit slicing and S-transform approaches.

3.3.1 Coefficient Scanning

In the coefficient scanning approach for progressive image transmission, the coefficients are transmitted in a prespecified order, usually from lower to higher since the lower order coefficients contain most of the image energy. A low resolution image is obtained by inverse transforming a few lower order coefficients. This low resolution image is then improved progressively by further transmission of higher order coefficients. With addition of more coefficients, the resolution is improved until an almost lossless reconstruction is achieved.

In the coefficient scanning approach, it is very important to choose the order in which

the coefficients are transmitted as it directly affects the performance of the scheme. One popular method is to reorder all AC coefficients in terms of the Normalized Mean Square Error (NMSE) of the reconstructed images. The reconstructed image for each AC coefficient is formed by inverse transforming the DC coefficient and the corresponding AC coefficient. The coefficients can also be reordered according to their entropy [46]. Another method, used in the JPEG-DCT algorithm, is the fixed ordering of coefficients [44]. In this algorithm, coefficients are scanned in a zig-zag order as shown in Fig. 3.3. Note that, as we traverse along the zig-zag scan, the coefficient energy generally decreases monotonically. Other popular scan patterns are the peano scan, peano-hilbert scan and similar space filling curves, etc [72]. These fixed patterns do not work well with all types of images. They are good for a broad range of images but may not be the best for any one type of image.

3.3.2 Bit Slicing

Tzou and Elnahas [46] describe a progressive image transmission scheme where all the transform coefficients are scalar quantized in a number of stages by using a set of incremental bit maps. Here, the total number of bits assigned to each coefficient is equal to the number of bits per pixel (in spatial domain). An example of the incremental bitmap is shown in Fig. 3.4, where the incremental bit rate per stage is 1 bit/coefficient. The sequence of bits allocated to each coefficient can be read from the corresponding locations in the map. This scheme of bit slicing can be viewed as slicing the full bit assignment map of the coefficients. The scheme has two disadvantages: there is no compression as each coefficient is eventually quantized at the same rate as the original image and the total number of bits assigned to each coefficient is the same, i.e there is no optimal bit allocation strategy.

3.3.3 S-Transform

The S-Transform is a hierarchical approach to represent images, and has been used for progressive image transmission [51, 52]. An image to be transmitted is first decomposed into spatially nonoverlapping blocks of 2×2 pixels. Hadamard transform is then applied to these blocks. The DC coefficients are assembled to form an approximation of the original image at 1/2 the resolution. The process is repeated until a single pixel remains, corresponding to the mean value of the image. At the receiver , a set of successive approximations are reconstructed in the reverse process of decomposition.

3.3.4 Multistage Vector Quantization

Multistage vector quantization is a technique in which the transform coefficients with higher energy are allocated more bits compared to the coefficients with lower energy [34]. In other words, it is an optimal bit allocation mechanism as the coefficients are assigned bits in accordance with their variances. In this technique, the image first undergoes a block orthogonal transform. An optimal bit allocation map is found at a fixed bit rate as outlined in [2]. The optimal bit allocation map is then sliced into a set of *bit allocation planes*, $\{B_{k,ij},\ k = 0, 1, \cdots,\}$ by applying a set of thresholds, $\{T_k\}$. Here $B_{k,ij}$ denotes the number of bits allocated to coefficient (i, j) at stage k. We note here that $B_{k,ij}$ is not restricted to integer values. At each stage, all the components with a non-zero number of bits allocated to the coefficient are combined to form a vector and then quantized. In other words, the transform coefficients are first vector quantized with the bit allocation plane B_0 and the residual errors are requantized with plane B_1 and so on. A seperate codebook is generated at each stage using a generalized Llyod clustering algorithm. This technique achieves high compression and is very efficient since it allows for fractional bit allocations.

3.4 Summary

We have reviewed the progressive image transmission techniques, in both the spatial and transform domains. Direct spatial techniques are simple but do not perform well unless powerful techniques such as vector quantization are used. Pyramidal approaches have a very comprehensive data representation structure and hence are well suited for progressive image transmission. Consequently, powerful techniques for PIT can be implemented using this approach.

Transform domain techniques offer high efficiency, since most of the energy in natural images is concentrated in a few low order coefficients. Some of the techniques for PIT in transform domian include coefficient scanning, S-transform and bit slicing. Pyramidal approaches can be combined with transform domain techniques to result in high performance PIT schemes.

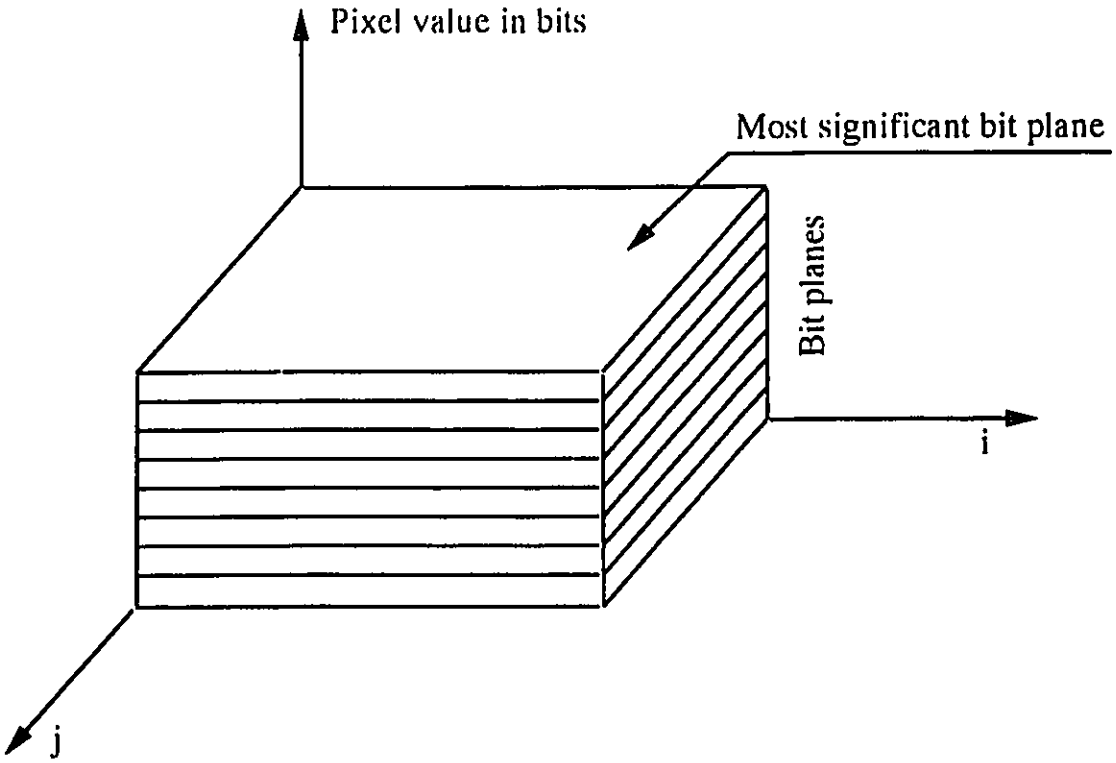


Figure 3.1: A Bit plane representation of an 8 bit image

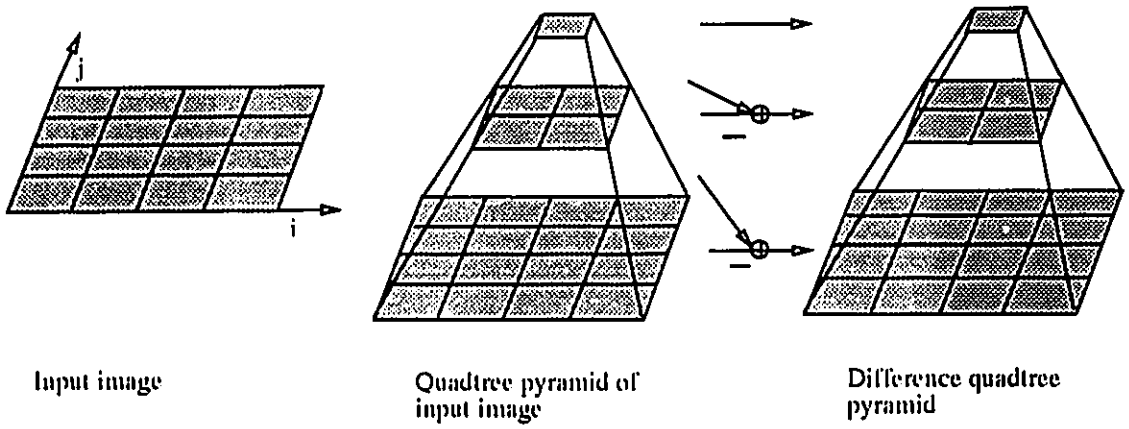


Figure 3.2: Example of quadtree pyramid structures

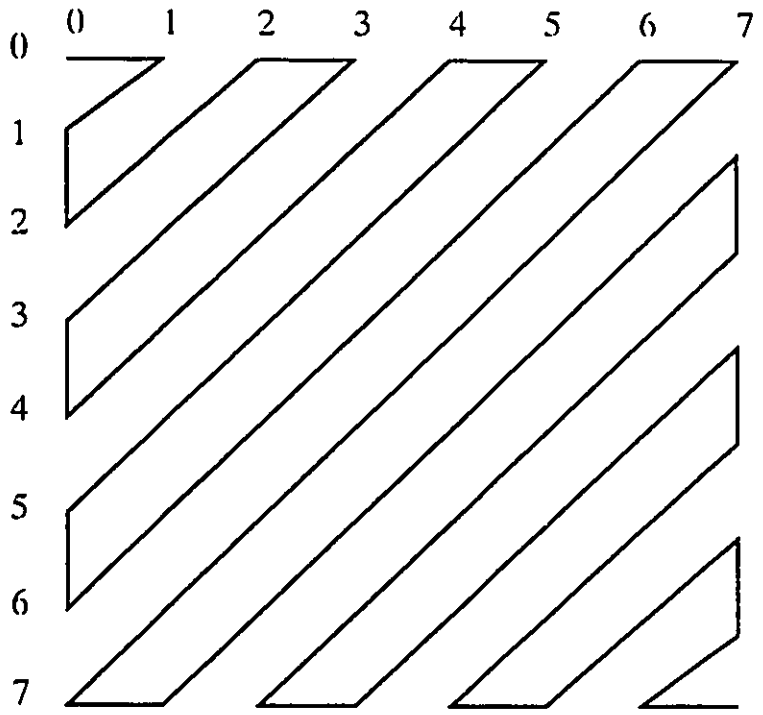


Figure 3.3: The zig-zag scan path of coefficients used in JPEG-DCT

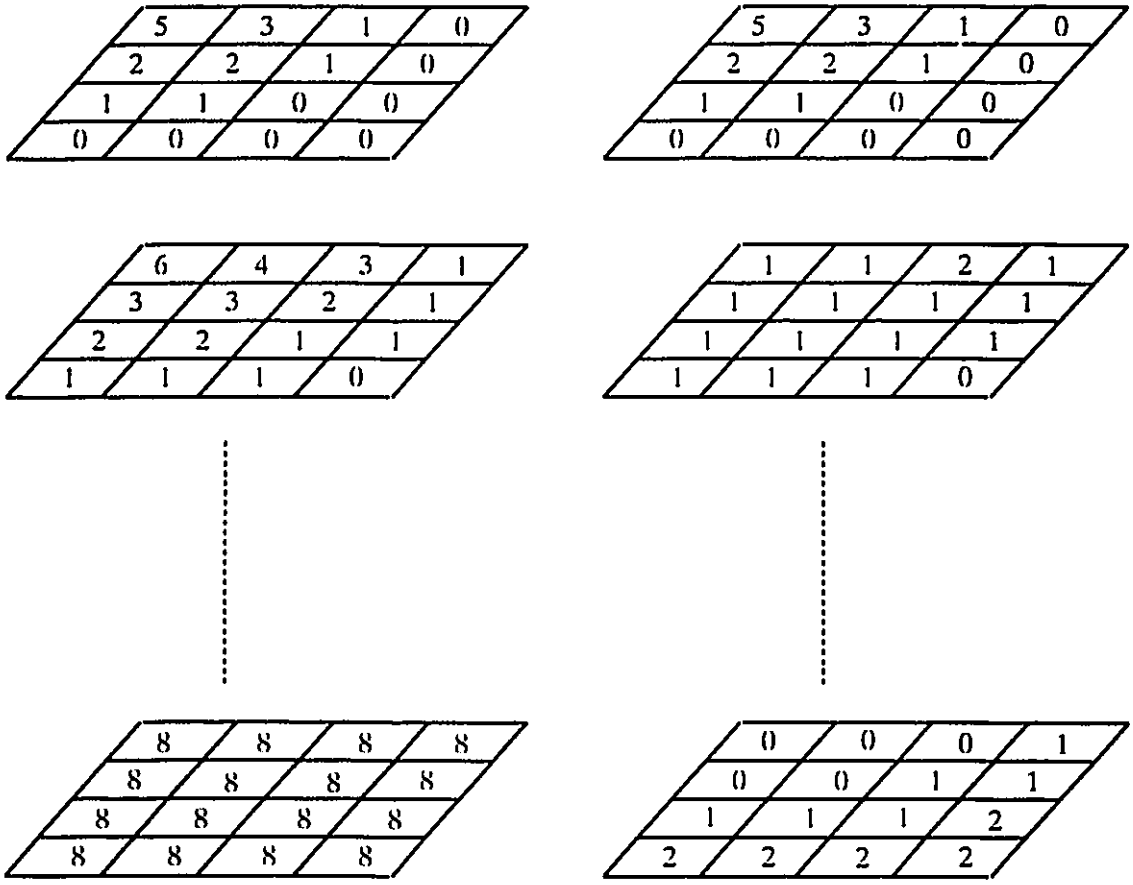


Figure 3.4: Bit maps (left) and incremental bit maps (right) for a 4×4 block and an incremental rate of 1 bits/pixel

Chapter 4

Evaluation environment for Progresive Image Transmission

Scientific visualization is an emerging field with applications in the areas of computer graphics, image processing, computer networks, digital video, user interface cognitive sciences, etc. It deals with representing data and information in such a way as to gain understanding and insight into the experimental observations [55]. It is as important, if not more so, to understand and present data in a user-friendly manner as it is to conduct an experiment. Visualization is thus a valuable tool for scientific research and investigation.

In this chapter, we present a software tool for progressive image transmission. This tool built using the *Khoros* software environment, offers a set of standard menus from which the user can choose image types, distortion criteria, coding schemes and bit rates. This makes possible fast and convenient evaluation of progressive image transmission (PIT) schemes.

In order to evaluate any image coding application, the evaluator has to be able to

perform simultaneous objective and subjective evaluations of the algorithms. The process is normally cumbersome because of the lack of a single platform from which the evaluation can be carried out. The researcher has to seperately evaluate the coding schemes for each of the distortion criteria. This renders simultaneous comparison extremely difficult. In this chapter, we present a software tool which offers a unifed platform for evaluating image coding schemes with special emphasis on PIT. This tool permits the researcher to choose image types, distortion criteria, coding schemes and bit rates. Hence this tool makes possible fast and convenient evaluation of different PIT schemes. It also illustrates the interaction between the image type, coding scheme and bit rate requirement. This tool does not require apriori knowledge of any of the algorithms and hence can also serve as a demonstration tool. In addition, the tool also offers an open development environment to which new algorithms can be added with ease. Under one platform, the tool permits simultaneous evaluation of two or more PIT schemes using both objective and subjective quality measures. It is an open architecture platform enabling addition of new modules with ease. Thus the proposed tool offers a simple, convenient and flexible platform for evaluating PIT schemes.

4.1 KHOROS Software environment

Khoros is a software development environment, developed at University of New Mexico, which runs on UNIX workstations and X windows [53, 54]. It was built to provide an effective environment for research in image processing by supporting rapid prototyping of applications. It provides a visual programming environment where applications can be connected as data flow graphs [54] and is hence a very intuitive user interface. Fig. 4.1 shows an example of the Khoros workspace. Khoros environment is also equipped with a library of signal and image processing routines which enable

the user to perform a variety of image processing tasks. In addition, Khoros also supports distributed processing for fast execution of compute intensive algorithms. Most importantly, it permits quick development of new applications and provides a generic structure upon which higher level applications can be built. We note that the underlying principle of open architecture makes it suitable for different applications. At present, Khoros is domain specific and is tuned to environmental monitoring and medical applications. It does not have an image coding library. Khoros comes with more than 250 standard image and signal processing routines and is available via anonymous ftp from University of New Mexico.

The visual programming environment of Khoros, cantata, is a dataflow engine. It represents the program as a dataflow graph and fires a routine as soon as all its inputs are ready. It exchanges information between modules via temporary files. Since, reading to and writing from temporary files can be slow, cantata slows down the execution of complex programs. Khoros also comes with a software development environment (SDE) which permits fast development of new algorithms and toolboxes. It has two automatic code generators which reduce the burden of designing a good user interface.

4.2 Software Tool for Progressive Image Transmission

We now present the software tool for PIT. This tool was built using the Khoros application software development environment. It offers a visual tool for evaluating the various PIT schemes. An example workspace of the tool is shown in Fig. 4.2.

4.2.1 Details of the Software

This tool offers three main menu selections. The first menu offers a choice of different types of images which can be used as inputs for the image coding algorithms. This menu is essential for proper evaluation of the coding schemes since the performance of the algorithms varies with images of different types. The second menu offers different algorithms that can be used for coding the selected image. In the current version of the tool, three high performance PIT schemes have been implemented; namely, the JPEG-DCT algorithm, a vector quantization based algorithm and an algorithm based on Gabor decomposition. We note that new algorithms can be added to this menu with ease. The steps to be executed for the development and installation of new algorithms are outlined in section 4.2.2. The third menu consists of various criteria for evaluating the image quality. For example, the current implementation allows for three objective and two subjective evaluation criteria. The three objective criteria are as follows:

- $Normalized\ mean\ square\ error(NMSE) = \frac{\left[\sum_{i=1}^{M}\sum_{j=1}^{N}(u_{ij}-v_{ij})^2\right]}{\left[\sum_{i=1}^{M}\sum_{j=1}^{N}(u_{ij})\right]}$ (4.1)

where u_{ij} is the original image pixel and v_{ij} is the reconstructed image pixel.

- $Peak\ Signal\ to\ Noise\ Ratio(PSNR) = 10\log_{10}(\frac{255\times 255}{MSE})$ (4.2)

where

$$MSE = \frac{1}{MN}\times\sum_{i=1}^{M}\sum_{j=1}^{N}(u_{ij}-v_{ij})^2 \tag{4.3}$$

- $Maximum\ Absolute\ Error(MAE) = \max(|u_{ij}-v_{ij}|)\ \forall i = 1,\cdots,M\ and\ j = 1,\cdots,N$ (4.4)

The subjective evaluation criteria that the toolkit offers are:

- Image difference: It offers both the normalized difference image and the absolute difference image. The normalized difference image is obtained by remapping the

difference image pixel values on a normalized scale from 0 to 255.

• Image zooming: It offers simultaneous interactive zooming of multiple images. This is very useful in observing distortions such as blocking introduced by the different coding schemes.

4.2.2 Development of new algorithms using the tool

We now explain the process of developing and installing new algorithms in the tool menus. In order to use the user-interface code generators that Khoros offers, a detailed user interface specification (UIS) file has to be generated. This UIS file contains the necessary information to generate the user interface widget. From the UIS file, the code generators create the program shell. The functionality of the program is then added to this shell. Once the new algorithm is tested and ready, it can be installed as outlined below:

- Choose one of the three menus where the algorithm fits best.
- Set the configuration file of that algorithm to point to the correct directories (one of the already existing configuration files may be used for guideline).
- Run the installation utility or copy the files manually.
- Add a widget specification for the program in the toolkit UIS.
- Write the manual files for future users.

4.3 Evaluation of PIT Schemes Using the Tool

We now show how the tool can be used for evaluating and demonstrating the various PIT schemes.

4.3.1 Evaluation of PIT Schemes

In order to evaluate any image coding scheme, the researcher has to choose images of different types as input to the coding algorithm. This can be done by selecting the proper menu item from the images menu. An icon (a glyph) corresponding to the image appears in the workspace. This glyph is then connected to the glyph of the coding algorithm. The user then chooses one or more of the desired objective and subjective quality measures. Hence, a block diagram representing the sequence of actions to be executed is created step by step. Since this is a visual representation, it is very intuitive and easy to implement. The visual representation of the block diagram is as shown in Fig. 4.1. The user executes the signal flow graph by a simple mouse click on the tool. The tool operates on the data flow concept and fires the routines when possible. It also gives visual feedback by highlighting the blocks that are being currently executed.

4.3.2 Usage of the tool

The evaluations of the algorithms proposed in this thesis have been carried out using this tool. In this section, we present an illustrative evaluation. The JPEG-DCT baseline algorithm is powerful and performs well for a variety of image types. In the JPEG-DCT algorithm, the input image is first transformed using DCT. The transform coefficients are then quantized using a perceptually weighted quantization matrix. This retains only the most important transform coefficients and the majority of the coefficients tend to be zero. In order to achieve a good compression, a zig-zag scanning of the quantized coefficients is employed followed by run-length and Huffman encoding. In the JPEG-DCT algorithm, progressive transmission is achieved using the coefficient scanning methodology. Simulations using the JPEG-DCT algorithm were carried out on the Lenna image of size 512×512 pixels, quantized to 8 bits per

pixel. The results are reported in Table 4.1 and the corresponding rate distortion curve is plotted in Fig. 4.3. The original image, JPEG-DCT encoded image and the respective zoomed images are shown in Fig. 4.4. The corresponding normalized diffrerence image is shown in Fig. 4.5.

It can be seen from Fig. 4.5 that most of the edge information is not transmitted by the JPEG-DCT algorithm. Since edge information correspond to high frequencies, we conclude that the JPEG-DCT passes the low frequencies in the image and blocks the high frequencies. Since most of the energy in natural images are in the lower frequencies, high performance is achieved if low frequencies are transmitted with higher fidelity. Perceptually, the human eye is more sensitive to low frequencies and can tolerate coarse quantization in high frequencies. This fact is exploited in the design of perceptually weighted quantization matrices in the JPEG-DCT algorithm. The interactive zoom utility of the tool was utilized to zoom the eye region of the Lenna image. In addition, the zoomed JPEG-DCT image shows blockiness. In JPEG-DCT, the image is split into blocks and each block is encoded independent of the others. Hence, JPEG-DCT exhibits blockiness which is clearly apparent at lower bit rates.

4.4 Summary

In this chapter, we presented a software tool for progressive image transmission. This tool provides a convenient platform for performing evaluations and developing new algorithms for PIT schemes. We note that this tool presents a convenient visual interface to the user and hence is very intuitive to use. This tool could also be used as a demonstration tool for image coding schemes. Hence, this tool is a convenient means of performing evaluations of new algorithms.

An example evaluation of the JPEG-DCT was illustrated using this tool. Effects such

as blockiness were observed using the zoom facility of the tool.

- UNIX is a trademark of AT&T Bell Labs
- Khoros is a trademark of the University of New Mexico
- X Window System is a trademark of MIT.

Table 1

bpp	PSNR(dB)
0.09	19.7
0.25	20.75
0.51	21.85
0.73	23.14
0.92	24.00
1.21	26.17

Table 4.1: Simulations results for the Lenna image using JPEG-DCT algorithm

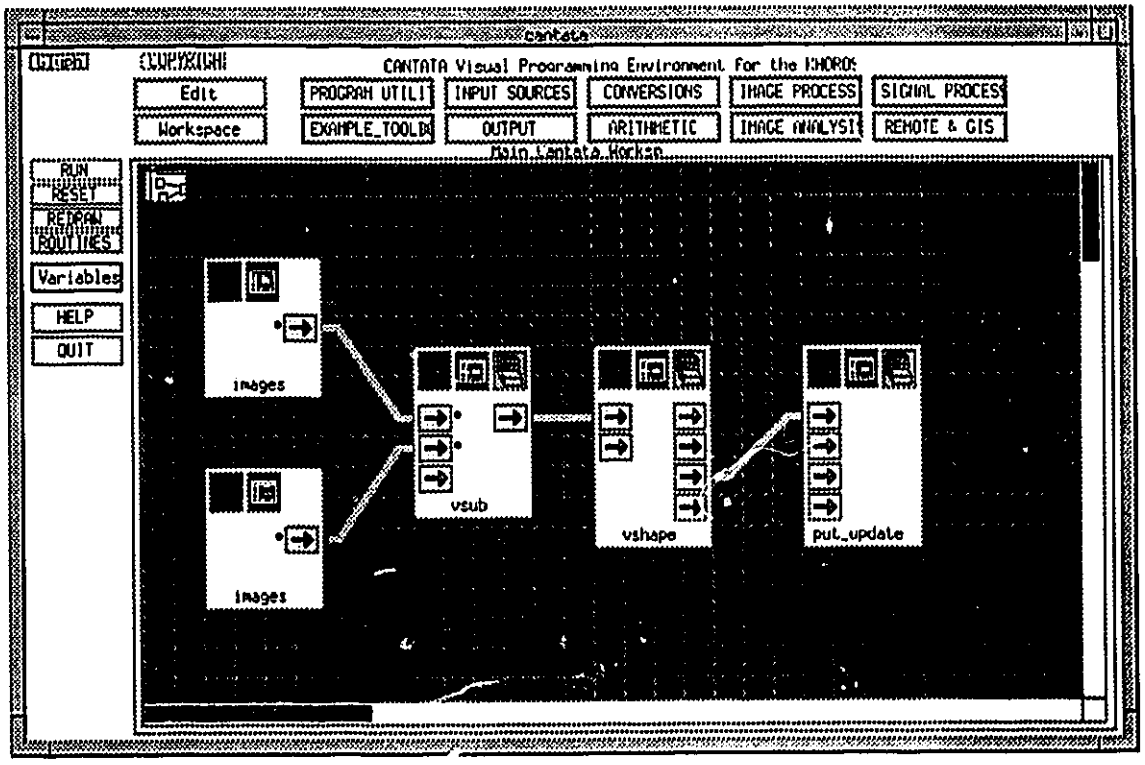


Figure 4.1: An example of the Khoros workspace

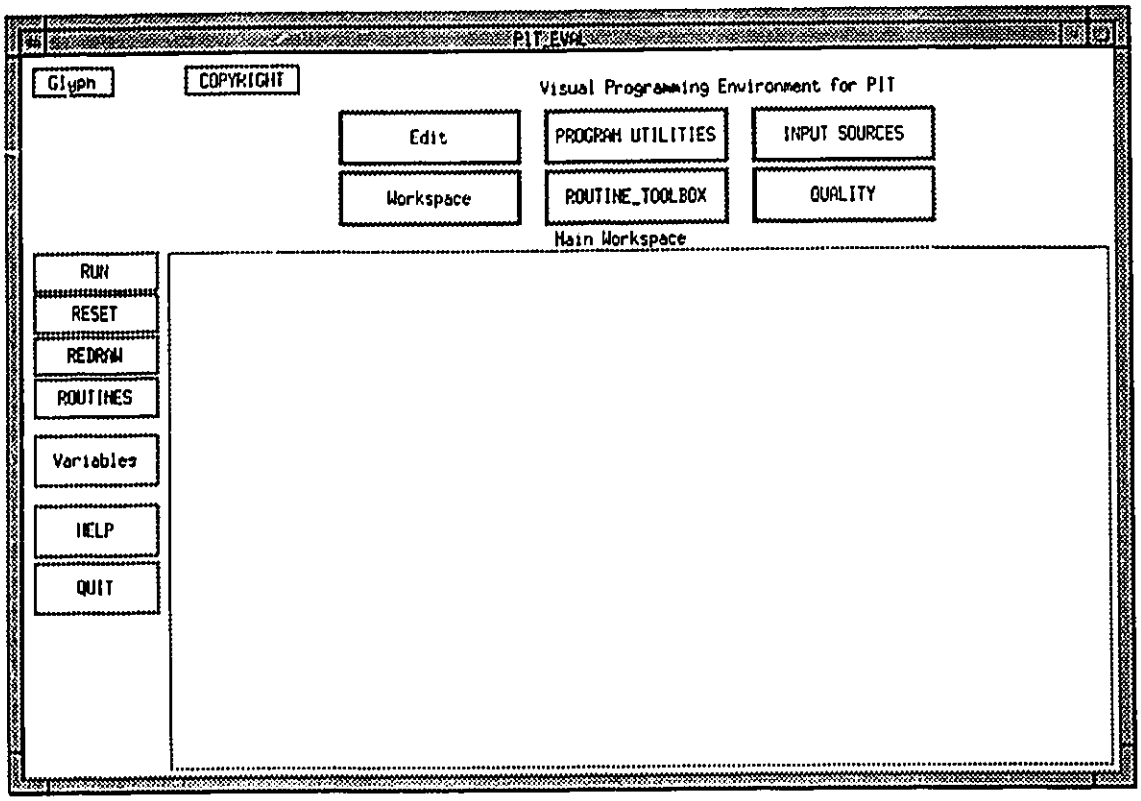


Figure 4.2: The PIT tool workspace

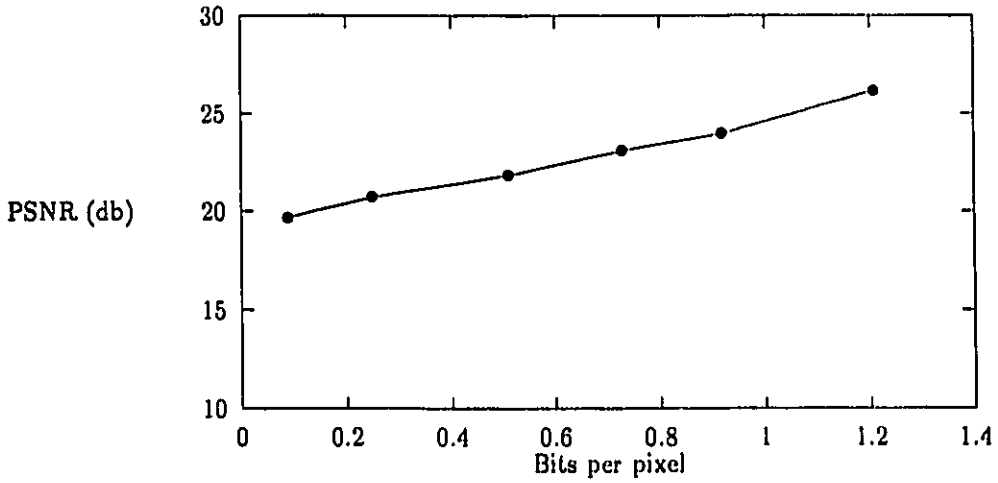


Figure 4.3: Rate distortion curve (PSNR vs bit rate) for JPEG-DCT encoded Lenna image

Figure 4.4: An example of the interactive zooming operation

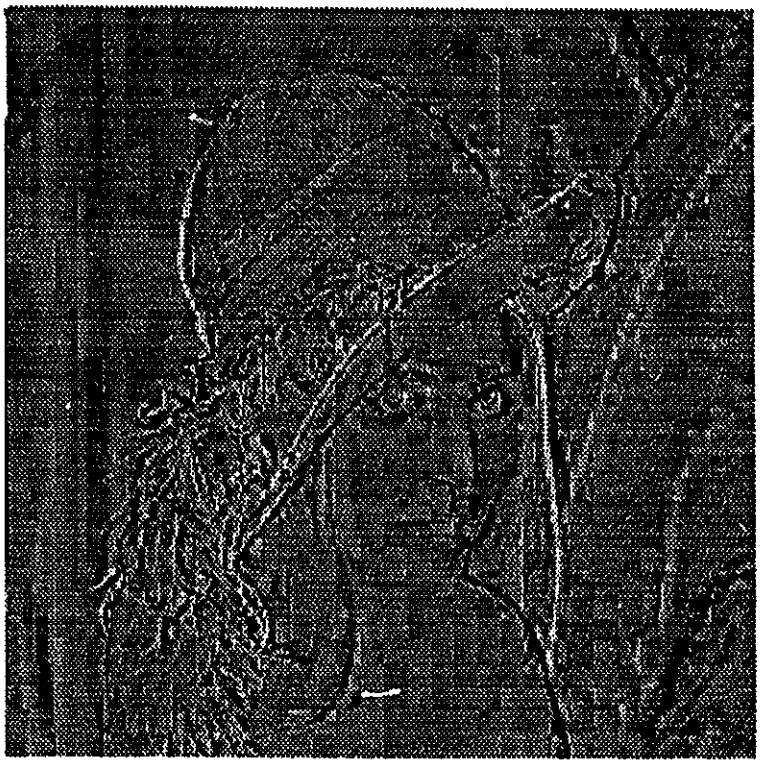


Figure 4.5: Difference image of JPEG-DCT encoded Lenna at 0.28 bpp

Chapter 5

Progressive Image Transmission using Gabor Decomposition

In this chapter we present two algorithms for progressive image transmission using Gabor decomposition.

In the first algorithm, the image is transformed into the joint spatial-frequency domain using Gabor decomposition. The coefficients are then reordered and transmitted using an optimal bit allocation startegy. Progressive image transmission is achieved by selective transmission of coefficients. In the first approximation, only one coefficient is transmitted. Upon viewer demand, more coefficients are transmitted. A lossless entropy coder is used to transmit the final residual error image to achieve lossless progressive transmission. We note that the proposed algorithm performs better in terms of subjective quality compared to the JPEG-DCT algorithm.

In the second algorithm, the image data is reorganized in the form of a pyramidal data structure. Gabor decomposition is then used to code the top level of the pyramid. The decomposed image is then quantized, arithmetic encoded and transmitted. In the

subsequent passes of the algorithm, this process is reversed and a low resolution image is computed. This image is then interpolated to give the reconstructed image at the current level. The difference between the original image and the reconstructed image at the current level is encoded using Gabor decomposition. and transmitted. Once the lowest level of the pyramid is reached, the residual error image is transmitted using a lossless entropy coding scheme. This algorithm performs better than the JPEG-DCT algorithm in objective and subjective quality.

In the following section, image compression using Gabor decomposition is presented in brief. The two algorithms are then detailed followed by simulation results and discussion.

5.1 Image compression using Gabor decomposition

We now present results of previously published research [61]. The compression system used is as shown in Fig. 5.1. In this algorithm, the image is first decomposed using a 4 level Gabor decomposition. That is, the image is split into 13 subbands as shown in Fig. 5.2. A quantization matrix is then used to quantize the coefficients of decomposition. This matrix is based upon the human visual system characteristics (Fig. 5.2). After quantization, a space filling peano curve [72] is used to scan the quantized coefficients which are then arithmetic encoded. The results of these simulations are reported using the rate distortion curves shown in Fig. 5.3 [61].

It can be seen from Fig. 5.3 that the Gabor algorithm outperforms the JPEG-DCT, in terms of objective performance at lower bit rates. This is particularly significant for progressive image transmission applications, since it is desired that the image be recognized at very low bit rates. We note that the subjective performance of the

Gabor algorithm is better than the JPEG-DCT because Gabor functions model the human visual system [61].

5.2 Algorithms for PIT using Gabor decomposition

We recall from section 5.1 that the Gabor algorithm performs better than the JPEG-DCT algorithm, especially at lower bit rates. Since Gabor decomposition models the human visual system, the subjective quality of the compressed image is expected to be better than other PIT algorithms.

We now present the two algorithms for PIT. The first algorithm follows the coefficient scanning methodology (CST) and the second algorithm uses a pyramidal representation (PGT) for PIT. Simulation results for both the algorithms will be presented in section 5.3.

5.2.1 Coefficient scanning algorithm (CST)

The image compression scheme used follows that of Fig. 5.1. This algorithm achieves progressive image transmission by transmitting a few lower order coefficients which reconstruct a coarse image. Transmission of additional coefficients gradually improves the reconstructed image quality until the desired fidelity is achieved.

The image is first transformed into the joint spatial-frequency domain using Gabor decomposition. The coefficients of the decomposition are then re-ordered using a peano curve [72]. An optimal bit allocation strategy as given in equation 2.38 is used to allocate the number of bits per coefficient. For the sake of convenience, the

equation is repeated below:

$$B_{i,j} = B + \frac{1}{2}\log_2 \epsilon_{ij}^2 \sigma_{ij}^2 - \frac{1}{N \times N} \sum_{k,l=0}^{N-1} \frac{1}{2} \log_2 \epsilon_{kl}^2 \sigma_{kl}^2$$
$$i, j = 0, 1, \ldots N-1. \tag{5.1}$$

where ϵ_{ij}^2 is the quantizer correction factor and σ_{ij}^2 is the coefficient variance. Once the coefficients are re-ordered and quantized, they are then arithmetic encoded for further reduction in the number of bits transmitted.

Prior to the transmission of the coefficients, the quantization information is sent. The coefficients are then transmitted one at a time on demand. At the receiving end, the inverse transform of these partially transmitted coefficients is computed. A first order interpolation algorithm is then employed to reconstruct the image. Fig. 5.4 shows the flowchart of the CST algorithm.

In pseudo code form the CST algorithm can be outlined as:

read original image;
transform into spatial-frequency domain using GD;
reorder coefficients using peano scan;
perform optimum bit allocation;
for all coefficients {
    transmit next coefficient;
    wait for feedback;
    if "transmit more coefficients"
      loop;
    elseif "quit"
       exit loop;
    }

We note here that since the image has only real data and the decomposition kernel is a complex matrix, the resulting coefficients have conjugate symmetry. Thus, only

half the number of the coefficients have to be transmitted in order to fully reconstruct the image.

5.2.2 Pyramidal Gabor algorithm (PGT)

A PIT scheme based on residual error feed forward method is now presented (PGT). In this scheme, the image data is first reorganized into a pyramidal data structure. This pyramid is a mean quadtree (refer section 3.2.1) representation (see Fig. 3.2). The basic principle of this scheme is recursive application of Gabor decomposition to the residual error images from previous iterations. A block diagram representation of the algorithm is shown in Fig. 5.5. At each stage, a fraction of the image information is transmitted and utilized in reconstructing successive approximations to the final image. If E_k and $\hat{E}_k$ represent, respectively, the input and the output images of the coder at iterative stage k, the steps involved in each iteration are as follows:

The input image, $E_k, k = 0, 1, \cdots$ (original or residual error), is first decomposed using Gabor decomposition. The image size varies from iteration to iteration as the algorithm traverses along the pyramid data structure. That is, if the image size at iteration k is $2^l \times 2^l$, then the image size at iteration $k+1$ is $2^{l+1} \times 2^{l+1}$.

The Gabor coefficients at each stage are quantized using a visually adaptive quantization matrix as ahown in Fig. 5.2. The quantized coefficients are then arithmetic encoded.

The difference image E_{k+1} which is the difference between the image at the lower level of the mean quad tree and the interpolated version of $\hat{E}_k$ is found. This image is taken as the input to the next stage of iteration. When the lowest level of the pyramid is reached, a decision has to be made as to whether the residual image is to be entropy coded or passed through

one more iteration of Gabor decomposition.

The decision is made as follows: we define a lossless coding rate at stage k , $R_t(k)$,

$$R_t(k) = R_p(k) + H_e(k), \tag{5.2}$$

where $R_p(k)$ is the total progressive bit rate required for transmission up to stage k, and $H_e(k)$ is the entropy of the residual error image at stage k. In other words, $R_t(k)$ is the total bit rate required for losslessly encoding the image using k stages of Gabor decomposition followed by an entropy coder. Theoretically the system must be switched to the entropy coder at any time the following two conditions hold:

- There is no coding redundancy at any stage, implying that the bit rate required for transmission at stage m, ΔR_m, is exactly equal to the image information transmitted at stage m, ΔH_m^t,

$$\Delta R_m = \Delta H_m^t, m = 0, 1, \cdots \tag{5.3}$$

- The entropy coder can losslessly encode the residual error (still correlated), at stage k, at a bit rate $H_e(k)$.

These conditions are a direct consequence of the requirement that the total bit rate for lossless encoding of an image be equal to the image entropy, H,

$$R_t(k) = R_p(k) + H_e(k) = \sum_{m=0}^{k} \Delta R_m + H_e(k) = H^t(k) + H_e(k), \tag{5.4}$$

where $H^t(k)$ is the total progressive information transmitted.
Since an entropy coder operating on a pixel-by-pixel basis is employed in the final stage, equation (5.2) is modified as follows,

$$R_t(k) = R_p(k) + H_e^1(k) \tag{5.5}$$

where $H_e^1(k)$ is the first order entropy of the residual error image at stage k. In the practical case, $R_t(k)$ is not a constant equal to the entropy of the image, H, since:

1) Practical lossy coding systems (such as vector quantization, Gabor decomposition) introduce some redundancy, and hence equation (5.3) has to be modified as follows,

$$\Delta R_m = \Delta H_m^t + \Delta R_m^r, m = 0, 1, \cdots \tag{5.6}$$

where ΔR_m^r is the coding redundancy introduced at stage m. Therefore, the total progressive bit rate upto stage k, $R_p(k)$, will be larger than the total progressive information transmitted upto stage k, $H^t(k)$, due to the accumulated coding redundancy up to stage k, $R^r(k)$.

2) Since the statistical dependency might still exist in the residual error image, the first order entropy of the residual image, $H_e^1(k)$, is larger than, or equal to, the real entropy, $H_e(k)$.

$$H_e^1(k) = H_e(k) + H_e^r(k) \geq H_e(k) \tag{5.7}$$

Since each stage partially removes the statistical dependency of the image, the first order entropy of the residual image will tend to the real entropy.
The total bit rate for losslessly encoding the image can, therefore, be expressed as follows,

$$\begin{aligned} R_t(k) &= R_p(k) + H_e^1(k) \\ &= H^t(k) + R^r(k) + H_e(k) + H_e^r(k) \\ &= H^t(k) + H_e(k) + R^r(k) + H_e^r(k) \\ &= H + (R^r(k) + H_e^r(k)), \end{aligned} \tag{5.8}$$

where H is the entropy of the image and the redundancy is due to non-ideal encoding and the statistical dependency of the residual errors. It is clear that since H is a constant, the curve of $R_t(k)$ is controlled only by the last two terms: $R^r(k)$ and $H_e^r(k)$.

Note that $R^r(k)$ and $H_e^r(k)$ are, respectively, monotonic-increasing and monotonic-decreasing functions of the number of stages, and consequently a function of the total progressive bit rate, $R_p(k)$.

We now demonstrate that the curve of $R_t(k)$ with respect to the total progressive bit rate, $R_p(k)$, is concave. To do this, we need to prove that the derivative of $R_t(k)$ with respect to $R_p(k)$ is a monotonic increasing curve crossing zero, i.e. an increasing curve from negative to positive values. From equations (5.5) and (5.7),

$$\begin{aligned} R_t(k) &= R_p(k) + H_e^1(k) \\ &= R_p(k) + (H_e(k) + H_e^r(k)) \\ &= R_p(k) + (H - H^t(k)) + H_e^r(k) \end{aligned} \tag{5.9}$$

where $H_e(k) = H - H^t(k)$. Taking derivatives,

$$\frac{dR_t(k)}{dR_p(k)} = 1 - \frac{dH^t(k)}{dR_p(k)} + \frac{dH_e^r(k)}{dR_p(k)} \tag{5.10}$$

In discrete form,

$$\frac{\Delta R_{t,m}}{\Delta R_m} = 1 - \frac{\Delta H_m^t}{\Delta R_m} + \frac{\Delta H_{e,m}^r}{\Delta R_m} \tag{5.11}$$

where $\Delta R_{t,m}$ and $\Delta H_{e,m}^r$ represent, respectively, the variations of $R_t(k)$ and $H_e^r(k)$ at stage $k = m$. Since the first term in equation (5.11) is a constant equal to 1, we need to analyze only the last two terms. The following assumptions and observations are in order:

(I) From equation (5.6), $\frac{\Delta H_m^t}{\Delta R_m} = \frac{\Delta H_m^t}{\Delta H_m^t + \Delta R_m^r}$. Therefore, this term is positive and less than 1.

(II) Since each stage partially removes the statistical dependency of the residual errors, $H_e^r(k)$ is a decreasing function of $R_p(k)$ and therefore $\frac{\Delta H_{e,m}^r}{\Delta R_m}$ is negative.

(III) Since the residual errors tend to become more decorrelated, it is reasonable to assume that the same coding technique (Gabor decomposition) should code the residual errors less efficiently. This means that

$\Delta R_m^r(k)$ increases and, therefore, $\frac{\Delta H_m^t}{\Delta R_m} = \frac{\Delta H_m^t}{\Delta H_m^t + \Delta R_m^r}$ decreases; while at the same time, $\left|\frac{\Delta H_{e,m}^r}{\Delta R_m}\right|$ decreases.

(IV) As the number of stages, or the total progressive bit rate of $R_p(k)$, increases, the residual errors tend to be statistically independent. Therefore, $H_e^r(k)$ approaches zero and

$$\frac{\Delta H_{e,m}^r}{\Delta R_m} \rightarrow 0, \tag{5.12}$$

which implies that,

$$\frac{\Delta R_{t,m}}{\Delta R_m} \rightarrow 1 - \frac{\Delta H_m^t}{\Delta H_m^t + \Delta R_m^r} > 0, \tag{5.13}$$

i.e the derivative (equation 5.11) is positive at some point.

(V) In addition, it is reasonable to assume that the coding system is very efficient for the original (highly correlated) image such that,

$$\frac{\Delta R_{t,0}}{\Delta R_0} < 0 \tag{5.14}$$

that is, the derivative (equation 5.11) is initially less than zero.

Based upon the above analysis, it can be concluded that

$$\frac{\Delta R_{t,m}}{\Delta R_m} = 1 - \frac{\Delta H_m^t}{\Delta R_m} + \frac{\Delta H_{e,m}^r}{\Delta R_m} = 1 - \left(\frac{\Delta H_m^t}{\Delta R_m} + \left|\frac{\Delta H_{e,m}^r}{\Delta R_m}\right|\right) \tag{5.15}$$

is an increasing function of $R_p(k)$ which goes from negative to positive value. Thus, we have demonstrated that $R_t(k)$ is a concave curve as a function of $R_p(k)$. Consequently, the system should be switched to entropy coding when the minimum value of $R_t(k)$ is acheived.

5.3 Simulation results and discussions

Computer simulations for CST, PGT and JPEG-DCT were carried out using the two standard CCITT test images: Lenna and Mandril. The images are of size 512×512

pixels quantized to 8 bits/pixel. The first order entropies for the Lenna and the Mandril image are: 7.2770 and 7.2520 bits/pixel, respectively. The original images are shown in Figs. 5.8 and 5.9.

The scaling parameters D and Ω of the Gabor kernel were chosen as 16 and $\pi/8$, respectively. We recall from section 2.4.1 that for image compression applications $\Omega = \frac{2\pi}{D}$ (which is the critical sampling case) is used. The parameter D specifies the width of the Gabor window and a value of $D = 16$ implies that majority of the Gabor window lies within 16 image pixels. In comparison to the JPEG-DCT algorithm, where a rectangular window of 8 pixels is used, this choice implies an overlap of 4 pixels on each side of the chosen block. The independent JPEG group's shareware released in January 1992 is used in this set of experiments.

Simulation results for the Lenna image are tabulated in tables 5.1 - 5.3. The rate distortion curves for the Lenna image ae shown in Fig. 5.6. Simulation results for the Mandril image rae tabulated in tables 5.4 - 5.6 and the corresponding rate distortion curves ae shown in Fig. 5.7.

5.3.1 Comparisons with JPEG-DCT

The reconstructed Lenna images at 0.08 and 0.32 bits/pixel for the JPEG-DCT algorithm are shown in Figs. 5.10 and 5.11, respectively. The reconstructed images for the CST algorithm at corresponding bit rates are shown in Figs. 5.12 and 5.13. Similarly, the reconstructed Lenna images for the PGT algorithm are shown in Figs. 5.14 and 5.15. In addition, the reconstructed Lenna image at 0.022 bits/pixel using the PGT algorithm is shown in Fig. 5.16. Results for the Mandril test image are reported in Figs. 5.17 - 5.22.

It can be seen that the PGT algorithm performs better than the JPEG-DCT algorithm in terms of PSNR at all bit rates. However, the CST algorithm performs

better than the JPEG-DCT at low bit rates. At medium bit rates, the CST algorithm has a performance comparable to that of the JPEG-DCT algorithm. From the reconstructed images, we see that there is a significant improvement in performance of both the CST and the PGT algorithms at lower bit rates. This is particularly important in progressive image transmission applications where faster image identification is required. In terms of subjective performance, we note that both the CST and the PGT algorithms perform better than the JPEG-DCT algorithm. We note that at a very low bit rate of 0.022 bits/pixel, the PGT algorithm can provide a recognisable image (refer Fig. 5.16). This superior subjective performance of the Gabor decomposition based algorithms is a direct consequence of the human visual system model upon which the Gabor functions are based. Hence, we can conclude that Gabor decomposition is a promising approach for progressive image transmission applications.

5.4 Summary

In this chapter, two schemes for progressive image transmmission have been presented. In the first algorithm (CST), the coefficient scanning methodology is employed to acheive progressive image transmission. The second algorithm (PGT) represents the image in a pyramidal data structure and encodes different levels of the pyramid in a recursive manner to achieve progressive transmission.

Simulation results indicate that CST performs better than the JPEG-DCT algorithm for lower bit rates. In addition, the subjective quality of the reconstructed images are higher. The PGT algorithm has a superior performance compared to the JPEG-DCT algorithm at all bit rates. The superior subjective performance of the proposed algorithms (CST and PGT) is a direct consequence of the close correspondence between the Gabor functions and the human visual system.

We recall that the computational complexity of Gabor decomposition requires special purpose architectures to implement the proposed algorithms in real-time. In chapter 6, an architecture for implementing the CST and PGT algorithms in real-time is designed.

bpp	PSNR(dB)
0.08	23.63
0.15	25.91
0.19	27.24
0.23	27.47
0.25	28.32
0.27	29.62
0.30	29.98
0.32	30.63
0.33	31.18
0.35	31.33

Table 5.1: Simulation results for Lenna image using JPEG-DCT algorithm

bpp	PSNR(dB)
0.054	22.95
0.078	23.28
0.1	23.51
0.2	25.94
0.3	28.21
0.4	30.1

Table 5.2: Simulation results for Lenna image using the CST algorithm

bpp	PSNR(dB)
0.022	20.95
0.055	24.07
0.082	26.13
0.14	28.51
0.231	29.67
0.269	31.60
0.32	31.78

Table 5.3: Simulation results for Lenna image using the PGT algorithm

bpp	PSNR(dB)
0.08	19.1
0.15	20.15
0.21	20.73
0.27	21.12
0.31	21.45
0.36	21.69
0.41	21.80
0.44	22.01
0.48	22.30

Table 5.4: Simulation results for Mandril image using the JPEG-DCT algorithm

bpp	PSNR(dB)
0.046	18.8
0.082	19.3
0.12	19.6
0.21	20.6
0.3	20.45
0.4	20.80

Table 5.5: Simulation results for Mandril image using the CST algorithm

bpp	PSNR(dB)
0.02	12.38
0.085	19.28
0.16	21.55
0.22	22.17
0.28	22.6
0.313	22.96
0.37	23.11
0.41	23.18
0.45	23.23
0.50	23.36

Table 5.6: Simulation results for Mandril image using the PGT algorithm

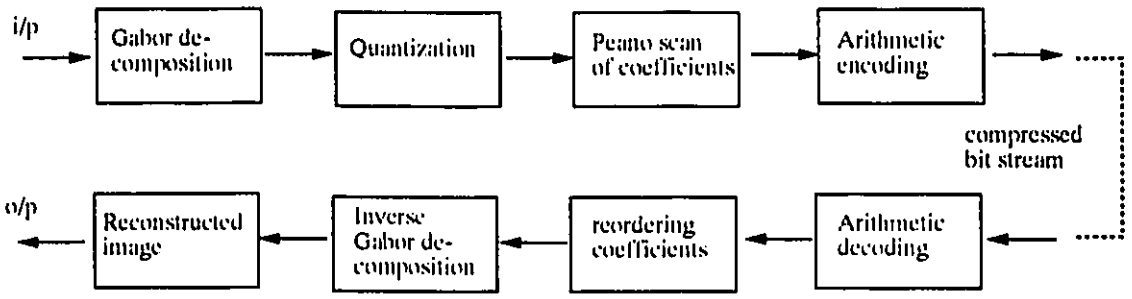


Figure 5.1: Image compression system for Gabor decomposition

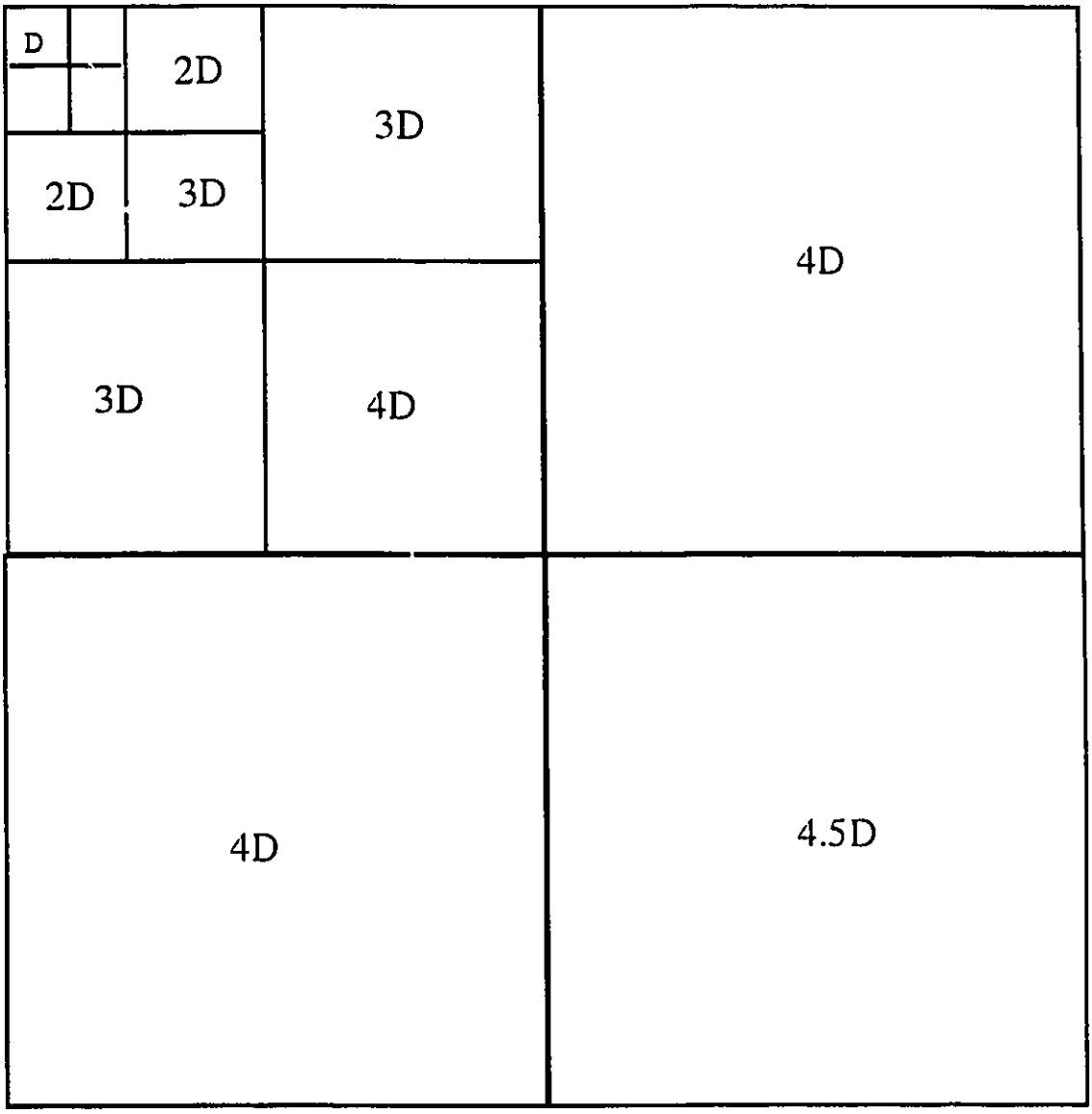


Figure 5.2: Splitting of the image into 13 bands and subsequent quantization tuned to human visual system characteristics

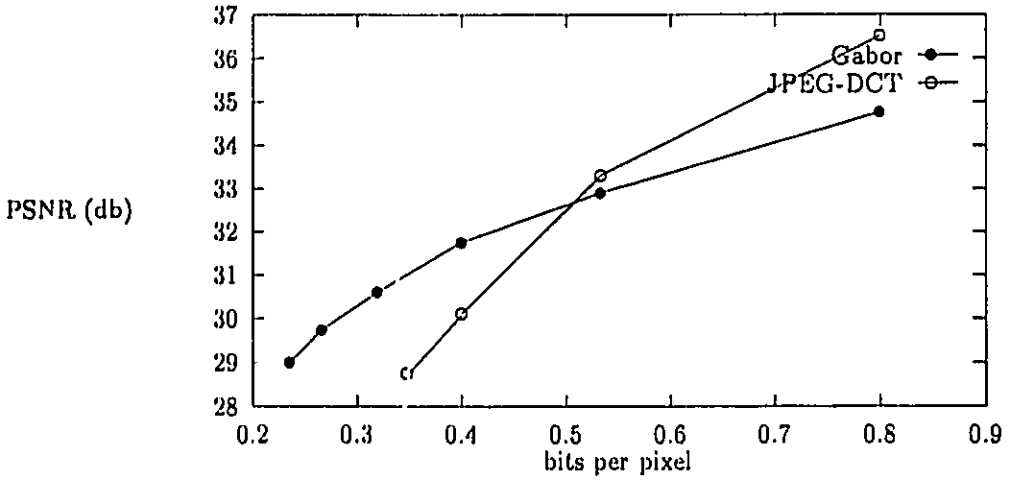


Figure 5.3: Rate distortion curve (PSNR vs bit rate) for the Lenna image encoded using JPEG-DCT and Gabor algorithms

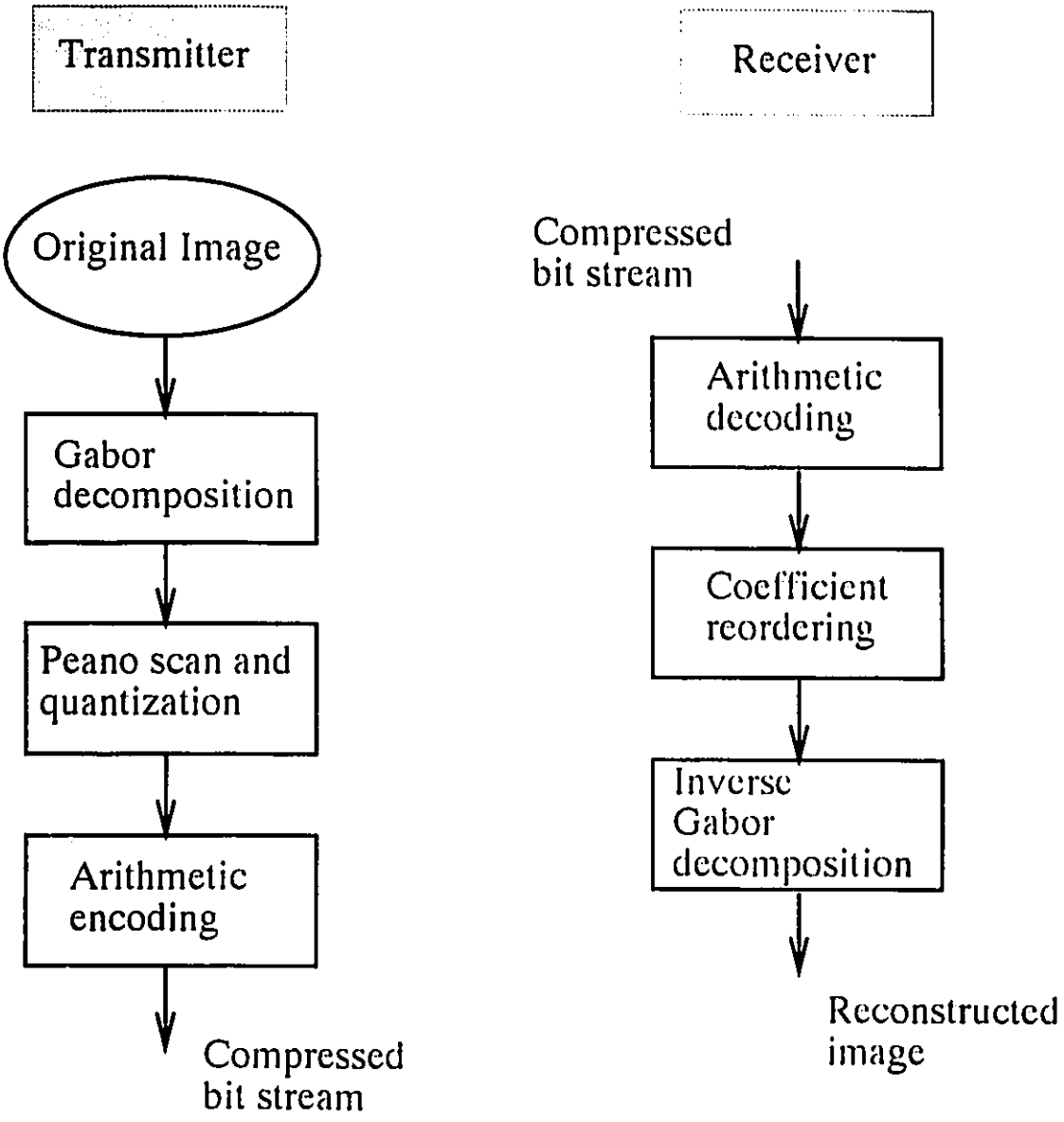


Figure 5.4: Flowchart of the CST algorithm

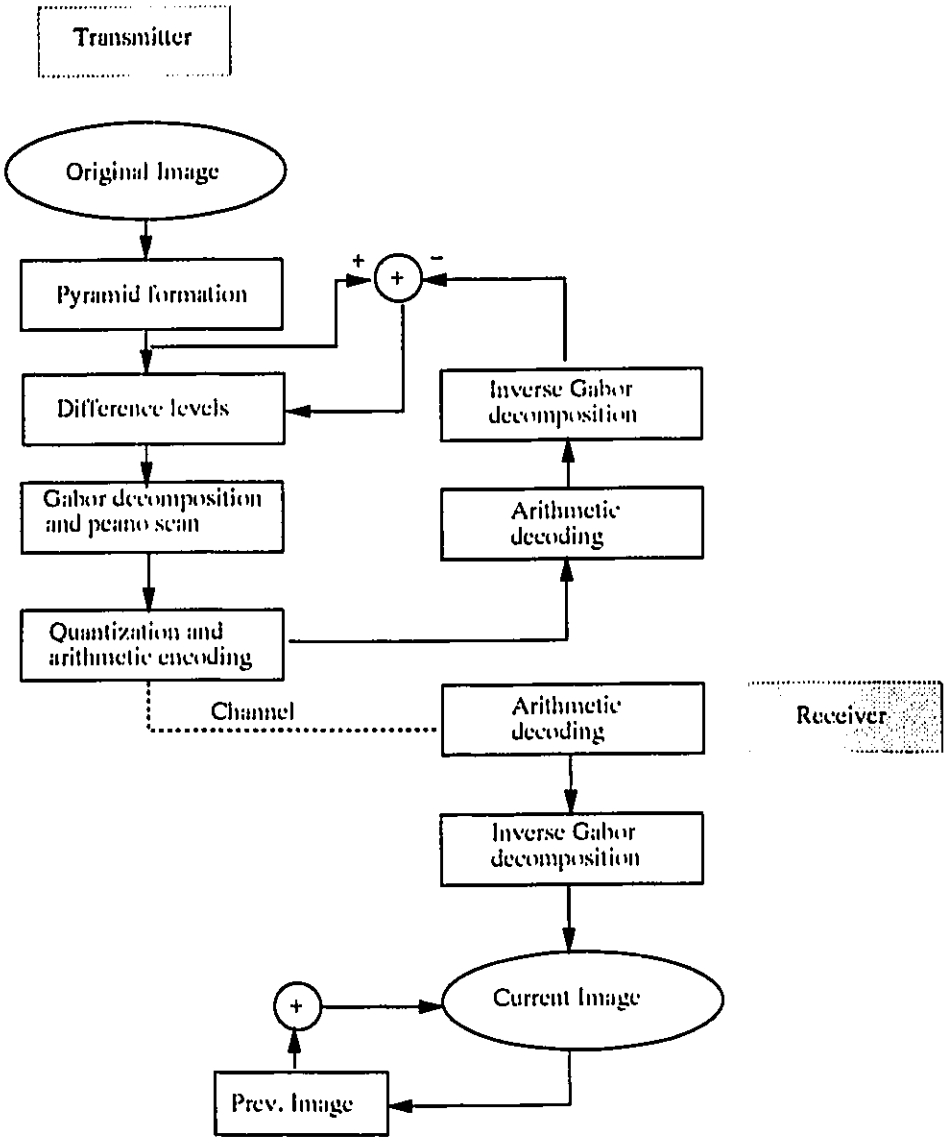


Figure 5.5: Flowchart of the PGT algorithm

PSNR (db)

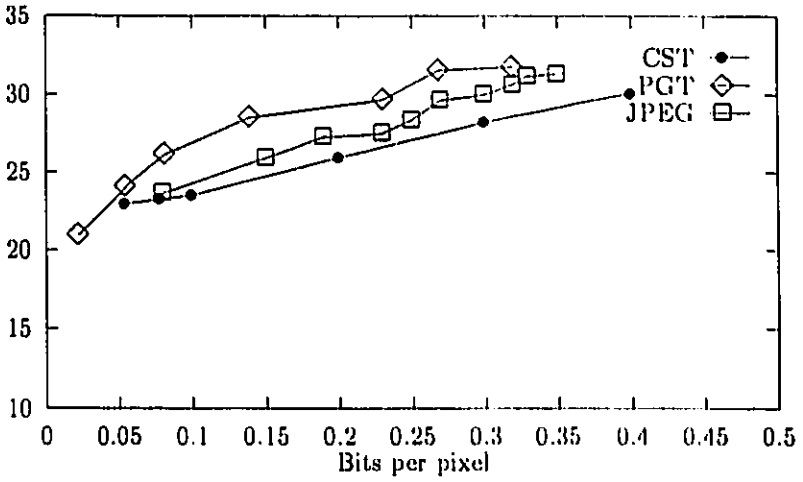


Figure 5.6: Rate distortion curve (PSNR vs bit rate) for the Lenna image encoded using JPEG-DCT, CST and PGT algorithms

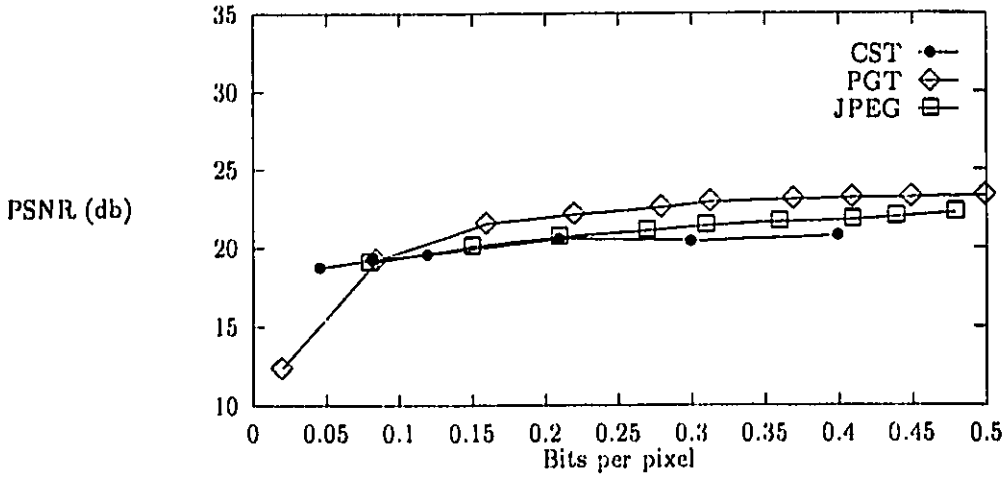


Figure 5.7: Rate distortion curve (PSNR vs bit rate) for the Mandril image encoded using the JPEG-DCT, CST and PGT algorithms

Figure 5.8: The original Lenna image

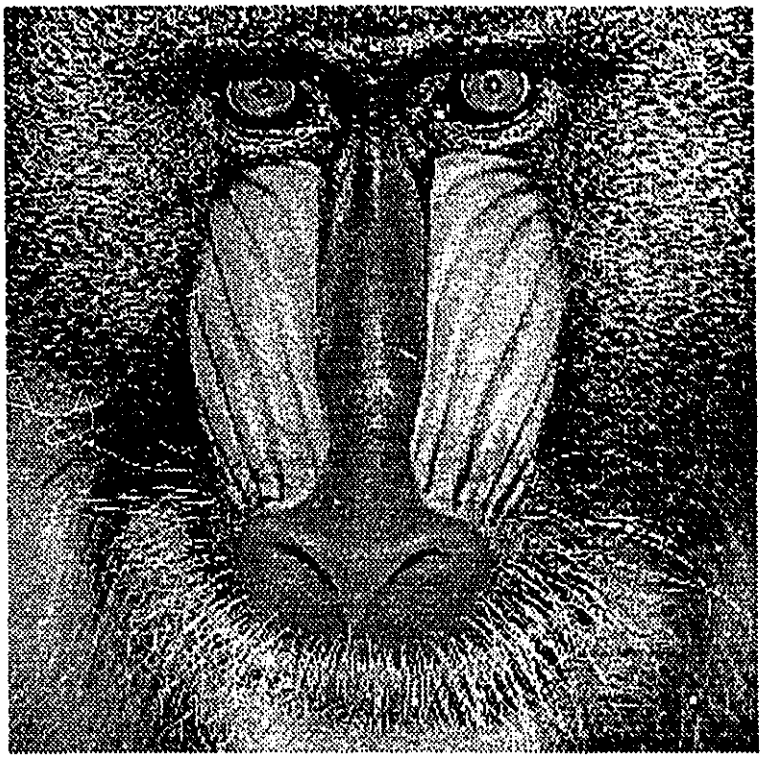


Figure 5.9: The original Mandril image

Figure 5.10: JPEG-DCT encoded Lenna image reconstructed at 0.08 bits per pixel

Figure 5.11: JPEG-DCT encoded Lenna image reconstructed at 0.32 bits per pixel

Figure 5.12: CST encoded Lenna image reconstructed at 0.078 bits per pixel

Figure 5.13: CST encoded Lenna image reconstructed at 0.3 bits per pixel

Figure 5.14: PGT encoded Lenna image reconstructed at 0.06 bits per pixel

Figure 5.15: PGT encoded Lenna image reconstructed at 0.31 bits per pixel

Figure 5.16: PGT encoded Lenna image reconstructed at 0.022 bits per pixel

Figure 5.17: JPEG-DCT encoded Mandril image reconstructed at a bit rate of 0.08 bits per pixel

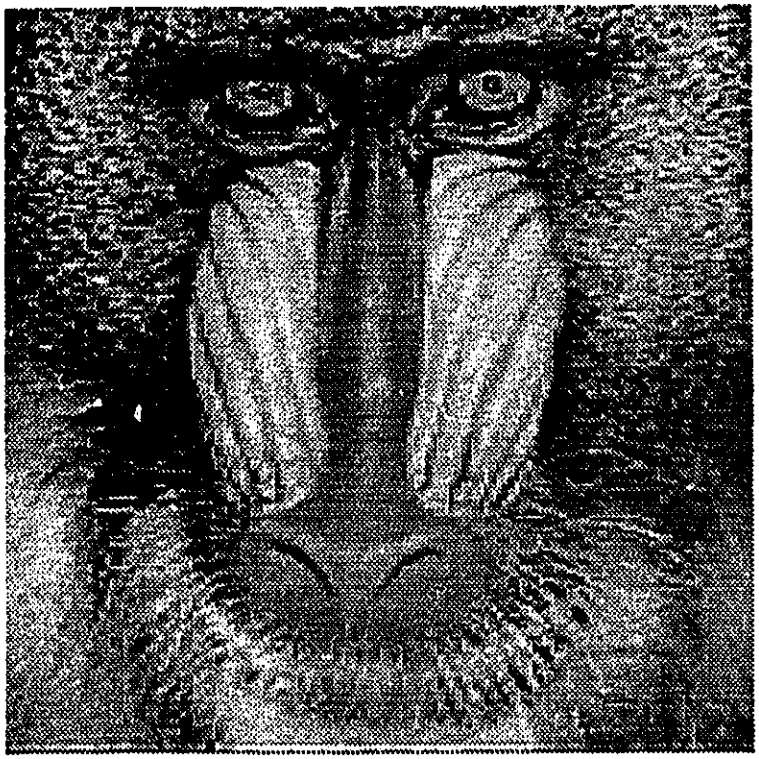


Figure 5.18: JPEG-DCT encoded Mandril image reconstructed at 0.31 bits per pixel

Figure 5.19: CST encoded Mandril image reconstructed at 0.08 bits per pixel

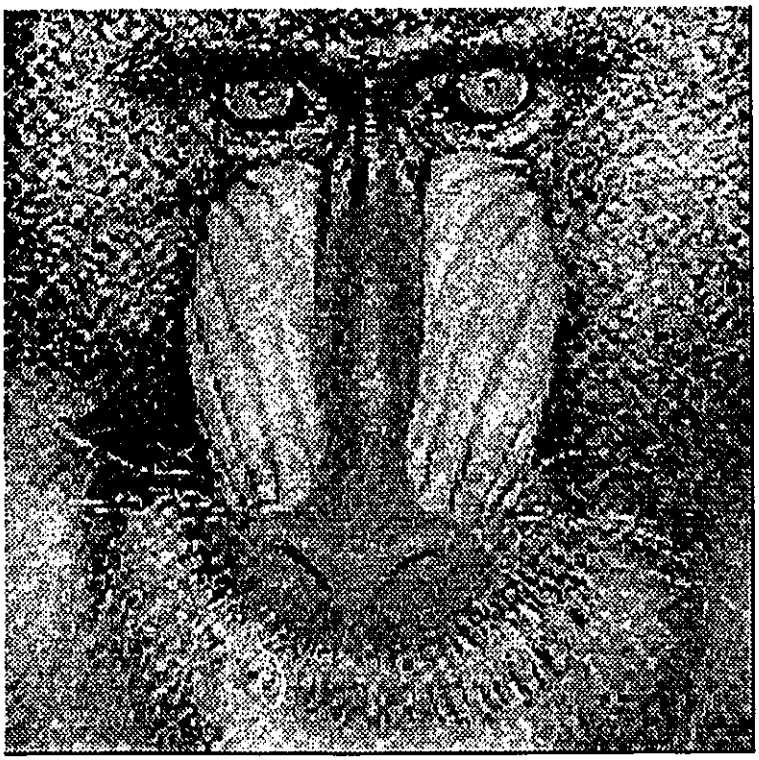


Figure 5.20: CST encoded Mandril image reconstructed at 0.3 bits per pixel

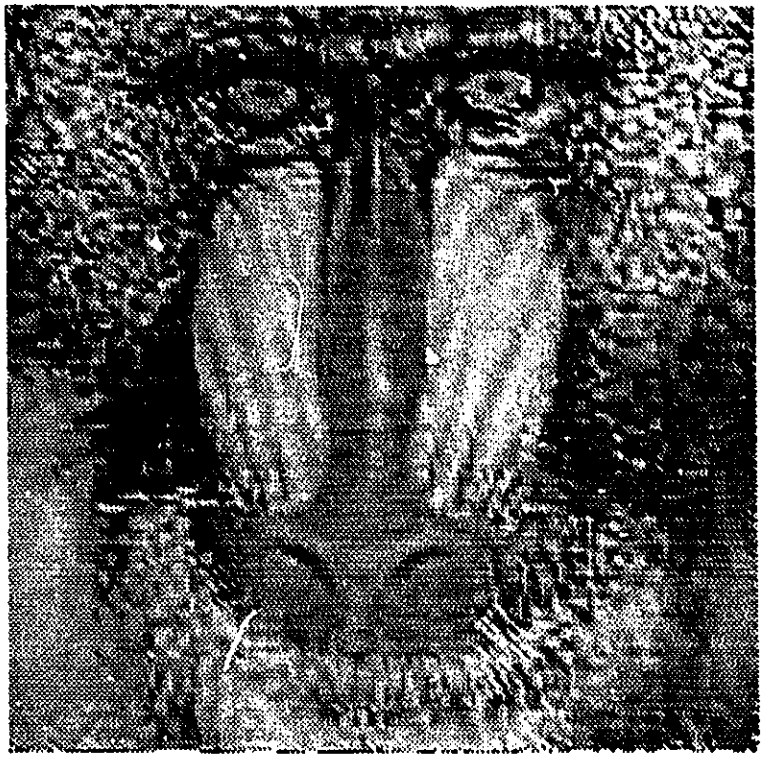


Figure 5.21: PGT encoded Mandril image reconstructed at 0.07 bits per pixel

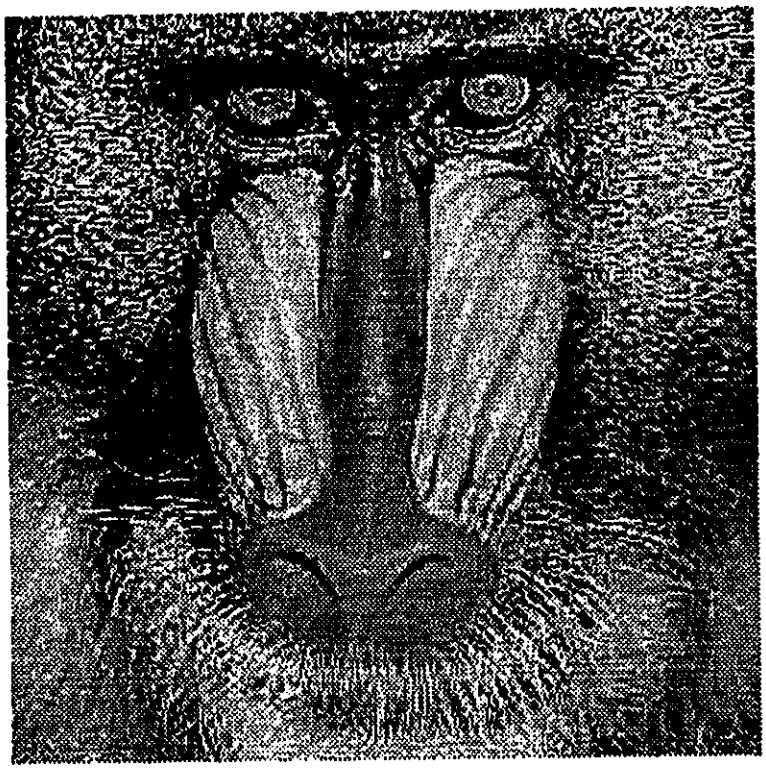


Figure 5.22: PGT encoded Mandril image reconstructed at 0.3 bits per pixel

Chapter 6

Codec for PIT using Gabor decomposition

In chapter 5, we presented two algorithms for PIT using Gabor decomposition. These algorithms perform better compared to the JPEG-DCT in terms of objective and subjective quality. However, Gabor decomposition requires special purpose hardware for real-time implementation.

In this chapter, we present a combined systolic array - content addressable memory architecture for Gabor decomposition (SCAM). Based on this SCAM architecture, codecs for executing the proposed algorithms are then designed. We recall from section 2.4 that Gabor decomposition can be computed as a multiplication between a transform matrix and a vector of image data. Systolic arrays are attractive for matrix multiplication problems [66, 68] and content addressable memories (CAM) offer fast means of data access [67]. Hence, a combined systolic array - content addressable memory design offers a powerful architecture which is well suited for the computationally intensive Gabor decomposition.

For an $n \times n$ image, the proposed architecture consists of a linear systolic array of n processing elements each with a local CAM of size $2p$ where p is the bandwidth of the Gabor kernel (refer section 2.4.3). Simulations and computational complexity calculations show that this architecture can achieve real-time performance with current VLSI technology. In addition, this architecture is modular and regular and hence can be implemented in VLSI as a codec.

We recall from section 2.5.2 that the architectures presented in the literature for Gabor decomposition are expensive to implement. The neural network architecture is complex and requires $3n^2$ elements for an $n \times n$ image. Moreover, it is iterative and may not always converge in real-time. In addition, the implementation of neural network architectures on silicon is an open-ended question. The ASP based implementation requires n^2 processing elements for an $n \times n$ image. In addition, the ASP design is complex in terms of inter-APE communications, control flow, etc. Hence, it is difficult to implement these architectures in VLSI as a codec.

6.1 Systolic arrays and content-addressable memories

In this section we present a brief review of systolic arrays and content addressable memories.

6.1.1 Systolic arrays

Systolic array is a general methodology for mapping high-level computations onto hardware structures [66]. Information in a systolic system flows between processing elements in a pipelined fashion, and communication with the external devices occurs

only at the boundary cells.

The basic principle of a systolic array is illustrated in Fig. 6.1 [66]. By replacing a single processing element (PE) with an array of PEs, a higher computation throughput can be achieved without increasing the memory bandwidth. This is because each unit of input that is retrieved from the memory is operated upon by different PEs in turn before returning the data to memory. Also, all the PEs work simultaneously on different data and hence high efficieny can be achieved.

Systolic arrays are particularly suitable for compute bound problems such as pattern matching, matrix multiplication, motion compensation, etc [68, 69]. In addition, other advantages of systolic arrays are modular expansibility, regular data and control flows and simple & uniform cells or PEs, etc. These advantages of a systolic array together with the modelling of Gabor decomposition as a matrix multiplication operation form the basis for the architecture proposed in this chapter.

6.1.2 Content-addressable memory

Content-addressable memories (CAM) are collections of elements having data storage capabilities, and can be accessed simultaneously and in parallel on the basis of their data content rather than by the specific address or location [67]. The access mechanism is based on an external search argument which is compared with the data stored in all the cells. Traditionally, CAMs have been employed in the design of high speed pattern recognition machines. A set of templates T is stored as words in the CAM array and the input pattern C is used as the search argument. The search argument is then compared in parallel with all the words and, if an exact match is obtained with a particular word, the corresponding output is activated and the result is recorded in a register. Hence CAMs are attractive for pattern matching, fast data retrieval, etc.

Systolic arrays are powerful for compute bound problems such as Gabor decomposition. CAMs complement the systolic array by providing fast and efficient data access. The high computational performance of a systolic array combined with fast data accessability of the CAM provides a compact architecture which is capable of executing Gabor decomposition in real-time.

6.2 Systolic array architecture for real-time Gabor decomposition

We recall from section 2.4.3 that the Gabor kernel is a banded matrix. That is, the kernel can be expressed as:

$$G_1 = \begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{bmatrix} \tag{6.1}$$

This matrix has a bandwidth $p = 3$ where the bandwidth is defined as the maximum number of non-zero elements in any row / column of the matrix.

6.2.1 Systolic array for banded matrix multiplication

For a banded matrix as given in equation 6.1, we present a a systolic array to implement the multiplication.

$$A = G_1 * X \tag{6.2}$$

The design is illustrated in Fig. 6.2, where,

$$\mathbf{X} = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{bmatrix} \tag{6.3}$$

is the image data matrix and G_1 is as given in equation 6.1.

In Fig. 6.2, there is one PE for each element in the banded matrix. This contrasts with the ASP implementation where one PE is required for each pixel of the image. Note that this is made possible by the banded Gabor kernel which is predominantly filled with zeros. For an $n \times n$ image, the proposed architecture requires only $n \times p - 2$ PEs ($p < n$), where p is the bandwidth of the Gabor kernel. Whereas, the ASP implementation uses $n \times n$ PEs. For example, with $n = 512$ and $p = 128$, 65534 PEs are required in the systolic array design as compared to 262144 PEs in the ASP implementation. In addition, the proposed systolic array has simpler data and control flows and hence is easier to implement.

In spite of the substantial reductions in the number of PEs required for executing Gabor decomposition, the design still uses a large number of PEs and is hence difficult to implement in VLSI as a codec. This points to the need for further reductions in the number of PEs to make VLSI implementation possible. The fast data retrieval mechanism of the CAM permits further reductions in the required number of PEs by combining the functions of several PEs into one PE.

6.2.2 CAM based Systolic array for Gabor decomposition (SCAM)

We now present a CAM based systolic array for Gabor decomposition. This design employs a linear systolic array in contrast to the 2 dimensional array design presented

in section 6.2.1. The SCAM architecture requires $2n$ PEs for an $n \times n$ image. Hence, this results in an $O(N)$ reduction in the number of processors required as compared to the ASP based implementation.

Fig. 6.3 shows the overview of the proposed SCAM architecture which comprises of 3 basic modules. The frame buffer module is used for storing the image data before executing the algorithm. Each systolic array module is capable of executing 1-D Gabor decomposition and hence two such modules are required to execute the 2-D seperable Gabor decomposition (refer section 2.4.3). For proper data input to the second systolic array module, a transpose operation is required in between the two 1-D decomposition operations. The transpose buffer module is a parallel transposer [70] capable of executing the transpose operation in one clock cycle and hence does not affect the total time required for computing the Gabor decomposition.

The systolic array module consists of a linear array of n processing elements (PE) for a transform matrix of size $n \times n$. Each PE is a simple inner product processor and has a local CAM of size $2p$. The inner product operation executed by the PE is illustrated in Fig. 6.4. Each PE takes in the partial sum c_{ik} from the previous PE and adds to it the product of image data b_{jk} with the corresponding Gabor kernel element, A_{ij}. This result is then passed to the next PE in line. Hence, the final result is as follows,

$$c_{ik} = \sum_{j=m}^{j=m+p} A_{ij} b_{jk}. \tag{6.4}$$

The CAM cell organization is shown in Fig. 6.5. and comprises of the following five data fields:

i) value of the coefficient (**V**)

ii) row number (i) of the coefficient (**R**)

iii) column number (j) of the coefficient (**C**)

iv) ready to output tag (**RT**)

v) end of computation tag (**ET**)

The row and the column number fields are used as search keys to retrieve the coefficient value in a computation. The **RT** tag indicates that the partial results are available to be sent to the next PE in line. Therefore the output of any PE to the next stage is the partial result value, the row number, the column number and the **RT** tag. The **ET** tag indicates that the result of the current computation is the final output at the end of one computation cycle.

Each PE has $2p$ basic CAM cells where p is the bandwidth of the transform matrix: p cells each are used for storing the transform matrix and intermediate values, respectively. The SCAM architecture to decompose an image of size 4×4 is shown in Fig. 6.6. The product of two matrices is computed as follows: If $c = A \times b$ where A, b *and* c are matrices of size $n \times n$, c_{ij}, the element in matrix c at location i, j is the inner product of row i of matrix A with column j of matrix b. We note that if one of the matrix is a banded matrix, then the number of multiplication and summation operations required to compute the inner product is reduced considerably. The banded transform matrix is stored in the CAM cells present in each PE, as shown in Fig. 6.6. Only the nonzero elements of the transform matrix are stored in the CAM cells of each PE. Since not all coefficients are required in the computation of every inner product, several computations can be executed in parallel. Hence the overall decomposition can be executed in real-time. The timing analysis is detailed in section 6.3.

The sequence of steps executed by each PE_i *for* $i = 1, \cdots, n$ in pseudocode form is given below:

begin

for $i = 1, \cdots, n$

  If data from PE_{i-1}

    Receive partial product from PE_{i-1} (c_{ik});

    Check row no. "i" of the received data (i.e check R field);

Retrieve coefficient A_{ij} from CAM and perform the inner product operation (match R values in CAM);

If **ET** = 1

output result to transpose/output buffer;

else

output result to PE_{i+1};

endif

else

Check **RT** *tag;*

If there is coefficient A_{ij}

multiply image data with coefficient A_{ij};

output result to PE_{i+1}

endif

endif

endfor

end

Since each of the PE in the SCAM module is a simple inner product processor and the SCAM has a regular and modular nature, it is easy to implement this design as a codec in VLSI. In addition, as compared to the ASP implementation, this design requires $O(N)$ fewer processors. Hence it is a very efficient design in terms of chip real-estate.

6.3 Performance analysis of the proposed systolic array

Two different snapshots of the execution process at time cycles 1 and 2 are shown in Fig. 6.7. The Gabor kernel is as given in equation 6.1.

Fig. 6.8 shows the cell occupancy diagram for the systolic array for a general Gabor kernel of size $n \times n$ with a bandwidth of p. The assumption made in the occupancy diagram is that $n < 2p$. This assumption does not affect the timing analysis as the time taken to compute each term of a 1-D decomposition is p cycles, where one inner product is computed in each cycle.

The time taken to execute the Gabor decomposition for an image of size $n \times n$ is computed as follows. If the bandwidth of the Gabor kernel is p, each term of the decomposition requires p inner product operations. Assuming one inner product operation can be executed in one cycle, this takes p cycles. During these p cycles, the remaining $p-1$ computations can be fed into the pipeline. Hence, the systolic array takes $2p+1$ cycles to compute the 1-D decomposition. Since each of the two decompositions can be executed in parallel, the architecture can effectively execute the 2-D Gabor decomposition in $2p+1$ cycles with a latency of $4p+2 = 2 \times (2p+1)$ cycles (the time between the input image is fed into the array and the availability of the corresponding output). If the decomposition has to be executed by a single instruction single data (SISD) machine, it takes $2pn - n - 2$ cycles which is the total number of inner product operations required to compute the 2-D Gabor decomposition.

The speedup obtained using this array can be calculated as follows:

The speedup, S_p of a parallel machine is defined as:

$$\frac{\textit{the time taken by a SISD machine to execute a task}}{\textit{the time takenby the parallel machine to execute the same task}}$$

Hence the speedup, S_p is given by,

$$S_p = \frac{2pn - n - 2}{2p + 1} \tag{6.5}$$
$$= \frac{(2p+1)n - 2n - 2}{2p+1}$$
$$= n - \frac{2(n-2)}{2p+1}$$

We note that the speedup, S_p is always less than n, the total number of processing elements in the array. The efficiency of parallelism is defined as:

$$\frac{S_p}{n}, \tag{6.6}$$

which is

$$\frac{the\ achieved\ speedup}{the\ number\ of\ processors\ in\ parallel}.$$

The speedup, S_p, increases as p increases with respect to n. This analysis follows the intuitive reasoning (refer Fig. 6.8) from the occupancy diagram. We note that p PEs (out of a total n PEs) commence computation simultaneously at cycle number 1. If p is higher, then the cell occupancy ratio is higher i.e. more number of PEs are occupied from the onset of computation and the array reaches peak performance (all n PEs in operation) in fewer cycles. This results in a higher speedup. Fig. 6.9 shows the plot of S_p versus p for 512 PEs. From Fig. 6.9 it can be seen that the architecture achieves high speedup ratios for a wide range of p values. Hence, this architecture performs well for a wide range of Gabor kernel matrices.

For example, with $n = 512$ and $p = 128$ (512 PEs) the SCAM has a speedup, $S_p = 508$ compared to the SISD machine. This represents an efficiency of parallelism of 99%. If the duration of each clock cycle is assumed to be $100ns$ [71], for $p = 128$, the SCAM

architecture executes the Gabor decomposition in $13ms$ which satisfies the real-time requirement (< 33 ms).

6.4 Codec for progressive image transmission using Gabor decomposition

In the previous two sections, we designed a systolic array - content addressable memory architecture for Gabor decomposition. We note that this architecture can perform 2-D seperable Gabor decomposition in real-time. We now present codec designs for executing the CST and PGT algorithms presented in chapter 5.

6.4.1 Codec for PIT using the CST algorithm

We now present a codec design based on the SCAM architecture presented in section 6.2.2. We recall the block diagram of the CST algorithm from Fig. 5.4. The codec design for executing this algorithm is shown in Fig. 6.10.
The transmitter (T_x) end of the codec has three modules, namely:

- Image database module: This contains the image information to be progressively transmitted.
- Gabor decomposition module: This is the SCAM architecture module (Fig. 6.3) for executing Gabor decomposition.
- Peano scan and arithmetic encoder module: This module performs the coefficient quantization, scanning and arithmetic encoding operations.

Similarly, the receiver (R_x) end of the codec also has three modules:

- Arithmetic decoder and reorder module: This performs the arithmetic decoding and reordering of the Gabor coefficients.
- Inverse Gabor decomposition module: This is functionally the same as the Gabor decomposition module except that the kernel memory (CAM) of the SCAM stores the inverse matrix.
- Reconstructed image module: This stores/buffers the reconstructed image.

In addition to these, there is a control link (via the communication channel) between the user interface of the receiver with the image database of the transmitter which is required for selecting different images based on the user's request. Note that there is a direct link between the image database and the entropy encoder block in the transmitter end of the codec. This is for the final iteration stage where lossless transmission of the residual image is desired. Correspondingly, there is a direct link between the entropy decoder and the image reconstruction block in the receiver end of the codec. We note that the Gabor decomposition block in the transmitter executes the decomposition only once per image but the corresponding receiver block has to perform the inverse operation for each iteration stage. This necessiates the use of a powerful hardware for this algorithm.

6.4.2 Codec for PIT using the PGT algorithm

We recall from section 5.2.2 and Fig. 5.5 that the pyramidal Gabor technique for progressive image transmission is a recursive approach. Fig. 6.11 is a block diagram representation of the codec for executing this algorithm.
Since this is a recursive technique where the encoder and decoder work in synchronism, the transmitter end of the codec also comprises of the decoder module as folllows:

- Image database module: This block stores the images to be trans-

mitted to the receiver.

- Pyramid structure module: This computes and stores the mean quadtree pyramid of the input image.
- Gabor decomposition module: This is the SCAM architecture module which executes Gabor decomposition of the input image.
- Quantization and arithmetic encoder module: This generates the compressed bit stream after quantization and arithmetic encoding of the Gabor coefficients.
- Arithmetic decoder module: This is the inverse of the arithmetic encoder module. It decodes the compressed bit stream to give the Gabor coefficients.
- Inverse Gabor decomposition module: This is functionally same as the SCAM module except that it stores the inverse Gabor matrix.

In addition, the codec has a kernel memory block where the Gabor kernels for different image sizes are stored. We note that if the SCAM module is designed to support the final image size, the same module can be used for real-time decomposition of smaller image sizes.

The receiver (R_x) end of the codec correspondingly comprises of the following modules.

- Aritemetic decoder module: This decodes the compressed bit stream and reorders the coefficients.
- Inverse Gabor decomposition module: This is the SCAM module with the inverse Gabor matrix.
- Reconstructed image module: This buffers the current reconstructed image.
- Image memory module: This buffers the reconstructed image from the previous iteration.

We note here that the transmitter end of the codec executes both the forward and the inverse decomposition for each iteration. The receiver end of the codec executes only the inverse decomposition for each iteration. This justifies the use of the powerful SCAM architecture for this algorithm.

6.5 Summary

We have presented a combined systolic array - content addressable memory architecture for real-time Gabor decomposition. The proposed architecture is capable of executing Gabor decomposition in real-time. It achieves a very high efficiency of parallelism (99 %) and thus offers a cost effective solution. This architecture can be implemented in VLSI as a codec because of its simple and modular construction. Two codecs, based on the SCAM architecture were detailed. These codecs implement the CST and the PGT algorithms for PIT, respectively.

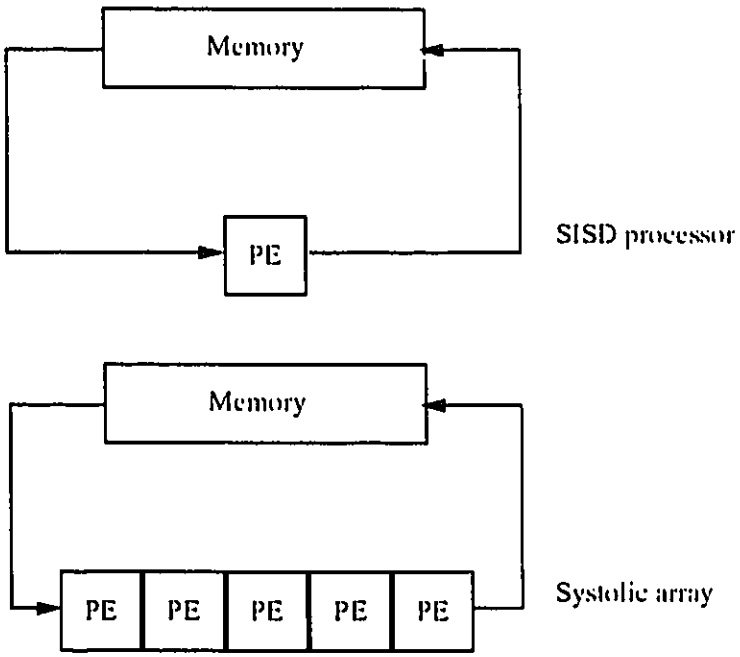


Figure 6.1: The principle of systolic arrays

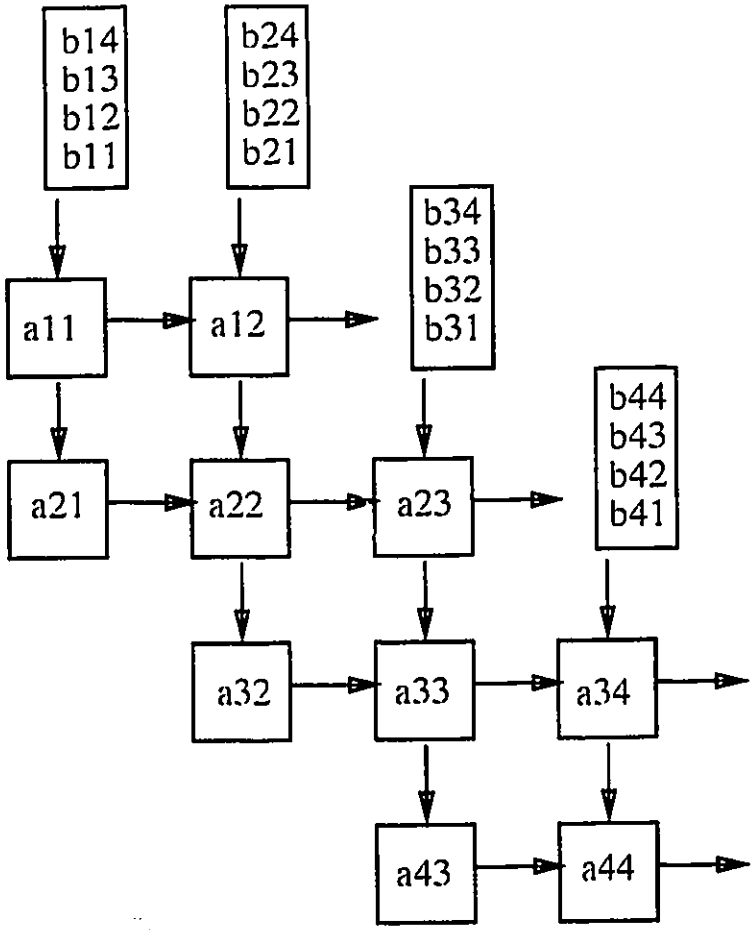


Figure 6.2: Systolic array for banded matrix multiplication

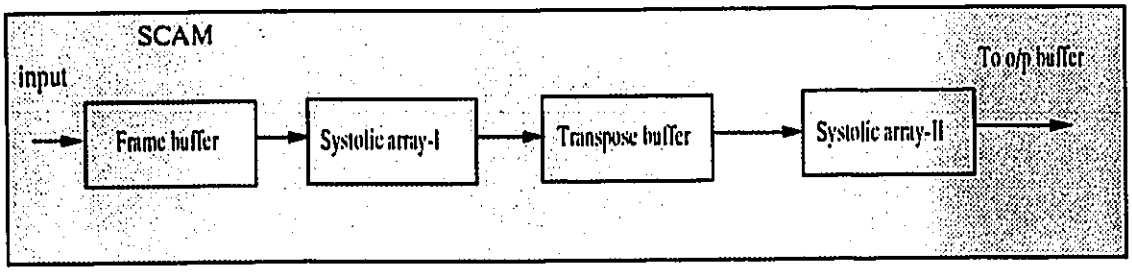


Figure 6.3: Overview of the proposed SCAM module

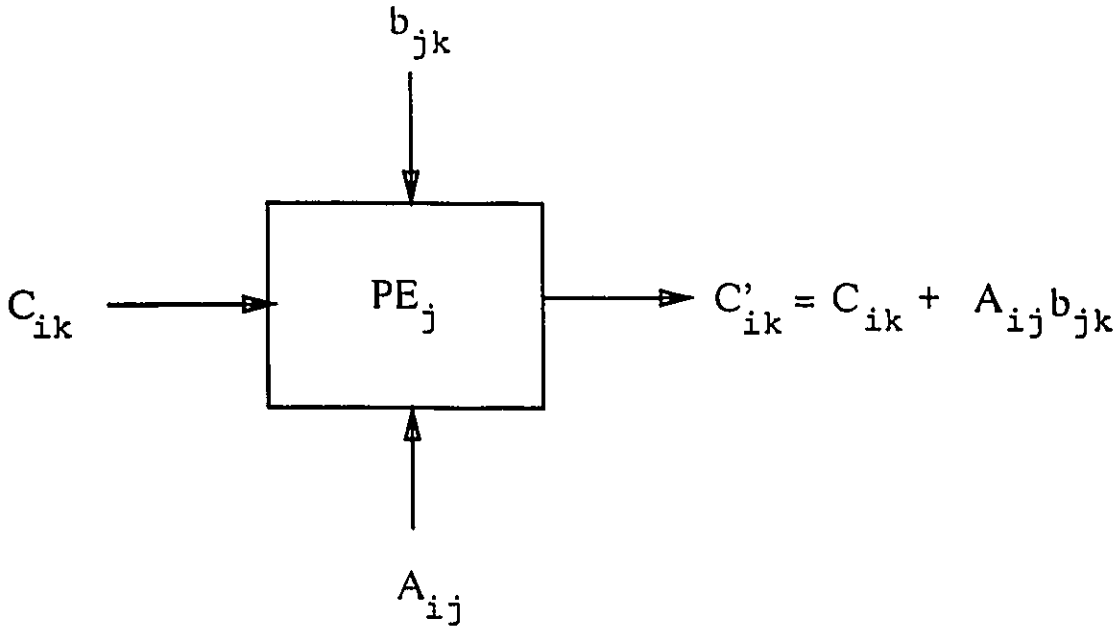


Figure 6.4: Example of the inner product operation

Value (V)	row (R)	column (C)	Ready to output (RT)	End of computation (ET)

Figure 6.5: Layout of the basic CAM cell

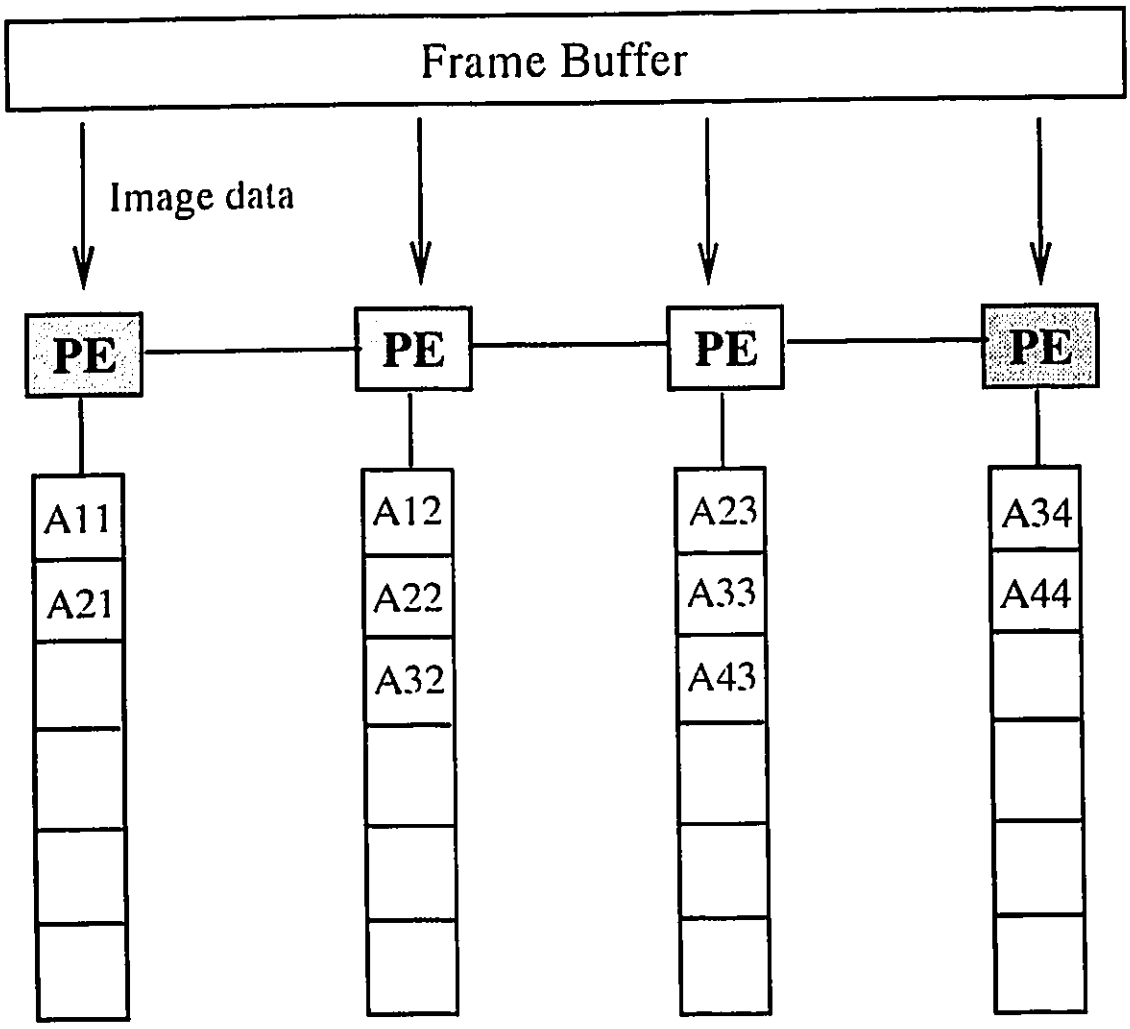


Figure 6.6: SCAM module for real-time Gabor decomposition

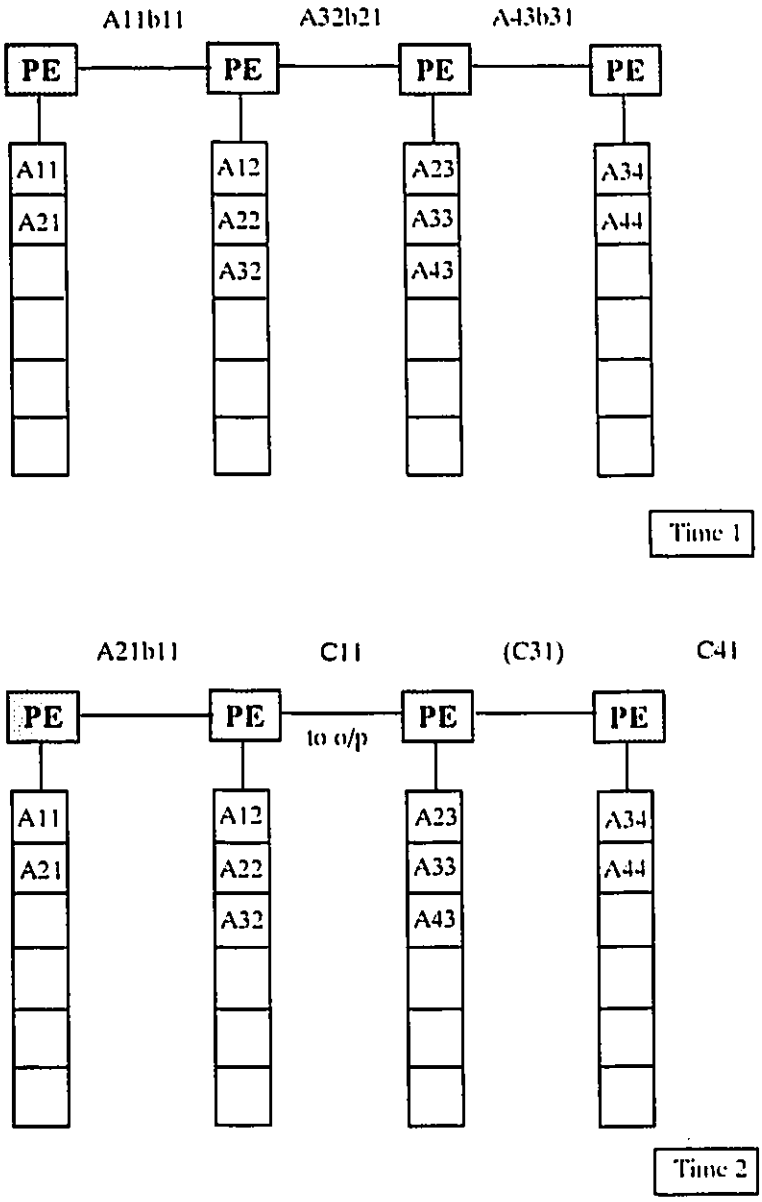


Figure 6.7: Snapshots of the SCAM module at time 1 and time 2

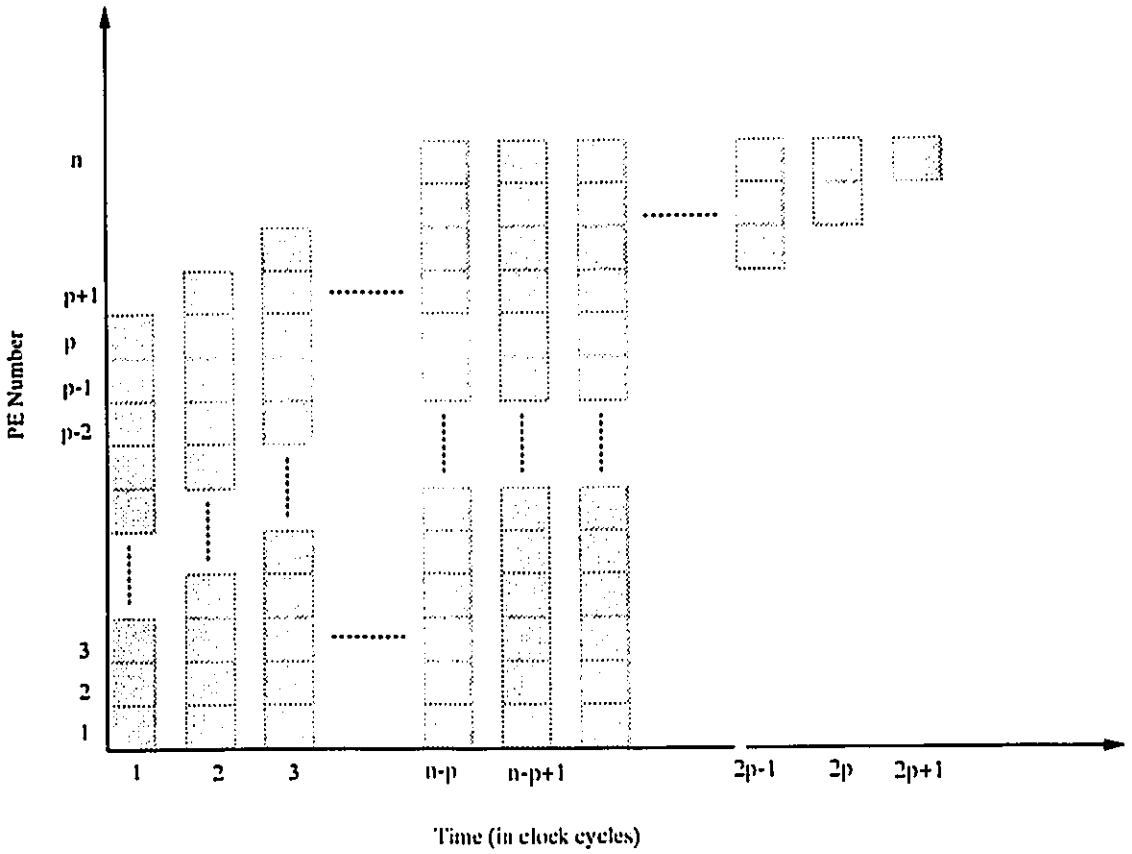


Figure 6.8: Cell occupancy diagram for the SCAM module

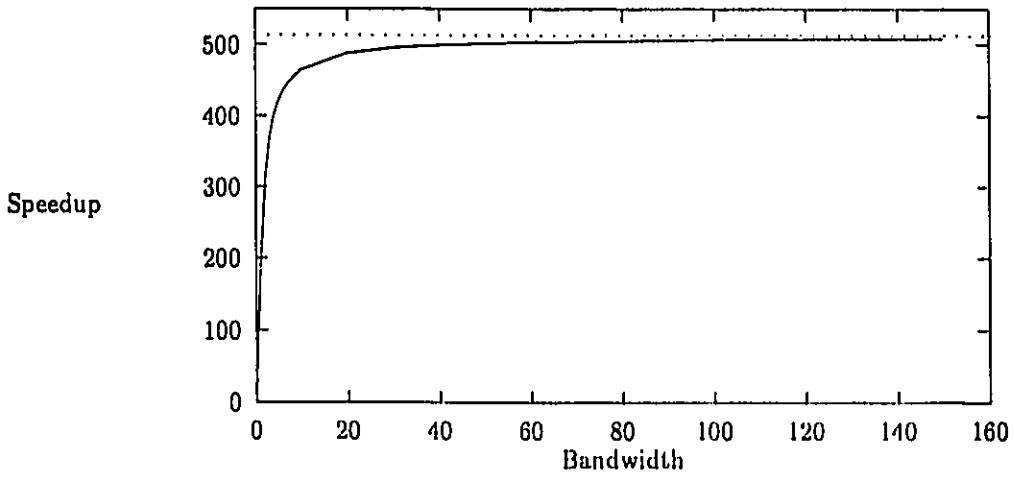


Figure 6.9: Speedup S_p versus Gabor kernel bandwidth p for 512 PEs

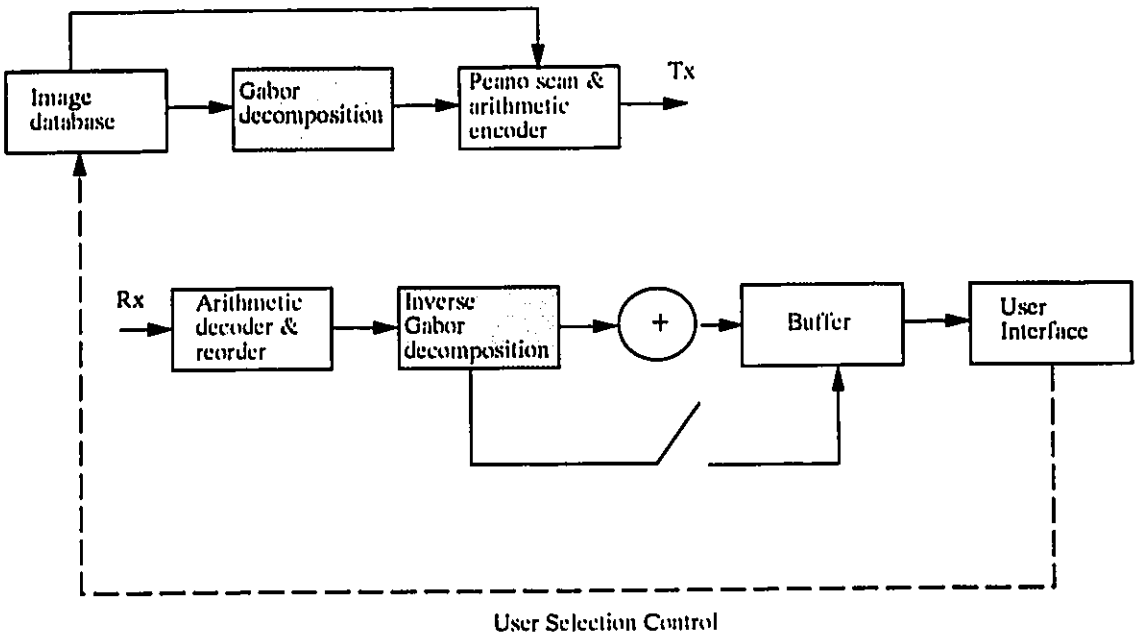


Figure 6.10: Codec for PIT by the CST algorithm using the SCAM architecture for Gabor decomposition

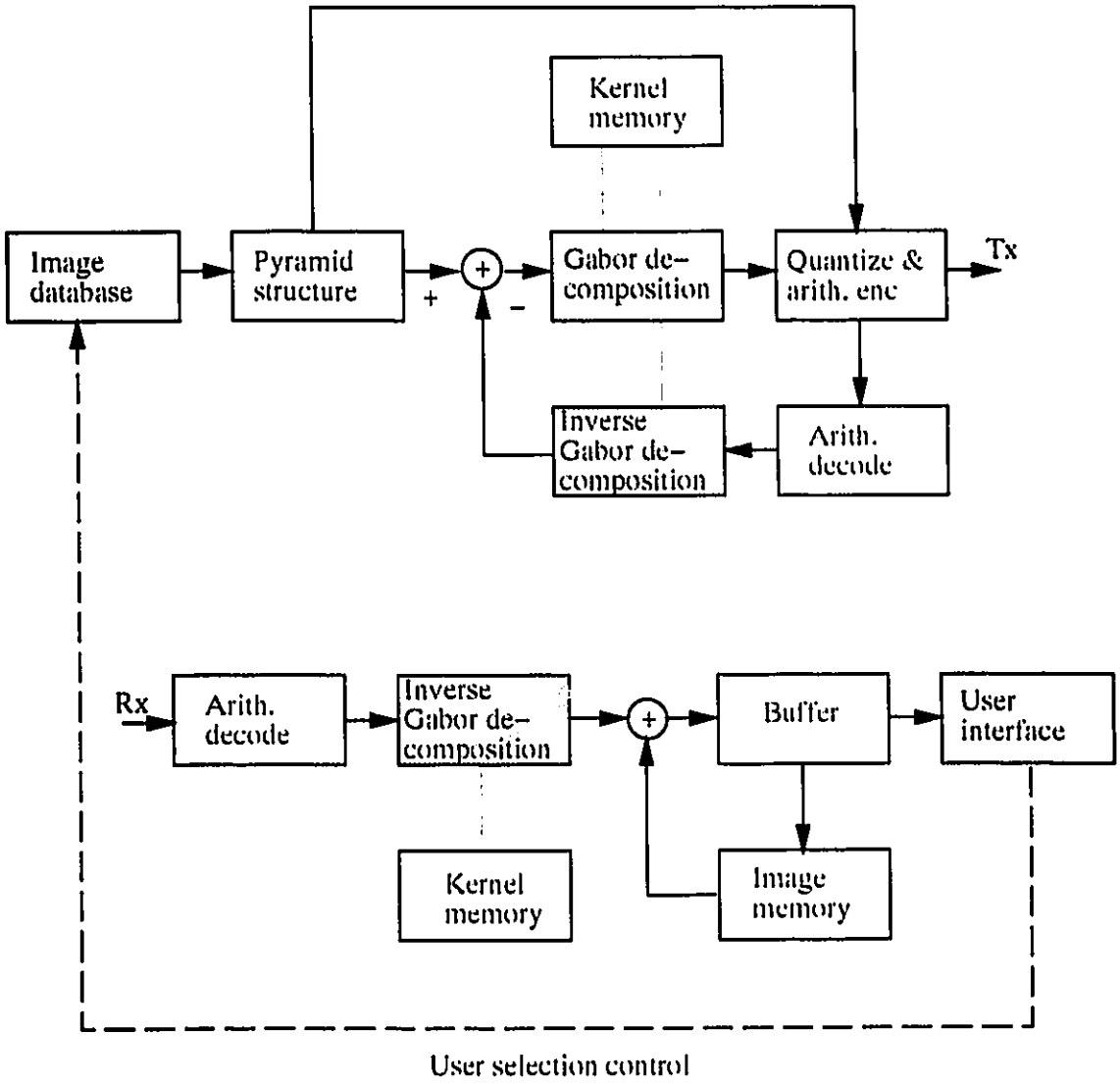


Figure 6.11: Codec for PIT by the PGT algorithm using SCAM architecture for Gabor decomposition

Chapter 7

Conclusions and Future work

7.1 Conclusions

In this thesis, a software tool for progressive image transmission has been presented. This tool is a user-friendly, unified platform for performing evaluations of multiple progressive image transmission schemes in terms of both objective and subjective criteria. The tool permits addition of new algorithms to the standard menus with ease. This tool can also be used to demonstrate progressive image transmission applications. The relationship between image types, bit rates and types of algorithms can be demonstrated using this tool. The software development environment used to build this tool permits rapid development of new algorithms. Example usages of this tool have been demonstrated in this thesis.

Two algorithms for progressive image transmission using Gabor decomposition have been presented. The first algorithm (CST) follows the coefficient scanning methodology. After transforming the input image into the joint spatial-frequency domain, the CST algorithm uses a peano scan combined with arithmetic encoding for efficient

transmission of the coefficients. It has a better subjective performance compared to the JPEG-DCT algorithm. The second algorithm (PGT) represents the image in a pyramid data structure and encodes different levels of the pyramid progressively using Gabor decomposition. Since the algorithm has an error feed forward mechanism, it is a very robust algorithm. This algorithm has a better objective and subjective performance compared to the JPEG-DCT algorithm.

Gabor decomposition is very expensive computationally. Hence, in order to implement the proposed algorithms in real-time, special architectures are required. In this thesis, a systolic array - content addressable memory (SCAM) architecture has been presented which is capable of executing Gabor decomposition in real-time. The SCAM architecture requires an $O(N)$ fewer processing elements to implement the Gabor decomposition compared to the ASP architecture. The high computational performance of the systolic arrays combined with the fast data retrieval of the CAM hardware is responsible for the efficient performance of this SCAM architecture. This architecture is easy to build as a codec in VLSI because of its simple and modular design. Performance analysis shows that this architecture achieves an efficiency of parallelism of 99 %.

7.2 Future work

The software tool is not designed to handle color image compression. Also, a database of recently coded images using various techniques will be useful in applications such as teaching. A higher level visual representation of the program will be of definite advantage and make the user interface very simple.

Gabor decomposition is promising as a technique for image compression. The proposed algorithms are based on seperable Gabor decomposition. Higher performance algorithms may be designed by using nonseperable Gabor decomposition. Moreover,

the proposed algorithms are for still image compression. Recently, digital video is gaining importance. Extensions of these algorithms for progressive video transmission applications is an area of considerable interest. Scalable video transmmission, i.e. simultaneous transmission of high definition and NTSC video information is a new area of interest. The PGT algorithm is particularly suited for such applications since, it has the structure to represent a higher and lower resolution version of the same signal. With the introduction of ATM, packet video is gaining popularity. Suitability of these algorithms for packetization is a possible reaserch direction.

Further reductions in the number of PEs in the proposed SCAM architecture may be possible by exploiting the properties of matrix multiplication. That is, combinations of the two 1-dimensional decomposition operations as one offers the possiblity of reduction in chip real-estate by half. Moreover, the CAM is a complex hardware. Substitution of the CAM hardware by other storage mechanisms such as a stack may provide a low cost VLSI implementation of the proposed design.

VLSI implementation of the proposed codecs is an important problem. The transfer of technology from concept to implementation opens up a new area of future research. Other issues include scalability of the proposed architecture for HDTV images.

Bibliography

[1] R. G. Gallagher, *Information Theory and Reliable Communication*, John Wiley, New York, 1968.

[2] N. S. Jayant and Peter Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video*, Prentice-Hall, Englewood Cliffs, New Jersey, 1984.

[3] C. E. Shannon, "Coding theorems for a discrete source with a fidelity criterion," *IRE National Convention Record*, Part 4, pp. 142-163, 1959.

[4] L. D. Davisson, "Rate-Distortion Theory and Applications," *Proceedings of the IEEE*, pp. 156-164, July 1972.

[5] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[6] J. O. Limb, "Distortion Criteria of the Human Viewer," *IEEE Transactions on Systems, Man and Cybernetics*, pp. 778-793, December 1979.

[7] J. A. Saghri, P. S. Cheatham and A. Habibi, "Image quality measure based on a human visual system model," *Journal of Optical Engineering*, pp. 813-818, July 1989.

[8] N. Abramson, *Information Theory and Coding*, McGraw-Hill, 1963.

[9] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the I.R.E*, pp.1098-1101, September 1952.

[10] R. G. Gallagher, "Variations on a theme by Huffman," *IEEE Transactions of Information Theory*, pp. 668-674, November 1978.

[11] T. S. Huang, "Run-length coding and its extensions," *Picture Bandwidth Compression,* pp. 443-448, Gordon and Breach, New York, 1972.

[12] D. Spencer, T. Huang, "Bit plane encoding of continuous pictures," *Symp. on Comput. Process. in Communications,* Polytechnic Institute of Brooklyn, New York, April 1969.

[13] I. H. Witten, R. M. Neal and J. G. Cleary, " Arithmetic Coding for Data Compression," *Communications of the ACM,* vol. 30, no. 6, pp. 520-540, June 1987.

[14] G. R. Seabrook, "Arithmetic coding - An Alternative VLC Strategy for Video Coding," *IEE Third Int. Conference on Image Processing,* University of Warwick, U.K., pp. 613-617.

[15] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* pp. 674-693, July 1989.

[16] I. Daubechies, "Orthonormal bases for compactly supported wavelets," *Communications of Pure and Applied Mathetatics,* pp. 909-996, November 1988.

[17] I. Daubechies, "Orthonormal bases of wavelets with finite support - connection with discrete filters," *Proceedings of International Conference on Wavelets,* pp. 38-66, 1987.

[18] I. Daubechies, "The wavelet transform, time-frequency localization and signal analysis," *IEEE Transactions on Information Theory,* vol. 36, no. 5, pp. 961-1005, May 1990.

[19] N. M. Nasrabadi, R. A. King, "Image coding using Vector Quantization: A Review," *IEEE Transactions on Communications,* pp. 957-971, August 1988.

[20] R. K. Jurgen, "Digital Video," *IEEE Spectrum magazine,* pp. 24-30, March 1992.

[21] M. Leonard, "Codec Chip Set Drives Standard Algorithms for Desktop Multimedia and Videoconferencing," *Electronic Design,* pp. 45-54, April 2, 1992.

[22] P. H. Ang, P. A. Ruetz and D. Auld, LSI Logic Corporation, "Video Compression makes big gains," *IEEE Spectrum magazine,* pp. 16-19, October 1991.

[23] Data Sheets, "CCITT Video Compression Chipset," *LSI Logic Corporation*, Milipitas, California.

[24] J. G. Daugman, "Complete Discrete 2-D Gabor Transforms by Neural Netwroks for Image Analysis and Compression," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 1169-1179, July 1988.

[25] F. Dufaux, T. Ebrahimi, M. Kunt, "A Massively Parallel Implementation For Real-Time Gabor Decomposition," *Visual Communications and Image Processing*, Boston, USA, November 1991.

[26] J. J. Modi, *Parallel algorithms and matrix computation*, Clarendon Press, Oxford, 1988.

[27] S. Panchanathan and M. Goldberg, "A Content-Addressable Memory Architecture for Image Coding Using Vector Quantization," *IEEE Transactions on Signal Processing*, vol. 39, pp. 2066-2078, September 1991.

[28] S. Panchanathan and M. Goldberg, "A Systolic Array Architecture for Image Coding Using Adaptive Vector Quantization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 1, pp. 222-229, June 1991.

[29] L. Wang and M. Goldberg, "Lossless progressive image transmission by residual error vector quantization," *Proc. IEEE Int. Montech'86 Conf. on antennas and Communications*, pp. 173-176, Montreal,Quebec, 1986.

[30] L. Wang and M. Goldberg, "Progressive image transmission using vector quantization on images in pyramid form," *IEEE Trans. Commn.*, Vol. Com-37, Dec. 1989.

[31] S. L. Tanimoto and T. L. Pavlidis, "A hierarchical data structure for picture processing," *Computer Graphics and Image Processing*, vol. 4, pp. 104-119, 1975.

[32] S. L. Tanimoto, "Image transmission with gross information first," *Computer Graphics and Image Processing*, vol. 9, pp. 72-76, January 1979.

[33] L. Wang, " Ph.D Thesis," *Dept. of Electrical Eng. University of Ottawa*, Ottawa, Canada, Jan. 1988.

[34] L. Wang and M. Goldberg, "Progressive image transmission by multistage vector quantization on images in pyramid form," *Proc. IEEE Int. Conf. on Communications*, Toronto, Ontario, pp. 419-423, June 1986.

[35] K. R. Sloan Jr., S. L. Tanimoto, "Progressive refinement of raster images," *IEEE Transactions on Computers*, vol. C-28, pp. 871-874, November 1979.

[36] K. Knowlton, "Progressive transmission of grey scale and binary pictures by simple, efficient, and lossless encoding scheme," *Proceedings of IEEE*, vol. 68, pp. 885-896, July 1980.

[37] F. S. Hill Jr., S. Walker Jr and F. Gao, "Interactive image query system using progressive image transmission," *Computer Graphics*, vol. 17, pp. 323-330, July 1983.

[38] A. Sanz, C. Munoz and N. Garcia, "Approximation quality improvement techniques in progressive image transmission," *IEEE Journal on Selected Areas in Communications*, vol. SAC-2, pp. 339-373, March 1984.

[39] H. M. Dreizen, "Content-driven progressive transmission of grey-scale images," *IEEE Transactions on Communications*, vol. COM-35, pp. 289-296, March 1987.

[40] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. COM-31, pp. 532-540, April 1983.

[41] J. Naor and S. Peleg, "Hierarchical image representation for compression, filtering and normalization," *Pattern Recognition Letters*, pp.43-46, October 1983.

[42] W. D. Hoffman and D. E. Troxel, "Making progressive transmission adaptive," *IEEE Transactions on Communications*, vol. COM-34 , pp.806-813, August 1986.

[43] A. Tran, K. M. Liu, K. H. Tzou and E. B. Vogel, "An efficient pyramid image coding system," *IEEE International Conference on ASSP*, Dallas, Texas, pp. 744-747, April 1987.

[44] K. N. Ngan, "Image display techniques using cosine transform," *IEEE Transactions on ASSP*, vol. ASSP-32, pp. 173-177, February 1984.

[45] E. Dubois and J. L. Moncet, "Encoding and progressive transmission of still pictures in NTSC composite format using transform domain methods," *IEEE Transactions on Communications,* vol. COM-34, pp. 310-319, March 1986.

[46] K. H. Tzou and S. E. Elnahas, "An optimum progressive transmission and reconstruction scheme for transformed image," *Proceedings of IEEE Int. Conf. on Commn.,* Toronto, Ont., pp. 413-418, June 1986.

[47] S. E. Elnahas, K. H. Tzou, J. R. Cox, R. L. Hill and R. G. Host, "Progressive coding and transmission of digital diagnostic pictures," *IEEE Transactions on Medical Imaging,* vol. MI-5, pp. 73-83, June 1986.

[48] L. Wang and M. Goldberg, "Lossless progressive image transmission by residual vector quantization," *Proceedings of the Thirteenth Biennial Symposium on Communications,* pp. c.1.9-c.1.12, Kingston, Ont., June 1986.

[49] L. Wang and M. Goldberg, "Progressive image transmission by residual error vector quantization in transform domain," *Proceedings of IEEE International Phoenix Conf. on Computers and Communications,* pp. 178-182, Scottdale, Arizoa, Februray 1987.

[50] L. Wang and M. Goldberg, "Progressive image transmission using vector quantization on images in quadtree form," *Proc. Int. Conf. on Digital Signal Processing,* Florence, Italy, Sept. 1987.

[51] T. H. Wendler and D. Meyer-Ebrecht, "Proposed standard for variable format picture processing and a codec approach to match diverse imaging devices," *SPIE,* vol. 318, (part 1), Picture archiving and communications systems (PACS) for medical applications, pp. 298-305, 1982.

[52] P. Boucher and M. Goldberg, "Transform image coding by vector quantization," *Ninth Symp. on Sig. Proc. and App.,* pp. 629-633, Nice, France, May 1983.

[53] Danielle Argiro, John Rasure, "An X Windows Based Applications Programming System," *XHibition '92,* San Jose, June 1992.

[54] Khoros Group, "Volumes 1,2 and 3 of the Khoros 1.0 Manual," University of New Mexico, January 1992.

[55] R. A. Earnshaw, N. Wiseman, "An Introductory Guide to Scientific Visualization," *Springer-Verlag*, Berlin, Germany, 1992.

[56] M. Porat and Y. Y. Zeevi, "The generalized Gabor scheme of image representation in biological and machine vision," *IEEE Trans. Pattern Anal. and Mach. Intelligence*, vol. 10, pp. 452-468, July 1988.

[57] J. G. Daugman, " Uncertainity relation for resolution in space,spatial frequency, and orientation optimized by two-dimensional visual cortical filters," *J. Optical Society of America A*, vol. 2, pp. 1160-1169, July 1985.

[58] S. Marcelja, "Mathematical description of the responses of simple cortical cells," *J. Optical Society of America*, vol. 70, pp. 1297-1300, November 1980.

[59] J. G. Daugman, " Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 36, pp. 1169-1179, July 1988.

[60] T. Ebrahimi, M. Kunt, "Image Compression by Gabor expansion," *Optical Engineering*, vol. 30, pp. 873-880, July 1991.

[61] T. Ebrahimi, "Ph.D. Thesis," *Ecole Polytechnique Federale de Lausanne*, Laussanne, Switzerland 1992.

[62] M. J. Bastiaans, "Gabor's expansion of a signal into Gaussian elementary signals," *Proceedings of the IEEE*, vol. 68, pp. 538-539, April 1980.

[63] K. Assaleh, Y. Zeevi and I. Gertner, "On the realization of the Zak-Gabor representation of images," *Visual Communications and Image Processing '91*, vol. 1606, pp. 532-540, November 1991.

[64] H. Kritikos and P. Farnum, "Approximate evaluations of Gabor expansions," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 17, no. 6, pp. 978-981, June 1987.

[65] F. Dufaux, T. Ebrahimi, M. Kunt, "A Massively Parallel Implementation For Real-Time Gabor Decomposition," *Proceedings of Visual Communications and Image Processing'91*, November 1991.

[66] H. T. Kung, "Why Systolic Architectures," *IEEE Computer*, pp. 37-46, January 1982.

[67] T. Kohonen, "Content addressable memories, second edition," *Springer Verlag*, Berlin, 1987.

[68] S-Y Kung, R. E. Owen and J. G. Nash, "VLSI signal processing II," *Editors, IEEE Press*, New York 1986.

[69] R. Charng and F-C Lin, "Linear Content-Addressable Systolic Array for Sparse Array Multiplication," *Chapter 19, VLSI signal Processing II, IEEE Press*, New York 1986.

[70] S. Panchanathan, "A Universal Architecture for Matrix Transposition," *IEE Proceedings, Part-E:Computer and Digital Techniques*, September 1992.

[71] P. A. Ramamoorthy and B. Potu, "Bit-serial systolic chip set for real-time image coding," *Proceedings of ICASSP*, vol. 2, pp. 756-759, April 1987.

[72] Benoit B. Mandelbrot, "The Fractal geometry of nature," *Publisher* , 1983.

[73] G. Iyengar and S. Panchanathan, "A systolic array architecture for real-time gabor decomposition," *Proceedings of VCIP'92*, vol. 1818, pp. 1006-1014, Boston, USA, Nov. 1992.

[74] G. Iyengar, S. Panchanathan and M. Goldberg, "Progressive Image Transmission using Gabor decomposition," *Proceedings of CCECE'92*, vol. 1, pp. TA.4.22.1-TA.4.22.4, Toronto, Canada, Sept. 1992.