MODEL BASED IMAGE COMPRESSION FOR TELEMEDICINE

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree of

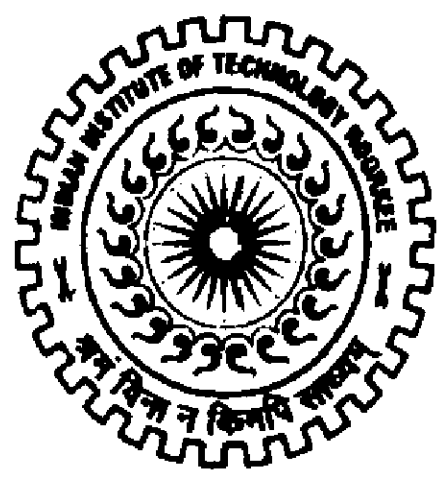
MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

SANDEEP JAIN



DEPARTMENT OF ELECTRONICS & COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE -247 667 (INDIA)

JUNE, 2007

CANDIDATE'S DECLARATION

I hereby declare that the work, which is being presented in the dissertation entitled "**MODEL BASED IMAGE COMPRESSION FOR TELEMEDICINE**" towards the partial fulfillment of the requirement for the award of the degree of **Master of Technology** in **Computer Science and Engineering** submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee (India) is an authentic record of my own work carried out during the period from July 2005 to June 2006, under the guidance of **Dr. Ankush Mittal , Associate Professor, Department of Electronics and Computer Engineering, IIT Roorkee.**

I have not submitted the matter embodied in this dissertation for the award of any other degree or diploma.

Date: 08-06-2007 **(Sandeep Jain)**

Place: Roorkee

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

Date: **(Dr. Ankush Mittal)**

Place: Roorkee Associate Professor

Department of Electronics and Computer Engineering

IIT Roorkee – 247 667

ACKNOWLEDGEMENTS

I would like to extend my heartfelt gratitude to my guide Dr. Ankush Mittal, Associate Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee, for his able guidance, regular source of encouragement and assistance throughout this dissertation work. It is his vision and insight that inspired me to carry out my dissertation in the field of Medical Imaging. I would state that the dissertation work would not have been in the present shape without his guidance and I consider myself fortunate to have done my dissertation under him.

I extend my sincere thanks to Dr. D. K. Mehra, Professor and Head of the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee for providing facilities for the work.

Finally, I would like to thank my parents and all my friends for their valuable suggestions and timely support.

SANDEEP JAIN

ABSTRACT

Image compression plays a key role in medical image archiving and transfer. Current compression schemes produce high compression rates, if loss of quality is acceptable. However, deficiency in quality cannot be accepted in the field of medicine. Any loss in information may lead to wrong decisions. Therefore a compression scheme is required which provides high compression ratio and is lossless. The scheme proposed in this dissertation work is lossless and provides high compression ratio. The proposed approach is model based and requires two major operations: registration of model and input images, and compression of residual image.

A novel Quadtree based Adaptive Block Partitioning with Rearrangement (QABPR) compression scheme has been proposed. The idea of using quad-tree as a primary data structure came after seeing the effective usage of quad-trees in fractal based compression algorithms. The idea of adaptive block partitioning is taken from various segmentation algorithms. Rearrangement was done to exploit the redundancies in blocks with similar intensities. Results have shown that proposed compression scheme works better than other compression schemes such as Huffman coding, GIF, JPEG-LS and JPEG2000.

Registration is done using minimization of RIS (Residual Image Size). Main idea of the proposed registration scheme is to use a criterion for alignment, which leads to maximum compression ratio. Results shows that proposed registration improves the results of compression significantly over mutual information based registration.

CONTENTS

CHAPTER 1

INTRODUCTION

Digital medical imaging provides powerful tools for diagnosis, treatment and surgery, hence becomes a major aspect of modern healthcare delivery. It is a fast growing area with the richest source of information and variety of modalities such as computerized tomography (CT), magnetic resonance imaging (MRI), positron emission tomography (PET), and many others. Digital systems have become an integral part of CT, MRI, PET and Ultrasound imaging and even traditionally non-digital techniques (e.g. film X-rays) are gradually evolving into computerized imaging

Hospital and clinical environments are moving towards computerization and digitization, resulting in large amounts of digital medical image data. Even though rapid progress has been made in mass storage density and computer network performance, the volume of digital imagery produced by hospitals has been increasing as well. Studies have shown that the radiology department of a large hospital can produce more than 20 terabits of image data per year [1,2]. This is the result of the high image resolution used in radiology, and the large number of images required for each examination. For example, a CT exam produces an average of 12 Mbytes of imaging data (30 images, 512x512 pixels each, 12 bits/pixel), whereas Digital Subtraction Angiography produces 21 Mbytes of imaging data (15 images/exam, 1024x1024 pixels each, 8 bits/pixel). In addition to the inherently digital modalities, digital data are also generated by digitizing X-ray images. A digitized chest X-ray is 4096x4096 pixels, 12 bits/pixel, which requires 24 Mbytes of storage space per image [2]. The demand for transmission bandwidth and storage space in the digital radiology environment continues to exceed the capabilities of available technologies. Therefore Compression techniques are essential in archival and communication of medical image.

Digital Image Compression can address these problems by reducing the data storage and transmission requirements. Many compression methods have been developed and have been evaluated for the medical environment. These compression methods usually reduce the size

of the data 2-4 times with no information loss, or more than 10 times with some information loss [2]. Despite the higher compression ratios of lossy compression methods, their use in medical imaging is limited because of concerns on losing image details [1,3,4]. Another reason for avoiding lossy compression is the development of computer aided diagnosis techniques. Computerized analysis of an image can use even the smallest details (e.g., very smooth variations in pixel intensities) which are often invisible to the eye. Compression methods should not lose any of these potentially important image details. For this reason, in medical imaging lossless compression is more important than lossy compression.

1.1 Telemedicine: An Overview

The use of advanced telecommunications and information technologies has been investigated for many years in an effort to improve health care. In particular, the focus has been centered on telemedicine, which is also referred to as telehealth in some arenas. Literally, telemedicine means medicine at a distance. Telemedicine has been defined as the electronically transmitted rapid exchange of medical information between sites of clinical practice for the purposes of relief and/or education [4]. Telemedicine is also defined as the use of electronic information and communication technologies to provide and support health care when distance separates the participants.

A broader definition is the use of telecommunication technologies to provide medical information and services. Telemedicine includes diagnosis, treatment, monitoring, and education of patients by using systems that allow ready access to expert advice and patient information no matter where the patient or relevant information is located [4, 5]. The use of telemedicine systems in hospitals, clinics, long-term care facilities and home care is becoming well established and is evolving in effectiveness and efficiency.

Telemedicine may be seen as a valuable tool for providing care to underserved areas, more efficient use of existing medical resources, and a way to attract patients living outside a hospital's area [6]. For telemedicine to be successful there must be an ability to clearly transmit a clinical situation, including clinical information of diagnostic quality, to a clinician

located far from the point of need and the ability for that clinician to effectively communicate concerns, additional requirements needed for diagnosis, or the provision of a diagnosis back to the point of need.

Telemedicine encompasses a wide variety of technologies ranging from the telephone to high technology equipment that enables health care professionals to provide health care away from the point of service. It includes interactive video equipment, fax machines, and computers along with satellites and fiber optics. Some of the leading goals of telemedicine are to enable an increase of the availability of services, instant access to data, access and exchange of data, a high quality of service, and reduction in the cost for the health service. Telemedicine is used in the following areas [4]:

1. Teleconsulting and assistance
    (i) Remote Supervision
    (ii) Remote Consulting
    (iii) Medical Videoconferencing
2. Virtual medical library
    (i) Document Distribution via WWW
    (ii) Medical Databases
    (iii) Research Databases
3. Virtual medical stores
    (i)Accounting
    (ii) Product databases
    (iii) Service brokers

The wide scope of applications for telemedicine includes patient care, education, research, and public health to diagnose, deliver care, transfer health data, provide consultation, and educate health professionals. In addition to these applications, telemedicine is utilized in a growing number of medical specialties by health providers.

1.2 Motivation

Medical expertise is not available in several areas of India. Many patients are sent elsewhere at considerable expense. In a number of these cases the treatment could have been carried out by the local doctor with advice from a specialist. There is a tendency for specialists to concentrate in the big cities making medical care in suburban and rural areas difficult. It is possible to transfer medical data (reports, mages, and videos) by using state of the art telecommunication tools.

The physical size of typical medical images, in terms of the number of bytes of data one image contains, is large e.g., a 1024 × 1024 image with 8 bits of data per pixel contains a megabyte of data [1,3] . To send these images over a low-bandwidth network requires considerable time. Image compression facilitates the transmission of images over the image network. Since compressed images contain fewer data, the transmission of a compressed image requires a smaller percentage of the capacity of the communications channel. This allows for a more efficient utilization of the communication network and increases the overall network throughput and decreases system response. Compression of the image translates directly into shorter transmission times.

Compression also helps in storage of medical data. To store these images over a long period of time requires a lot of storage. Consequently medical image compression helps to alleviate these large demands for medical image data storage and transmission capacity

Image quality and accuracy are of primary importance in medical environment. The process of compressing the image, and then reconstructing it, cannot induce errors, artifacts, or a loss of detail in the image [2,3,7]. If stray lines were to be seen in a reconstructed X-ray image owing to a flaw in the implementation of a particular compression algorithm, it is possible that the line could be interpreted as a hairline fracture rather than as the artifact that it is. There are other artifacts that can occur as well. Different image compression algorithms can yield different types of undesirable effects if the algorithms are not properly implemented [7]. Therefore a compression scheme with high compression ratio and without loss of quality

is required for medical images. A new compression scheme has been proposed in this report which provides a higher compression ratio without loss of quality.

1.3 Problem Statement

The aim of this thesis is to design and implement a new model based image compression framework for medical images. The Proposed framework provides high compression ratio without losing any information. The results show that proposed compression framework improves compression ratio considerably over current state of the art compression techniques.

The work done under this thesis can be listed as follows:

1. Design and implementation of a new registration technique that improves compression ratio and is simple to implement.
2. Design and implementation of a new compression technique for residual medical images.
3. Comparison of performance with other compression techniques like RLE, Huffman, GIF, JPEG-LS and JPEG2000.

1.4 Organization of the Report

The report is divided into seven chapters including this introductory chapter. The rest of this thesis is organized as follows.

Chapter 2 gives an overview of medical imaging issues and medical imaging representation standards. DICOM and PACS are introduced in this chapter. It also introduces registration and compression and presents their classifications.

Literature review and research gaps are presented in Chapter 3. This chapter gives an introduction to standard compression and registration techniques.

The proposed framework for compression discussed in Chapter 4. An Architecture diagram for the model based compression framework is explained in this chapter. This chapter contains an overview of the both compression and registration techniques presented in this dissertation work.

Chapter 5 provides design and implementation details of the proposed scheme.

Chapter 6 Results are shown in this chapter and performance of the proposed scheme is discussed. It also provides a comparative analysis of Maximization of mutual information and the proposed registration criterion.

Chapter 7 concludes the dissertation and gives some suggestions for future work.

CHAPTER 2

PRELIMINARIES

Medical imaging refers to the techniques and processes used to create images of the human body for clinical purposes. Medical imaging is a critical and rapidly expanding field. Over the past decade research into the processing and analysis of medical image data has begun to flourish

Medical imaging modalities and research tools provide anatomical, pathological, and functional information about human body [8]. The information is then utilized in different ways based on the selected treatment modality, i.e., for precision surgery, monitoring the effects of drugs etc.

The field of Medical imaging is of increasing importance in modern medicine. Applications of Medical imaging include Computer Assisted Diagnoses (CAD) and Computer Assisted Surgery (CAS), which strongly depend on MIP methods.

2.1 Medical Imaging Issues

A digital medical image is a digital image acquired by a certain radiological procedure. It is a two-dimensional M x N array of non-negative integers f(x,y), where $1 \leq x \leq M$ and $1 \leq y \leq N$ are the coordinates of anatomical structures in the image. The image segment represented by the coordinates (x,y) is called a picture element, or a pixel, and f (x, y) is its functional value or gray level. The radiological procedure can be X-rays, ultrasound, computerized tomography, nuclear magnetic resonance, or another digital modality. Depending on the digitization procedure or the radiological procedure, the gray level can range from 0 to 255 (8-b), 0 to 511 (9-b), 0 to 1023 (10-b), 0 to 2047 (11-b), and 0 to 4095 (12-b). These gray levels represent some physical or chemical properties of the object structure. As an example, in an image obtained by digitizing an X-ray film, the gray level value of a pixel denotes the optical density of the square area of that film. For X-ray computerized tomography (XCT),

the gray level value represents the relative linear attenuation coefficient of the tissue. For magnetic resonance imaging (MRI), it corresponds to the magnetic resonance signal response of the tissue [1].

A two-dimensional (2-D) radiological image has a size of M x N x k b, where k is the no of bits required for each pixel. The size of an image and the number of images taken in one patient examination vary with modalities [9]. Table 2.1 lists the average number of megabytes per patient examination generated by radiological imaging technologies.

Modality	Image Dimension	Gray Level	Avg No of Images per Exam	Avg Mbytes per Exam
CT	512 X 512	12	30	16
MRI	256 X 256	12	50	6.5
Ultrasound	512 X 512	6	36	9.5
PET	128 X 128	16	62	2
SPECT	128 X 128	8	50	0.8
Digitized Film	2048 X 2048	12	4	32

Table 2.1: List of radiological image properties

Main issues of medical image processing are discussed below

Restoration

Image restoration is a process of image enhancement, in which certain defects related to the physical acquisition process are removed. Two classical examples of image restoration are bias correction and noise reduction. MRI images are commonly corrupted by a multiplicative bias (a problem that occur in image modalities which leads to different intensity values for the same type of object at different spatial positions in image), which needs to be removed in order to obtain equivalent intensity values for the same tissue across the image (e.g. a similar intensity for all points in the brain corresponding to white matter) [10]. Bias correction

methods have been created mainly based on a classification process in which each point is classified according to the tissue it contains. Many techniques have been created to reduce the noise arising from the image acquisition process. Linear filtering methods apply low-pass filters which reduce noise, but tend to soften edges, thus resulting in fuzzy images. On the other hand, anisotropic diffusion techniques are very effective to smooth an image while preserving important discontinuities and their results are remarkable.

Segmentation

Segmentation is the process of extracting points, lines or regions, which are then used as inputs for complementary tasks such as registration, measurement, movement analysis, visualisation, etc. Although there is no general solution to the segmentation problem, there exist a set of mathematical tools and algorithms that can be combined to solve specific problems. Main methods for segmentation are thresholding, deformable models, multi-scale analysis, mathematical morphology and discrete topology and differential operators [11].

Interpolation

Interpolation is the process of determining the values of a function at positions lying between its samples. It achieves this process by fitting a continuous function through the discrete input samples. This permits input values to be evaluated at arbitrary positions in the input, not just those defined at the sample points.

Interpolation plays a crucial role in medical image processing. It is required for tomographic reconstruction, for medical image visualization (in 2D or 3D), for image registration (rigid-body or elastic matching), and feature extraction.

The interpolation techniques are divided into two categories, deterministic and statistical interpolation techniques. The difference is that deterministic interpolation techniques assume certain variability between the sample points, such as linearity in case of linear interpolation. Statistical interpolation methods approximate the signal by minimizing the estimation error.

Atlases, Morphometry and Statistical Analysis

Morphometry is the quantitative study of the geometry of shapes, and in particular it involves the computation of mean shapes and the analysis of variations about such mean. The definition of statistics about shape requires an appropriate formalization, since in general they are applied on differential manifolds that are not vector spaces (e.g. lines, planes, frames, oriented points, spatial rotations, etc) [10]. The applications concerned include inter-patient comparison.

Surgery simulation

This research field aims at the definition of geometrical and biomechanical models of organs and soft tissues to simulate in real time their deformation, cutting and stitching. Real time constraints imply that images must [12].

Medical robotics

The processes involved in passing from a simulation to the concrete realization of a robot-assisted surgical intervention have received a lot of attention in the last few years [12]. Two other major issues in medical image processing are medical image registration and medical image compression. These issues are discussed in section 2.3 and 2.4.

2.2 Medical Image Modalities

Medical imaging modalities can be divided into two major categories: anatomical and functional modalities. Anatomical modalities, depicting primarily morphology, include X-ray, CT, MRI and US etc. Functional modalities, depicting primarily information on the metabolism of the underlying anatomy, include SPECT, PET, which together make up the nuclear medicine imaging modalities.

X-Ray

Transmitting X-ray photons through the body onto an X-ray sensitive detector forms images. Some of the photons interact with tissue in the body and are scattered or absorbed; other

photons make it to the detector and are recorded. The resulting image is a transmission image showing the amount of X-ray attenuation along a path through the body (alternatively, the image can be formatted as an attenuation image showing how much of the energy was absorbed in the body). Dense materials absorb more X-ray photons than less dense materials [13]. Different absorption characteristics allow us to distinguish different materials (and provide contrast) in the image.

Magnetic resonance imaging (MRI)

MRI uses radio waves and a strong magnetic field to provide remarkably clear and detailed pictures of internal organs and tissues. This technique has proved very helpful to radiologists in diagnosing tumors of the brain as well as disorders of the eyes and the inner ear [14]. It requires specialized equipment and expertise and allows evaluation of some body structures that may not be as visible with other imaging methods. Feature extraction in MRI images is very difficult task because edge detectors may be sensitive to noise [15].

Computed Tomography (CT)

CT allows imaging of hard tissues while MRI allows imaging of soft tissues [15]. CT scans create images inside of the body that look like "slices". The scans are the most useful diagnostic study available for a number of disorders, such as head, spine, or chest injuries. The process begins with an x-ray that rotates around a stationary object, in this case a body, and passes a diverging beam through the object onto an array of several hundred detectors. The detectors then report the attenuation, or the strength of radiation each receives, to a computer which calculates tissue densities. The computer generates a picture based on the measured strength of the beam passing through a slice of the body. Dense tissue absorbs different amounts of radiation than soft tissue, making it possible to calculate tissue density by knowing the strength of the x-ray as it passes through the body at varying angles. Each pixel of the image is then assigned a numerical value depending on the calculated density for its corresponding unit of tissue. This value is then translated into a shade of gray [16]. The different shades of gray represent different tissues or organs within the body and make up the image that we see.

Positron Emission Tomography (PET)

PET is a method of medical diagnostics to provide information about processes in the human body. The data measured by a tomograph must be reconstructed and transformed into a graphical representation that can be evaluated. PET is a variant of tomography procedures to show metabolism processes in the human organism. The patient gets an injection of a so called tracer which is a substance that plays an important role in the metabolism of a human being. A tracer is radio-active and consists of isotopes with a short half-life period [16]. The radiation caused by the gamma rays is measured by the tomograph.

Single Photon Emission Computed Tomography (SPECT)

Similar to PET, SPECT uses radioactive tracers and a scanner to record data that a computer uses to construct two- or three-dimensional images of active brain regions (Ball). SPECT tracers are considered to be more limited than PET scanners in the kinds of brain activity they have the ability to monitor. The tracers of SPECT are longer lasting than those of PET, which allows for different, longer lasting brain functions to be examined, but this also requires more time for the SPECT to be completed [16]. The resolution of a SPECT is poor (about 1 cm) compared to that of PET. SPECT is often chosen over PET simply as a cost issue, for less equipment is involved and fewer staff is required to perform the tests.

Ultrasound

This medical imaging technique uses high-frequency sound waves and their echoes. The technique is similar to the echolocation used by bats, whales and dolphins. These portable machines display the distances and intensities of the echoes on the screen, forming a two-dimensional image. The main advantage is that certain images can be observed without using radiation. This lends itself well to obstetrics and gynecology, but ultrasound is also used in cardiology and urology. The great disadvantage of ultrasonography is that it produces very noisy images. It may therefore be hard to distinguish smaller features [8].

2.3 Medical Image Representation

Picture Archiving and Communication System (PACS)

One of the major burdens for a radiology department is providing adequate storage for films that provide a record of various types of imaging procedures, e.g., CT (Computed Tomography), and X ray. For legal reasons, in most of the hospitals, films need to be stored for 5 to 10 years as part of the associated medical examination and record **[1]**. To preserve the films for that long a period of time, they must be kept in a cool, humidity-controlled, and low-light environment. Given the number of examinations requiring an X ray, or a series of X rays, that are performed on a daily basis, the number of films that must be stored and maintained for 5 to 10 years is a burden for the hospital in terms of floor space taken for storage.

Picture Archiving and Communication System (PACS) give a framework for a more efficient method for the storage and transmission of medical images. One of the major capabilities provided by DIN/PACS is the ability to electronically store and retrieve digitized X-ray images, as well as images from other modalities. Once the images are in a digital format, they can be stored in a variety of ways that are more easily managed than films. PACS creates a single powerful work flow engine that allows radiologists to access comprehensive patient information at a single workstation via a single login anywhere in the enterprise. These systems operate in conjunction with MRI, CT, Digital X-ray, PET and Nuclear Medicine systems and eliminate the need to process hard-copy films (in turn, eliminating the need for toxic chemicals and storage of image history).

Image compression facilitates PACS as an economically viable alternative to analog film-based systems by reducing the bit size required to store and represent images, while maintaining relevant diagnostic information. It also enables fast transmission of large medical images over a PACS network to display workstations for diagnostic, review, and teaching purposes

Digital Imaging and Communications in Medicine (DICOM)

DICOM standard has been developed due to the emergence of different medical imaging modalities, which requires some standards for the purpose of storage and transmission. The

DICOM standard was created by the National Electrical Manufacturers Association (NEMA). Its aim is to support the distribution and viewing of medical images from CT, MRI and other medical modalities. The DICOM format is an extension of the older NEMA standard. A DICOM file contains a header and the image data. The header stores information about the patient's name, the type of scan, position and dimension of image and lots of other data. The image data part contains all the image information. DICOM supports following compression standards.

(1) RLE
(2) JPEG-LS
(3) JPEG 2000
(4) Zip/gzip (for non-image objects)

2.4 Medical Image Compression

The quick transmission of medical data (doctor's notice, annotations, cardiograms, single images or image series, audio documents, etc.) over the telephone network is an important aspect for telemedicine in the future. The problems lies in the transmission of large amounts of data in short time. Problem can be solved by compression of images on the server side, the transmission of these coded images over the data network and the decompression of the images on the client side. In the area of telemedicine, compression plays a key role.

Compression is often described as the removal of redundant information from a set of data that describes some object or some event. Image compression is a process that reduces the amount of data needed to represent an image while maintaining the "information" content of the image description. One must have a measure of the information content of an image, or any other set of data, in order to derive a useful compression algorithm and to be able to determine the effectiveness of the algorithm.

Entropy is a term borrowed from thermodynamics and used in digital communications to describe a measure of uncertainty or disorder in a system or series of events.

If X = {xl, x2 . . . . . xn}, which represents a sequence of events or values, then the entropy (designated as H) of this sequence is.

$$H(x) = -\sum_{i=1}^{N} P(x_i) \log_2 P(x_i) \qquad (2.1)$$

Where P(x_i) is the probability of event x_i occurring. The term - $\log_2 P(x_i)$ is the information conveyed (measured in bits) when event x_i occurs. It can be seen that the sum of the amounts of information conveyed by x_1 . . . . . x_n is directly proportional to the amount of uncertainty or disorder that exists in this sequence, a property that is measured in terms of entropy. It can be shown that the entropy is maximized when each of the events x_i has an equal probability of occurring; that is, if each event has the same probability, then it is just as likely that one event could occur as the next, and the sequence is said to have a maximum amount of disorder. The idea behind this discussion of entropy is that information is related to predictability or the correlation between expectation and outcome.

The concepts of information and entropy are the basis for image compression. Image compression is a process that reduces the amount of data needed to represent an image while maintaining the information content of the image. A basic characteristic of digital images is the existence of correlation between the gray level values of the pixels that comprise the image. It is this correlation that the image-compression process exploits to arrive at a compressed version of an image.

There are two basic approaches to compressing an image. Both seek to preserve the information content of the original image, but the two approaches are quite different in nature.

The first category of methods yields a compressed image from which the original can be exactly reconstructed. There exists a direct, causal relationship between the gray-level values in the original image, the reconstructed image, and the representation of the image in the

compressed format. This approach can be called "data preserving." These are deterministic methods of coding an image such that no information is lost and an exact replica of the original image can be generated from the compressed image. Examples of this compression types are Run Length Encoding, Huffman Coding, Differential Pulse Code Modulation (DPCM), JPEG-LS and JPEG 2000 etc. The compression ratio for images compressed using one method or another, varies from one image to the next

A lossy image compression is one where compressing data and then decompressing it retrieves data that may well be different from the original, but is close enough to be useful in some way. Examples of lossy image compression are JPEG (DCT based coding), JPEG 2000(Wavelet based coding) etc.

2.5 Medical Image Registration

Model based compression schemes involve registration as a first step before compression. Registration enhances compression by reducing the entropy of the image.

Registration refers to the process of finding a transformation matrix that best aligns a floating image to a reference image. For registration two functions are required.

(a) For similarity criteria
(b) For optimization

Similarity criterion is used to determine whether two images are aligned or not. Similarity criterion should be minimum or maximum when two images are best aligned.

Optimization procedure is used to search a transformation matrix for which Similarity criterion is optimum

Information gained from two images acquired during treatment is usually of a complementary nature; proper integration of useful data obtained from the separate images is

often desired. A first step in this integration process is to bring the modalities involved into spatial alignment, a procedure referred to as registration [17,18]. After registration, a fusion step is required for the integrated display of the data involved.

Besides multimodality registration, important application areas exist in monomodality registration. Examples include treatment verification by comparison of pre- and post-intervention images, comparison of images during and between seizures. Because of the high degree of similarity between these images, solving the registration is usually an order of magnitude easier than in the multimodality applications [18]. In this dissertation work registration is used for compression purpose.

Classifications of Registration

The classification of registration methods used in this section is based on the criteria formulated by V.D Elsen (1993) [18]. A version considerably augmented and detailed is presented. Nine basic criteria are used, each of which is again subdivided in one or two levels [19]. The nine criteria and primary subdivisions are:

Dimensionality

According to whether all dimensions are spatial, or that time is an added dimension, this criterion is subdivided into spatial dimensions only and times series (more than two images) with spatial dimensions. In either case, the problem can be further categorized depending on the number of spatial dimensions involved: 2D-2D, 2D-3D, and 3D-3D [18]. In this work registration of 2D images is done.

Nature of registration basis

This category can be further divided into image-based and non-imaged based. Image-based registration can also be subdivided into extrinsic, i.e. based on foreign objects introduced into the imaged space, and intrinsic methods, i.e. based on the image information as generated by the patient. If the imaging coordinate systems of the two scanners involved are somehow calibrated to each other, registration of multimodal images can be non-imaged based, but it is usually necessary that the scanners are brought into the same physical location, and the

patient remains motionless between both acquisitions [18]. In this work registration is image based.

Nature of transformation

The nature of transformation can be divided into rigid, affine, projective, and curved as shown in figure 2.1. An image coordinate transformation is called rigid, when only translations and rotations are allowed. If the transformation maps parallel lines onto parallel lines it is called affine. If it maps lines onto lines, it is called projective. Finally, if it maps lines onto curves, it is called curved of elastic [18,19]. Only rigid transformations are applied, in this work.

Domain of transformation

A transformation is called global if it applies to the entire image, and local if subsections of the image each have their own transformations defined as shown in figure 2.1. In this work registration is global.

Interaction

There are three levels of interaction which are automatic, interactive and semi-automatic that can be recognized according to the interaction between user and software [19]. Automatic registration is done for CT and MRI images. Semiautomatic registration is done for SPECT images.

Optimization procedure

The parameters that make up the registration transformation can either be computed directly, i.e. determined in an explicit fashion from the available data, or be searched for, i.e. determined by finding an optimum of some function defined on the parameter space. So the optimization can be further divided into parameters computed and parameters searched for.

Some of the most commonly used optimization procedures are Powell's method, Downhill simplex method, Levenberg-Marquardt optimization, simulated annealing Genetic methods

and Quasi-exhaustive searching [19]. Downhill simplex method is used here for minimization of RIS.

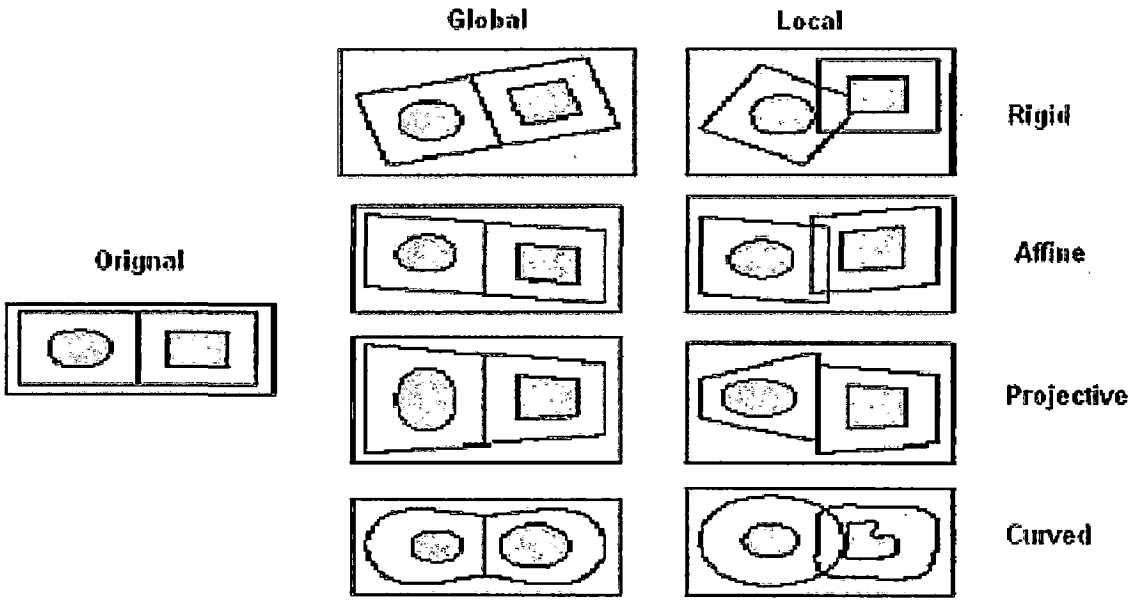


Figure 2.1 Domain of Transformation and Nature of Transformation

Modalities involved

Four classes of registration tasks can be recognized based on the modalities that are involved as shown in table 2.2 [19]. Registration can be monomodal, multimodal, modality to modal or patient to modality. Registration, in this work is modality to model.

Subject

Based on the images involved in a registration task, there are three classes of subject, which are intra-subject (using two images of a single patient), inter-subject (using two images of different patients) and atlas (one image acquired from a single patient and the other image constructed from an image information database which is obtained using imaging of many subjects). Inter-subject registration is required here.

Object

The objects include head, thorax, abdomen, pelvis and perineum, limbs, spine and vertebrae, and so on.

Type of Registration	Modalities for which registration is to be done.	Description
Monomodality	Auto-radiographic; CT to CTA; MR; PET;Portal; SPECT; US; X-ray; etc.	Images to be registered belong to the same modality.
Multimodality	CT-MR; CT-PET; CT-SPET; PET-MR; PET-US; SPECT-MR etc.	Images to be registered stem from two different modalities.
Modality to model	CT; MR; SPECT; X-ray	Only one image is involved and the other modality is a model.
Patient to modality	CT; MR; PET; Portal; X-ray	Only one image is involved and the other modality is the patient himself

Table 2.2 Types of registration based on Modalities

CHAPTER 3

LITERATURE REVIEW AND RESEARCH GAPS

3.1 Literature Review

Various methods, both for lossy and lossless image compression are proposed in the literature. The recent advances in the lossy compression techniques include different methods such as vector quantization, wavelet coding, neural networks and fractal coding. Although these methods can achieve high compression ratios (of the order 50:1, or even more) they do not allow reconstructing exactly the original input data. Lossless compression techniques permit the perfect reconstruction of the original image, but the achievable compression ratios are only of the order 2:1, up to 4:1.

The first lossy and lossless still image coding standard JPEG appeared by the end of 1980's and had a remarkable performance at that time. The JPEG file format was named for the committee that wrote the standard. It is the best-known and most popular compression standard. The standard uses a block discrete cosine transform (DCT). The DCT was shown to be very close to the optimal decomposition for the class of natural images, performing almost perfect decorrelation of image pixels [7]. The standard uses an ad-hoc Huffman or arithmetic entropy coding of transform coefficients. The version with arithmetic coding exploits binarization of the coefficients and a simple heuristic context modeling. It was originally designed to deal with color images, as it can store 24 bits per pixel of color data (instead of 8 bits per pixel) and restore millions of colors. Compression level is usually expressed in ratios (2:1, 10:1, 20:1, and so on).

After the success of the series of JPEG standards, the committee decided it was appropriate to revisit the lossless coding mode within JPEG. This mode had been a late addition to the standard, and was not really close to 'state of the art' techniques. In addition, it really was not closely related to the block based DCT techniques.

Looking at user requirements, particularly those of the medical imaging business which was concerned about potential large errors being introduced through lossy compression, the scope of the new standard was defined as effective lossless and near lossless compression of continuous-tone, grey scale and color still images. By near lossless, it was agreed that a scheme was needed that guaranteed a maximum error between the original image data and the reconstructed image data. A number of contenders for a suitable algorithm were proposed, and the committee carried out an extensive set of measurements to determine the best contender. The winner of these objective tests was agreed to be the LOCO (LOw COmplexity Lossless Compression) algorithm from HP Labs.A major side effect of undertaking this standards activity was that some of the other contenders such as CALIC, FELICS, and Ricoh's CREW algorithm in particular had some very attractive features - for example, in the ability to provide a single code stream which could provide lossy and lossless images without additional processing. Although outside the direct scope of JPEG-LS, these features and the discussions they provoked directly led to the development of the architectural approach of the new JPEG 2000 standard.

The JPEG standard remains the most popular image compression algorithm until the introduction of a (discrete) wavelet transform (DWT) in the middle of 1980's [21,22] Wavelet transform (DWT) launched a new era in image compression. Besides being good for decorrelation of image pixels, the new transform provides more functionality to the compression algorithm, such as embedded coding, where any initial part of the compressed bit stream can be used to reconstruct the image with the quality in proportion to the length of this part. A benchmark method called the Embedded Zerotree Wavelet (EZW) coding was introduced in [23] and further developed in the SPIHT (Set Partitioning in Hierarchical Trees) algorithm [24]. The use of integer-to-integer (reversible) wavelet transform [22] allowed for progressive coding up to the lossless image reconstruction [25]. JPEG 2000 is the new standard for wavelet compression issued by the JPEG Committee. Among the advantages of JPEG 2000 over the other compression algorithms are: (i) A superior lossy performance at low bit rates. (ii). Integrated lossy and lossless operation with scalable lossy-to-lossless decoding option. (iii) Progressive transmission. (iv) with a broader range of image types.

3.2 Image Compression Methods

3.2.1 Entropy Based

In information theory an entropy encoding is a data compression scheme that assigns codes to symbols so as to match code lengths with the probabilities of the symbols. Typically, entropy encoders are used to compress data by replacing symbols represented by equal-length codes with symbols represented by codes where the length of each codeword is proportional to the negative logarithm of the probability. Therefore, the most common symbols use the shortest codes.

According to Shannon's source coding theorem, the optimal code length for a symbol is $\log_b P$, where b is the number of symbols used to make output codes and P is the probability of the input symbol. Most common entropy encoding techniques are Huffman coding and arithmetic coding. These methods are limited to compression ration 1:2 to 1:3 [26]. These methods are not used directly for image compression. Entropy based methods are used as a post-processing step.

3.2.2 Transformation based

The image transformation, sometimes also referred as decorrelation, is used to reduce the dynamic range of the signal, to eliminate redundant information, and to provide a suitable representation for efficient entropy coding. A transformation should satisfy three conditions. First, all transform coefficients become statistically independent. Decorrelation with concomitant entropy reduction is important so that the transformed outputs are suitable for entropy encoding. Second, the energy of the transformed image is compacted into a minimum number of coefficients. Efficient energy compaction requires good localization of the sampling functions in both the spatial and the frequency domains. Third, the transform coefficients are concentrated in minimum frequency or transform scale region. As shown in figure 3.2, transformation based compression involves 3 steps.

1. Transformation: Over the years, a variety of linear transforms have been developed which include Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT and many more, each with its own advantages and disadvantages

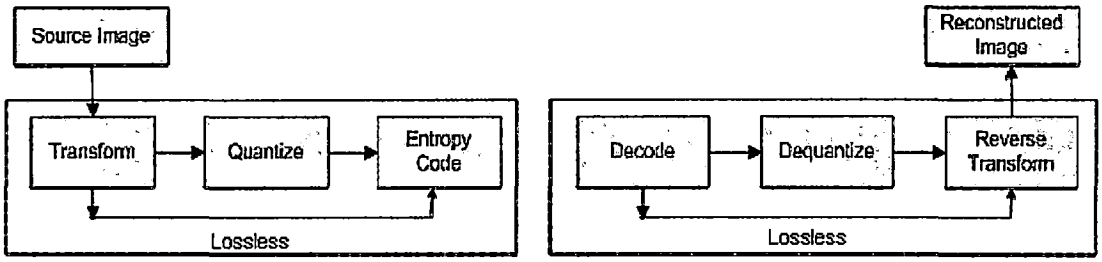


Figure 3.2 General framework of transform based compression.

2. Quantization: A quantizer simply reduces the number of bits needed to store the transformed coefficients by reducing the precision of those values. Since this is a many-to-one mapping, it is a lossy process and is the main source of compression in an encoder. Quantization can be performed on each individual coefficient, which is known as Scalar Quantization (SQ). Quantization can also be performed on a group of coefficients together, and this is known as Vector Quantization (VQ). Both uniform and non-uniform quantizers can be used depending on the problem at hand. For an analysis on different quantization schemes

3. Encoding: An entropy encoder further compresses the quantized values losslessly to give better overall compression. It uses a model to accurately determine the probabilities for each quantized value and produces an appropriate code based on these probabilities so that the resultant output code stream will be smaller than the input stream. The most commonly used entropy encoders are the Huffman encoder and the arithmetic encoder, although for applications requiring fast execution, simple run-length encoding (RLE) has proven very effective.

3.2.3 Fractal Based

Fractal compression is a lossy compression method used to compress images using fractals. The method is best suited for photographs of natural scenes (trees, mountains, ferns, clouds). The fractal compression technique relies on the fact that in certain images, parts of the image resemble other parts of the same image. Michael Barnsley led development of fractal compression in 1987.

Fractal compression appeared to be a promising technology in the late 1980s, when in some circumstances it appeared to compress much better than JPEG, its main competitor at that time [1]. However, fractal compression never achieved widespread use. Fractal compression is much, much slower to compress and much slower to decompress than JPEG. Furthermore, its patents were not widely licensed. Also, the improved compression ratio is an illusion.

3.2.4 Prediction Based

These compression algorithms include 2 steps as follows:

1 The value of an image pixel is predicted as a weighted average of the preceding neighboring points.

2 The difference between actual value and predicted value is represented using arithmetic coding.

Predictive coding, such as the differential pulse code modulation (DPCM) predicts the next sample based on the previously coded samples and codes the error between the prediction and the actual sample. The main objective of DPCM is to narrow down the range of coded samples. However, images are not stationary in texture, and some predictors may work in certain areas, but not in others. Hence, the concept of adaptive predictors was introduced so that the coder and decoder could choose from a set of predictors at the cost of transmitting the parameters needed for the prediction. This method of coding needs to limit prediction error for efficient coding and achieves lossless compression in coding the entire error; lossy coding is achieved by limiting the range of the error value.

3.3 Research Gaps

Medical imaging requires lossless image compression. Most of the proposed algorithms are either lossy or have low compression ratio. For transmission of medical images in low-bandwidth regions, a compression algorithm with high compression ratio and without loss of quality is required.

M. J. Zukoski [27] proposed a new model based approach for medical image compression. In his work, he proposed a model based framework for medical images. He used JPEG 2000 compression for residual images and used maximization of mutual information as a registration criterion.

The model based compression proposed by M. J. Zukoski is not completely lossless. According to his approach, input image needs to be registered (aligned). Registration involves transformation of input image such as rotation, scaling, etc. and information may loss in these operations. Model based compression can be made lossless by aligning model image and sending transformation parameters with residual image.

His work improves compression ratio but he used mutual information based registration. Mutual Information based registration is best suited as a initial step for integration of information from multimodality images. For compression, model image and input image are of same modality and here registration is not used as initial step of integration. Therefore a better registration criterion can improve compression ratio. Furthermore time requirements of mutual information based registration are high.

JPEG 2000 compression is based on Wavelet technology, which although renders highly compressed images better than regular JPEG, is a quite a complex technology with complex coding. This increases memory requirements and decreases performance. JPEG 2000 files generally load two to three times slower than regular JPEG files. Furthermore JPEG 2000 is a general purpose compression algorithm. There can be an algorithm which can exploit the enormous redundancy in residual images and perform fast for these images by avoiding areas of zero intensity.

CHAPTER 4

AN OVERVIEW OF MODEL BASED FRAMEWORK FOR MEDICAL IMAGE COMPRESSION

Work done in this dissertation, addresses the problem of obtaining high compression ratios for the lossless coding of images. In medical imaging it is not acceptable to lose any information. Discarding of small image details that might be an indication of pathology could alter a diagnosis, with severe human and legal consequences. In several medical applications, like coronary angiography, where one has to measure sub-millimeter blood vessel diameters, lossy coding methods are obviously inadequate. More generally, for measurement (quantitative) images, one usually cannot afford information loss due to compact coding. A model-based framework is proposed for medical image compression.

4.1 Overview

Architecture of proposed system for model based image compression is shown in Figure 4.4. A model image is searched for the input image (image that needs to be compressed). The obtained model image is then aligned to the input image. Once the images are aligned, the registered model image is subtracted from the input image, generating a residual image. It needs to be noted that model image is aligned instead of input image. Aligning model image instead of input image makes the compression framework completely lossless. The only extra information that needs to be sent is scaling factor and rotation angle which will take only few bytes. The residual image is compressed using proposed QABPR compression technique. Overall framework for compression involves 6 major blocks. These blocks are explained in following sections

4.2 Selection of Model Image

A prestored database of common image types is stored. Images serve as templates and the collection serves as atlas.

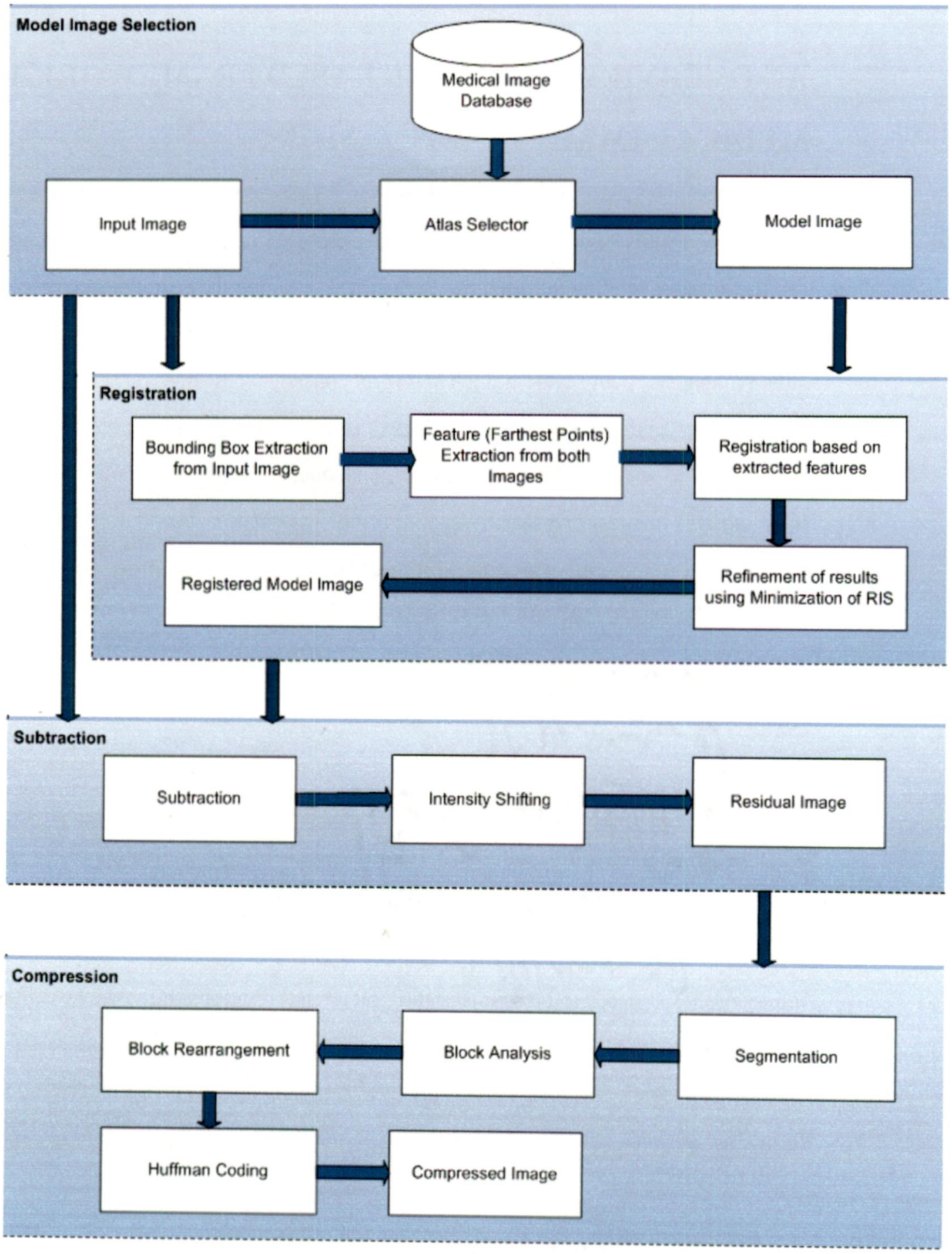
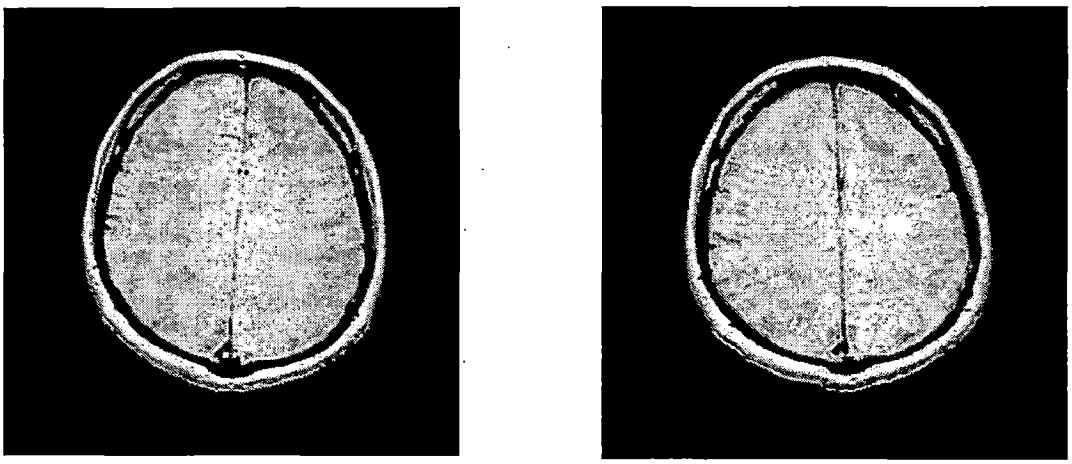


Figure 4.4: Architecture of the proposed model based compression scheme

When an image is taken, gender and age can be determined from the header information stored with image. The header will also indicate what body part is being imaged (head, chest, foot, etc). Atlas selector uses this information to find appropriate model image from atlas.



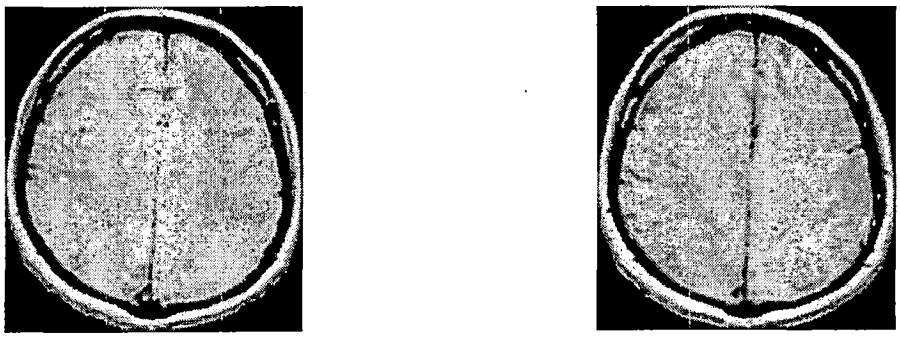
(a) (b)

Figure 4.1: MRI images of brain: (a) Model image (b) Image to be compressed

4.3 Registration

After selection of model image, the model image is registered with the input image. It will require a lot of time, to search for all possible transformations and is practically infeasible. Therefore, coarse followed by fine registration is used in the model based compression scheme. For coarse registration, features are extracted from both the images and initial transformation matrix is calculated with the help of these features.

Results of registration are refined using minimization of RIS. According to proposed criterion, two images are best aligned, when size of their compressed residual image is minimized. Registration is explained in next chapter.



(a) (b)

Figure 4.2: (a) Model Image after bounding box extraction (b) Uncompressed image after registration

4.4 Image Subtraction and Intensity Shifting

Registered model image is subtracted from the input image. Difference image contain negative values. To avoid these negative values intensities are shifted by minimum of all values and minimum is stored. These residual images are very low entropy images and contain excessive redundancy.

4.5 Header Construction

As discussed in section 4.2, the input image contains information about patient and the information about the part of body. This information, transformation parameters obtained from registration in section 4.3 and intensity shift obtained in section 4.4 are clubbed together and work as a header for the model based compression scheme.

4.6 Compression

Obtained residual image is compressed using proposed Quadtree based Adaptive Block Partitioning with Rearrangement (QABPR) compression method. QABPR compression decomposes image in blocks recursively. And analyze these blocks. After analysis step blocks are either stored or are rearranged. Rearranged blocks are further decomposed. Compression is explained in next chapter

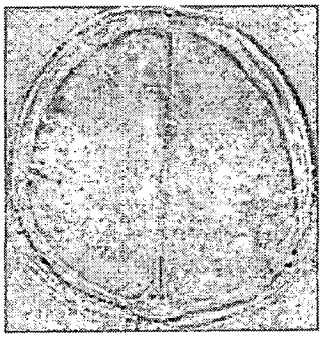


Figure 4.3: Image obtained after subtraction and intensity shift

4.6 Reconstruction

Reconstruction is straightforward. The compressed residual image is first uncompressed. The patient information and information about the part of body is extracted from the header of residual image. Atlas selector uses this information and selects appropriate model image. The transformation parameters are extracted from the header and transformations are applied to the model image. Therefore, an aligned model image is obtained and the obtained aligned image is added to the received residual image.

CHAPTER 5

A UNIFIED REGISTRATION AND COMPRESSION SCHEME

As discussed in previous chapter, the model based compression scheme comprises of two major steps registration and compression. Images are first registered, and then residual images are compressed. Design and implementation of these two steps are discussed in detail, in following sections.

5.1 Registration

Registration is often used to align two or more images of one or more than one modality. Multimodality registration is used to get more information about an object. For example SPECT images provide functional information of brain while MRI images provide anatomical information. In order to get more precise information like what metabolic activity is going in a particular part of brain, these images need to be aligned. Another common use of registration is to measure healing process by registering series of images after operation.

Many methods have been proposed for registration. These method include Landmark based, Intensity difference based, Correlation based, Segmentation based and Mutual Information based. Among all these methods, Mutual information based registration is the most widely used and discussed registration method [17,18].

5.1.1 Mutual Information Based Registration

Mutual information is a measure of how well one image explains the other, and is maximized at the optimal alignment [17]. Mathematically mutual information can be defined as.

Let A and B be two input images

$$I(A,B) = H(A) + H(B) - H(A,B) \tag{5.1}$$

Where I(A,B) is mutual information, H(A) is entropy of 1st image A, H(B) entropy of 2nd image B, H(A,B) is entropy of Joint Histogram of images A and B. Entropy of an image can be calculated using eq. 2.1.

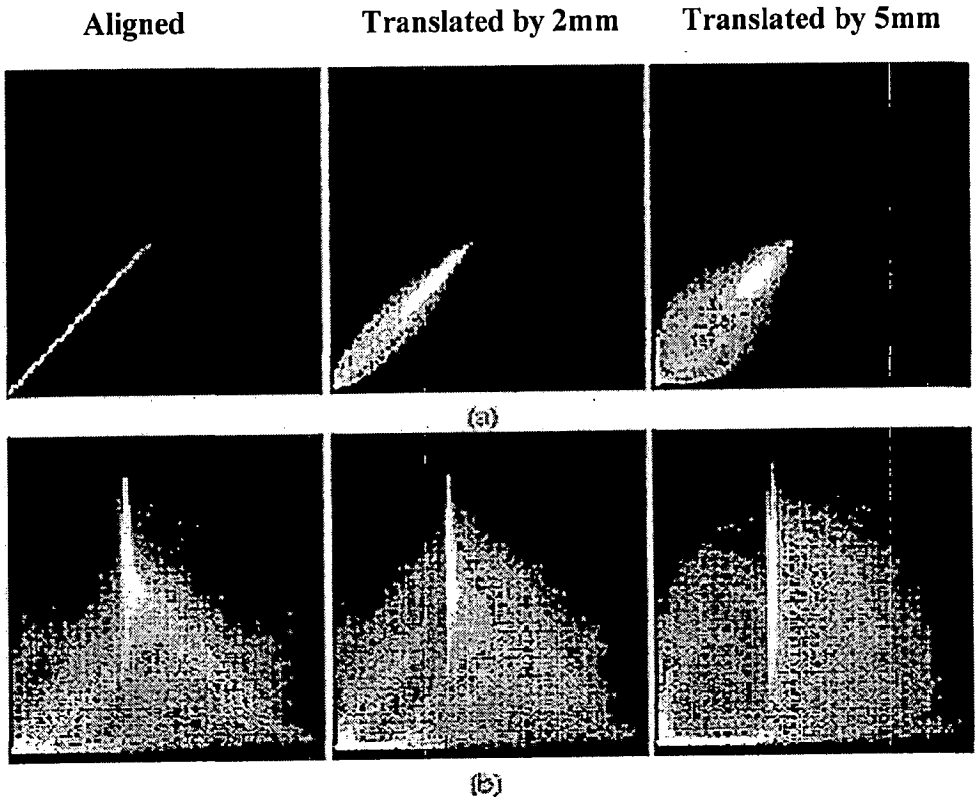


Figure 5.1: Example 2D histograms of the head images (a) identical MR images, (b) MR and CT images

Mutual information is maximized when entropy of joint histogram is minimal or when the joint histogram is crispiest. When two identical MR images are aligned histogram is crispiest as shown in figure 5.1(a). A straight line in joint histogram is obtained here since intensities are linearly related (Images are of same modality). When input images are CT and B again histogram is again crispiest when images are best aligned as shown in figure 5.1(b). Here straight line is not obtained since intensities of CT and MR images are not linearly related.

Mutual information is the most promising and most discussed technique for multimodality image registration. The main advantage of mutual information based registration is that intensities of input images need to be linearly relate. Main drawback of mutual information based registration is that it is slower than other techniques and practically infeasible for exhaustive search of transformation matrix [14, 15].

5.1.2 Residual Image Size based Registration

In Model based Compression Scheme, registration is used for compression purpose. Therefore, goal is to achieve maximum compression ratio. A registration criterion is used here, that lead to maximum compression ratio.

In this work images are first registered using coarse registration, and then fine registration is applied. For coarse registration farthest point based approach is used. For fine registration minimization of RIS is used. Figure 5.2 shows the whole registration process. Step involved in registration are discussed in following sections.

Bounding Box Extraction

Bounding box of an image is the minimal rectangle containing that image. Bounding box of both images is extracted, in order to avoid unnecessary background part of image. For bounding box extraction, both images are converted to binary using threshold segmentation. Bounding box extraction removes unnecessary background from the images. Another advantage of calculating bounding box is, no translation is required. Only bounding boxes are now needed to be registered.

Registration based on farthest points

Looking for all possible transformation for minimum RIS will require a lot of time and is practically infeasible. Therefore coarse registration followed by fine registration is done. For coarse registration farthest points are determined in both bounding boxes. For model images it is better to store bounding box. Based on the farthest points scaling factor and rotation

angles are calculated and model image is transformed. Rotation angle is calculated using eq. 5.3. As discussed in the above section there is no need to calculate translation parameters. Scaling factor is calculated as the ratio between distances of farthest points. Up to the point images are registered based on the farthest points. For avoiding errors second farthest points are also calculated. And one of them is decided on the basis of RIS. Results are refined using minimization of RIS.

However it should be noted that this method is restricted to rigid transformations and cannot be applied for other objects where using the farthest points we cannot determine approximate transformation matrix. The method has been applied for several CT and MRI images and works. For SPECT and ultrasound image it does not work. Therefore a semiautomatic method for registration is used for these images. Certain points are selected by the radiologist and based on these points images are aligned and results are refined using minimization of RIS.

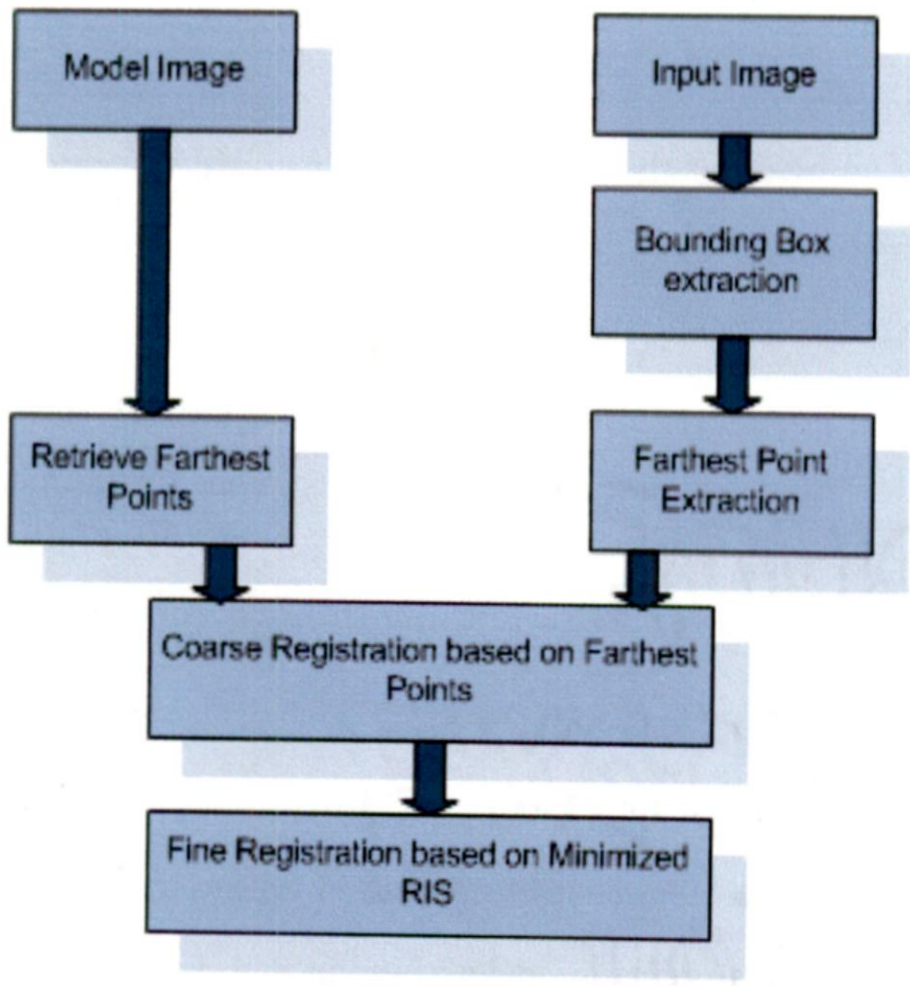


Figure 5.2: Block Diagram for Registration

Refinement of registration using Minimization of RIS

After coarse registration, fine registration is done using minimization of RIS. For calculation of RIS, compression method can directly be used. In compression method, 2 images are input instead of one residual image. Residual image is calculated and image is compressed using same compression code. An additional line for calculation of size of compressed image is added. This size value is returned.

5.2 Compression of Residual Image

The major difference between the other compression approaches and the proposed approach is, this approach is for residual medical images, and residual medical images contain excessive amount of redundant information. In residual images, there are many areas of zero intensity. Medical images themselves contain large redundancy [28] and here residual of medical images is taken. Therefore target images are with huge redundancy.

In medical images grey level values within a block vary gradually rather than abruptly [27]. This characteristic is also called smoothness. Residual images have many blocks of different sizes, with constant or nearly constant intensity level. Residual image is partitioned into variable sized blocks using quad-tree decomposition (figure 5.3). For each block, level of smoothness is evaluated by the difference in pixel's maximum and minimum gray level within the block.

If the difference in pixel's maximum and minimum gray level is 0 within the block then the pixels have the same level of gray, thus to compress this block, only minimum value is stored If the difference is less than threshold value then the block is represented by the offset from minimum gray level value or otherwise the block is further divided.

While decomposing image a rearrangement step is also performed for blocks bigger than a threshold value. This sorting saves space since after sorting, 1 or more blocks with small range may be obtained, and there would be no need to decompose these blocks further. The

Huffman entropy coding method is performed as a post-processing step to further reduce the coded file size.

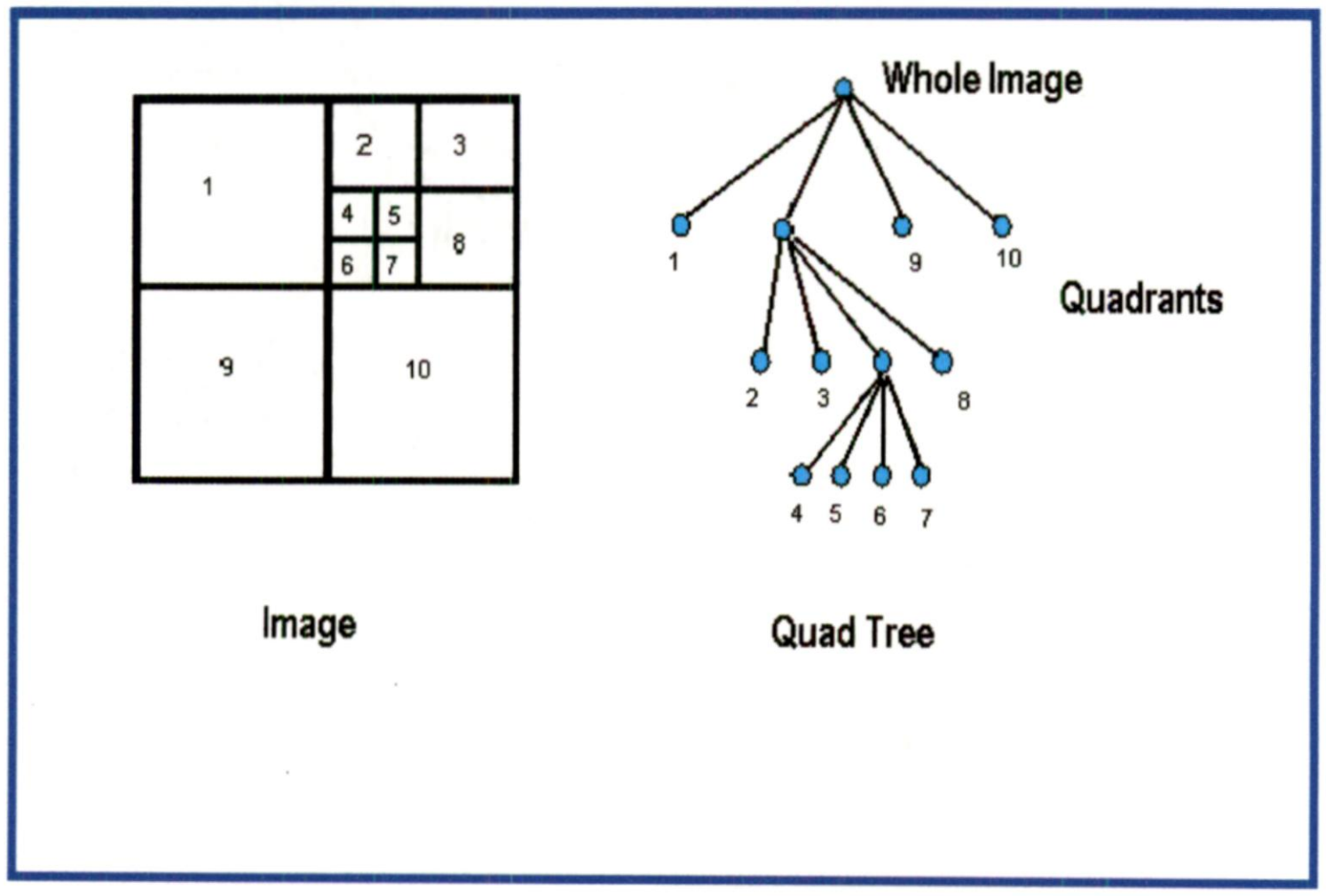


Figure 5.3 Quadtree decomposition of the image

The level of smoothness, in blocks, is evaluated by the difference in pixel's' maximum and minimum gray level within the block. If the difference in pixel's maximum and minimum gray level is 0 within the block then the pixels have the same level of gray, thus to compress this block, only block minimum value is required to store. Even if the difference is not 0, it is usually very small. Therefore this block can be represented by the offset value between the minimum gray level value and neighboring pixels in the block. Each offset can be *N*-bit encoded as shown in Eq. 5.2.

$$Diffi = \max(B_i) - \min(B_i) \tag{5.2}$$

$$N = floor(\log_2(Diff_i)) + 1 \tag{5.3}$$

If the value is smaller than fixed threshold value, it is compressed only with minimum gray level value and corresponding offsets. Figure 5.4 shows the whole QABPR compression process. The whole compression process of residual image can be divided in following operations.

Segmentation of Residual image

Proposed approach for compression, segments images in variable sized blocks, based on the block properties. The segmentation process used for decomposition is directly related to encoding characteristic and is simple. Therefore speed up the compression. Difference between maximum and minimum intensity values for a block is calculated. If it is less than a threshold value, then block is not decomposed further otherwise block is further decomposed using quad-tree decomposition.

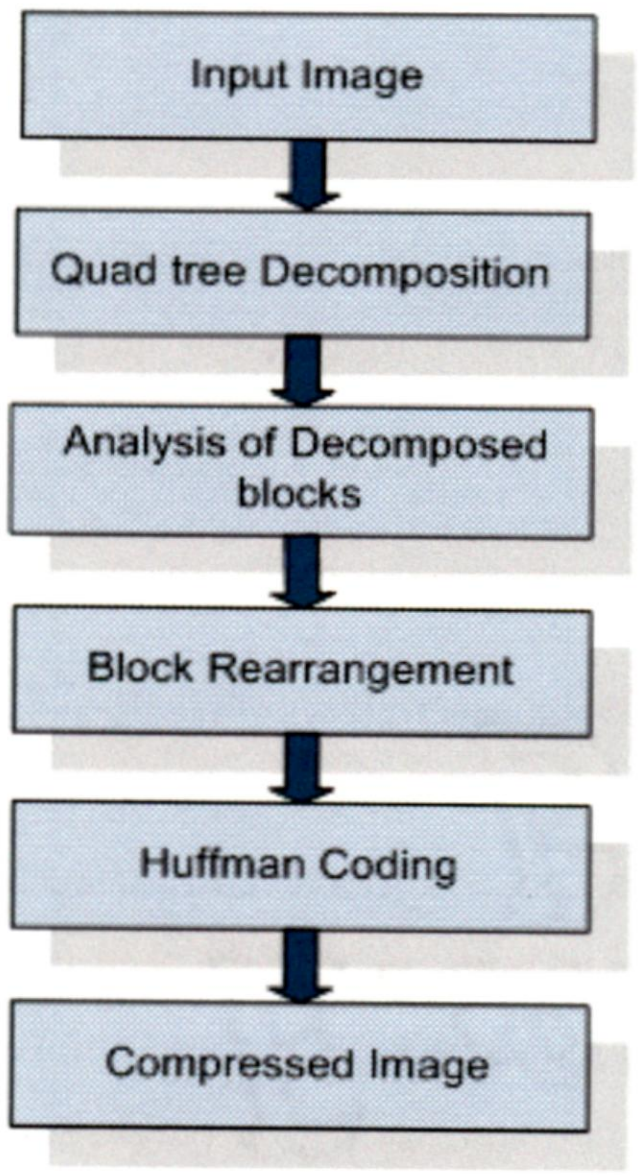


Figure 5.4: Block Diagram for compression

Rearrangement of Blocks

Before decomposition of block in 4 sub-blocks, block is divided in 16 equal sized sub-blocks and these 16 sub-blocks are sorted according to their average intensities. Rearrangement of blocks put blocks of similar intensities together and some blocks of below threshold value may be obtained and compression is improved. For rearrangement different no of block were tried. Division in 16 sub-blocks gives the optimal results (figure 6.1).

Entropy Coding

Blocks are divided recursively until a block of below threshold value is reached or block size is reduced to 2x2. For blocks of size 2x2 minimum value and offset values are stored. All the blocks with different bit sizes, calculated from eq 5.1 and 5.2 are stored in different temporary files. Therefore, image is stored in 8 different temporary files. Finally all these files are merged into a single output file and all temporary files are deleted. Before merging, Entropy coding is applied to all temporary files. RLE, Huffman, and (RLE + Huffman) were tried for entropy coding. Best results were obtained with Huffman coding.

5.3 Algorithms

The whole compression procedure is implemented in two parts. First part is registration process. Separate algorithms for both registration and compression are discussed.

5.3.1 Registration Algorithm

Inputs to registration procedure are model image M and the image I, which is to be compressed. Examples of M and I are shown in figure 5.4 (a) and (b). Both images are first converted to binary and then bounding box is extracted from both images. After cropping of binary images, original images are cropped, and then these cropped images are multiplied to remove unnecessary background noise. Farthest points are extracted in steps 6, 7 and 8. After farthest point extraction transformation parameters are calculated in steps 9 and 10. These transformations are applied to model image in step 11Second farthest points are also

calculated to make this approach more robust. After calculation of second farthest points, RIS is calculated for all four combinations. The pairs for which RIS is minimized are chosen. All these steps perform coarse registration.

Initial Input:

i. A 2-D Ineteger martix M (Model Image).

ii. Another 2-D Ineteger martix I (Image to be Compresssed).

Final Output:

i. A 2-D Ineteger martix RM (Registered model image)

Proceure

1. Convert M and I to binary matrices. Let M_Bin and I_Bin be the binary images.

2. Calculate coordinates of bounding boxes of both matrices M_Bin and I_Bin using thresholds. Crop bounding boxes of M_Bin and I_Bin. Cropped Binary images are shown in figure 5.4(c) and (d).Let cropped images be $M\_Bin\_Cropped$ and $I\_Bin\_Cropped$

3. Fill the cropped images. Let cropped and filled matrices be $M\_Bin\_Cropped\_Filled$ and $I\_Bin\_Cropped\_Filled$. Cropped filled binary images are shown in figure 5.5 (e) and (f).

4. \Crop M and I using bounding box coordinates calculated in step 2. Let cropped images be $M\_Cropped$ and $I\_Cropped$. Multiply $M\_Cropped$ with $M\_Bin\_Cropped\_Filled$ to and $I\_Cropped$ with $I\_Bin\_Cropped\_Filled$ to cleat them. Let clean Images be $M\_Cropped\_clean$ and $I\_Cropped\_clean$. These clean images are shown in figure 5.5 (g) and (h).

5. Initialize $M\_LeftSet$, $M\_RightSet$, $I\_LeftSet$ and $I\_RightSet$ to null. These sets will contain set of extreme left and extreme right points for model and input images.

6. For all rows in $M\_Bin\_Cropped\_Filled$
   i. Scan $M\_Bin\_Cropped\_Filled$ from left to right until a white point is reached. Add coordinates of this point to $M\_LeftSet$.
   ii. Scan $M\_Bin\_Cropped\_Filled$ from right to left until a white point is reached. Add coordinates of this point to $M\_RightSet$

7. Find the farthest points by taking one point from $M\_LeftSet$ and one from $M\_RightSet$. Let [a, b] be the farthest points where $a \in M\_LeftSet$ and $b \in M\_RightSet$.

8. Repeat steps 5 and 6 for $I\_Bin\_Cropped$. Let [c, d] be the farthest points in this matrix. Let $\vec{V}_{ab}$ be the vector, representing lines joining a and b, and $\vec{V}_{cd}$ be the vector representing the line joining c and d.

9. Calculate scaling factor S using following equation.

$$S = \frac{|ab|}{|cd|} \tag{5.4}$$

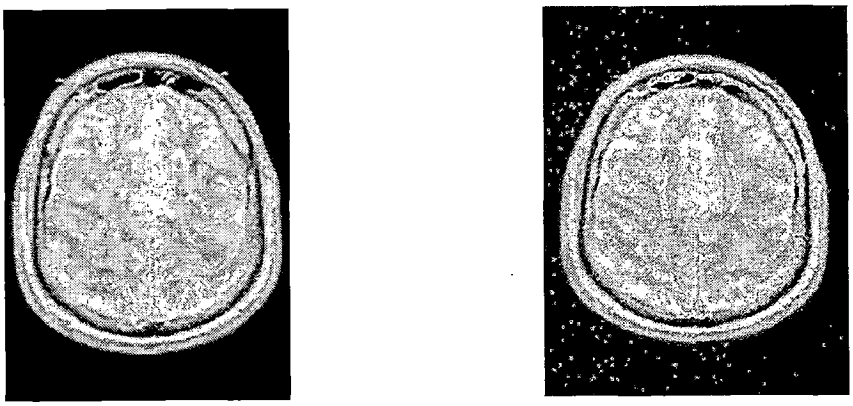
10. Calculate rotation angle as shown in eq. 5.3 and again calculate bounding box of rotated and scaled second image.

$$\theta = \cos^{-1} \frac{(\vec{V}_{ab}.\vec{V}_{cd})}{(|\vec{V}_{ab}||\vec{V}_{cd}|)} \tag{5.5}.$$

11. Apply following operations on $M\_Cropped$.

- Rotate by angle θ (Calculated in step 8).
- Scale by S (Calculated in step 9).
- Calculate bounding box and crop (Use steps 1, 2 and 3). Let the transformed image be RM. RM is shown in figure 5.5 (i)

After coarse registration fine registration is applied to model image. For fine registration residual image size is used as a registration criterion. As discussed in section 5.1.3, algorithm for fine registration is directly derived from compression algorithm, discussed in next section. Algorithm Therefore, algorithm for RIS based registration is not written here.

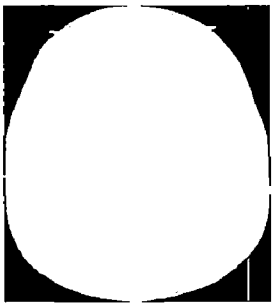


(a) (b)

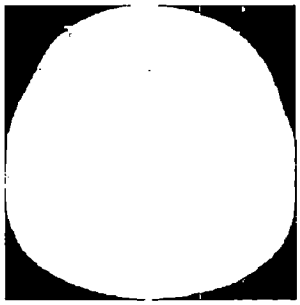
(a) Model image **(b)** Image to be compressed

(c) (d)

(c) Binary model image **(d)**Binary input image

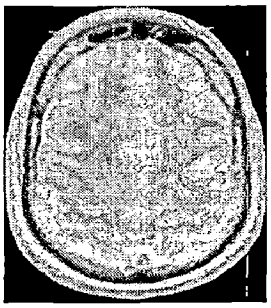


(e)

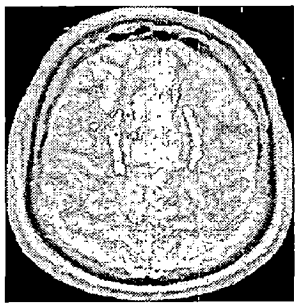


(f)

(e) Binary filled model image **(f)** Binary filled input image

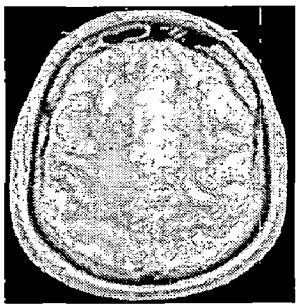


(g)

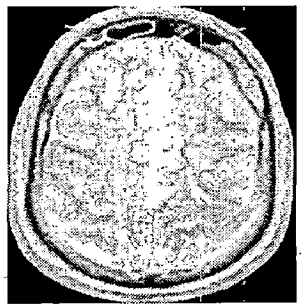


(h)

(g) Clean model image **(h)** Clean input image



(i)



(j)

(i) Model image after 1st stage registration **(j)** Model image after second stage registration.

Figure 5.5: Step for registration in model based compression scheme.

After registration, registered model image is subtracted from input image. Intensity of the subtracted image is shifted for making all values nonnegative. This subtracted and shifted image is shown in figure 5.6

Figure 5.6: Residual Image

5.3.2 Compression Algorithm

Residual image is compressed using proposed QABPR algorithm. In step 1, size of input image is made power of two for quad –tree decomposition. Added entries are filled with value 0. A procedure *Transform_Img* is called in step 2. Entropy encoding is done in steps3 and 4.

Initial Input:

A 2-D Ineteger martix R_{rxc} of size rxc (Residual Image).

Final Output:

A file F containing compressed image, R_{rxc}

Procedure

1. Make size of R_{rxc} power of 2.
   i. If $r \leq c$ then $n = r$

ii. Else $n=c$

iii. Calculate new size *ns*

$$ns = 2^{\lceil \log_2 n \rceil}$$

iv. Fill rows from r+1 to ns with values 0.

v. Fill columns c+1 to ns with values 0.

2. Call *Transform_Img* to segment image in variable-sized-blocks.

   [*block_min* , *block_no_of_bits*] = *Transform_Img (R)*

3. Apply first run length encoding then Huffman coding on *block_min* and *block_no_of_bits*

4. Group blocks according *block_no_of_bits* values. There will be 8 groups with different bit requirements. Entropy encodes these groups separately.

Algorithm for *Transform_Img*

Transform_Img is a recursive procedure. *Transform_Img* segments image in variable size blocks and calculates number of bit requirements for each block.

Transform_Img also involve rearrangement of blocks (see step 4).

Initial Input:

A 2-D Ineteger martix M of size nxn (Residual Image).

Final Output:

i. A1-D array *block_min* (contains minimum value in each block).

ii. A 1-D array *block_no_of_bits* (contains minimum no. of bits required to encode each block).

Procedure

If $n \geq 2$.

1. *block_min (Index) = min(M,n)*

2. *block_no_of_bits (Index) = floor(log₂(max(M) - min(M))) + 1*

3. Subtract *block_min (Index)* from all intensities of the current block

4. If *block_min (Index)* ≥ *Thres* and $n > 2$ then
   - *i.* Divide *M* in 16 equal sized blocks
   - *ii.* Sort these blocks according to their average intensities.
   - *iii.* *Index = Index+1*
   - *iv.* Take first 4 blocks from the sorted blocks, make a block of size n/2 and call *Transform_Img* for this.
   - *v.* Repeat above two steps (iii and iv) for rest 12 sub-blocks.

5. End if

End if

5.4 Implementation Details

Model based compression is implemented using MATLAB. Image processing toolbox is used for basic image operations. The Image Processing Toolbox provides a comprehensive set of reference-standard algorithms and graphical tools for image processing, analysis, visualization, and algorithm development. Most toolbox functions are written in the open MATLAB language. Optimization toolbox is also used for Downhill simplex method.

5.5 Implementation Issues

5.5.1 Scalability

Scalability generally refers to a quality reduction achieved by manipulation of the bitstream or file (without decompression and re-compression). Other names for scalability are progressive coding or embedded bitstreams. Despite its contrary nature, scalability can also be found in lossless codecs, usually in form of coarse-to-fine pixel scans. Scalability is especially useful for providing variable quality access viewers [29]. There are several types of scalability:

- *Quality progressive* or layer progressive: A compression scheme is said to be quality progressive if bit-stream successively refines the reconstructed image. Proposed compression scheme can be made quality progressive. Following changes are required, for making Compression and decompression quality progressive.
  - **(i)** Conversion from recursive to iterative procedure.
  - **(ii)** Compression should be in reverse order. Starting from 2x2 blocks, image can be constructed in progressive manner but this reverse order will increase time taken by compression.
- *Component progressive*: First encode grey; then color. Generally medical images are gray scale images, as shown in table 2.1. Therefore component progressive scalability does not apply to medical images.

5.5.2 Steps that led to the final design

Proposed compression scheme is for residual medical images. At the beginning of this work, these images were analyzed. It was observed that intensity values in these images vary smoothly and these images have many areas of constant and nearly constant intensity values. Initial design was started with taking JPEG as a base. Images were divided in 8x8, fixed sized blocks. An effort was made to obtain more zeros than are obtained after JPEG transformation step. For getting zeros, minimum intensity values, no of bits required for each block and offset values were stored and values were folded in 8 bits. It was observed that by

taking variable sized blocks, better compression ratio can be achieved. For entropy encoding of blocks of same sizes, Huffman coding, RLE and Huffman followed by RLE, were tried. Huffman coding performed better than other entropy coding schemes. Therefore Huffman coding was chosen as final entropy coding scheme. Rearrangement step was added to put together blocks of similar intensities. Rearrangement step further improved compression ratio.

5.5.3 Efficiency

The Model based Compression scheme is both space and time efficient. Coarse registration followed by fine registration is used to avoid user interaction. For coarse registration, farthest point based approach is used. Extraction of farthest point is simple and time efficient.

For fine registration minimization of RIS criterion is used. Registration based on minimization of RIS improves compression ratio and time efficiency as well. Calculation of Mutual information for images is a complex process.

For compression of residual images quad-tree decomposition is used. Since residual images have huge redundancy, many blocks of variable size are not decomposed and are not processed further. Other lossless compression methods use transform and do not take advantage of this redundancy.

5.5.4 Failures and Successes

The Model based Compression Framework comprises of two major parts, alignment and compression.

For alignment coarse registration followed by fine one is implemented. Coarse registration is done by extracting farthest points from both images. Farthest points based approach is simple and therefore improves efficiency of registration.

Farthest points based approach works well for anatomical images on the other hand it fails for functional images since functional images do not have rigid structures. Therefore a semiautomatic approach is used for theses images. For fine registration RIS is used as a registration criterion. RIS is based on compression characteristics of images and therefore, successfully improves compression ratio significantly. This fact is proved by results (see table 6.2).

For compression quadtree based adaptive block partitioning with rearrangement compression (QABPRC) scheme is used. QABPRC is based on the fact that residual images have large redundancy. QABPRC works better than current standard compression techniques, but QABPRC does not work well for normal medical images except for CT images. QABPRC gives better results for CT images however for MRI and other images it fails and gives compression ratio between GIF and JPEG200.

CHAPTER 6

RESULTS AND DISCUSSION

This chapter analyzes the results of compression scheme, applied to several CT and MRI images. In all cases the images were compressed, decompressed, and then compared with the original to ensure that the implementation was truly reversible Results show that proposed compression performs much better than other standard lossless compression algorithms. Since standard compression algorithms are available only for 8 bit grey scale image or for color images, Residual images are down-sampled to 8 bit for comparison purpose. Results in table 6.1 shows that proposed compression works much better than JPEG 2000 and JPEG-LS. Results in table 6.1 are taken for images shown in figure 6.2. Results in table 6.1 shows that more the redundancy is more is the compression ratio.

Image No	GIF	RLE	Huffman	JPEG-LS	JPEG 2000	Proposed
MR I1	3.23	1.85	4.68	4.14	4.45	5.39
MRI 2	3.78	1.90	4.83	4.50	4.69	6.70
MRI 3	2.17	1.55	4.66	2.83	3.33	5.71
CT 1	2.64	1.65	3.94	3.52	4.44	4.63
CT 2	3.30	2.40	4.16	4.63	5.14	6.97
CT 3	3.03	2.35	4.01	4.54	4.91	6.84

Table 6.1: Comparison of compression algorithms for residual medical images

Registration of model and input image is a primary step for compression. MI (Mutual Information) is the most used and the most discussed criterion for registration. RIS is a new registration criterion, used in Model Based Compression Scheme. RIS based registration improves compression ratio considerably. Table 6.2 shows improvements in compression, by the use of RIS as a registration criterion rather than MI. Images for these results are shown in figure 6.2

Compression Ratio with MI as a Registration criterion	**Compression Ratio with RIS as a Registration criterion**
4.27	5.39
5.93	6.70
4.82	5.71
4.07	4.63
5.82	6.97
6.10	6.84

Table 6.2: Comparison of registration methods for compression

Figure 6.2 shows results of registration for CT and MRI images. First coarse then fi registration is applied to input image. Figure 6.2(c) shows images after registration. Resu show that RIS aligns images well.

QABPR compression involves rearrangement of blocks. For deciding no of blocks to rearranged, compression algorithm was tested for different no of blocks. Results shows th blocks should be decomposed in 16 sub-blocks.

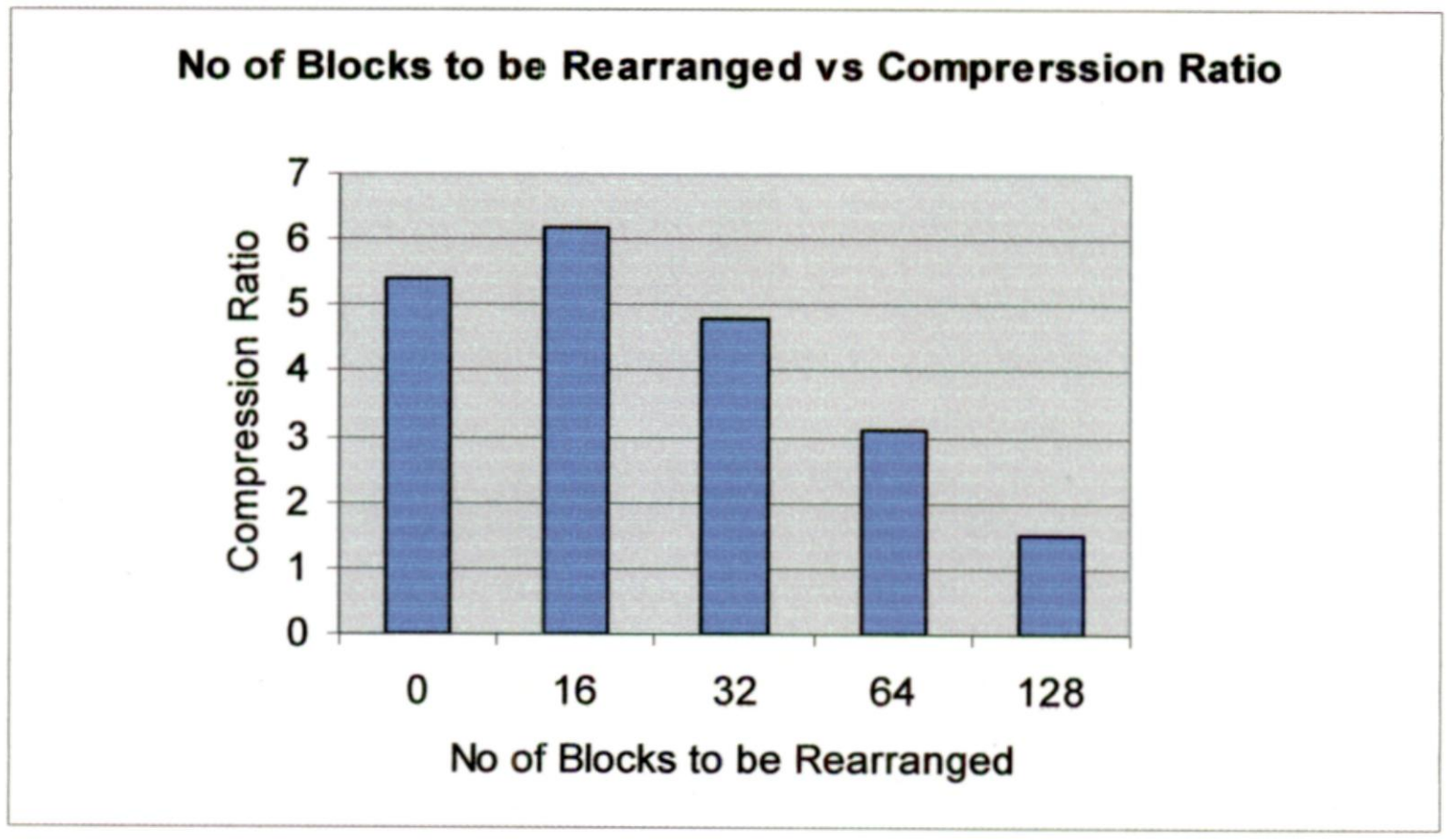
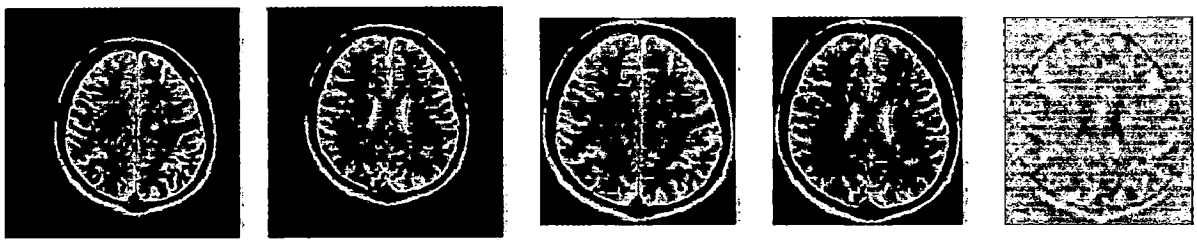
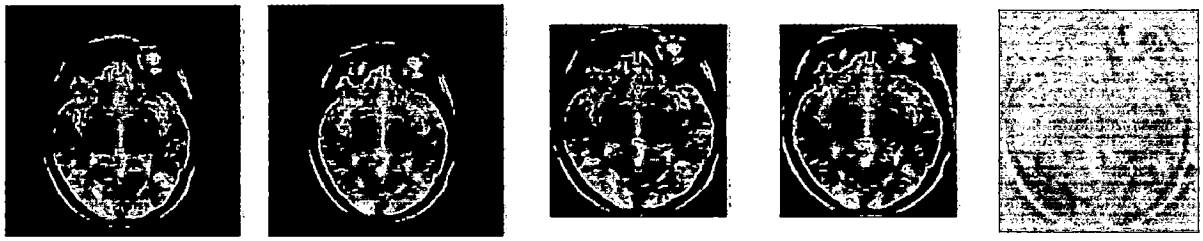


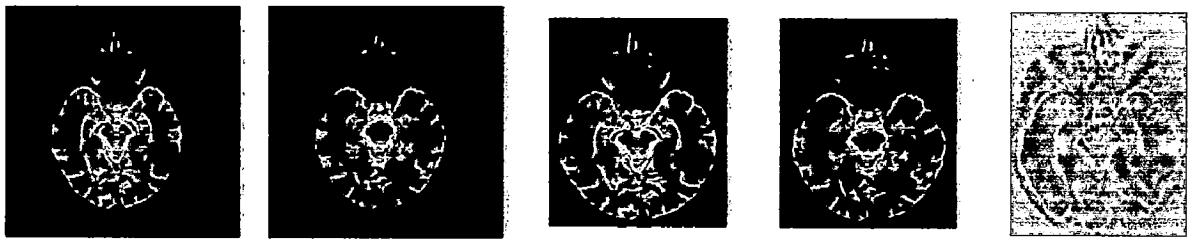
Figure 6.1: Compression ratio for different no of blocks to be rearranged



Slice 15 of MRI brain images



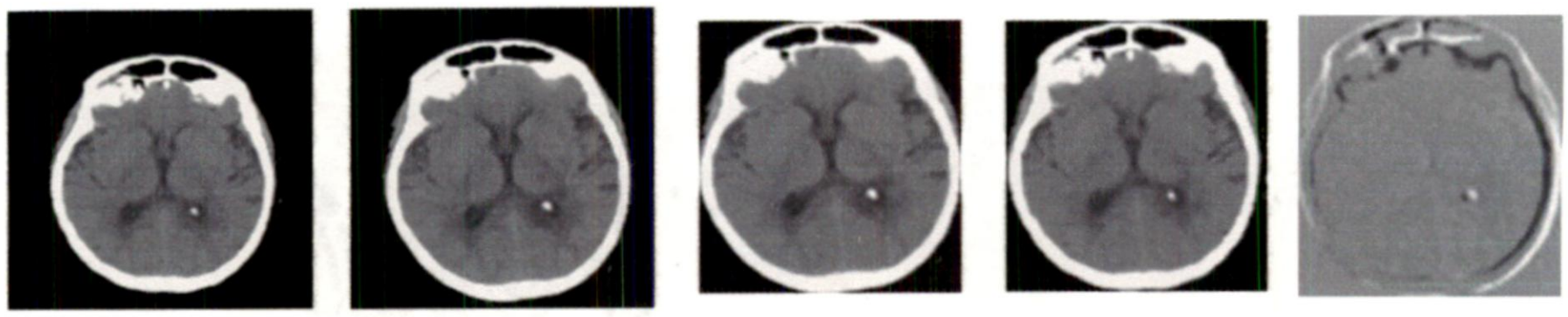
slice 11 of brain MRI images



Slice 20 of MRI brain images

CT image of knee

Slice 11 of CT brain image



Slice 10 of brain image

(a) (b) (c) (d)

Figure 6.2: Steps of the Model Based Compression framework for some CT and MRI images **(a)** Model Image **(b)**Uncompressed Image **(c)** Images After Registration **(d)** Residual Image

Proposed compression algorithm was tested for several medical images. Results have proven that QABPR compression performs much better than standard compression algorithms (see figure 6.3).

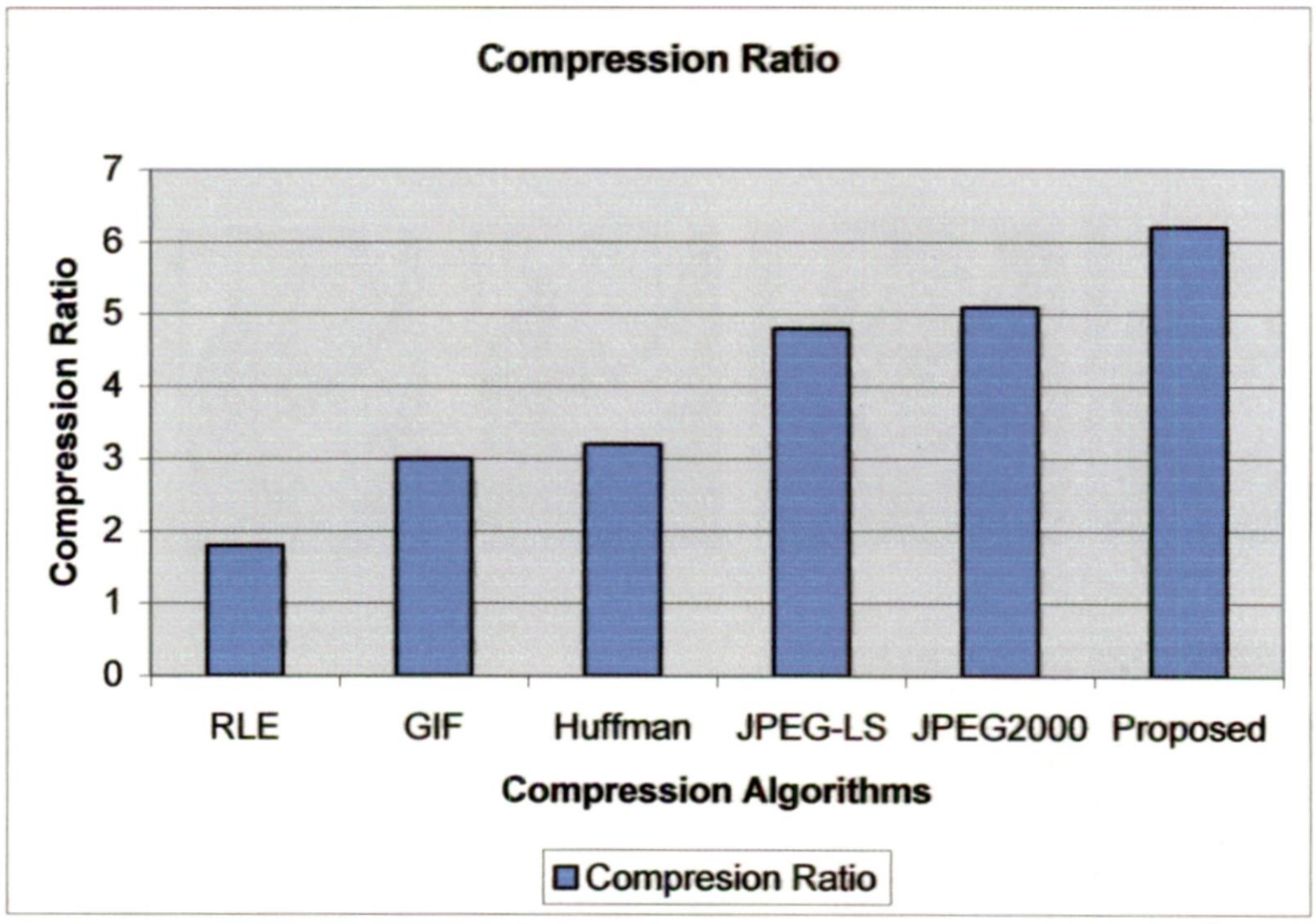


Figure 6.3 Average compression ratio achieved by different algorithms

CHAPTER 7

CONCLUSION AND FUTURE WORK

In this dissertation a novel Model based Compression Framework is proposed for medical images. The system comprises of two parts, image registration and residual image compression. A new approach for registration and compression is proposed.

Image Registration has been used in medical imaging for integration. Here registration is used for compression. Coarse registration followed by fine registration is done. A new criterion for registration is implemented for registration. For coarse registration farthest points based approach is used. Calculation of farthest point is simple and avoids user interaction. For fine registration RIS is used. The new criterion is simple and leads to higher compression ratio, as shown in table 6.2.

Residual image compression is done using proposed Quadtree based Adaptive Block Partitioning with Rearrangement (QABPR) compression. The compression is based on the fact that residual images have large amount of redundancy (see figure 6.2 (c)). QABPR compression segments in variable size blocks and encode them based on block properties. Results have shown that compression scheme performs much better than other state of the art compression schemes e.g. GIF, JPEG-LS, and JPEG2000.

Future Work

Presented model based compression scheme can be enhanced in following ways.

1. Compression scheme can be improved by using a variable value for threshold, for blocks of different sizes.

2. Since image generated by various modalities are 3D. 3D image compression is valuable in medical imaging. Compression and registration work can be easily scaled to 3D images.

3. Presented automatic registration fails for SPECT and PET images. Registration can be made automatic for theses images by extracting some other features rather than farthest points.

4. Compression ratio can further be improved by using more exhaustive analysis of blocks and use of more compression techniques, according to block properties.

5. Registration procedure can be made more robust by adding deformable transformations, and this addition will also improve compression ratio since image would be better aligned.

REFERENCES

[1] S. Wong, L. Zaremba and H.K. Huang, "Radiologic Image Compression-A Review". Proc. IEEE, Volume 83, 1995, Pages 194–219.

[2] K. Karadimitriou and J. M. Tyler, "Min-max Compression Methods for Medical Image Databases" ACM SIGMOD Record, Volume 26, Issue 1, 1997, Pages 47–52.

[3] D. Mateika, R. Martavičius, "Analysis of the Compression Ratio and Quality in Medical Images", Information Technology and Control, Volume 35, Issue4, 2006, Pages 78.-87.

[4] N. Fatma and E. Derya , "Theory and Applications of Telemedicine", Journal of Medical Systems, Volume 26, Issue 3, 2002, Pages 199-216.

[5] S. Zach, T. Aviv, "Telemedicine Overview and Summary", IEEE, 1996, Pages 409.412.

[6] http.//www.thamburaj.com/telemedicine.htm.

[7] D. A Koff, H. Shulman, "An Overview of Digital Compression of Medical Images. Can We Use Lossy Image Compression in Radiology?" Canadian Association of Radiologists Journal, Volume 57, Issue 4, 2006, Pages 211.217.

[8] "Mathematical Methods in Medical Image Processing", Bulletin (New Series) of The American Mathematical Society, Volume 00, Issue 0, 1999, Pages 102-109.

[9] I. K. Indrajit, J D Souza, R. Singh, A. Shekar, "Radiology Image Management in a Teaching Hospital Network Scenario. Initial Experience", Medical Journal Armed Forces India, Volume 59, Issue 3, 2003, Pages 234.238.

[10] R. Guillemaud and M. Brady, "Estimating the Bias Field of MR Images", IEEE Transactions on Medical Imaging, Volume 16, Issue 3, june 1997, Pages 238.242.

[11] Y. J. Zhang, "A Review of Recent Evaluation Methods for Image Segmentation", IEEE, Volume 1, 200l, Pages 148.151.

[12] M. H. Gross, "Surgery Simulation-A Challenge for Graphics and Vision", Vision, Modeling and Simulation 99 Workshop, Erlangen, Germany, Issuevember 1999, Pages 1-3.

[13] T. Taxt, A. Lundervold , J. Strand and S. Holm, "Advances in Medical Imaging", 14th Internationall Conference on Pattern Recognition , Brisbane, Australia, August 1998 (ICPR 98), Volume 1, Pages 505-509.

[14] M. Holden, D. L. G. Hill, E. R. E. Denton, J. M. Jarosz, T. C. S. Cox, T Rohlfing, J. Goodey, and D. J. Hawkes "Voxel Similarity Measures for 3-D Serial MR Image Registration" IEEE Transaction on Medical Imaging, Volume 19, Issue 2, Feb 2000, Pages 94-102.

[15] C. Huang, C. Jiang, W. Sung, "Medical Image Registration and Fusion with 3D CT and MR data of Head", Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems (CBMS06), 2006, Pages 1-4.

[16] http.//www.radiologyinfo.org/index.cfm?bhcp=1

[17] P. Josien, W. Pluim, J. B. A. Maintz and M. A. Viergever, "Mutual Information Based Registration of Medical Images. A Survey" IEEE Transactions on Medical Imaging, Volume XX, 2003, Pages 1-15.

[18] J. B. A. Maintz and M. A. Viergever, "A Survey of Medical Image Registration", Oxford University Press, Volume 2, Issue 1, 1997. Pages 1–25.

[19] W. Rui and L. Minglu, "An Overview of Medical Image Registration "IEEE Proceedings of the Fifth International Conference on Computational Intelligence and Multimedia Applications, 2003, Pages 1-6.

[20] D. Cai,"Adaptive Block Partitioning in Fractal Image Coding", IEEE Region 10th Annual Conference, Speech and Image Technologies for Computing and Telecommunications (TENCON97), Volume 2, Pages 565-568

[21] A. Fidler, B. Likar, F. Pernus and U. Skaleric, "Comparative evaluation of JPEG and JPEG2000 Compression in Quantitative Digital Subtraction Radiography", Dentomaxillofacial Radiology, Nature Publishing Group, Issue 6, Volume 31, 2002, Pages 379-384

[22] F. Sheng, A. Bilgin, P. J. Sementillit, M. W. Marcellin, "Lossy and Lossless Image Compression Using Reversible Integer Wavelet Transforms", International Conference on Image Processing, IEEE, Pages 876-878.

[23] J. M. Shapiro, "Embedded Image Coding using Zerotrees of Wavelet Coefficients," IEEE Transaction on Signal Processing, Volume 41, 1993, Pages 3445-3448.

[24] A. Said and W. Pearlman, "A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical trees," IEEE Transaction on Circuits and Systems Video Technology, Volume 6, Issue 3, 1996, Pages 243-250.

[25] A. Munteanu, 1 J. Cornelis, G. Auwera, P. Cristea, "Wavelet Based Lossless Compression Scheme with Progressive Transmission Capability", John Wiley & Sons, Inc., New York, USA, Volume 10, Pages 77 -84.

[26] J. Kivija¨rvia, T. Ojalaa, T. Kaukorantaa, A. Kubab, L. Nyulb, O. Nevalainena, " A Comparison of Lossless Compression Methods for Medical Images", IEEE Data Compression Conference, Volume 0, 1999, Pages 718-721

[27] M. J. Zukoski, T. Boult, T. Iyriboz, "A Novel Approach to Medical Image Compression", International Journal on Bioinformatics Research and Applications, Volume 2, Issue 1, 2006, Pages 89-101.

[28] S. Kil, J. S. Lee†, D. F. Shen, J. G. Ryu, E. H. Lee, H. Min, S. H. Hong, "Lossless Medical Image Compression using Redundancy Analysis", IJCSNS International Journal of Computer Science and Network Security, Volume 6, Issue 1A, 2006, Pages 50-55.

[29] H. Jingsong , W. Xufa, Z. Min, W. Jiying, and F. Qiansheng, "New research on Scalability of Lossless Image Compression by GP Engine" , ASA/DoD Conference on Evolvable Hardware, 2005, Pages 160-164

[30] R. C. Gonzalez and R. E. Woods, "Digital Image Processing", Second Edition, Pearson Education, 2004.

APPENDIX A: SOURCE CODE LISTING

EncodeImage.m

```
function flag = EncodeImage(A,filepath,globali,ns,r,c);
  filepath = str2mat(filepath);
  ComprFile_path([1:globali]) = filepath([1:globali]);
  ComprFile_path(globali+1:globali+4) = ['b','b','c','d'];
  fid1 = fopen(ComprFile_path,'wb');
  fwrite(fid1,r,'uint16');
  fwrite(fid1,c,'uint16');
  ComprFile_path(globali+1:globali+2) = ['b','c'];
  N = CaclulateSize(ns);
  blkmin(1:N) = 0;
  blk_nofbits(1:N) = 8;
  [A,blkmin,blk_nofbits,idx] =
  compress_img(A,blkmin,blk_nofbits,ns,ns,1);
  disp('Stage 1');
  WriteArray(fid1,blk_nofbits,N,2);
  WriteArray(fid1,blkmin,N,2);
  disp('Stage 2');
  fid = getfids(20,ComprFile_path,globali,1);
  x =
  WriteImage(A,ns,ns,blk_nofbits,blkmin,N,ComprFile_path,globali,1,fid
  );
  disp('Stage 3');
  for i = 1:20
       fclose(fid(i));
  end
  EncodeAndWrite(ComprFile_path,globali);
  disp('Stage 4');
  FinalWrite(fid1,ComprFile_path,globali);
  fclose(fid1);
  fclose('all');
end
```

getImage.m

```
function [A,ns,r,c] = getImage(filepath)
 A = imread(filepath);
 [r,c] = size(A);
 if r>c
     n = r;
 else
     n = c;
 end
 nb = ceil(log2( double (n) ));
 ns = power(2,nb);
 ns = uint32(ns);
 A([1:r],[c+1:ns]) = 0;
 A([r+1:ns],[1:ns]) = 0;
```

End

Divide_block.m

```
function [B] = Divide_block(A)
   sz1 = size(A,2);
   range1(1:16) = 0;
   sz = sz1/4
   idx = 1;
   for m = 0:3
       for n = 0:3
           temp = A(m*sz+[1:sz],n*sz+[1:sz]);
           range1(idx) = max(max(temp))-min(min(temp));
           idx = idx+1;
       end
   end
   [R,IDX] = sort(range1);
   B = zeros(sz1);
   ix = 1;
   for m = 0:1
       for n = 0:1
           x = IDX(ix)-1;
           x = double(x);
           R = floor(x/4.0)+1;
           R = uint8(R);
           C = mod(x,4)+1;
           C = uint8(C);
           R = (R-1)*sz+1;
           C = (C-1)*sz+1;
           B(m*sz+[1:sz],n*sz+[1:sz]) = A(R:R+sz-1,C:C+sz-1);
           ix = ix+1;
       end
   end
   for m = 0:1
       for n = 2:3
           x = IDX(ix)-1;
           x = double(x);
           R = floor(x/4.0)+1;
           R = uint8(R);
           C = mod(x,4)+1;
           C = uint8(C);
           R = (R-1)*sz+1;
           C = (C-1)*sz+1;
           B(m*sz+[1:sz],n*sz+[1:sz]) = A(R:R+sz-1,C:C+sz-1);
           ix = ix+1;
       end
   end
   for m = 2:3
       for n = 0:1
           x = IDX(ix)-1;
           x = double(x);
           R = floor(x/4.0)+1;
           R = uint8(R);
           C = mod(x,4)+1;
```

```
            C = uint8(C);
            R = (R-1)*sz+1;
            C = (C-1)*sz+1;
            B(m*sz+[1:sz],n*sz+[1:sz]) = A(R:R+sz-1,C:C+sz-1);
            ix = ix+1;
        end
    end
    for m = 2:3
        for n = 2:3
            x = IDX(ix)-1;
            x = double(x);
            R = floor(x/4.0)+1;
            R = uint8(R);
            C = mod(x,4)+1;
            C = uint8(C);
            R = (R-1)*sz+1;
            C = (C-1)*sz+1;
            B(m*sz+[1:sz],n*sz+[1:sz]) = A(R:R+sz-1,C:C+sz-1);
            ix = ix+1;
        end
    end
    imshow(uint8(B));
end
```

CaclulateSize.m

```
function [N] = CaclulateSize(ns)
    ns = double(ns);
    x = log2(ns);
    N = (power(4,x)-1)/3;
End
```

---

getfids.m

```
function fid = getfids(n,ComprFile_path,globali,mode)
    if mode == 1
        for i = 1:9
            a = num2str(i);
            ComprFile_path(globali+3:globali+5) = ['2','0',a(1)];
            fid(i) = fopen(ComprFile_path,'wb');
        end

        for i = 10:n
            a = num2str(i);
            ComprFile_path(globali+3:globali+5) = ['2',a(1:2)];
            fid(i) = fopen(ComprFile_path,'wb');
        end
    else
        for i = 1:9
        a = num2str(i);
        ComprFile_path(globali+3:globali+5) = ['2','0',a(1)];
        fid(i) = fopen(ComprFile_path,'rb');
```

```
        end

        for i = 10:n
            a = num2str(i);
            ComprFile_path(globali+3:globali+5) = ['2',a(1:2)];
            fid(i) = fopen(ComprFile_path,'rb');
        end
end
```

---

compress_block.m

```
function [blockmin_value,block_nofbits_value] =
compress_block(ip_blk,blk_size)
   THERSHOLD = 3;
   max1 = ip_blk(1,1);
   min11 = ip_blk(1,1);
   for ii = 1:blk_size
       for jj = 1:blk_size
           if max1 < ip_blk(ii,jj)
               max1 = ip_blk(ii,jj);
           end
           if min11 > ip_blk(ii,jj)
               min11 = ip_blk(ii,jj);
           end
       end
   end
   blockmin_value = min11;
   if max1-min11==0
       block_nofbits_value = 0;
       return;
   end
   block_nofbits_value = uint8(floor(log2( double (( abs(max1-min11) )
   ) ))+1);
   if block_nofbits_value == 7
       block_nofbits_value=8;
   end
end %End of Function
```

---

compress_img.m

```
function [A,blockmin,block_nofbits,idx] =
compress_img(A,blockmin,block_nofbits,ns,sz,idx)
   thers = 2;
   if sz >= 2
       [blockmin_value,block_nofbits_value] = compress_block(A,sz);
       A = A - blockmin_value;
       if block_nofbits_value > thers
               blockmin(idx)  = blockmin_value;
               block_nofbits(idx) = block_nofbits_value;
               if sz > 2
                  blk_size = sz/2;
                  m = uint32(0);
```

```
                    n = uint32(0);
                    for m = 0:1
                        for n = 0:1
                            idx = idx+1;
                            [A(
   m*blk_size+[1:blk_size],n*blk_size+[1:blk_size]),blockmin,block_nofb
   its,idx] = compress_img(A(
   m*blk_size+[1:blk_size],n*blk_size+[1:blk_size]),blockmin,block_nofb
   its,ns,blk_size,idx);
                        end
                    end
                end
        end
end
```

EncodeAndWrite.m

```
function [flag] =  EncodeAndWrite (ComprFile_path, globali)
    for i = 1:20
     a = num2str(i);
    if i<10
        ComprFile_path(globali+3:globali+5) = ['2','0',a(1)];
    end
    if i>9 && i <21
        ComprFile_path(globali+3:globali+5) = ['2',a(1:2)];
    end
    fid(i) = fopen(ComprFile_path,'rb');
    if i < 9
    bitstr = ['ubit',num2str(i),'=>ubit8'];
    end
    if i>=9 && i<17
    bitstr = ['ubit',num2str(i),'=>uint16'];
    end
    if i>=17 && i<33
    bitstr = ['ubit',num2str(i),'=>uint32'];
    end
    temp = fread(fid(i),bitstr);
    fclose(fid(i));
    delete(ComprFile_path);
    sz = size(temp,1);
    if sz > 0
        ComprFile_path1(1:globali) = ComprFile_path(1:globali);
        if i<10
            ComprFile_path1(globali+1:globali+5) =
   ['b','n','2','0',num2str(i)];
            fid2 = fopen(ComprFile_path1,'wb');
        else
            a = num2str(i);
            ComprFile_path1(globali+1:globali+5) =
   ['b','n','0',a(1),a(2)];
            fid2 = fopen(ComprFile_path1,'wb');
        end
     temp1 = temp';
     WriteArray(fid2,temp1,sz,1);
```

```
    fclose(fid2);
    end
   end
   flag =1;
end
```

Huffman.m

```
function [Info,h_enc_arr] = Huffman(iparr,sz)

   clear symb;
   symb = [];
   p = 1/sz;
   symb(1) = iparr(1);
   prob(1) = p;
   idx = 1;
   for i = 2:sz
       flag = 0;
       for j = 1:idx
           if iparr(i) == symb(j)
               prob(j) = prob(j)+p;
               flag =1;
           end
       end
       if flag==0
           idx = idx+1;
           symb(idx) = iparr(i);
           prob(idx) = p;
       end
   end
   if idx ~= 1
       [dict,avglen] = huffmandict(symb,prob);
       enc_arr = [];
       dictLength = size(dict,1);
       Info.NoOfSymb = dictLength;
       idx = 0;
       for i = 1 : dictLength
           a = dict(i,2);
           b = cell2mat(a);
           Info.symbl(i) = cell2mat(dict(i,1));
           Info.nob(i) = size(dict{i,2},2);
           Info.Symbenc_arr([idx+1:idx+size(b,2)]) = b;
           idx = idx + size(b,2);
       end
       h_enc_arr = huffmanenco(iparr,dict);
       Info.len = size(h_enc_arr,2);

   else
       for i = 1:sz
           h_enc_arr = [0];
       end
       Info.NoOfSymb = 1;
       Info.len = sz;
       Info.nob = 1;
```

```
        Info.Symbenc_arr = 0;
        Info.symbl = iparr(i);
    end
end
```

FileSize.m

```
function sz = FileSize(filepath)
    fid  = fopen(filepath,'rb');
    temp = fread(fid,'ubit8');
    sz = size(temp,1)/1024;
end
```

---

rle_encode.m

```
function [rle_count_arr,rle_val_arr] = rle_encode(encoded_arr,arr_size)
     clear rle_count_arr
     clear rle_val_arr
     prev_value = encoded_arr(1);
     rle_count_arr(1) = 1;
     rle_val_arr(1) = encoded_arr(1);
     count = 1;
     j = 1;
      for i = 2:arr_size
          if encoded_arr(i) == prev_value
              count = count+1;
              if i == arr_size
                  rle_count_arr(j) = count;
                  rle_val_arr(j) = prev_value;
              end
          else
             rle_count_arr(j) = count;
             rle_val_arr(j) = prev_value;
             j = j+1;
             count =1;
             prev_value = encoded_arr(i);
              if i == arr_size
                  rle_count_arr(j) = count;
                  rle_val_arr(j) = prev_value;
              end
          end
      end
 end
```

WriteBlock.m

```
function x = WriteBlock(A,blk_nofbits_val,
blk_min_val,blk_size,ComprFile_path,globali,fid1)
```

```
    j = uint32(0);
    k = uint32(0);

    nobstr = ['ubit',num2str(blk_nofbits_val)];
    % fprintf(1,'blk_size = %d\n',blk_size);
    for j = 1:blk_size
        for k =1:blk_size
                    isy = A(j,k);
                    fwrite(fid1,isy,nobstr);
        end
    end
    x = 1;
end
```

WriteHuffmanCode.m

```
function [flag] = WriteHuffmanCode(fid,Info,h_arr,nob)
    flag = 0;
    x = ftell(fid);
    fwrite(fid,nob,'ubit8');
    fwrite(fid,Info.NoOfSymb,'ubit32');
    nobstr = ['ubit',num2str(nob)];
    bc = 0;
    fwrite(fid,Info.nob,'ubit8');
    fwrite(fid,Info.len,'ubit32');
    c = fwrite(fid,Info.symbl,nobstr);
    bc = bc + c*nob;
    c = fwrite(fid,Info.Symbenc_arr,'ubit1'); %We can calculate size of
    bc = bc+c;
    c = fwrite(fid,h_arr,'ubit1');
    bc = bc+c;
    x = 8-mod(bc,8);
    temp(1:x) = 0;
    fwrite(fid,temp,'ubit1');
    flag = 1;
end
```

WriteImage.m

```
function idx =
WriteImage(A,ns,blk_size,blk_nofbits,blkmin,N,ComprFile_path,globali,id
x,fid);
global cnt;
if blk_size >= 2
    x = blk_nofbits(idx);
    if (blk_size == 2&& x>0)
        idx = uint32(idx);
        ns = uint32(ns);
            if x~=0
                WriteBlock(A,x,blkmin(idx),blk_size,ComprFile_path,gl
                obali,fid(x));
            end
```

```
        return;
    else
      if blk_size > 2 && x >0
         %CalculateChildIndex(ns,blk_size,idx);
        blk_size = blk_size/2;
        m = uint32(0);
        n = uint32(0);
        idx = idx+1;
          idx =
        WriteImage(A(1:blk_size,1:blk_size),ns,blk_size,blk_nofbits,blkm
        in,N,ComprFile_path,globali,idx,fid);
        idx = idx+1;
        idx =
        WriteImage(A(1:blk_size,blk_size+[1:blk_size]),ns,blk_size,blk_n
        ofbits,blkmin,N,ComprFile_path,globali,idx,fid);
        idx = idx+1;
        idx =
        WriteImage(A(blk_size+[1:blk_size],[1:blk_size]),ns,blk_size,blk
        _nofbits,blkmin,N,ComprFile_path,globali,idx,fid);
        idx = idx+1;
        idx =
        WriteImage(A(blk_size+[1:blk_size],blk_size+[1:blk_size]),ns,blk
        _size,blk_nofbits,blkmin,N,ComprFile_path,globali,idx,fid);
      end
    end
end
```

FinalWrite.m

```
function [flag] =  FinalWrite(fid1,ComprFile_path, globali)
for i = 1:20
        if i<10
            ComprFile_path(globali+1:globali+5) =
            ['b','n','2','0',num2str(i)];
            fid2 = fopen(ComprFile_path,'rb');
        else
            ComprFile_path(globali+1:globali+5) =
            ['b','n','0',num2str(i)];
            fid2 = fopen(ComprFile_path,'rb');
        end
       if fid2 > 0
          fwrite(fid1,i,'ubit8');
          [arr,sz] =  ReadArray(fid2,1);
          arr = arr';
          WriteArray(fid1,arr,sz,1);
          fclose(fid2);
          delete(ComprFile_path);
      end
end
```

gui.m

```
function varargout = gui(varargin)
   gui_Singleton = 1;
   gui_State = struct('gui_Name',       mfilename, ...
                      'gui_Singleton',  gui_Singleton, ...
                      'gui_OpeningFcn', @gui_OpeningFcn, ...
                      'gui_OutputFcn',  @gui_OutputFcn, ...
                      'gui_LayoutFcn',  [] , ...
                      'gui_Callback',   []);
   if nargin && ischar(varargin{1})
       gui_State.gui_Callback = str2func(varargin{1});
   end
   if nargout
       [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
   else
       gui_mainfcn(gui_State, varargin{:});
   end
end

function gui_OpeningFcn(hObject, eventdata, handles, varargin)
   handles.output = hObject;
   guidata(hObject, handles);
end


function varargout = gui_OutputFcn(hObject, eventdata, handles)
   varargout{1} = handles.output;
   function pushbutton1_Callback(hObject, eventdata, handles)
   global IMG3;
   global filepath1;
   data = get(gcbf,'Userdata');
       pathname     = 'D:\Code\';
   [filename, pathname] = uigetfile([pathname '*.bmp'], 'Load image
   1');
   filepath1 = [pathname, filename]
   IMG3 = imread(filepath1);
   axes(handles.A01);
   set(handles.A01,'HandleVisibility','ON');
   image(IMG3);
   colormap(gray(256));
   axis equal; %sets the aspect ratio. (Show the image in its right
   ratio)
   axis tight; %Sets the axis limits to the arrange of the data.
   axis off; % Turn off all axis labeling
   set(handles.A01,'HandleVisibility','OFF');  ,
end

function pushbutton3_Callback(hObject, eventdata, handles)
   global IMG4;
   data = get(gcbf,'Userdata');
       pathname     = 'D:\Code\'
   [filename, pathname] = uigetfile([pathname '*.bmp'], 'Load image
   1');
```

```
    IMG4 = imread([pathname, filename]);
    axes(handles.A02);
    set(handles.A02,'HandleVisibility','ON');
    image(IMG4);
    axis equal; %sets the aspect ratio. (Show the image in its right
    ratio)
    axis tight; %Sets the axis limits to the arrange of the data.
    axis off; % Turn off all axis labeling
    set(handles.A02,'HandleVisibility','OFF');
end


function Register1_Callback(hObject, eventdata, handles)
    global IMG3;
    global IMG4;
    global Ghisto;
    global filepath1;
    [n1,n2] = size(IMG3)
    [n3,n4] = size(IMG4)
    disp('.....................................................
    ....');
    disp('START OF PROGRAM
    ................................................');
    disp('.....................................................
    ....');
    imwrite(IMG3,'temp.bmp');

        axes(handles.A21);
        set(handles.A21,'HandleVisibility','ON');
        image(IMG3);
        colormap(gray(256));
        axis equal;
        axis tight;
        axis off;


        axes(handles.A22);
        set(handles.A22,'HandleVisibility','ON');
        image(IMG4);
        colormap(gray(256));
        axis equal;
        axis tight;
        axis off;
        imwrite(IMG4,'temp1.bmp');
        clear IMbwc1;
    clear IMbwc2;
    [IMbwc1,xmin_i1,xmax_i1,ymin_i1,ymax_i1] = redsize(IMbw1);
    [IMbwc2,xmin_i2,xmax_i2,ymin_i2,ymax_i2] = redsize(IMbw2);
    IMbwc1 = im2bw(IMbwc1);
    tic
    [n1,n2] = size(IMbwc1);
    [n5,n6] = size(IMbwc2);
    index1 = 1;
    index2 =1;
    for ii = 2:n1
        for kk =1:n2
```

```
        if IMbwc1(ii,kk)==1
            pts1(index1,1)=kk;
            pts1(index1,2)=ii;
            index1 = index1+1;
            break;
        end
    end
    for jj = n2:-1:1
       if IMbwc1(ii,jj) == 1
          IMbwc1(ii,kk:1:jj) = 1;    %%%%%%%%%% FILL  STEP
          pts2(index2,1) = jj;
          pts2(index2,2) = ii;
          index2 = index2+1;
          break;
       end
    end      %%%%%%%%end of jj loop
end %%%%%%%%%%end of ii loop
IMbwc1 = logical(IMbwc1);
l1x1 = pts1(1,1); l1x2 = pts2(1,1); l1y1 = pts1(1,2); l1y2 =
pts2(1,2);
l1x3 = pts1(2,1); l1x4 = pts2(2,1); l1y3 = pts1(2,2); l1y4 =
pts2(2,2);
d1 = 0;
d2 = 0;
for ii = 1:index1-1
    for jj = 1:index2-1
        d=(pts1(ii,1)-pts2(jj,1))*(pts1(ii,1)-
pts2(jj,1))+(pts1(ii,2)-pts2(jj,2))*(pts1(ii,2)-pts2(jj,2));
        if d>d1
            l1x1 = pts1(ii,1);
            l1y1 = pts1(ii,2);
            l1x2 = pts2(jj,1);
            l1y2 = pts2(jj,2);
            d1 = d;
        end
    end
end

for ii = 1:index1-1
    for jj = 1:index2-1
d=(pts1(ii,1)-pts2(jj,1))*(pts1(ii,1)-pts2(jj,1))+(pts1(ii,2)-
pts2(jj,2))*(pts1(ii,2)-pts2(jj,2));
       l1x5 = pts1(ii,1);
       l1y5 = pts1(ii,2);
       l1x6 = pts2(jj,1);
       l1y6 = pts2(jj,2);
if (d>d2&&d<d1)&&(abs(l1x5-l1x1)>10)&&(abs(l1y5-
l1y1)>10)&&(abs(l1x6-l1x2)>10)&&(abs(l1y6-l1y2)>10)&&(abs(l1x5-
l1x2)>10)&&(abs(l1y5-l1y2)>10)&&(abs(l1x6-l1x1)>10)&&(abs(l1y6-
l1y1)>10)
                l1x3 = l1x5;
                l1y3 = l1y5;
                l1x4 = l1x6;
                l1y4 = l1y6;
                d2 = d;
        end
```

```
    end
end
df = sqrt(d1);
);

set(handles.A11,'HandleVisibility','ON');
axes(handles.A11);
imwrite(mat2gray(IMbwc1),'binary1.gif');
binary1 = imread('binary1.gif');

% colormap(gray(2));
image(binary1);
hold on
for jj = 1:index2-1
 if pts2(jj,1) >0
    y =  pts2(jj,2);
    x = pts2(jj,1);%-5;
    plot(x,y,'Color','r','LineWidth',40);
  end
end
for ii =1:index1-1
  y = pts1(ii,2);
  x = pts1(ii,1);%+5;
  plot(x,y,'Color','r','LineWidth', 40);
end
plot([l1x1 l1x2],[l1y1 l1y2],'Color','g','LineWidth', 4);
plot([l1x3 l1x4],[l1y3 l1y4],'Color','g','LineWidth', 4);
axis equal;
axis tight;
hold off
axis off;
set(handles.A11,'HandleVisibility','OFF');
 temp1 = d;
m11 = (l1y2-l1y1)/(l1x2-l1x1);
m12 = (l1y4-l1y3)/(l1x4-l1x3);
theta11 = atan(m11)*180/3.14159265359;
theta12 = atan(m12)*180/3.14159265359;
IMbwc2 = im2bw(logical(IMbwc2));
[n3,n4] = size(IMbwc2);
index1 = 1;
index2 = 1;
for ii = 1:n3
    for kk =1:n4
        if IMbwc2(ii,kk)==1
            pts1(index1,1)= kk;
            pts1(index1,2)= ii;
            index1 = index1+1;
         break;
        end
    end %%%%%%%%%%%End of kk loop
    for jj = n4:-1:1
       if IMbwc2(ii,jj) == 1
            IMbwc2(ii,kk:jj) = 1;
            pts2(index2,1)=jj;
            pts2(index2,2)=ii;
            index2 = index2+1;
```

```
        break;
        end
    end      %%%%%%%%end of jj loop
end %%%%%%%%%%end of ii loo
d1 =0;
d2 =0;
l2x1 = pts1(1,1); l2x2 = pts2(1,1); l2y1 = pts1(1,2); l2y2 =
pts2(1,2);
l2x3 = pts1(1,1); l2x4 = pts2(1,1); l2y3 = pts1(1,2); l2y4 =
pts2(1,2);
for ii = 1:index1-1
    for jj = 1:index2-1
       d=(pts1(ii,1)-pts2(jj,1))*(pts1(ii,1)-
pts2(jj,1))+(pts1(ii,2)-pts2(jj,2))*(pts1(ii,2)-pts2(jj,2));
        if d>d1
            d1 = d;
            l2x1 = pts1(ii,1);
            l2y1 = pts1(ii,2);
            l2x2 = pts2(jj,1);
            l2y2 = pts2(jj,2);
        end
    end
end


for ii = 1:index1-1
    for jj = 1:index2-1
       d=(pts1(ii,1)-pts2(jj,1))*(pts1(ii,1)-
pts2(jj,1))+(pts1(ii,2)-pts2(jj,2))*(pts1(ii,2)-pts2(jj,2));
       l2x5 = pts1(ii,1);
       l2y5 = pts1(ii,2);
       l2x6 = pts2(jj,1);
       l2y6 = pts2(jj,2);
       if (d>d2&&d<d1)&&(abs(l2x5-l2x1)>10)&&(abs(l2y5-
l2y1)>10)&&(abs(l2x6-l2x2)>10)&&(abs(l2y6-l2y2)>10)&&(abs(l2x5-
l2x2)>10)&&(abs(l2y5-l2y2)>10)&&(abs(l2x6-l2x1)>10)&&(abs(l2y6-
l2y1)>10)
                l2x3 = l2x5;
                l2y3 = l2y5;
                l2x4 = l2x6;
                l2y4 = l2y6;
                d2 = d;
         end
    end
end
ds = sqrt(d1);
set(handles.A12,'HandleVisibility','ON');
axes(handles.A12);
% colormap(gray(2));
imwrite(mat2gray(IMbwc2),'binary2.gif');
binary2 = imread('binary2.gif');
image(binary2);
hold on
plot([l2x1,l2x2],[l2y1, l2y2],'Color','g','LineWidth', 4)
plot([l2x3,l2x4],[l2y3, l2y4],'Color','g','LineWidth', 4)
for jj = 1:index2-1
```

```
 if pts2(jj,1)-5 >0
    y =  pts2(jj,2);
    x = pts2(jj,1);%-5;
    plot(x,y,'Color','r','LineWidth',40);
  end
end
for ii =1:index1-1
  y = pts1(ii,2);
  x = pts1(ii,1);%+5;
  plot(x,y,'Color','r','LineWidth', 40);
end

axis equal;
axis tight;
hold off
axis off;
set(handles.A12,'HandleVisibility','OFF');
toc
m21 = (l2y2-l2y1)/(l2x2-l2x1);
m22 = (l2y4-l2y3)/(l2x4-l2x3);
theta21 = atan(m21)*180/3.14159265359;
theta22 = atan(m22)*180/3.14159265359;
angle1 = (theta11-theta21);
angle2 = (theta12-theta22);
angle3 = (theta12-theta21);
angle4 = (theta11-theta22);

clear IMGc1;
clear IMGc2;
IMGc1 = imcrop(IM1,[xmin_i1,ymin_i1,xmax_i1-xmin_i1,ymax_i1-
ymin_i1]);
imwrite(IMGc1,'temp.gif');
IMGc2 = imcrop(IM2,[xmin_i2,ymin_i2,xmax_i2-xmin_i2,ymax_i2-
ymin_i2]);

  clear IMGcr2;
  clear IMbwofGcr2;
  scale1 = ds/df;
  x = [1.00,0];
  IMGcr2 = imrotate(IMGc2,-angle1,'bilinear');
  IMbwofGcr2 = im2bw(IMGcr2,level2);
  [IMbwofGcr2,xmin,xmax,ymin,ymax] = redsize(IMbwofGcr2);
  [n3,n4] = size(IMbwofGcr2);
  clear IMGcrc2;
  for ii = 1:n3
       for jj = 1:n4
            IMGcrc2(ii,jj) = uint8(IMGcr2(ymin+ii-1,xmin+jj-1));
       end
    end
    [n1,n2] = size(IMGc1);
    clear IMGcc1;
    clear IMbwofcrcs2;
    IMbwofcrcs2 = imresize(IMbwofGcr2,[n1 n2],'bilinear');
    % IMGcc1 is clean IMGc1
    [n1,n2] = size(IMGc1);
    [n3,n4] = size(IMGcrc2);
```

```
clear IMGcrcs2;
IMGcrcs2 = imresize(IMGcrc2,[n1 n2],'bilinear');

imwrite(IMGcrcs2,'temp1.gif');
x = [0,1];
minf = MInfo(x);
IMGcrcs2_180 = imrotate(IMGcrcs2,180);
imwrite(IMGcrcs2_180,'temp1.gif');
minf_180 = MInfo(x);
if minf_180<minf
    clear IMGcrcs2;
    IMGcrcs2 = IMGcrcs2_180;
    IMGcrc2 = imrotate(IMGcrc2,180);
    minf = minf_180;
end


clear IMGcr22;
clear IMbwofGcr22;
x = [1.00,0];
IMGcr22 = imrotate(IMGc2,-angle2,'bilinear');
IMbwofGcr22 = im2bw(IMGcr22,level2);
[IMbwofGcr22,xmin,xmax,ymin,ymax] = redsize(IMbwofGcr22);
[n3,n4] = size(IMbwofGcr22);
clear IMGcrc22;
for ii = 1:n3
   for jj = 1:n4
                IMGcrc22(ii,jj) = uint8(IMGcr22(ymin+ii-
     1,xmin+jj-   1))
      end
end
[n1,n2] = size(IMGc1);
clear IMbwofcrcs22;
IMbwofcrcs22 = imresize(IMbwofGcr22,[n1 n2],'bilinear');
% IMGcc1 is clean IMGc1
[n1,n2] = size(IMGc1);
[n3,n4] = size(IMGcrc22);
clear IMGcrcs22;
IMGcrcs22 = imresize(IMGcrc22,[n1 n2],'bilinear');
imwrite(IMGcrcs22,'temp1.gif');
x = [0,1];
minf2 = MInfo(x);
if minf2<minf
    clear IMGcrcs2;
    clear IMGcrc2;
    IMGcrcs2 = IMGcrcs22;
    IMGcrc2 = IMGcrc22;
    minf = minf2;
end
IMGcrcs22_180 = imrotate(IMGcrcs22,180);
imwrite(IMGcrcs22_180,'temp1.gif');
minf2_180 = MInfo(x);
if minf2_180<minf
    clear IMGcrcs22;
    clear IMGcrc2;
    IMGcrcs2 = IMGcrcs22_180;
```

```
    IMGcrc2 = imrotate(IMGcrc22,180);
    minf = minf2_180;
end



clear IMGcr23;
clear IMbwofGcr23;
x = [1.00,0];
IMGcr23 = imrotate(IMGc2,-angle3,'bilinear');
IMbwofGcr23 = im2bw(IMGcr23,level2);
[IMbwofGcr23,xmin,xmax,ymin,ymax] = redsize(IMbwofGcr23);
[n3,n4] = size(IMbwofGcr23);
clear IMGcrc23;
for ii = 1:n3
   for jj = 1:n4
        IMGcrc23(ii,jj) = uint8(IMGcr23(ymin+ii-1,xmin+jj-1));
     end
end
[n1,n2] = size(IMGc1);
clear IMbwofcrcs23;
IMbwofcrcs23 = imresize(IMbwofGcr23,[n1 n2],'bilinear');
[n1,n2] = size(IMGc1);
[n3,n4] = size(IMGcrc23);
clear IMGcrcs23;
IMGcrcs23 = imresize(IMGcrc23,[n1 n2],'bilinear');
imwrite(IMGcrcs23,'temp1.gif');
x = [0,1];
minf3 = MInfo(x);
if minf3<minf
    clear IMGcrcs2;
    clear IMGcrc2;
    IMGcrcs2 = IMGcrcs23;
    IMGcrc2 = IMGcrc23;
    minf = minf3;
end
IMGcrcs23_180 = imrotate(IMGcrcs23,180);
imwrite(IMGcrcs23_180,'temp1.gif');
minf3_180 = MInfo(x);
if minf3_180<minf
    clear IMGcrcs22;
    clear IMGcrc2;
    IMGcrcs2 = IMGcrcs23_180;
    IMGcrc2 = imrotate(IMGcrc23,180);
    minf = minf3_180;
end
clear IMGcr24;
clear IMbwofGcr24;
x = [1.00,0];
IMGcr24 = imrotate(IMGc2,-angle4,'bilinear');
IMbwofGcr24 = im2bw(IMGcr24,level2);
[IMbwofGcr24,xmin,xmax,ymin,ymax] = redsize(IMbwofGcr24);
[n3,n4] = size(IMbwofGcr24);
clear IMGcrc24;
for ii = 1:n3
   for jj = 1:n4
```

```
            IMGcrc24(ii,jj) = uint8(IMGcr24(ymin+ii-1,xmin+jj-1));
        end
    end
    [n1,n2] = size(IMGc1);
    % IMGcrc2 is clean IMGcr2
    %figure,imshow(IMbwc1);
    clear IMbwofcrcs24;
    IMbwofcrcs24 = imresize(IMbwofGcr24,[n1 n2],'bilinear');
    % IMGcc1 is clean IMGc1
    [n1,n2] = size(IMGc1);
    [n3,n4] = size(IMGcrc24);
    clear IMGcrcs24;
    IMGcrcs24 = imresize(IMGcrc24,[n1 n2],'bilinear');
    imwrite(IMGcrcs24,'temp1.gif');
    x = [0,1];
    minf4 = MInfo(x);
    if minf4<minf
        clear IMGcrcs2;
        clear IMGcrc2;
        IMGcrcs2 = IMGcrcs24;
        IMGcrc2 = IMGcrc24;
        minf = minf4;
    end
    IMGcrcs24_180 = imrotate(IMGcrcs24,180);
    imwrite(IMGcrcs24_180,'temp1.gif');
    minf4_180 = MInfo(x);
    if minf4_180<minf
        clear IMGcrcs24;
        IMGcrcs24 = IMGcrcs24_180;
        IMGcrc24 = imrotate(IMGcrc24,180);
        minf = minf4_180;
    end
    axes(handles.A21);
    set(handles.A21,'HandleVisibility','ON');
    image(IMGc1);
    colormap(gray(256));
    axis equal;
    axis tight;
    axis off;
    axes(handles.A22);
    set(handles.A22,'HandleVisibility','ON');
    image(IMGcrcs2);
    colormap(gray(256));
    axis equal;
    axis tight;
    axis off;
    imwrite(IMGcrc2,'temp1.gif');

%%%%%%%%%%%%%%%%%%SECOND STAGE STARTS HERE %%%%%%%%%%%%%%%%%%%%%%%%
tic
    clear x0
    x0=[0; scale1]
    ang =0;
    min = minf;
    for ii = -10:10
        x = [ii,1];
```

```
        f = MInfo(x);
        if f<min
            ang = ii;
            min = f;
        end
    end
IMG4 =imrotate(IMG4, ang,'bilinear');
IMG4 = imresize(IMG4,[n1,n2],'bilinear');
set(handles.A31,'HandleVisibility','OFF');
set(handles.A32,'HandleVisibility','OFF');
set(handles.A01,'HandleVisibility','OFF');
set(handles.A02,'HandleVisibility','OFF');
set(handles.A11,'HandleVisibility','OFF');
set(handles.A12,'HandleVisibility','OFF');
set(handles.A21,'HandleVisibility','OFF');
set(handles.A22,'HandleVisibility','OFF');

                tic
               x0 = [0,1];
               [x, fval]=fminsearch(@MInfo,x0);
               IMG4=imresize(IMG4, x(2), 'bilinear');
               IMG4=imrotate(IMG4, x(1),'bilinear');
               IMG4 = imresize(IMG4,[n1 n2],'bilinear');

    axes(handles.A31);
    set(handles.A31,'HandleVisibility','ON');
    image(IMG3);
    colormap(gray(256));
    axis equal;
    axis tight;
    axis off;
    axes(handles.A32);
    set(handles.A32,'HandleVisibility','ON');
    image(IMG4);
    colormap(gray(256));
    axis equal;
    axis tight;
toc
    a = int16(IMG4)-int16(IMG3);
    a = a+255;
    a = a/2;
    a = uint8(a);
    imwrite(a,filepath2);


axes(handles.AFinal);
set(handles.AFinal,'HandleVisibility','ON');
image(a);
axis equal;
axis tight;
axis off;
set(handles.AFinal,'HandleVisibility','ON');
set(handles.A31,'HandleVisibility','on');
set(handles.A32,'HandleVisibility','on');
set(handles.A01,'HandleVisibility','on');
set(handles.A02,'HandleVisibility','on');
```

```
    set(handles.A11,'HandleVisibility','on');
    set(handles.A12,'HandleVisibility','on');
    set(handles.A21,'HandleVisibility','on');
    set(handles.A22,'HandleVisibility','on');
end


function pushbutton33_Callback(hObject, eventdata, handles)
    global filepath2;

     [A,ns,r,c] = getImage(filepath2);
    globali =1;
    while(filepath2(globali)~='.')
       globali = globali+1;
    end
    tempstr = 'bbcd'
    filepath3 = [filepath2(1:globali),tempstr];
    EncodeImage(A,filepath3,globali,ns,r,c);
    orgsz = Filesize(filepath2);
    crsz =.Filesize(filepath3);
    cr = orgsz/crsz;
    set(handles.txtsf,'String',num2str(cr));
    clear filepath3;
    clear filepath2;
end


% --- Executes on button press in Refresh.
function Refresh_Callback(hObject, eventdata, handles)

    axes(handles.A02);
    image(256);
    axis equal;
    axis tight;
    axis off;

    axes(handles.A01);
    image(256);
    axis equal;
    axis tight;
    axis off;

    axes(handles.A12);
    image(256);
    axis equal;
    axis tight;
    axis off;

    axes(handles.A11);
    image(256);
    axis equal;
    axis tight;
    axis off;

    axes(handles.A21);
    image(256);
```

```
   axis equal;
   axis tight;
   axis off;

   axes(handles.A22);
   image(256);
   axis equal;
   axis tight;
   axis off;

   axes(handles.A32);
   image(256);
   axes(handles.A31);
   image(256);
   axis equal;
   axis tight;
   axis off;


   axes(handles.AFinal);
   image(256);
   axis equal;
   axis tight;
   axis off;

   set(handles.txtangle,'String','');
   set(handles.txtsf,'String','');

   clc;
   clear all;

end
```

Minfo.m

```
function f=MInfo(x)
   global histo;
   clear IMGcc1;
   clear IMGcrc2
   IMGcc1 = imread('temp.bmp');
   IMGcrc2 = imread('temp1.bmp');
   clear IMGRe;
   clear IMGRR2;
   clear IM1;
   clear IM2;
   clear IMGR3;
   clear IMGRR2;
   clear IMGR
   clear IMGR31;
   IMGR=imrotate(IMGcrc2, x(1),'bilinear');
   IMGRR2=imresize(IMGR, x(2), 'bilinear');
   level2 = graythresh(IMGRR2);

   IM1 =im2bw(IMGRR2,level2);
```

```
[IM2,xmin,xmax,ymin,ymax] = redsize(IM1);
[n3 n4]=size(IM2);
for ii = 1:n3
    for jj = 1:n4
          IMGR31(ii,jj) = uint8(IMGRR2(ymin+ii-1,xmin+jj-1));
    end
end

level1 = graythresh(IMGcc1);
IM3 =im2bw(IMGcc1,level1);
[n1,n2]=size(IMGcc1);
[IM4,xmin,xmax,ymin,ymax] = redsize(IM3);
[n1 n2]=size(IM4);
for ii = 1:n1
    for jj = 1:n2
          IMGR32(ii,jj) = uint8(IMGcc1(ymin+ii-1,xmin+jj-1));
    end
end

IMGR3Re = imresize(IMGR31,[n1 n2],'bilinear');
[n1 n2]=size(IMGR32);
[n3 n4]=size(IMGR3Re);
N=256;
histo=zeros(N,N);
IM1 = uint8(IMGcc1);
IM2 = uint8(IMGR3Re);
if n3>n1
    for ii=1:n1;
        for jj=1:n2;
            histo(IM1(ii,jj)+1,IM2(ii,jj)+1)=
histo(IM1(ii,jj)+1,IM2(ii,jj)+1)+1;
        end
    end
else
    for ii=1:n3;
        for jj=1:n4;
            histo(IM1(ii,jj)+1,IM2(ii,jj)+1)=
histo(IM1(ii,jj)+1,IM2(ii,jj)+1)+1;
        end
    end
end
[r,c] = size(histo);
b= histo./(r*c);
imag2_in=sum(b);
imag1_in=sum(b');
Im1_entpy=0;
for i=1:c
    if( imag2_in(i)~=0 )
       Im1_entpy = Im1_entpy + -(imag2_in(i)*(log2(imag2_in(i))));
    end
end
Im2_entpy = 0;
for i=1:r
    if( imag1_in(i)~=0 )
       Im2_entpy = Im2_entpy  + -(imag1_in(i)*(log2(imag1_in(i))));
    end
```

```
    end
        h_xy = -sum(sum(b.*(log2(b+(b==0))))); % joint entropy
        f=-(Im2_entpy +Im1_entpy -h_xy);% Mutual information
end
```

---

redsize.m

```
function [im,xmin,xmax,ymin,ymax] = redsize(In)
    [n1,n2] = size(In);
    I = In;
    ymin = 1;xmin =1;ymax=n1;xmax =n2;
    for ii = 1:n1
        for jj =1:n2
            if I(ii,jj) ~= 0
                ymax = ii;
                break;
            end
        end
    end
    for jj =1:n2
        for ii = 1:n1
            if I(ii,jj) ~= 0
                xmax = jj;
                break;
            end
        end
    end
    for jj =n2:-1:1
        for ii = 1:n1
            if I(ii,jj) ~=0
                xmin = jj;
                break;
            end
        end
    end
    for ii = n1:-1:1
        for jj =1:n2
            if I(ii,jj) ~= 0
                ymin = ii;
                break;
            end
        end
    end
    %if xmax-xmin >30 && ymax-ymin>30
        for ii = ymin:ymax
            for jj = xmin:xmax
                im(ii-ymin+1,jj-xmin+1) = In(ii,jj);
            end
        end
end
```

Decode.m

```
function [A] = Decode(filepath)
   filepath = str2mat(filepath);
   globali = 1;
   while(filepath(globali)~='.')
      globali = globali+1;
   end
   ComprFile_path([1:globali]) = filepath([1:globali]);
   ComprFile_path(globali+1:globali+4) = ['b','b','c','d'];
   fid1 = fopen(ComprFile_path,'rb');
   r =  fread(fid1,1,'ubit16')
   c =  fread(fid1,1,'ubit16')
   if r>c
       n = r;
   else
       n = c;
   end
   nb = ceil(log2( double (n) ));
   ns = power(2,nb);
   ns = uint32(ns);
   N = CaclulateSize(ns);
   [blk_nofbits,NoOfBlks1] = ReadArray(fid1,2);
   [blkmin,NoOfBlks] = ReadArray(fid1,2);
   if NoOfBlks1 == NoOfBlks
       disp(' Working');
else
       disp('Eroor');
   end
   disp('stage 1');
   N = CaclulateSize(ns);
   A = zeros(ns);
   FinalRead(fid1,ComprFile_path,globali)
   disp('stage 2');
   DecodeAndRead(ComprFile_path, globali);
   disp('stage 3');
   ComprFile_path(globali+1:globali+2) = ['b','c'];
   fid = getfids(20,ComprFile_path, globali,2);
   A =
   ReadImage(A,ns,ns,blk_nofbits,blkmin,ComprFile_path,globali,1,fid);
   disp('stage 4');
   [A,idx] = decompress_img(A,blkmin,blk_nofbits,ns,ns,1);
   disp('stage 5');
   A = uint8(A(1:r,1:c));
   imwrite(A,'D:\Code\a.bmp');
   imshow(A)
   fclose('all');
   DeleteTempData(ComprFile_path, globali);
   disp('stage 6');
end
```

decompress_block.m

```
function decompressed_block =
decompress_block(ip_blk,blockmin_value,block_nofbits_value,blk_size)

   decompressed_block = zeros(blk_size);

   if block_nofbits_value == 0
       decompressed_block([1:blk_size], [1:blk_size])= blockmin_value;
   % Assign all the values to max1 value
   else
       decompressed_block =
   mydecode(ip_blk,block_nofbits_value,blk_size);
       decompressed_block = decompressed_block + blockmin_value  ;
   end
   end %End of Function


   function [A,blkmin,blk_nofbits,idx] =
   decompress_img(A,blkmin,blk_nofbits,ns,blk_size,idx)
   global cnt1;

    if blk_size >= 2
   %     A
       blkmin_value = blkmin(idx);
       blk_nofbits_value =  blk_nofbits(idx) ;

       if blk_nofbits_value > 0
           if blk_size > 2
              m = uint32(0);
              n = uint32(0);
              sz = blk_size/2;
              for m = 0:1
                  for n = 0:1
                      idx = idx+1;
                      [A(
   m*sz+[1:sz],n*sz+[1:sz]),blkmin,blk_nofbits,idx] = decompress_img(A(
   m*sz+[1:sz],n*sz+[1:sz]),blkmin,blk_nofbits,ns,sz,idx);
                  end
              end
   %          A = mydecode(A,blk_nofbits_value,blk_size);
           end

       end
       A = A + blkmin_value ;
end
```

DecodeAndRead.m

```
function [flag] =  DecodeAndRead(ComprFile_path,globali)
   for i = 2:20
           if i<10
```

```
                ComprFile_path(globali+1:globali+5) =
    ['b','n','2','0',num2str(i)];
                fid2 = fopen(ComprFile_path,'rb');
            else
                ComprFile_path(globali+1:globali+5) =
    ['b','n','0',num2str(i)];
                fid2 = fopen(ComprFile_path,'rb');
            end
          if fid2 > 0

                [arr,sz] =  ReadArray(fid2,1);
                a = num2str(i);
                ComprFile_path(globali+1:globali+2) = ['b','c'];
                if i<10
                    ComprFile_path(globali+3:globali+5) =
    ['2','0',a(1)];
                end
                if i>9 && i <21
                    ComprFile_path(globali+3:globali+5) = ['2',a(1:2)];
                end
                fid(i) = fopen(ComprFile_path,'wb');
                bitstr = ['ubit',num2str(i)];
                fwrite(fid(i),arr,bitstr);
                fclose(fid2);
                end
    end
    flag = 1;
end
```

ReadBlock.m

```
function  A =
ReadBlock(A,blk_nofbits_val,blkmin_val,blk_size,ComprFile_path,globali,
fid)
    if blk_nofbits_val == 0
        A = zeros(blk_size);
        return;
    end

    nobstr = ['ubit',num2str(blk_nofbits_val),'=>ubit8'];
    % fprintf(1,'blk_size = %d\n',blk_size);
    for j = 1:blk_size
        for k =1:blk_size
                  isy =  (fread(fid,1,nobstr));
                  A(j,k) = isy;
        end
    end
    x = 1;
end
```

rle_decode.m

```
function [oparr,arrsize] = rle_decode(ipval,ipcount,arr_size )
   idx = 1;
   for i = 1:arr_size
       x = ipcount(i);
       oparr([idx:idx+x-1]) =  ipval(i);
       idx = idx+x;
   end
   arrsize = idx-1;
end
```

ReadHuffmanCode.m

```
function [Info,h_arr] = ReadHuffmanCode(fid)
   flag = 0;
   clear nob;
   clear nobstr;
   clear Info;
   clear h_arr;
   nob = fread(fid,1,'uint8');
   nobstr = ['ubit',num2str(nob)];
   Info.NoOfSymb = fread(fid,1,'ubit32');
   Info.nob = fread(fid,Info.NoOfSymb,'ubit8');
   Info.len = fread(fid,1,'ubit32');
           size_symbarr     = 0;
           size_symbarr     = sum(Info.nob);
   bc = 0;
   [Info.symbl,c]    = fread(fid,Info.NoOfSymb,nobstr);
   bc = bc+c*nob;
   [Info.Symbenc_arr,c] = fread(fid,size_symbarr,'ubit1');      %We can
   calculate size of this array by adding all no of bits
   bc = bc+c;
   [h_arr,c] = fread(fid,Info.len,'ubit1');
   bc = bc+c;
   x = 8-mod(bc, 8);
   temp = fread(fid,x,'ubit1');
   flag = 1;
end
```

ReadImage.m

```
function [A,idx] = ReadImage(A,ns,
blk_size,blk_nofbits,blkmin,ComprFile_path,globali,idx,fid)
   global cnt;
    x = blk_nofbits(idx);
       if (blk_size == 2)
            idx = uint32(idx);
            ns = uint32(ns);
            if x~=0
```

```
            A =
   ReadBlock(A,blk_nofbits(idx),blkmin(idx),blk_size,ComprFile_path,glo
   bali,fid(x));
            end
        end
        if blk_size > 2 && x>0
            %CalculateChildIndex(ns,blk_size,idx);
           blk_size = blk_size/2;
           idx = idx+1;
           [A(1:blk_size,1:blk_size),idx] =
   ReadImage(A(1:blk_size,1:blk_size),ns,
   blk_size,blk_nofbits,blkmin,ComprFile_path,globali,idx,fid);
           idx = idx+1;
           [A([1:blk_size],blk_size+[1:blk_size]),idx] =
   ReadImage(A([1:blk_size],blk_size+[1:blk_size]),ns,
   blk_size,blk_nofbits,blkmin,ComprFile_path,globali,idx,fid);
           idx = idx+1;
           [A(blk_size+[1:blk_size],[1:blk_size]),idx] =
   ReadImage(A(blk_size+[1:blk_size],[1:blk_size]),ns,
   blk_size,blk_nofbits,blkmin,ComprFile_path,globali,idx,fid);
           idx = idx+1;
           [A(blk_size+[1:blk_size],blk_size+[1:blk_size]),idx] =
   ReadImage(A(blk_size+[1:blk_size],blk_size+[1:blk_size]),ns,
   blk_size,blk_nofbits,blkmin,ComprFile_path,globali,idx,fid);
end
```

HuffmanDec.m

```
function [dec_arr,sz] = HuffmanDec(Info,h_arr);
   idx = 1;
   dict = cell(Info.NoOfSymb,2);
   Info.Symbenc_arr = Info.Symbenc_arr';
   for i = 1: Info.NoOfSymb
       dict{i,1} = double(Info.symbl(i));
       dict{i,2} = Info.Symbenc_arr(idx:idx+Info.nob(i)-1);
       idx = idx + Info.nob(i);
   end
   dec_arr = [];
   h_arr = double(h_arr);
   dec_arr = huffmandeco(h_arr,dict);
   sz = size(dec_arr,1);
end
```

MakeTable.m

```
function flag = MakeTable(filepath)
   g = size(filepath,2);
   filepath(g+1:g+3) = ['b','m','p'];
   orgsz = FileSize(filepath1);
   filepath1(g1+1:g1+3) = ['j','p','2'];
   csz = FileSize(filepath1);
   crwo = orgsz/csz;
```

```
filepath(g+1:g+3) = ['g','i','f'];
imwrite(A,filepath);
fs =  FileSize(filepath)
crgif = orgsz/fs;


filepath(g+1:g+3) = ['r','l','e'];
[r,c] = size(A);
cnt =1;
cnt = uint16(cnt);
for i = 1:r
    for j =1:c
        temp(cnt) = A(i,j);
        cnt = cnt+1;
    end
end
fid = fopen(filepath,'wb');
WriteArray(fid,temp,cnt-1,3);
fclose(fid);
fs =  FileSize(filepath);
crrle = orgsz/fs;

filepath(g+1:g+3) = ['h','m','n'];
fid = fopen(filepath,'wb');
x = uint8(cnt/3);
WriteArray(fid,temp,cnt-1,1);
WriteArray(fid,temp(1:x),x,1)
fclose(fid);
fs =  FileSize(filepath);
crhmn = orgsz/fs

filepath(g+1:g+4) = ['b','b','c','d'];
fs =  FileSize(filepath);
crbbcd = orgsz/fs
% orgsz = FileSize(filepath)
% crw = orgsz/fs;


filepath = filepath(1:g+3);
filepath(g+1:g+3) = ['j','p','2'];
fid = fopen(filepath,'rb')
if fid == -1
    crjp2 = -1
else
fclose(fid);
fs =FileSize(filepath);
crjp2 = bsz/fs;
end

 entry2 = xlsread('a.xls')
[r,k] = size(entry2);
entry2(r+1,:) = [crgif, crrle, crhmn, crjp2, crbbcd,crwo,crw];
s = xlswrite('a.xls', entry2)
close('all');
end
```

FinalRead.m

```
function [flag] =  FinalRead(fid1,ComprFile_path,globali)
   x = fread(fid1,1,'ubit8');
   for i = 1:20
        if x==i
            [arr,sz] =  ReadArray(fid1,1);
            if i<10
                  ComprFile_path(globali+1:globali+5) =
   ['b','n','2','0',num2str(i)];
                  fid2 = fopen(ComprFile_path,'wb');
             else
                  ComprFile_path(globali+1:globali+5) =
   ['b','n','0',num2str(i)];
                  fid2 = fopen(ComprFile_path,'wb');
            end
             arr = arr';
             WriteArray(fid2,arr,sz,1);
             fclose(fid2);
             x = fread(fid1,1,'ubit8');
       end
end
```