

P-2828



**CONTENT BASED IMAGE RETRIEVAL OF SPINE  
X-RAYS**

**A PROJECT REPORT**

*Submitted by*

**SRIHARI.R**

**71205104050**

**SASSHI KUMAR.K**

**71205104303**

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF ENGINEERING**

*in*

**COMPUTER SCIENCE AND ENGINEERING**

**KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE.**

*(Accredited by NBA, An ISO 9001:2000 Certified Institution)*




**ANNA UNIVERSITY: CHENNAI-600 025**

**APRIL 2009**

## BONAFIDE CERTIFICATE

Certified that this project report entitled "CONTENT BASED IMAGE RETRIEVAL OF SPINE X-RAYS" is the bonafide work of R.Srihari and K.Sasshi Kumar, who carried out the research under my supervision. Certified also, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.



SIGNATURE

Dr. S. Thangasamy

HEAD OF THE DEPARTMENT

Department of Computer Science  
& Engineering,  
Kumaraguru College of Technology,  
Coimbatore-641006.



SIGNATURE

Mrs. D. Chandrakala

SUPERVISOR

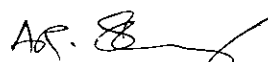
Assistant Professor

Department of Computer Science  
& Engineering,  
Kumaraguru College of Technology,  
Coimbatore-641006.

The candidates with University Register Nos. 71205104050 and 71205104303 were examined by us in the project viva-voce examination held on 28/04/2009.



INTERNAL EXAMINER



EXTERNAL EXAMINER

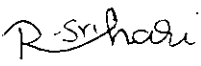
## DECLARATION

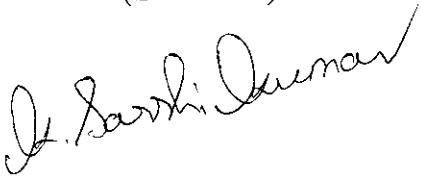
We hereby declare that the project entitled “**Content Based Image Retrieval Of Spine X-Rays**” is a record of original work done by us and to the best of our knowledge, a similar work has not been submitted to Anna University or any Institutions, for fulfillment of the requirement of the course study.

The report is submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Computer Science and Engineering of Anna University, Chennai.

Place : Coimbatore

Date : 28/04/2009

  
(Srihari.R)

  
(Sasshi Kumar.K)

## ACKNOWLEDGEMENTS

We express our hearty gratitude to our beloved Correspondent, **Prof. Dr. K. Arumugam, B.E., (Hons), M.S. (USA), M.I.E.**, for giving us this great opportunity to pursue this course.

We thank, **Dr. Joseph V. Thanikal, Ph.D.**, Principal, Kumaraguru College of Technology, Coimbatore, for being a constant source of inspiration and providing us with the necessary facilities to work on this project.

We would like to make a special acknowledgement and thanks to **Dr. S. Thangasamy, Ph.D.**, Dean, Professor and Head of Department of Computer Science and Engineering, for his support and encouragement throughout the project.

We extend our sincere thanks to **Dr. T. Arunachalam, M.Sc., M. Phil., Ph.D.**, Asst. Professor of Mathematics, Department of Science & Humanities for his valuable suggestions and guidance.

We express deep gratitude and gratefulness to our guide **Mrs. D. Chandrakala M.E.**, Assistant Professor, Department of Computer Science and Engineering, for her supervision, enduring patience, active involvement and guidance.

We would also like to thank our project coordinator, **Mrs.P.Devaki M.S.**, Assistant Professor, for her support during the course of our project.

We would like to convey our honest thanks to all **members of staff** of the Department for their enthusiasm and wealth of experience from which we have greatly benefited.

We also express our profound gratitude to our parents and friends for their moral support.

## **ABSTRACT**

R.Srihari – 05BCS50,  
K.Sasshi Kumar – 03BCS63.

There is a growing need for efficient visual information retrieval systems in the field of bio-medical engineering. To minimize the retrieval of irrelevant results, the system must take into account particular features of images that form the basic attributes of an image. Shape is the only feature that effectively describes various pathologies identified by medical experts and reliably found in image collection. The shape methods perform well on visual inspection of overall shape boundaries but they fall short in meeting the needs of determining similarity between the vertebral shapes based on pathology. This project proposes a new method for content-based image retrieval, wherein one of the basic characteristics of an image is described, namely, its shape. The required shape is segmented from the background and descriptions of these shapes are formed.

Our method allows users to provide query X-ray images which has the feature of something the user is expecting to retrieve from the patient X-ray images present in the database. Once the user introduces a query, area of interest is to be selected and morphological operations are performed and the image of the vertebra in the area of interest is obtained and is simplified

using discrete curve evolution method and then a tangent space representation is constructed for the simplified curve obtained as a result of discrete curve evolution. Similarity measures based on the normalized length of a line segment and the turning functions are used to recover similar images from the database. This application finds direct use in medical field for retrieving similar images to the given one to enable them to research about a specific feature.

## LIST OF SYMBOLS AND ABBREVIATIONS

CBIR	-	Content Based Image Retrieval of Spine X-rays
FV	-	Feature Vector
DCE	-	Discrete Curve Evolution
TSR	-	Tangent Space Representation
K	-	Relevance Measure
$s_x$	-	Segment x
$l(s)$	-	length of segment s
$\beta(s_1, s_2)$	-	Turn angle between segments $s_1$ and $s_2$
IDB	-	Image Database
FDB	-	Feature Vector Database

## LIST OF FIGURES AND TABLES

### LIST OF FIGURES

NAME	DESCRIPTION	PAGE NO.
Figure 3.1	Image Enhancement	14
Figure 3.2	X-Ray Preprocessing	15
Figure 3.3	Sobel Operator Example	17
Figure 3.4	Preprocessing	20
Figure 3.4	Discrete Curve Evolution Process	22
Figure 3.5	Tangent Space Representation	23
Figure 3.6	Feature Vector Extraction	26
Figure 3.7	Image Querying and Retrieval	27
Figure 4.1	Communication between various elements of the system	30
Figure 4.2	Overall Communication	31
Figure 4.2	Architecture of Image Querying and Retrieval	24



## LIST OF TABLES

NAME	DESCRIPTION	PAGE NO.
Table 4.1	Members of vertices retrieval module	32
Table 4.2	Members of elimination of vertices module	35
Table 4.3	Members of conversion_TSR module	37
Table 4.4	Members of Add image module	38
Table 4.5	Members of Find related image module	39

# TABLE OF CONTENTS

<b>BONAFIDE CERTIFICATE</b>	<b>II</b>
<b>DECLARATION</b>	<b>III</b>
<b>ACKNOWLEDGEMENT</b>	<b>IV</b>
<b>ABSTRACT</b>	<b>V</b>
<b>LIST OF SYMBOLS AND ABBREVIATIONS</b>	<b>VII</b>
<b>LIST OF FIGURES AND TABLES</b>	<b>VIII</b>
<b>1. Introduction</b>	<b>1</b>
Introduction	1
Existing Systems	4
Proposing Method	5
About The Implementation	6
<b>2. Background Study and Related Works</b>	<b>7</b>
Image Data Management	7
Related Works	8
<b>3. Methodology</b>	<b>14</b>
Preprocessing of X-Ray Images	14
Feature Extraction Method	21
Image Querying and Retrieval	27
<b>4. System Design</b>	<b>30</b>

Preprocessing	32
Vertices Retrieval	32
Feature Vector Extraction	33
Reduction of Vertices_DCE	34
Relevance Measure Computation	34
Elimination of Vertices	35
Conversion_TSR	36
Add Image	37
Find Related Images	38
View Records	40
<b>5. Programming Environment</b>	<b>41</b>
Hardware Requirements	41
Software Requirements	41
<b>6. Future Enhancements</b>	<b>42</b>
Enhancements to the UI	42
Enhancements to the Algorithm	43
<b>7. Conclusion</b>	<b>44</b>
<b>8. Appendix</b>	<b>45</b>
MatLab7 Code Samples	45
Screenshots	58
<b>9. References</b>	<b>69</b>

---

## Chapter I

### INTRODUCTION

---

#### 1.1. INTRODUCTION:

‘A picture is worth a thousand words’ says an old Chinese proverb. This saying is true to the core. One can easily convey the meaning of a thousand words, and much more, with some simple strokes of the hand. Needless to say, one must have some mastery of the art, but any creation, be it even in words, needs the creator to have a little mastery over the basics. The real beauty of a pictorial representation lies in its ability to mimic a lengthy lecture, since human beings find it easy to visualize something when it is drafted pictorially. With such a powerful tool for understanding the world around us, it comes not as much of a surprise when it is stated that we have tried to exploit it in all ways possible, ever since we arrived on Planet Earth.

We use images in every conceivable field of study. With ever increasing need for image recognition and processing in the fields of medicine, science, engineering, technology and what not, an efficient method for image matching is required. Surprisingly, not much is in the offering, when it comes to analyzing the similarity between two objects that convey visual information, in this case, two images. The general problem of developing algorithms for the automated or computer-assisted retrieval of

images by structural contents is a significant research challenge .This is particularly so in the case of biomedical images, where the structures of interest are commonly irregular, and may be partially occluded. Examples are the images created by digitizing film x-rays of the human cervical and lumbar spines, digitized color slides of the uterine cervix, color endoscopy, endoscopic ultra-sonography, etc. Text data in the form of patient or survey data is commonly associated with medical images. Systems in use allow retrieval of image data through a text based query. Content-based image retrieval aims to allow researchers and medical practitioner access to the images directly by their content. We envisage that the development of a system that provides such access would have many applications in education, research, clinical trials, diagnosis, etc. For example, a medical school faculty member who is an expert in degenerative spine disease could query for examples of severe disc space narrowing for both sexes for the cervical spine; or a clinician could use it for searching for images similar to patient's present image pathology or injury .Text-based image searching techniques have their limitations, there is a high probability for obtaining highly irrelevant results for one, and the sooner they are replaced with a much superior method, the better.

Various medical imaging techniques including X-rays, magnetic resonance imaging (MRI), computerized tomography (CT), and positron emission tomography (PET) have been increasingly producing vast amounts of patient images, which require a considerable amount of diagnostic work. There are thus numerous emerging needs for computerized decision support in medical research, especially in the area of medical imaging.

Content-Based Image Retrieval (CBIR) has also recently gained popularity for medical image management. Based on the characteristics of different pathologies visible in medical images, one or more relevant pathology described using image or anatomy color, shape, and texture features can be extracted and represented for computing similarity between images. Since images possessing similar features can be expected to share similar pathology, CBIR results can provide for a diagnostic reference through retrieval of visually similar images which could include even past diagnosis history (when available).

A critical limitation to CBIR techniques is the gap between high-level human visual perception and the low-level computerized features that CBIR relies on. And our project is one endeavor to better the techniques in vogue.

## **1.2. EXISTING SYSTEMS:**

Traditional methods for bio-medical image retrieval use a set of keywords as queries for patient data. Current search engines are inherently text based, using keywords to help the users to find bio-medical images. Though this is widely popular and is a vital part of the Internet, this method has its own limitations which prevent it from working more effectively. In this process, the keyword associated with an image is matched with the user query (which is also textual), and the image is returned if the two keywords match. Since any keyword can be associated with any image, consequently, the retrieved results have a high proportion of content unrelated to the user query. In many bio-medical undertakings, image processing has become an integral part and the current text based method, does not have much to offer in terms of efficiency.

### **1.3. PROPOSED METHOD:**

To overcome this problem, and to make inroads into the field of biomedical image retrieval and processing, many methods have been proposed that take into account certain features of images when making a comparison between any two of them. In the proposed method, one of the most fundamental attributes of an image, its shape, is considered for processing the images along with some features of the image like the protruding shape structure. The shape of the region of interest is taken to be the defining attribute that is compared with that of other images for the retrieval of relevant results. Since shape is nothing but the outline that an object presents to the external world, two objects that present identical outlines could well be taken to belong to the same category.



#### **1.4. ABOUT THE IMPLEMENTATION:**

This project addresses a shape description approach for the retrieval of in image retrieval along with text based queries and to increase the overall efficiency of the image retrieval process. The proposed model is implemented and evaluated using MatLab Version 7.5 under Microsoft Windows XP Operating System. The images and the information used are collected from the Internet and other electronic and bibliographical information sources. Image enhancement is done using ImageJ a public domain Java Image processing software. It can display, edit, analyze, process, save and print 8-bit, 16-bit and 32-bit images. It can read many image formats including TIFF, GIF, JPEG, BMP, DICOM, FITS and "raw". It supports "stacks", a series of images that share a single window. It is multithreaded, so time-consuming operations such as image file reading can be performed in parallel with other operations.

---

## **Chapter II**

### **BACKGROUND STUDY AND RELATED WORKS**

---

#### **2.1. IMAGE DATA MANAGEMENT:**

Even though the process of digitization enhances the usability and manipulation of images, it does not in any way make it easier to manage. Some form of cataloguing and indexing is still very much necessary. A workshop sponsored by the US National Science Foundation in 1992 reinforces the need for efficient storage and retrieval of images. In that workshop, the participants examined the various issues involved in managing and retrieving visual information, with the knowledge that visual information is likely to play a vital part in electronic communication. However, significant research advances, involving collaboration between numbers of disciplines, would be needed before image providers could take full advantage of the opportunities offered. Many critical areas like data representation, feature extraction and indexing, image query matching and user interfacing, still are not fully exploited and a lot of research needs to be done in these areas.

Of the various issues considered, one main issue was the difficulty of locating a desired image in a large and varied repository. Unlike a small collection of images, which can easily be browsed manually to find the desired image, more efficient techniques are needed with collections

containing hundreds or thousands of items. Moreover, the users come from different fields of science and technology, ranging from amateur to professional, all with different needs and different methods of approach. The same individual may have different needs at different times. Such users include professional groups, students, doctors and pathological investigators etc. An efficient method for management of image data is needed, that would satisfy the needs of all these people.

## **2.2. RELATED WORKS:**

The following is a list of different approaches, methods and related works which strive to arrive at a better model for our work-suite.

### *“IMAGE RETRIEVAL BY ONTOLOGICAL DESCRIPTION OF SHAPES (IRONS), EARLY RESULTS (IEEE 2004)”*

This paper addresses the problem of retrieving documents that contain visual information. Current visual information retrieval systems sometimes retrieve irrelevant documents or documents unrelated to the user’s query. This problem is caused by the use of low-level image descriptors; furthermore, these descriptors hardly have a semantic weight. In this work we address the image retrieval problem based on shape, since shape has a meaning by itself. On the other hand, an extension of the ontology concept, which is used in information retrieval based on text, is proposed in the image domain. Likewise, we present the Image Retrieval by Ontological Description of Shapes (IRONS) System. IRONS has been implemented and evaluated in order to analyze its utility, efficiency, and advantages comparing with well-known systems.

“USING CONTEXTUAL INFORMATION FOR IMAGE RETRIEVAL (IEEE 2001)”

Visual Information Retrieval presents many challenges for the computer vision community. The terabytes of visual information stored in digital image and video libraries will remain inaccessible if the problems of indexing and retrieval are not addressed. In this paper, we present techniques for content based image retrieval using higher level contextual information. The content is represented and queried using attributed relational graphs, with color attributes and relaxation labeling techniques. We present retrieval examples using both synthetic and real images of national flags. This, although a simplistic problem, highlights the shortcomings, and difficulties associated with content based retrieval systems.

“CONTENT-BASED RETRIEVAL AND IMAGE DATABASE TECHNIQUES: CONTENT-BASED VISUAL INFORMATION RETRIEVAL (DISTRIBUTED MULTIMEDIA DATABASES 2002)”

This chapter provides a survey of the state-of-the-art in the field of Visual Information Retrieval (VIR) systems, particularly Content-Based Visual Information Retrieval (CBVIR) systems. It presents the main concepts and system design issues, reviews many research prototypes and commercial solutions currently available and points out promising research directions in this area.

“STRUCTURING THE VISUAL CONTENT OF DIGITAL LIBRARIES USING CBIR SYSTEMS (IEEE 2000)”

In this article, the benefits of CBIR - content-based image retrieval - systems for retrieval and organization of visual content in databases of

digital libraries is described. Unlike text-based retrieval systems which work with manually annotated keywords as metadata, CBIR systems use automatically extracted numerical representations of perceptual features like color, texture or shape. This technique enables users of digital libraries to retrieve visual content without the help of textual metadata, which in some cases may be either not existent or not sufficient for a special purpose. It is our approach to support maintainers of digital libraries in organizing large image volumes using CBIR systems. Finding clusters of similar content or providing clusters with suitable keywords are new application contexts of CBIR. We propose a method for structuring visual content and evaluate retrieval results of CBIR systems with regard to their applicability to this method.

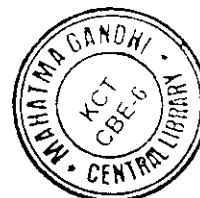
*“AN INTEGRATED APPROACH TO SHAPE AND COLOR-BASED  
IMAGE RETRIEVAL OF ROTATED OBJECTS USING HIDDEN MARKOV  
MODELS (HIDDEN MARKOV MODELS 2001)”*

An integrated approach to shape and color-based image retrieval, where the cues color and shape are both utilized in a local rather than a global way is presented in this paper. An experimental retrieval system has been developed, which enables the user to search a color image database intuitively for presenting simple sketches. In order to be able to perform elastic matching, which is especially needed in sketch-based image retrieval, objects in the images are represented by Hidden Markov Models. The use of streams (sets of features that are assumed to be statistically independent) within the HMM framework allows the integration of shape and color derived features into a single model, thereby allowing to control the influence of the different streams via stream weights. The approach has been

evaluated on a color image database containing 120 different isolated objects with arbitrary orientation and showed good retrieval results with several users. Furthermore, the use of HMMs allows efficient pruning and thus a fast retrieval even with large databases.

“SHAPE-BASED IMAGE RETRIEVAL APPLIED TO TRADEMARK IMAGES (KLUWER ACADEMIC PUBLISHERS 2004)”

In this chapter, a new shape-based, query-by-example, image database retrieval method is proposed, that is able to match a query image to one of the images in the database, based on a whole or partial match. The proposed method has two key components: the architecture of the retrieval and the features used. Both play a role in the overall retrieval efficacy. The proposed architecture is based on the analysis of connected components and holes in the query and database images. The features we propose to use are geometric in nature, and are invariant to translation, rotation, and scale. Each of the suggested three features is not new per se, but combining them to produce a compact and efficient feature vector is. We use hand-sketched, rotated, and scaled, query images to test the proposed method using a database of 500 logo images. We compare the performance of the suggested features with the performance of the moment's invariants. The suggested features match the moment's invariants in rotated and scaled queries and consistently surpass them in hand sketched queries. Moreover, results clearly show that the proposed architecture significantly increase the performance of the two feature sets.



*“SHAPE SIMILARITY IMAGE RETRIEVAL BY HYPOTHESIS AND TEST  
(IEEE 2004)”*

Image Retrieval by shape similarity systems usually either focus their attention on images with isolated objects (uniform backgrounds and no occluding objects) or perform time consuming exhaustive initializations to localize the portion of the image possibly containing the searched shape. We propose a Content Based Image Retrieval (CBIR) method merging classic alignment techniques for efficient shape localization with an innovative verification strategy able to deal with an inexact matching between the searched and the found shape.

*“ BIOMEDICAL INFORMATION SYSTEM FOR COMBINED CONTENT-  
BASED RETRIEVAL OF SPINE X-RAY IMAGES AND ASSOCIATED TEXT  
INFORMATION (IEEE 2002)”*

Bio-medical image processing poses serious challenges as unlike the normal image databases like trademark and others where images of different visibly differentiable shape features are available, in spine X-ray image retrieval all the images are more or less similar. So careful examination of the features should be done. The representation of the images also plays an important role in retrieval. Textual information can be combined with the queries so that this feature is also available for the required users.

“EVALUATING PARTIAL SHAPE QUERIES FOR PATHOLOGY-BASED RETRIEVAL OF VERTEBRA (IEEE 2004)”

In most bio-medical image analysis or retrieval we need not focus on the outline of the entire image. We need to analyze the specific parts which are of interest to us or those which are frequently examined to ascertain a disease. So only the features of those parts of the image are to be used for retrieval unlike other image retrieval systems where whole image is taken into consideration. In our spine x-ray image processing we need to concentrate on the interior vertebrae portions and not on the entire shape as it will give the entire vertebrae shape and will not be useful for analysis.

“A SPINE X-RAY IMAGE RETRIEVAL SYSTEM USING PARTIAL SHAPE MATCHING(IEEE 2008)”

In recent years, there has been a rapid increase in the size and number of medical image collections. Thus, the development of appropriate methods for medical information retrieval is especially important. In a large collection of spine X-ray images, maintained by the National Library of Medicine in USA, vertebral boundary shape has been determined to be relevant to pathology of interest. This paper presents an innovative partial shape matching (PSM) technique using dynamic programming (DP) for the retrieval of spine X-ray images. The improved version of this technique called corner-guided DP is introduced. It uses nine landmark boundary points for DP search and improves matching speed by approximately 10 times compared to traditional DP. The retrieval accuracy and processing speed of the retrieval system based on the new corner-guided PSM method have also been evaluated.



---

## Chapter III

### METHODOLOGY

---

#### 3.1 PREPROCESSING OF X-RAY IMAGES

##### 3.1.1 IMAGE ENHANCEMENT

The spine X-ray images that we take as input are scanned copies of the original X-ray film. The contrast and other features are not visible clearly and are not suitable for processing. So the contrast enhancement is done using ImageJ a public domain Java image processing program.



(a)



(b)

Figure 3.1. (a) original image and (b)enhanced image

The X-Ray image is now blurred using the blur filter.

To compute the filter coefficients, h:

1. Construct an ideal line segment with the desired length and angle, centered at the center coefficient of h.
2. For each coefficient location (i,j), compute the nearest distance between that location and the ideal line segment.
3.  $h = \max(1 - \text{nearest\_distance}, 0)$
4. Normalize h using the formula  $h = h / (\text{sum}(h(:)))$

The filter is created such that once convolved with an image, the linear motion of a camera by len pixels, with an angle of theta degrees in a counterclockwise direction is applied to the image. The coding for the filter is shown below:

```
LEN = 50;  
THETA = 15;  
PSF = fspecial('motion',LEN,THETA);  
bluring= imfilter(img1,PSF,'circular','conv');
```

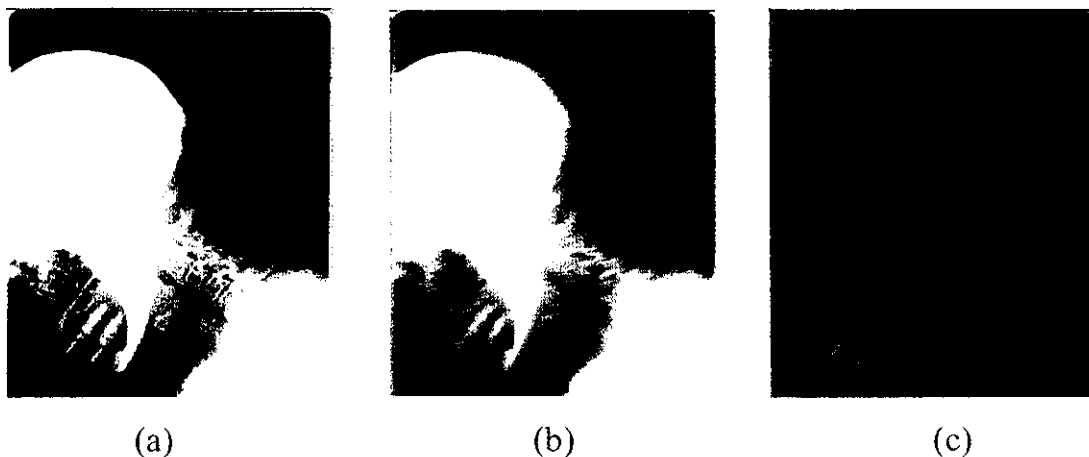


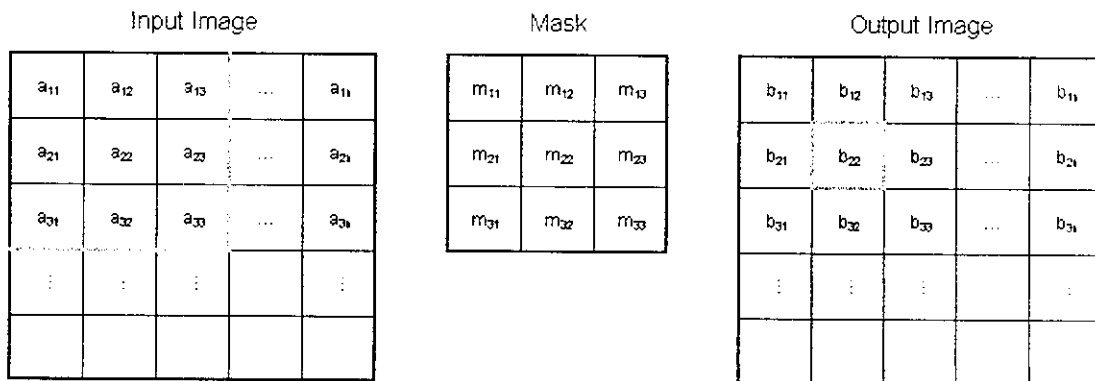
Figure 3.2. (a)Original Image (b)Blurred Image (c)Image obtained by subtracting original and blurred image

### **3.1.2 AREA OF INTEREST SELECTION AND EDGE DETECTION**

The area of interest can be selected from the subtracted image using a rectangle which can be extended or shrunk in all directions. So placement of the rectangle after altering its size is done on the interested vertebrae to be analyzed. Two reference points need to be selected one at the bottom of the vertebrae where the navigation starts and on the top of the vertebrae where the navigation ends. The number of vertebrae between the reference points is specified. The area of interest rectangle navigates on its own one vertebra by vertebra by calculating the average x and y displacements to be done to the position of the area of interest rectangle.

Edge detection is a terminology in image processing and computer vision, particularly in the areas of feature detection and feature extraction, to refer to algorithms which aim at identifying points in a digital image at which the image brightness changes sharply or more formally has discontinuities. Sobel operator has been used in the project for edge detection. The operator calculates the gradient of the image intensity at each point, giving the direction of the largest possible increase from light to dark and the rate of change in that direction. The result therefore shows how "abruptly" or "smoothly" the image changes at that point and therefore how likely it is that that part of the image represents an edge, as well as how that edge is likely to be oriented. Mathematically, the operator uses two  $3 \times 3$  kernels which are convolved with the original image to calculate approximations of the derivatives - one for horizontal changes, and one for vertical.

The mask is slid over an area of the input image, changes that pixel's value and then shifts one pixel to the right and continues to the right until it reaches the end of a row. It then starts at the beginning of the next row. The example below shows the mask being slid over the top left portion of the input image represented by the green outline. The formula shows how a particular pixel in the output image would be calculated. The center of the mask is placed over the pixel you are manipulating in the image. And the I & J values are used to move the file pointer so you can multiply, for example, pixel (a22) by the corresponding mask value (m22). It is important to notice that pixels in the first and last rows, as well as the first and last columns cannot be manipulated by a 3x3 mask. This is because when placing the center of the mask over a pixel in the first row (for example), the mask will be outside the image boundaries.



$$b_{22} = (a_{11} * m_{11}) + (a_{12} * m_{12}) + (a_{13} * m_{13}) + (a_{21} * m_{21}) + (a_{22} * m_{22}) + (a_{23} * m_{23}) + (a_{31} * m_{31}) + (a_{32} * m_{32}) + (a_{33} * m_{33})$$

Figure 3.3. Example to explain Sobel Operator

The output is the edge matrix A.

### 3.1.3 EDGE IMAGE ENHANCEMENT

The output of the edge detection is given as input here. The output is analyzed for pixels having 1's and the corresponding intensity values of the grayscale image are summed up and their average is found. A new black and white image is formed by putting 1's for the pixels in the original image (area of interest image) with values greater than average intensity and 0's otherwise. Then morphological operations like erosion, dilation, and thickening are performed. The closed areas in the image are identified. The numbers of pixels for those closed areas are found separately and if its less than the threshold the closed area is removed. The morphological operations are repeated to get the vertebra in the area of interest.

Preprocessing can also be represented by the following figure:

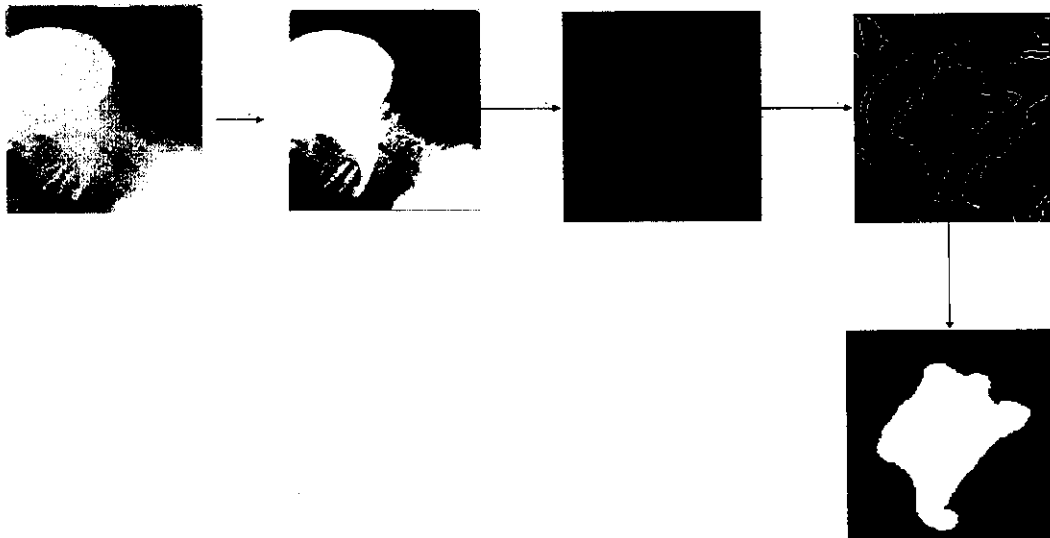


Figure 3.4 Overview of Preprocessing of X-Ray image

Algorithm for preprocessing is given below:

Step 1: Select an image from the image database and perform image enhancement.

Step 2: Select the area of interest and find the edges using Sobel operator.

Step 3: Find the average intensity for the 1's in the edge matrix.

Step 4: Form a new matrix(A) by comparing intensity of the pixel values( $x_{ij}$ ) in the original image and average intensity (y)with the following rule:

$$A[i,j]=1 \quad \text{if } x_{ij} \geq y$$

$$A[i,j]=0 \quad \text{Otherwise}$$

Step 5: Perform morphological operations like dilation, erosion, thicken, close, clean etc.

Step 6: Find the number of pixels( $z_i$ ) in each closed area where i is the number of closed areas.

Step 7: If  $z_i < \text{threshold}$  remove that closed area from the image.

Step 8: Until the gaps are filled continue step 5 and obtain the image of the vertebra in the area of interest.

## **3.2 FEATURE VECTOR EXTRACTION**

### **3.2.1 THE SHAPE DESCRIPTION METHOD:**

The proposed approach for spine X-ray image retrieval takes as input a query image from the user and does some processing on the query to extract the shape of the region of interest and represents it in a form invariant to translation, scaling and rotation. The vertebra image is first reduced to a simpler form by using the Discrete Curve Evolution method. This algorithm reduces the number of vertices of the query image, so that the shape of the region of interest is approximated to the extent that is set according to the level of accuracy required. This simplified curve is converted into another form known as Tangent Space Representation which is nothing but a graph, in which the normalized length of each line segment is represented along the x-axis and the turn angle between any two pair of segments by the y-axis values. A feature vector is formed from the length and turn angle values which is compared with the feature vector representations of the images stored in the database. If the difference between the two is below a set threshold value, then the image is retrieved from the database as a result.

The region of interest is differentiated from the background of the query image by using an algorithm that extracts the region of interest starting from the leftmost pixel having a value of 255.

Starting from the leftmost pixel, the possible vertices of the shape are identified. A pixel is a vertex of the shape if the direction of the next one changes from the current course of traversal. Thus all the vertices of the

shape are identified. Three pairs of vertices are taken from the start and the length of the two line segments ( $s_1, s_2$ ) joining them and is found and their slopes ( $m_1$  &  $m_2$ ) are also found. Then the angle between the line segments is also found using the formula.

$$A = (m_1 - m_2) / (1 + m_1 \cdot m_2)$$

$$\beta = \text{atand}(A)$$

### 3.2.2 DISCRETE CURVE EVOLUTION:

This technique reduces the set of vertices of a polygon to a subset of vertices, which contain relevant information about the original outline. In order to reduce the number of vertices, it is necessary to assign a relevance measure to every vertex, so that the least important vertex is removed. Once a vertex is removed, its neighboring vertices must be connected. This process is repeated until we obtain the desired shape simplification. Figure 1 shows the results of this technique.

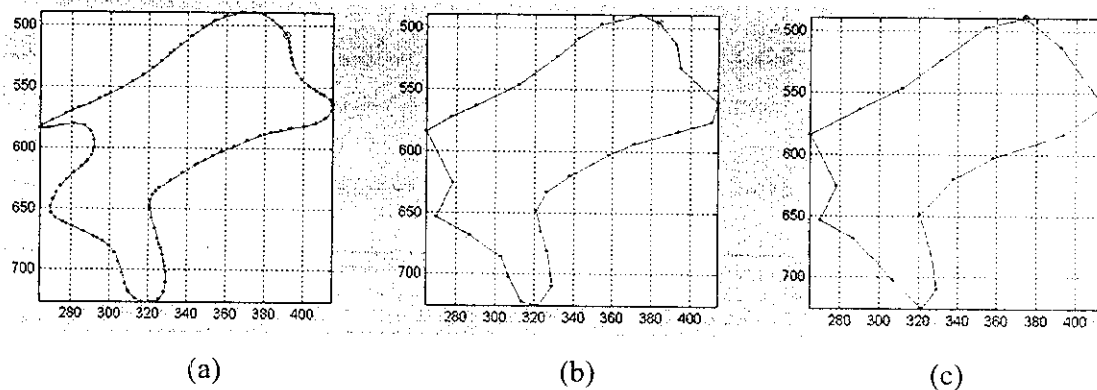


Figure 3.5. (a)Original image (b)&(c)Image obtained in the curve evolution process with 30, 20 vertices respectively.



The order of the vertex substitution is given by the relevance measure  $K$ , where  $\beta(s_1, s_2)$  is the turn angle at the common vertex of the segments  $s_1, s_2$ , and  $l$  is the length function normalized with respect to the total length of the polygonal curve  $C$ . The lower the value of  $K(s_1, s_2)$ , the less the contribution to the shape of the curve of arc  $s_1 \cup s_2$ . To stop the evolution it is necessary that a parameter defines the number of iterations.

$$K(s_1, s_2) = \frac{\beta(s_1, s_2)l(s_1)l(s_2)}{l(s_1) + l(s_2)}$$

### **3.2.3 TANGENT SPACE REPRESENTATION:**

Since, the polygonal representation is not a convenient form for calculating the similarity between two shapes, an alternative representation such as TSR, is needed. TSR stands for Tangent Space Representation. Using TSR a curve  $C$  is represented by the graph of a step function, the steps on  $x$ -axis represents the arc length of each segment normalized with respect to the total length of  $C$ , and the  $y$ -axis represents the turn angle between two consecutive segments in  $C$ . Tangent representation is invariant to rotation, scaling and translation. An example for a tangent space representation is given below.

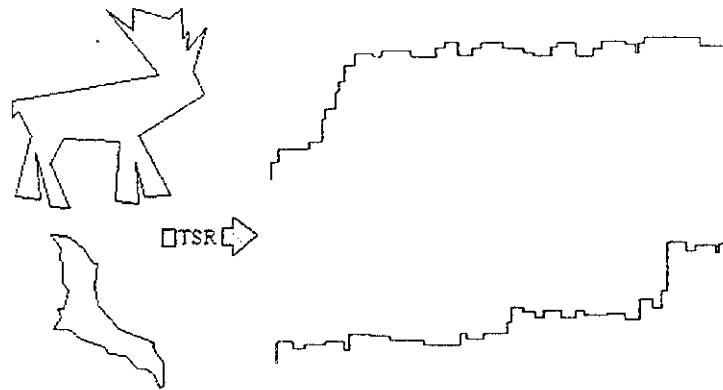


Figure 3.6 TSR

Each segment in the graph is called a step, the directional angle of the line segments is represented by the y-value of the step, and the length of each line segment normalized with respect to the length of the curve is represented by the x-value of the steps. This way the resultant TSR is invariant to scaling, rotation and translation.

The algorithm for feature vector extraction is shown below:

Step 1: Form a new matrix B with the following rules

$$B [i, j] = 255 \text{ if } A [i, j] = 1$$

$$B [I, j] = 0 \quad \text{Otherwise}$$

Step 2: Find the leftmost pixel (sx, sy) whose value is 255.

Step 3: Find the directionality (d) of the pixel.

Step 4: Find the position of the next pixel(x, y) in the found direction.

Step 5: Repeat step 3 for the found pixel.

Step 6: If d is the same goto step 4

$$\text{Else } B[x, y] = 127$$

Step 7: Continue this till the starting point (sx, sy) is reached.

Step 8: Now select first three pairs of vertices and find the length line joining the two pairs of vertices (s1&s2) and their slopes (m1&m2).

Step 9: Find the angle between them using the formula

$$A = (m_1 - m_2) / (1 + m_1.m_2)$$

$$\beta = \text{atand}(A)$$

Step 10: Find the relative measure.

$$K(s_1, s_2) = \frac{\beta(s_1, s_2)l(s_1)l(s_2)}{l(s_1) + l(s_2)}$$

Where  $\beta(s_1, s_2)$  is the angle between  $s_1$  and  $s_2$

$l(s_1)$  &  $l(s_2)$  are the lengths of  $s_1$  and  $s_2$ .

Step 11: Continue till all vertices are processed.

Step 12: Select the points with the highest relative measure (K) based on the number of vertices needed and form tangent space representation.

Step 13: Convert the tangent space representation to feature vector in the format  $y_{pos}$ ,  $x_{pos}$ , length, angle, relative measure and save it to the database.

Step 14: Repeat for each image in the database.

The feature vector algorithm can be represented by the following figure

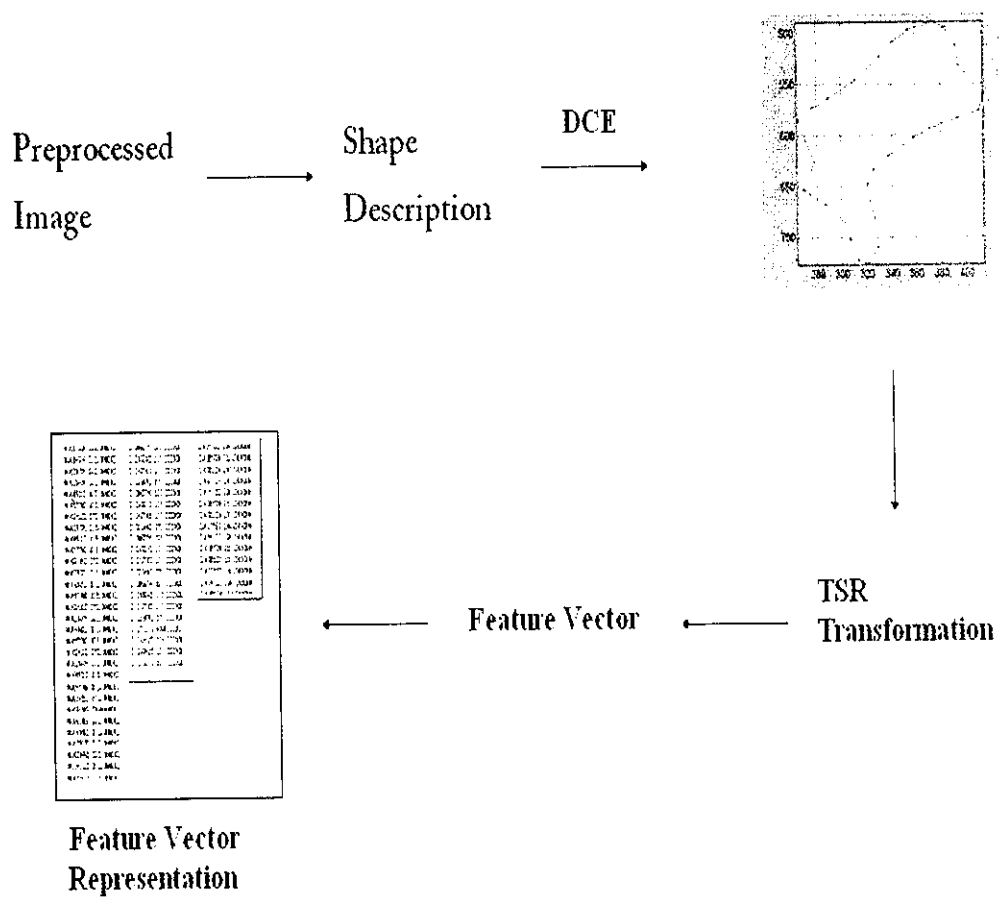


Figure 3.6 Feature Vector Extraction

### 3.3 IMAGE QUERYING AND RETRIEVAL:

In the image querying phase, the user supplies a spine X-ray which is to be retrieved. The user has to select the area of interest. The software then extracts the vertebra of interest using preprocessing. The software applies the discrete curve evolution method to the vertebra image and reduces the number of vertices to 50 and the tangent space representations for the above are formed. The feature vectors for each tangent space representation are formed. These feature vectors are compared with those that are stored in the database and the Euclidean distance between each pair of feature vectors is computed based on angle and length. If the distance is less than a preset threshold value, then the corresponding image is included in the list of images retrieved as results. This is shown in Figure 4.2.

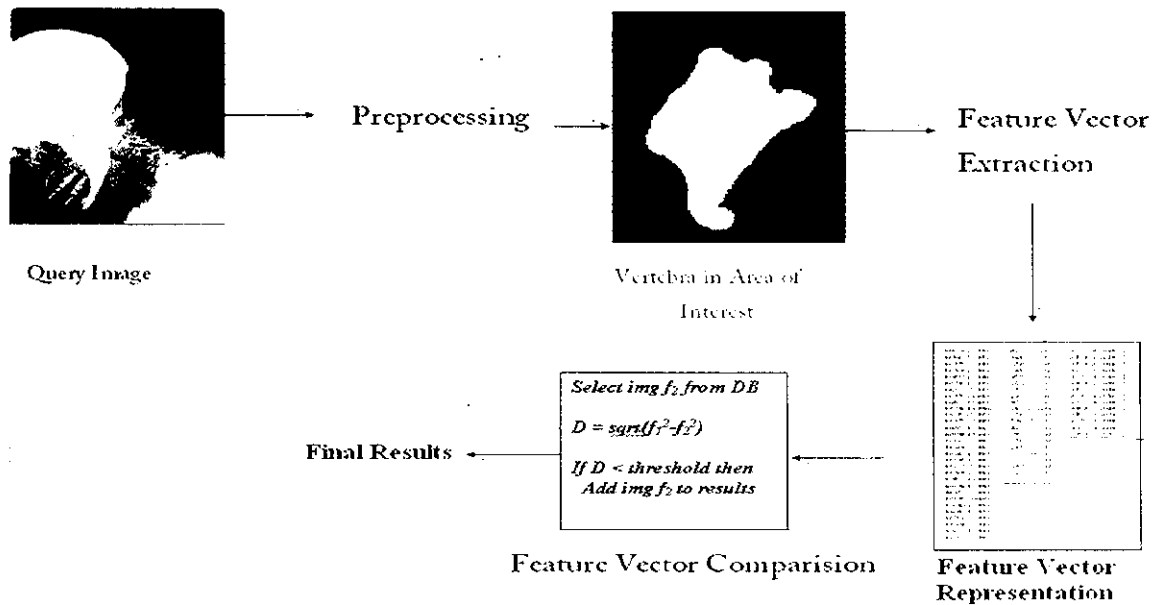


Figure 3.7 Image Querying and Retrieval

The algorithm for image retrieval and querying is represented below

Step 1: Get an X-ray as input.

Step 2: Perform preprocessing.

Step 3: Perform feature vector extraction and obtain the feature vector (fv1).

Step 4: Get the feature vector of an image stored in the database (fv2).

Step 5: Find the cumulative Euclidean distance between lengths and angles in fv1 and fv2 using the following method.

Repeat for each i (i varies till number of feature vectors in fv1)

```
length_minus = fv1 (i, 3)-fv2 (i, 3)
angle_minus = fv1 (i, 4)-fv2 (i, 4)
euclidean_length = euclidean_length +
    (length_minus*length_minus);
euclidean_angle = euclidean_angle +
    (angle_minus* angle_minus);
```

End

```
euclidean_length = sqrt (euclidean_length);
euclidean_angle = sqrt (euclidean_angle);
```

Step 5: If euclidean \_length and euclidean \_angle < threshold then

Add the image to related images list

Step 6: Repeat steps 2 to 5 for all images in the database.



---

## Chapter IV

### SYSTEM DESIGN

---

The system is composed of the following important parts.

Preprocessing

Vertices Retrieval

Feature Vector Extraction

Reduction of Vertices\_DCE

Relevance Measure Computation

Elimination of Vertices

Conversion\_TSR

Add Image

Find Related Images

View Records

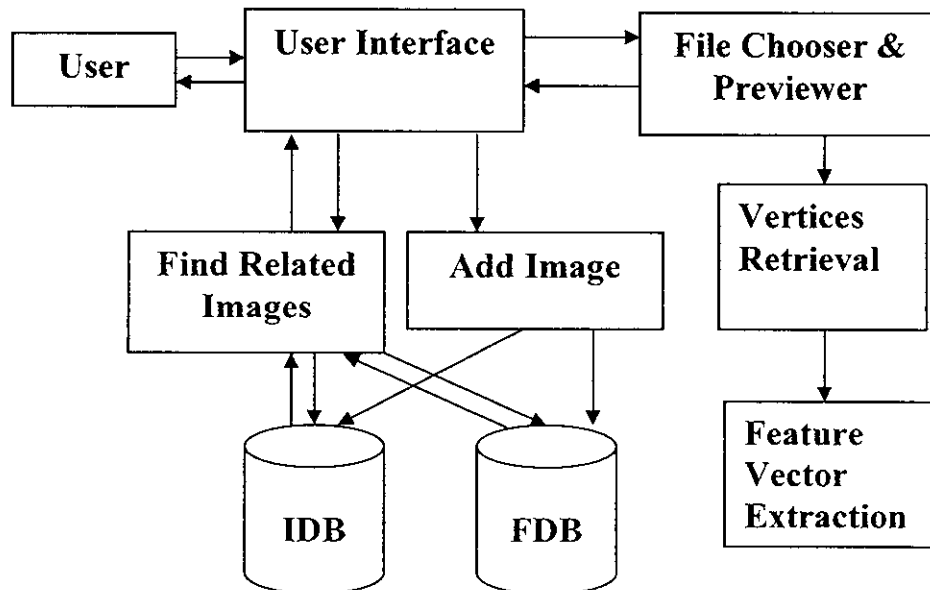


Figure 4.1. Communication between various elements of the system

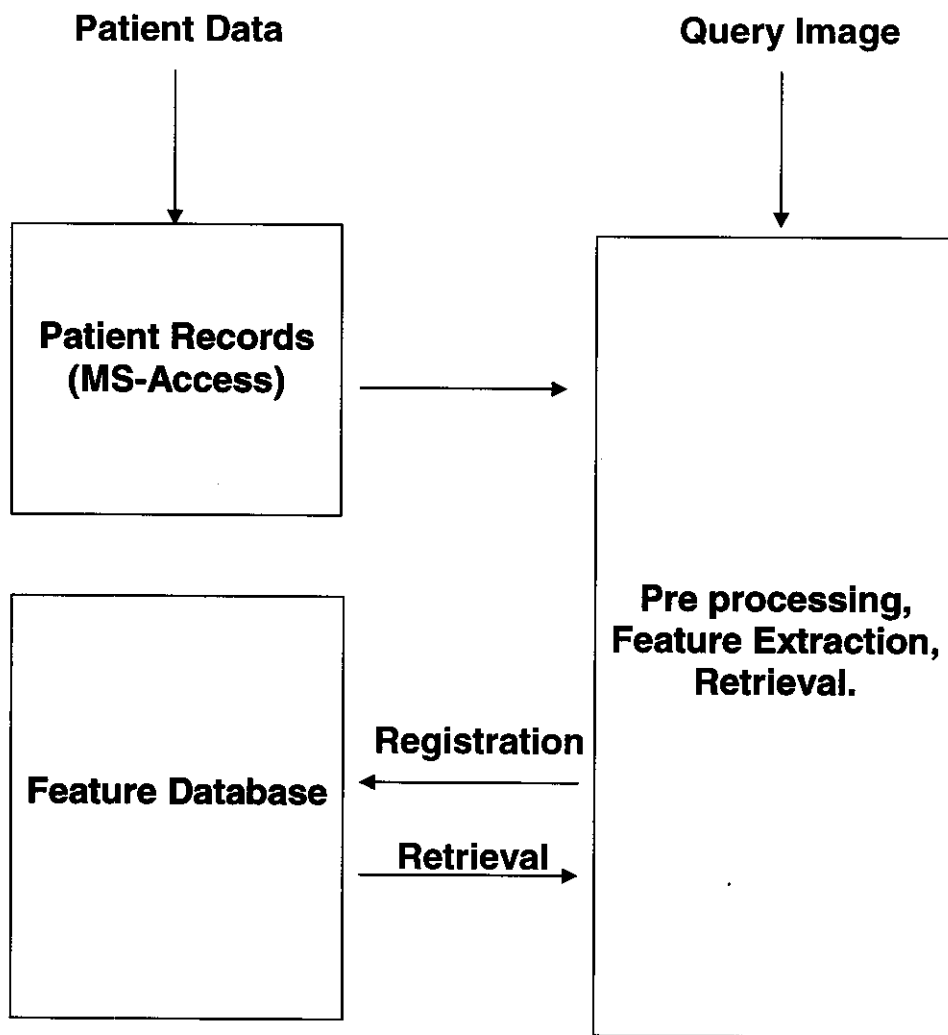


Figure 4.2. Overall Retrieval system

## 4.1 Preprocessing

This module gets the input X-ray image and asks the user to select the area of interest. It applies sobel operator to the image and obtains the edges. Further manipulations are done and then operations like erosion, dilation, thicken are used. The resulting image is filtered for large sections greater than a threshold and then again some morphological operations are performed and then the vertebra image in the are of interest is obtained.

## 4.2 Vertices Retrieval

This module segregates the object present in the input image from its background. It processes the image by probing for points which contribute to the outer shape of the object .It traverses the image from left to right column wise and finds the left most point (seed point) which contributes to the outer shape of the object. From the seed point it looks at all the eight directions and finds the immediate neighbor which contributes to the shape. The above process is repeated until the seed point is reached again, forming a closed curve.

The members within the method are given below

Member	Purpose
Image_x, image_y	Contains the x and y coordinates of the current pixel.
Image_size	Contains the number of rows and columns present in the image.
Image_gray	A matrix to size of the image contains the

	value 255 if the corresponding pixel in the image contributes to the object otherwise 0.
direction_matrix	Contains 8 values to traverse all the neighboring 8 pixels.
I_x_start, i_y_start	Contains the x and y coordinates of the vertex that has been first found.
I_x_new, i_y_new	Contains the x and y coordinates of newly found point which contributes to the outer shape of the object.
Image_prev_x, image_prev_y	Contains the x and y coordinates of previously found point.
edge_mat	Contains the x and y coordinates of the vertex, length, angle and relevance measure.

Table 5.1

### 4.3 Feature Vector Extraction

In this module, the feature vector which represents each image is extracted from the image. To do this, first the image is processed using Discrete Curve Evolution method, which simplifies the image by reducing the number of vertices it contains. Then the feature vector is extracted and the image is represented using Tangent Space Representation.

## Reduction of Vertices\_DCE

This process can be seen to be performing two functions. It computes the relevance measure  $K$  of a vertex to the whole shape of the object and based on its value eliminates the vertices which are least relevant to the object's shape.

### Relevance Measure Computation

The relevance measure  $K$  of a vertex is given by the formula,

$$K(s_1, s_2) = \frac{\beta(s_1, s_2)l(s_1)l(s_2)}{l(s_1) + l(s_2)}$$

The algorithm computes the length of a segment as it traverses that segment. After computing the lengths of two adjacent segments, it normalizes their combined length with respect to the total length of the object. This lets the algorithm be invariant to scaling since the ratio of lengths can be safely assumed to be almost equal even when the object is scaled to a different size.

The turn angle between two segments is computed using the formula,

$$\tan \beta = (m_1 - m_2)/(1 + m_1 \cdot m_2)$$

where,  $m_1$  and  $m_2$  are the slopes of segments  $s_1$  and  $s_2$ . When one of the segments is vertical then the above formula leads to an indeterminate form. In such cases, the slope of one segment is computed using the inverse tangent function separately and the value of  $\beta$  is computed by

adding or subtracting to the acquired angle according to the direction of the adjacent segment.

The higher the turn angle between two segments, the more relevant to the shape is the vertex that is formed at the junction of the two segments. The length of a segment and the turn angle it forms with the previous segment are taken to be the components of the feature vector which is stored along with the images in the database.

Some of the built-in functions that are used are

atand() – used to compute the inverse of tangent function.

abs() – used to find the absolute value .

### **Elimination of Vertices**

After computing the relevance measure for each of the vertices in the object, its value is compared with a predetermined threshold value. If the value of K is less than the threshold, then the corresponding vertex is discarded and the neighboring vertices are joined together.

The members within the method are given below

<b>Method</b>	<b>Purpose</b>
vertices_limit	Contains values 50, 30 and 17
vertices_deleted_count	Contains the count of number of vertices deleted
edge_mat_check	Contains the number of vertices present at that time
rel_mes_small	Contains the lowest relevance measure value

rel_mes_small_index	Contains the index of vertex that has lowest relevance measure value
X1,y1,x2,y2,x3,y3	Contains the x and y coordinates of consecutive three vertices in the array
M1,m2	Contains the slope of segment formed by the vertex 1 &2 and 2 &3
Total	Contains the sum of the length of segment present in the array
rel_mes_high	Contains the highest relevance measure value
rel_mes_high_index	Contains the index of vertex that has highest relevance measure value

Table 5.2

### Conversion\_TSR

This module creates a table in the format shown below.

**Length                      Angle**

where,

Length – the length of each segment.

Angle – the turn angle between two consequent segments.

The 50 vertices set are converted to the above specified format. The Starting vertex is the one which has the largest relevance measure amongst its respective set. The length is calculated from the formula,

$$\text{Length} = \text{sqrt}((x_2-x_1)^2 + (y_2-y_1)^2)$$

The members within the method are given below

<b>Member</b>	<b>Purpose</b>
Feature_vector	Contains only the length and angle of edge_mat
Feature_count	Contains the number of entries present in the feature_vector
Feature_show_50	Contains the length and angle of 50 vertices set
Tangent_space_50	Contains the TSR for 50 vertices set

Table 5.3

### **Add Image**

This module when activated adds the query image and its feature vector to their respective databases. Before adding, it reads the file 'test0.txt' and gets the count of number of files present in the database and compares all the images in the database whether the image has been already added to the database. If so it will not allow the user to add the image. If not it increases the file count value by one and saves it in the same file 'test0.txt'. It saves the image in the file 'test+(filecount)+.bmp' and the feature vector in 'test+(filecount)+.txt'.



The members within the method are given below

<b>Member</b>	<b>Purpose</b>
input_image_path	Contains the path of the query image
input_file	Contains the entries of query image in matrix format
database_file	Contains the entries of image in the database in matrix format
file_count	Contains the number of files present in the database

Table 5.4

### **Find Related Images**

This module compares the feature vector of query image with the feature vectors of each image in the database by retrieving the data from the files 'test+\*.txt'. Initially it finds the index of three vertices, which are having the largest three relevance measure values in each set. Also it finds the above three values for each and every image in the database. Starting from the index of those vertices the angles and length are compared and a difference of some value is accepted. This difference varies from set to set. Depending on the number of vertices, which are similar, the image in the database is related to the query image or not is found out. Totally nine comparisons are made for each and every set. In any one of the comparisons if the number of similar vertices are more than the threshold value then the image is selected for that set.

The members within the method are given below

<b>Member</b>	<b>Purpose</b>
Related_image_50	Contains the path of related images in the 50 vertices set
Related_image_50_count	Contains the number of images related to the query in the 50 vertices set
Related_image_50_discrete	Contains the x and y coordinates of 50 vertices set of images present in the database which are related to the query
Feature_vector1	Contains the feature vector for the image present in the database
eucledian_length	Contains the eucledian length for 50,30 and 17 vertices separately
eucledian_angle	Contains the eucledian angle for 50,30 and 17 vertices separately
largest_rel_50_index	Contains the index of three vertices, which possess the largest three relevance measure value in the 50 vertices set for the query image.
Largest_rel_dbase_50_index	Contains the index of three vertices, which possess the largest three relevance measure value in the 50 vertices set for the image present in the database.

Table 5.5

## **View Records**

In this module, the users are allowed to add patient details like name, age, gender and also select an image to be associated with that file. The system then copies the image file to the image database location and renames it based on the patient id. It also allows us to view the records in the database. The records are displayed along with the associated image.

---

## Chapter V

# PROGRAMMING ENVIRONMENT

---

### 5.1. HARDWARE REQUIREMENTS:

1. Processor : Intel Pentium IV
2. Processor Speed : 1.6 GHz
3. Memory :512MB SDRAM  
(At least 1024 MB recommended)
4. Graphics : 16-, 24-, or 32-bit OpenGL capable  
graphics adapter
5. DirectX : 9.0c (August 2005) or later

### 5.2. SOFTWARE REQUIREMENTS:

1. Operating System : Windows XP Service Pack 2 or 3
2. MATLAB 7.5(MATLAB R2007B)

---

## Chapter VI

### FUTURE ENHANCEMENTS

---

The application has been built with only basic functionalities, with the emphasis given on the working of the algorithm. Extended functionality can be provided to both the User Interface and to the algorithm used itself.

#### 6.1. ENHANCEMENTS TO THE UI:

Some of the features that can be added to the User Interface include:

- Enable the user to view the results in a separate window than that in which the query image is input.
- Enable the user to select the level of accuracy for the retrieval of results.
- Generate reports that provide a graphical representation of the various parameters involved in the retrieval process in the form of graphs, charts etc.,

## **6.2. ENHANCEMENTS TO THE ALGORITHM:**

Some of the enhancements that can be done to the algorithm in future are given below:

The algorithm can be modified in such a way that

- It works for complex images i.e., images which have a complex background such that preprocessing cannot improve the contrast.
- It works well for images which have disjoint vertebrae, so that the foreground can be divided into more than one part.
- The algorithm is highly scalable, since it uses the feature vector approach for representing the images present in the database. Any number of components can be added to the feature vector so that more features of the images are compared. This improves the quality of retrieved images.
- Another approach is to have varying threshold levels so that the user can choose one, according to the level of accuracy needed.

---

## Chapter VII

### CONCLUSION

---

The “Content Based Image Retrieval Of Spine X-rays” method was successfully implemented and evaluated using Matlab 7.5 on Windows XP operating system. The results arrived at are significant. The system can help answer many real-world demands.

Unlike traditional bio-medical information retrieval systems, in which implicit queries might return results of patients, this system always gives the best possible solutions for the selected image query. The design is user-friendly and comprehensive. Some basic amount of training is required and it can be learn with ease. The system is also easily specialized; the basic features of the system can be modified to suit various needs. By including information that is particular to a situation, the system can be made more specific and the efficiency improved.

The database used for the demonstration of the working of this algorithm was chosen carefully, so that the basic functionality of the algorithm is highlighted. Further research can strive to improve the scalability of the algorithm by having more built-in features that are customizable by the user.

Overall, the project was challenging to implement and provided deep insights into the field of image processing.

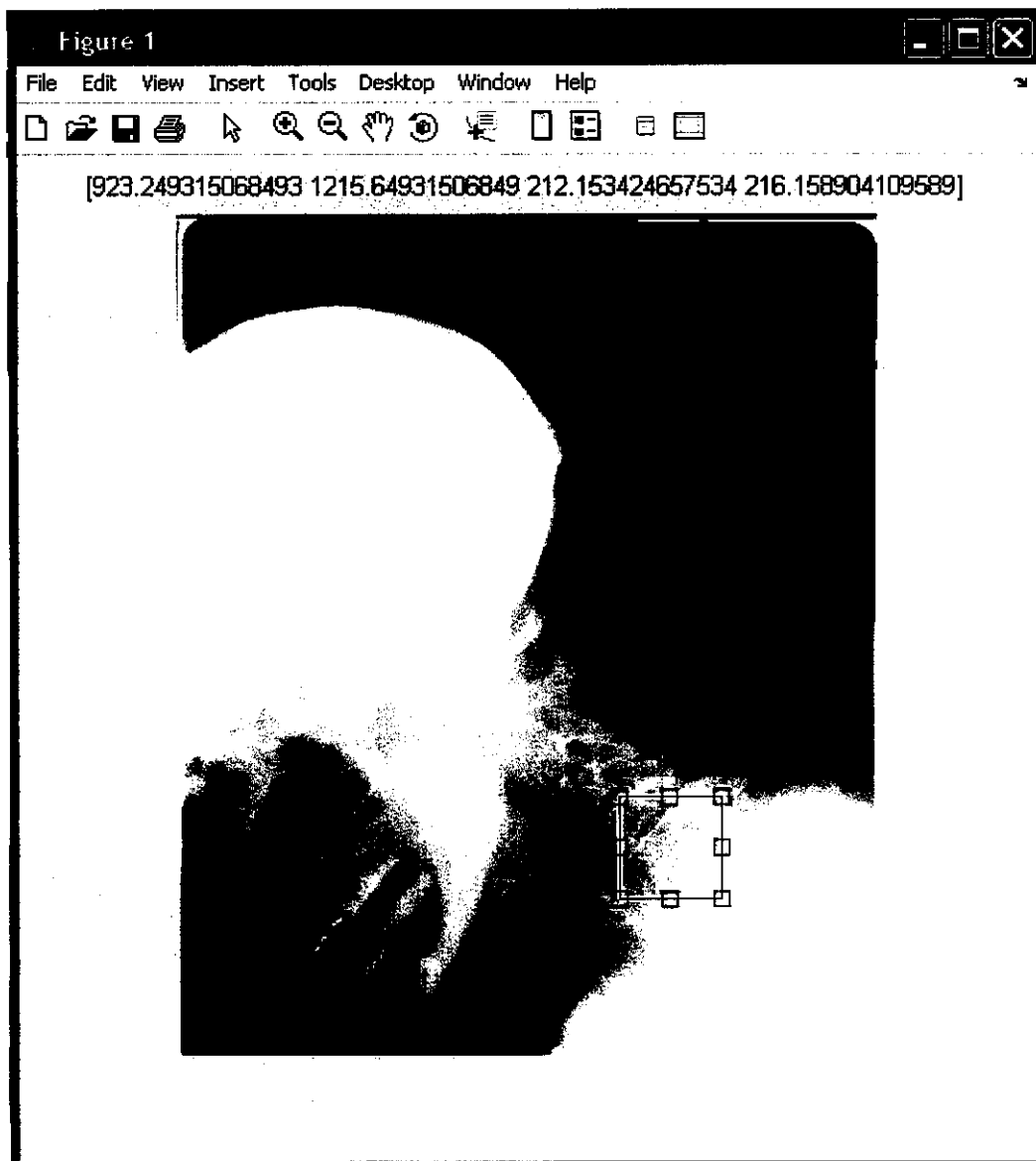
---

# CHAPTER VIII

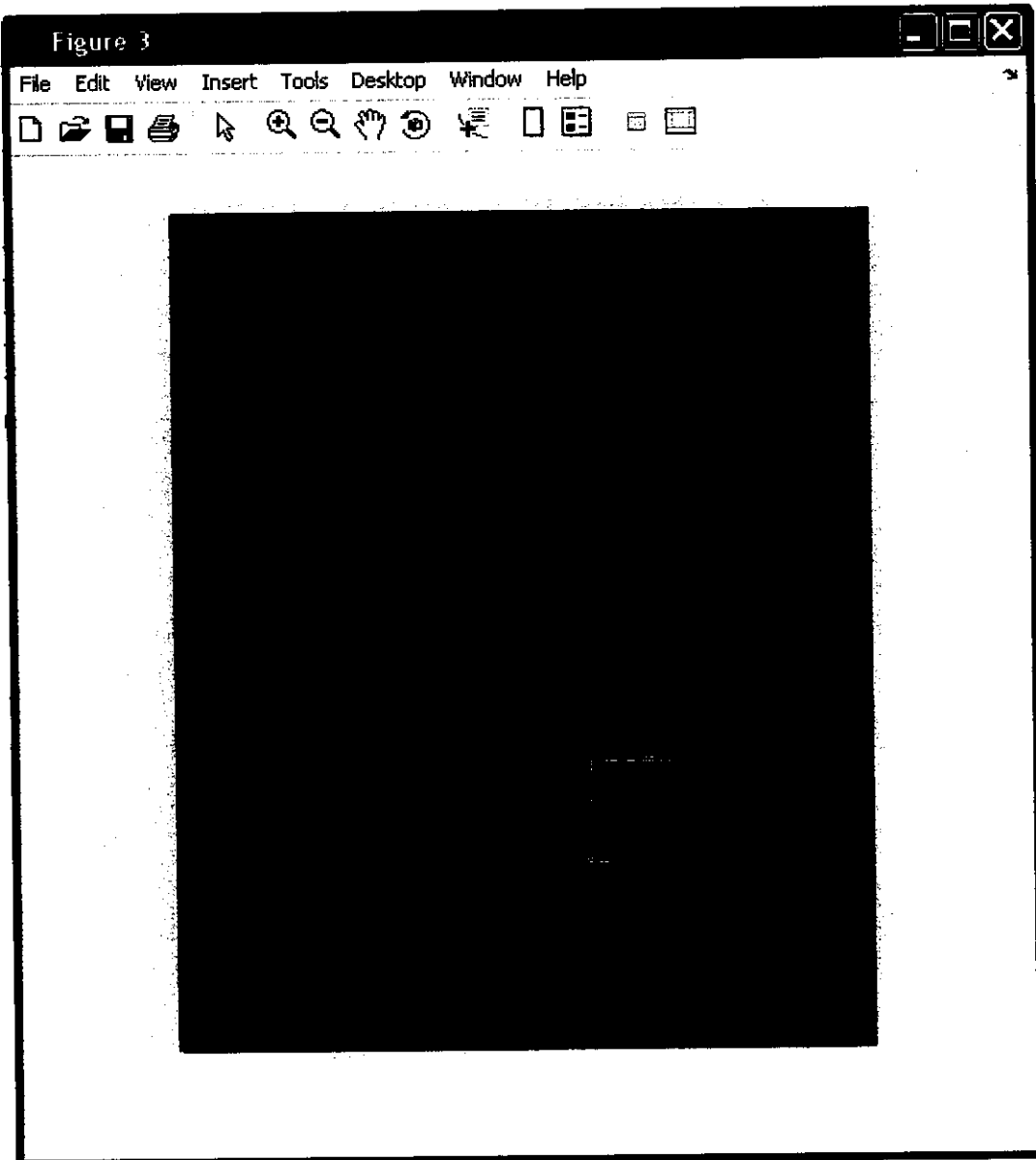
## APPENDIX

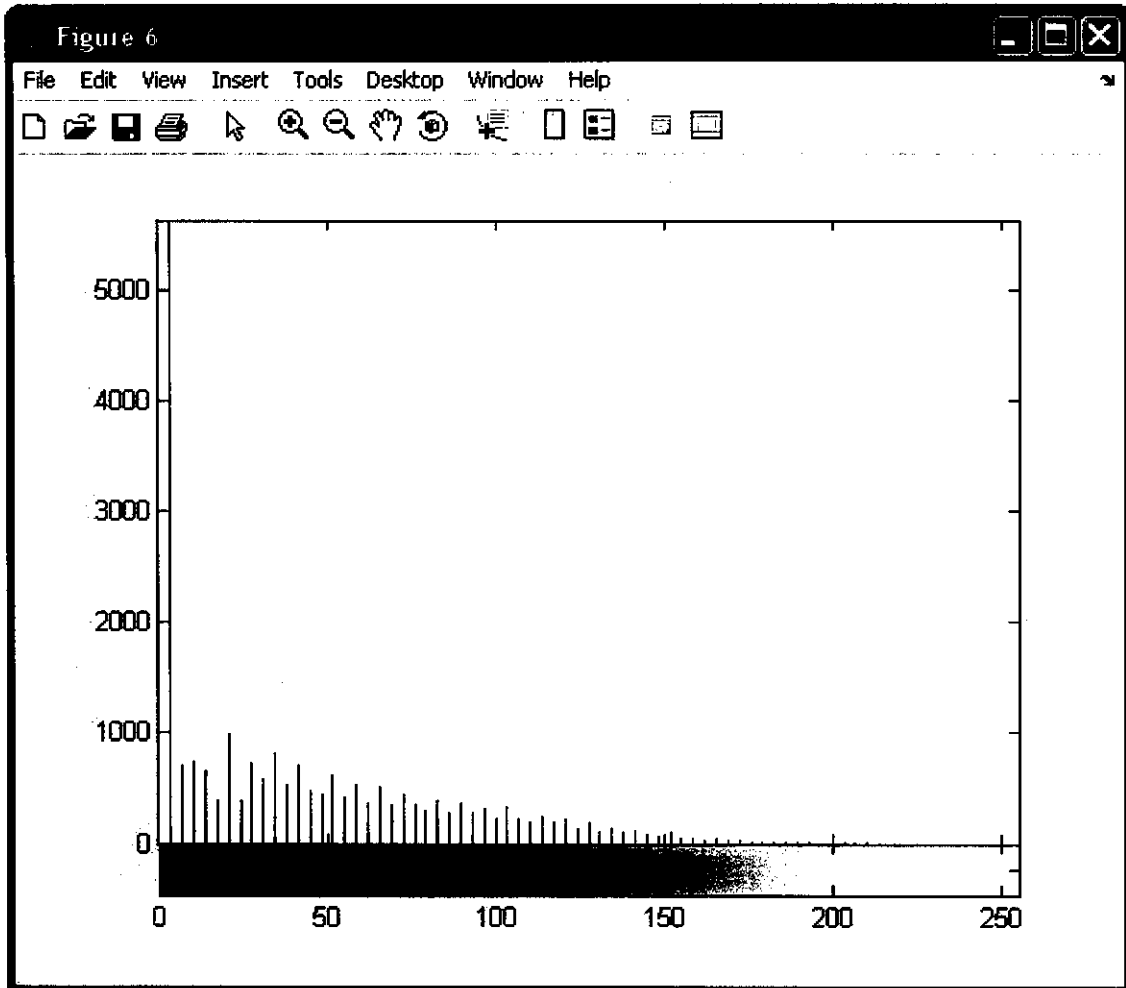
---

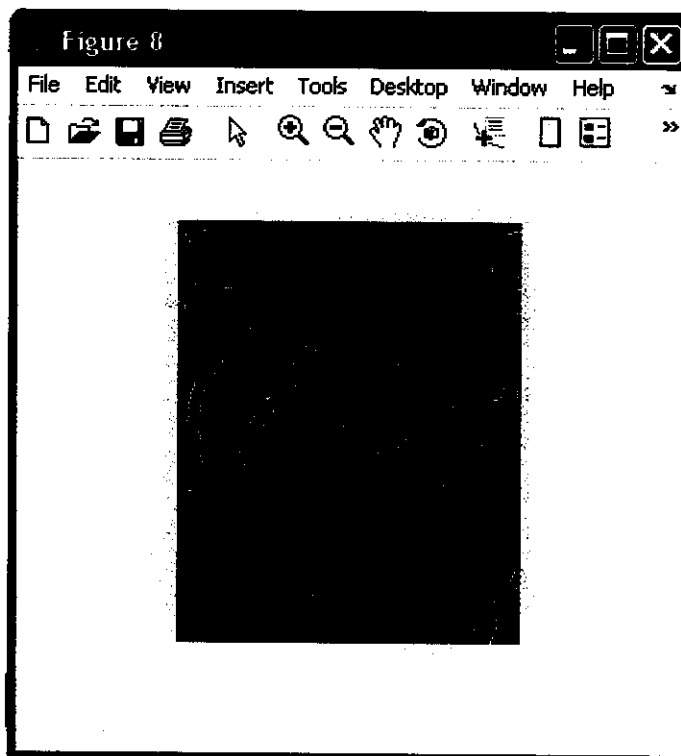
### 8.1 RESULTS:

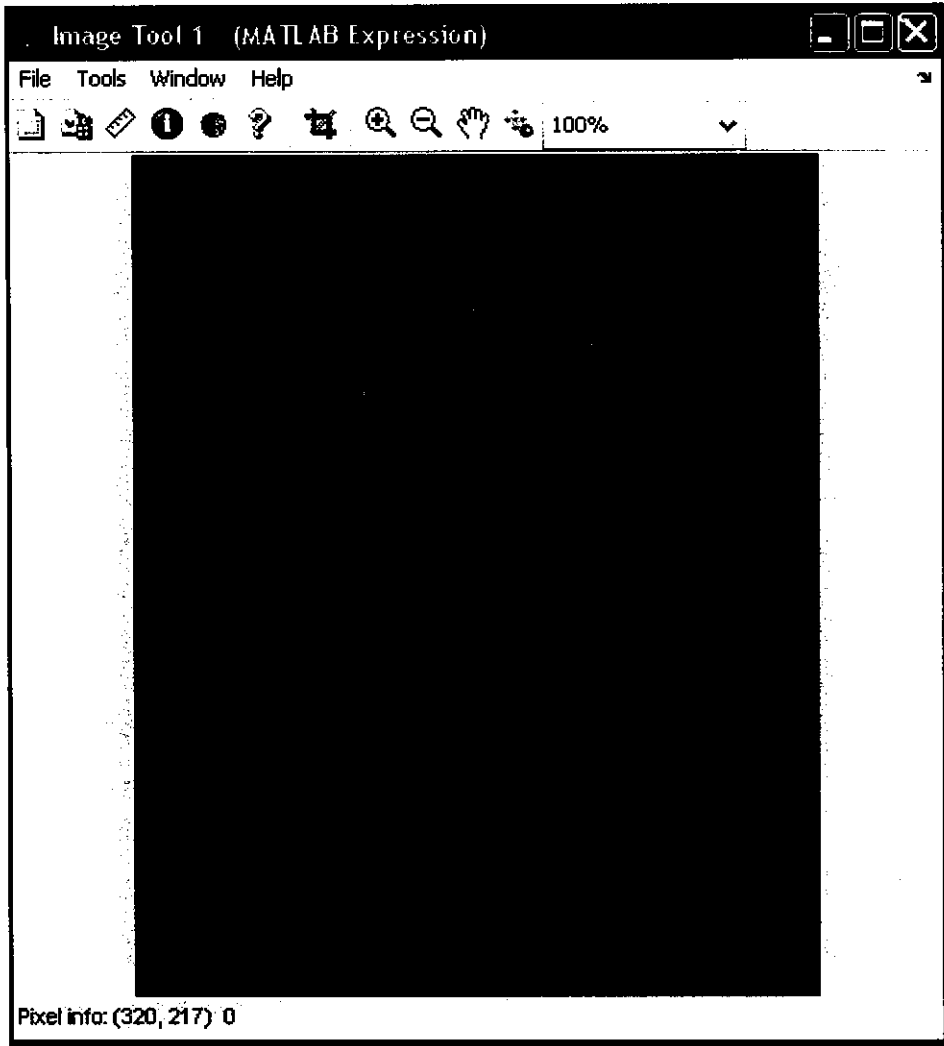


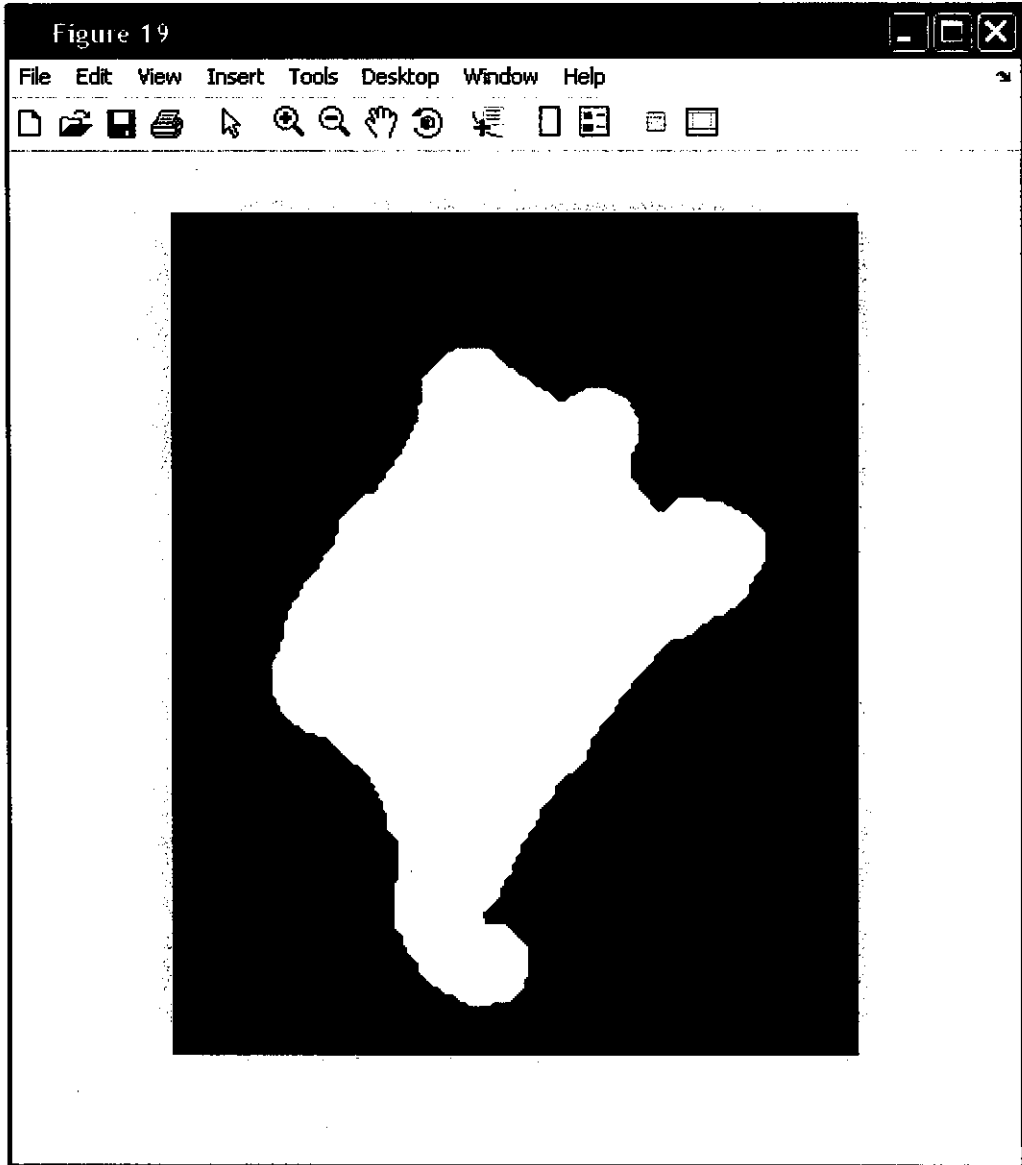


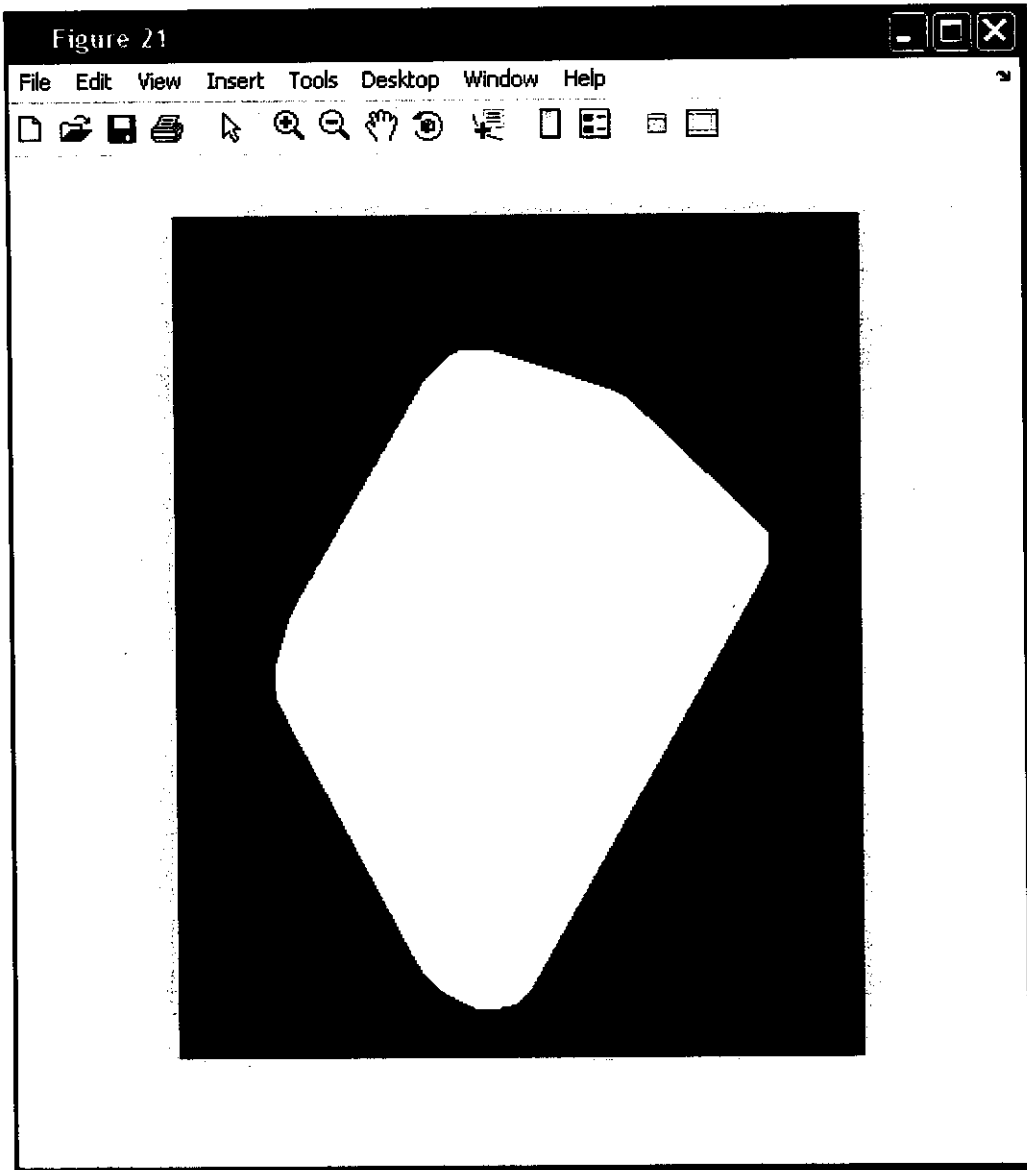


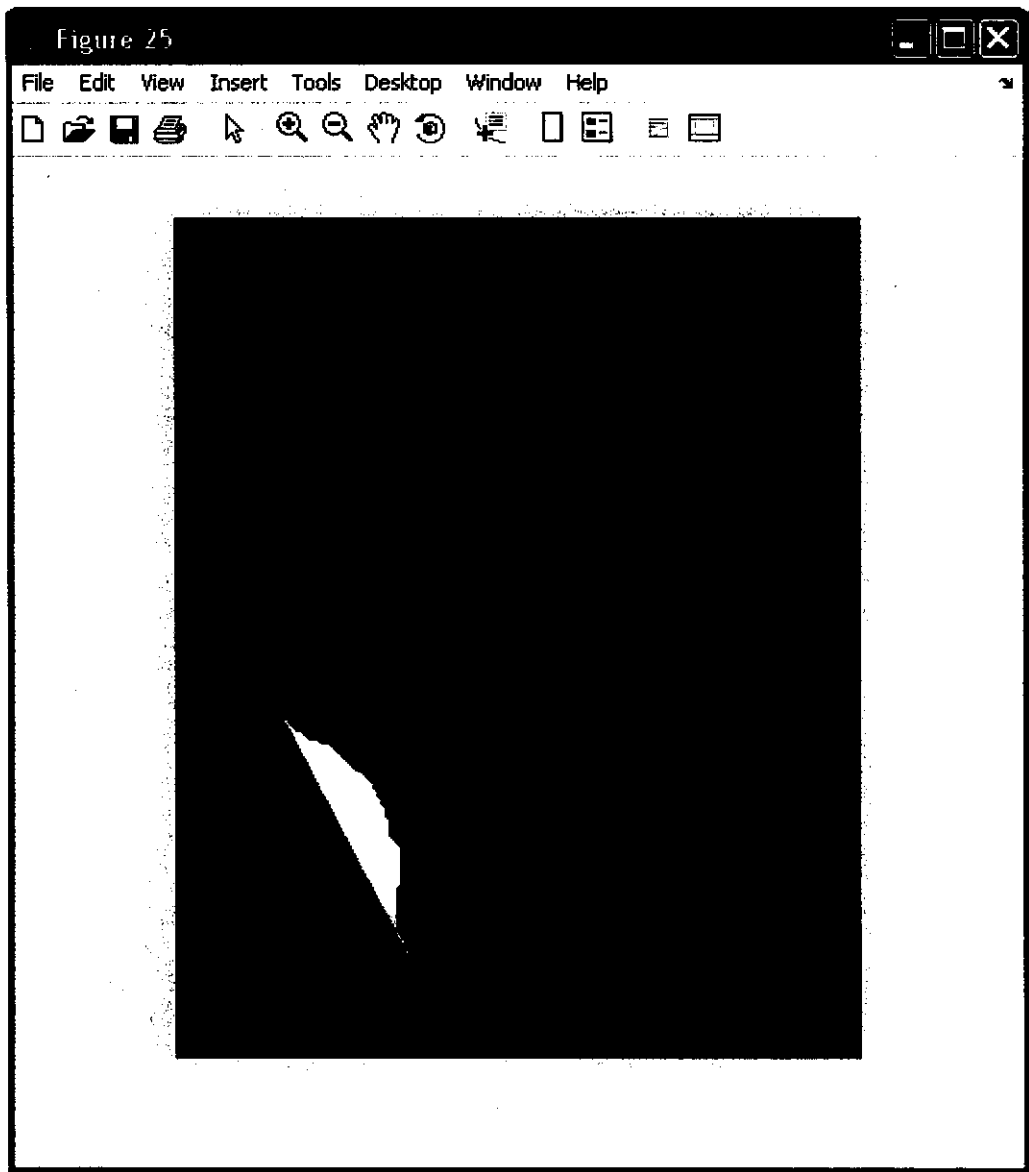


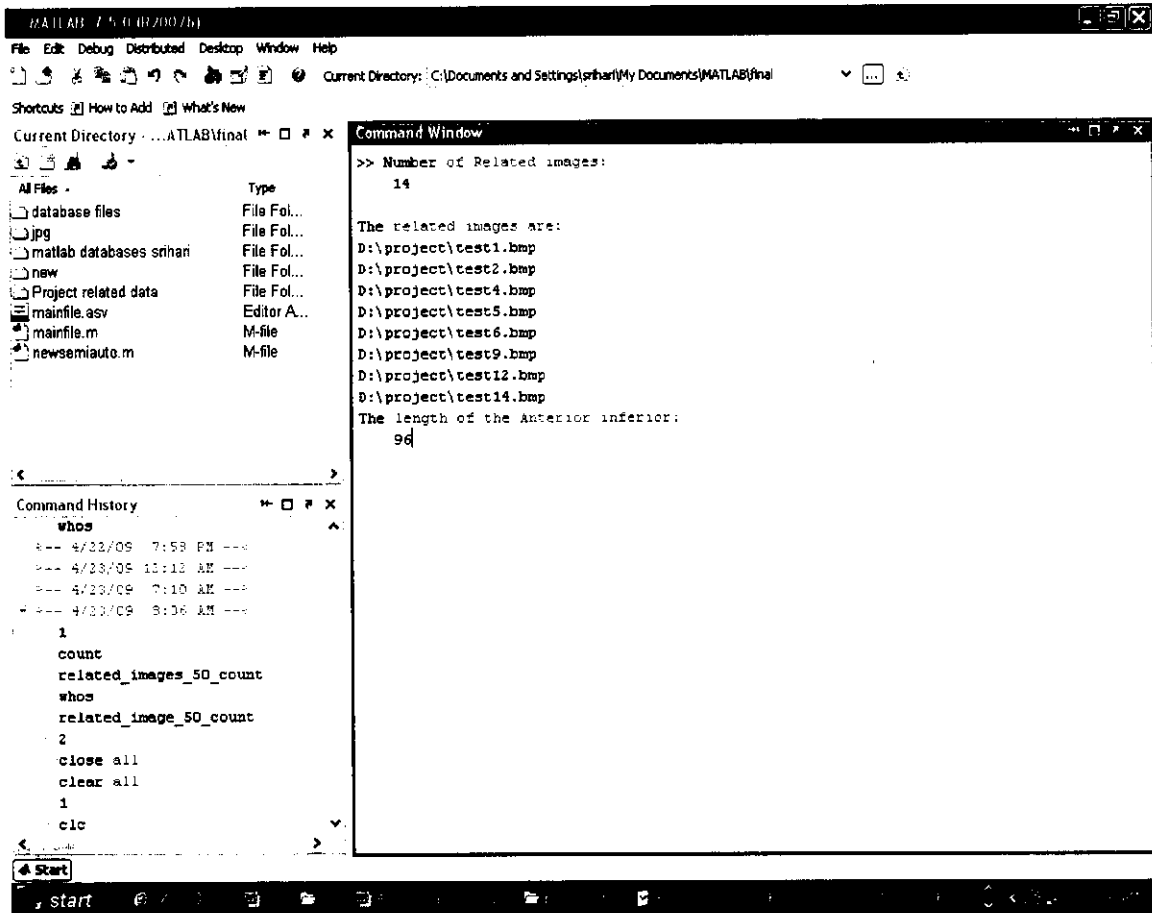














dbconnection

Patient No


Patient Name

Age

Gender

Image Path

Insert




dbconnection

Patient No


Patient Name

Age

Gender

Image Path  

Insert



## 8.2 MATLAB7.5 CODE SAMPLES

### SAMPLE CODE TO RETRIEVE VERTICES

```
// finds the leftmost part of pixel whose value equals to 255
for image_x = 1:image_size(2)
    for image_y = 1:image_size(1)
        if image_gray(image_y,image_x) == 255
            iFlag = 1;
            break
        end
    end
    if iFlag == 1
        break
    end
end
//end of finding
length = 1;
image_prev_x = image_x;
image_prev_y = image_y;
i_x_start = image_x;
i_y_start = image_y;
i_x_new = -1;
i_y_new = -1;
edge_index = 2;
edge_mat = [image_y,image_x,-1];
```

```

//finds the neighbouring pixel and sets the direction
for direction_index = 1 : 8
    if image_gray(image_y+direction_matrix(direction_index,1), image_x +
                    direction_matrix (direction_index,2)) == 255
        direction = direction_index;
        image_y = image_y + direction_matrix(direction_index,1);
        image_x = image_x + direction_matrix(direction_index,2);
        break;
    end
end
// end of the process

```

```

//finds the vertices by comparing the direction index with direction and
//writes it to the edge mat array

```

```

while (i_x_new ~= i_x_start) | (i_y_new ~= i_y_start)
    for direction_index = 1 : 8
        i_y = image_y + direction_matrix (direction_index,1);
        i_x = image_x + direction_matrix (direction_index,2);
        if image_gray(i_y,i_x) == 0
            if direction_index == 8
                d = 1;
            else
                d = direction_index + 1;
            end
            i_y_new = image_y + direction_matrix(d,1);
            i_x_new = image_x + direction_matrix(d,2);
        end
    end
end

```

```

if image_gray(i_y_new,i_x_new) == 255 &
    (image_prev_x ~= i_x_new | image_prev_y ~= i_y_new)
    if direction ~= d
        if direction == 1 | direction == 3 | direction == 5 | direction == 7
            length = length * sqrt(2);
        end
        edge_mat(edge_index,:) = [image_y,image_x,length];
        edge_index = edge_index + 1;
        length = 0;
        direction = d;
    end
    break;
end
end
end
end
image_prev_x = image_x;
image_prev_y = image_y;
image_x = i_x_new ;
image_y = i_y_new ;
length = length + 1;
end

```

## SAMPLE CODE TO EXTRACT FEATURE VECTOR

```
vertices_limit= [50];
vertices_deleted_count = 0;

for i = 1 : 1
    check = vertices_limit(i);
    edge_mat_check = edge_mat_limit;
    while edge_mat_check > check
        edge_mat(rel_mes_small_index,6) = 1;
        vertices_deleted_count = vertices_deleted_count + 1;
        edge_mat_check = edge_mat_check - 1;
        temp = rel_mes_small_index - 1;
        if temp == 0
            temp = edge_mat_limit;
        end
        while(edge_mat(temp,6) == 1)
            temp = temp - 1;
            if temp == 0
                temp = edge_mat_limit;
            end
        end
        y2 = edge_mat(temp,1);
        x2 = edge_mat(temp,2);
        index_2 = temp;
        temp = temp - 1;
        if temp == 0
```

```

temp = edge_mat_limit;
end
while(edge_mat(temp,6) == 1)
    temp = temp -1;
    if temp == 0
        temp = edge_mat_limit;
    end
end
y1 = edge_mat(temp,1);
x1 = edge_mat(temp,2);
temp = rel_mes_small_index + 1;
if temp == edge_mat_limit + 1
    temp = 1;
end
while(edge_mat(temp,6) == 1)
    temp = temp +1;
    if temp == edge_mat_limit + 1
        temp = 1;
    end
end
y3 = edge_mat(temp,1);
x3 = edge_mat(temp,2);
index_3 = temp;
temp = temp + 1;
if temp == edge_mat_limit + 1
    temp = 1;
end

```

```

while(edge_mat(temp,6) == 1)
    temp = temp +1;
    if temp == edge_mat_limit + 1
        temp = 1;
    end
end
end
y4 = edge_mat(temp,1);
x4 = edge_mat(temp,2);
y_diff = y3 - y2;
x_diff = x3 - x2;
length = sqrt( (y_diff*y_diff)+(x_diff*x_diff) );
edge_mat(index_3,3) = length;
length1 = edge_mat(index_2,3);
length2 = edge_mat(index_3,3);
length3 = edge_mat(temp,3);

..... FIND ANGLE .....

rel_mes = angle * length1 * length2 / (length1 + length2);
edge_mat(index_2,5) = rel_mes;
index_2 = index_3;
x1 = x2 ;y1 = y2 ; x2 = x3 ; y2 = y3 ; x3 = x4 ; y3 = y4;
length1 = length2;
length2 = length3;
end
rel_mes_small = Inf;
for edge_mat_index = 1 : edge_mat_limit

```



```

    if edge_mat(edge_mat_index,6) == 0
        if edge_mat(edge_mat_index,5) < rel_mes_small
            rel_mes_small = edge_mat(edge_mat_index,5);
            rel_mes_small_index = edge_mat_index;
        end
    end
end
end
end
edge_mat_index = 1;
index = 1;
total = 0;
rel_mes_high = 0;
rel_mes_high_index = 0;
while edge_mat_index <= edge_mat_limit
    if edge_mat(edge_mat_index,6) == 0
        edge_mat1(index,:) = edge_mat(edge_mat_index,:);
        total = total + edge_mat(index,3);
        if rel_mes_high < edge_mat1(index,5)
            rel_mes_high = edge_mat1(index,5);
            rel_mes_high_index = index - 1;
        end
        index = index + 1;
    end
    edge_mat_index = edge_mat_index + 1;
end
edge_mat = edge_mat1;
clear edge_mat1;

```

```

edge_mat_limit = index - 1;
vertices_deleted_count = 0;
rel_mes_small = Inf;
for edge_mat_index = 1 : edge_mat_limit
    if edge_mat(edge_mat_index,6) == 0
        if edge_mat(edge_mat_index,5) < rel_mes_small
            rel_mes_small = edge_mat(edge_mat_index,5);
            rel_mes_small_index = edge_mat_index;
        end
        edge_mod_index = mod(rel_mes_high_index + edge_mat_index, check);
        if edge_mod_index == 0
            edge_mod_index = check;
        end
        feature_vector(feature_count,1) = edge_mat(edge_mod_index,1);
        feature_vector(feature_count,2) = edge_mat(edge_mod_index,2);
        feature_vector(feature_count,3) = edge_mat(edge_mod_index,3)/total;
        feature_vector(feature_count,4) = edge_mat(edge_mod_index,4);
        feature_vector(feature_count,5) = edge_mat(edge_mod_index,5);
        feature_count = feature_count + 1;
    end
end
feature_vector(feature_count,:) = [-1,-1,-1,-1,-1];
feature_count = feature_count + 1;

```

## SAMPLE CODE TO ADD IMAGE

```
input_file = imread(input_image_path);
iFlag = 0;
for file_search_index = 1:file_count
    fname=['C:\MATLAB\Database\test'num2str(file_search_index).bmp'];
    database_file = imread(fname);
    if isequal(input_file,database_file)
        iFlag = 1;
        break;
    end
end
if iFlag == 0
    fname=['C:\MATLAB\Database\test0.txt'];
    file_count = file_count + 1;
    fid = fopen(fname,'w');
    fprintf(fid,'%d',file_count);
    fclose(fid);
    fname=['C:\MATLAB\Database\test' num2str(file_count) '.txt'];
    fid = fopen(fname,'w');
    for i = 1 : 100
        fprintf(fid,'%d %d %f %f %f\n',feature_vector(i,:));
    end
    fclose(fid);
    fid = fopen(input_image_path,'r');
    [array,count]= fread(fid,Inf);
    fclose(fid);
```

```

fname=['C:\MATLAB7\Database\test' num2str(file_count) '.bmp'];
fid = fopen(fname,'w');
count= fwrite(fid,array);
fclose(fid);
set(handles.pushbutton7,'enable','on');
set(handles.pushbutton8,'enable','off');
else
    msgbox('Image Already found in the Database','Error','warn','modal');
end

```

## **SAMPLE CODE TO FIND RELATED IMAGES**

```

for index = 1: file_count
    fname=['c:\MATLAB7\Database\test' num2str(index) '.txt'];
    fname1=['c:\MATLAB7\Database\test' num2str(index) '.bmp'];
    fid = fopen(fname,'r');
    for vertices_index = 1 : 100
        [feature_vector1(vertices_index,:),line_count] =
            fscanf(fid,'%d %d %f %f %f', [1 5]);
    end
    fclose(fid);
    rel_dbase_50_index(1,1) = 1;
    if(feature_vector1(2,5)>feature_vector1(3,5))
        a = 2; b = 3; c = 3; d = 2;
    else
        a = 3; b = 2; c = 2; d = 3;
    end
end

```

```

for vertices_index = 4:50
    if feature_vector1(vertices_index,5) >= feature_vector1(a,5)
        b = a;
        a = vertices_index;
    end
    if feature_vector1(vertices_index,5) <= feature_vector1(c,5)
        d = c;
        c = vertices_index;
    end
end
end
rel_dbase_50_index(2,1) = a;
rel_dbase_50_index(3,1) = b;
rel_dbase_50_index(4,1) = c;
rel_dbase_50_index(5,1) = d;
count_large = -1;
count_large_index1 = -1;
count_large_index2 = -1;
count_large_direction = -1;
eucledian_small_index = -1;
eucledian_small_direction = -1;
eucledian_small_angle = Inf;
count_25 = 0;
for direction = 1: 2
    for count1 = 1 : 5
        lar_quer_index = rel_50_index(count1,1) ;
        lar_dbase_index = rel_dbase_50_index(count1,1) ;
        count_50(count1,direction) = 0;
    end
end

```

```

eucledian_length = 0;
eucledian_angle = 0;
for vertices_index = 1 : 50
    feature_index = lar_quer_index + vertices_index ;
    if feature_index > 50
        feature_index = feature_index - 50;
    end
    if direction == 1
        feature1_index = lar_dbase_index + vertices_index ;
        if feature1_index > 50
            feature1_index = feature1_index - 50;
        end
    else
        feature1_index = lar_dbase_index - vertices_index ;
        if feature1_index < 1
            feature1_index = feature1_index + 50;
        end
    end
    if abs(feature_vector(feature_index,4) -
feature_vector1(feature1_index,4)) <= 30 &
abs(feature_vector(feature_index,3) -
feature_vector1(feature1_index,3)) < 0.5
        count_50(count1,direction) = count_50(count1,direction) + 1;
    end
    length_minus = feature_vector1(feature1_index,3) -
feature_vector(feature_index,3);

```

```

angle_minus = feature_vector1(feature1_index,4) -
feature_vector(feature_index,4);
euclidian_length = euclidian_length +
(length_minus*length_minus);
euclidian_angle = euclidian_angle + (angle_minus*angle_minus);
end
euclidian_length = sqrt(euclidian_length);
euclidian_angle = sqrt(euclidian_angle);
if count_50(count1,direction) > count_large
count_large_index1 = lar_quer_index;
count_large_index2 = lar_dbase_index;
count_large_direction = direction;
count_large = count_50(count1,direction);
end
if euclidian_angle < euclidian_small_angle
euclidian_small_angle = euclidian_angle;
euclidian_small_direction = direction;
euclidian_small_index1 = lar_quer_index;
euclidian_small_length = euclidian_length;
euclidian_small_index2 = lar_dbase_index;
end
if count_50(count1,direction) >= 25
count_25 = count_25 + 1;
end
end
end
end
end

```

## REFERENCES

1. Alberto Chavez-Aragon, Oleg Starostenko, "Ontological shape-description, a new method for visual information retrieval," *conielecomp*, p. 288, 14th International Conference on Electronics, Communications and Computers, 2004.
2. Alberto Chavez-Aragon, Oleg Starostenko, "Image Retrieval by Ontological Description of Shapes (IRONS), Early Results," *crv*, pp. 341-346, 1st Canadian Conference on Computer and Robot Vision (CRV'04), 2004.
3. L. Gregory, J. Kittler, "Using Contextual Information for Image Retrieval," *iciap*, p. 0230, 11th International Conference on Image Analysis and Processing (ICIAP'01), 2001.
4. Ioannis Andreou, Nikitas M. Sgouros, "Computing, explaining and visualizing shape similarity in content-based image retrieval", Volume 41, Issue 5, *Information Processing and Management: an International Journal*, September 2005.



5. Evaluating Partial Shape Queries for Pathology-based Retrieval of Vertebra ,IEEE 2004 Jason Shou, Sameer Antani, L. Rodney Long, George R. Thoma(from [www.nlm.nih.gov](http://www.nlm.nih.gov))
  
6. A spine X-Ray Image retrieval System Using Partial Shape Matching, IEEE 2008, Xiaoqian Xu, Dah-Jye Lee, Senior Member, IEEE, Sameer Antani, Member, IEEE, and L. Rodney Long, Member, IEEE(from [www.nlm.nih.gov](http://www.nlm.nih.gov))

**Books:**

1. Gonzalez R., Eddins and Woods B.E., “Digital Image Processing”, Addison Wesley, 1993.
  
2. Donald Hearn, M. Pauline Baker, “Computer Graphics”, Second Edition, Prentice-Hall of India, 1994.
  
3. Stephen J. Chapman, “MATLAB Programming for Engineers”, Third Edition, Thomas Learning, 2004.
  
4. Dr. B. S. Grewal, “Higher Engineering Mathematics”, Khanna Publishers, 2001.

**Websites:**

1. [www.imageprocessingplace.com](http://www.imageprocessingplace.com)
2. [www.math.uni-hamburg.de](http://www.math.uni-hamburg.de)
3. [www.mathworks.com](http://www.mathworks.com)
4. [www.nlm.nih.gov](http://www.nlm.nih.gov)