P-2851

IMAGE RETRIEVAL USING SHAPE SIMILARITY

A PROJECT REPORT

Submitted by

KAVITHA.M	**71205104018**
SATHYA.S	**71205104043**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY:CHENNAI 600 025

APRIL 2009

ANNA UNIVERSITY:CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this work report "**IMAGE RETRIEVAL USING SHAPE SIMILARITY**" is the bonafide work of KAVITHA.M and SATHYA.S who carried out the project work under my supervision.

SIGNATURE

Dr.S.THANGASAMY

DEAN

Computer Science and Engineering

Dept. Of Computer Science

Kumaraguru College of Technology

Coimbatore-641 006

SIGNATURE

Mrs.AMUTHA VENKATESSH ,M.E.

SUPERVISOR

Senior Lecturer

Computer Science and Engineering

Dept. Of Computer Science

Kumaraguru College of Technology

Coimbatore-641 006

The candidates with University Register Nos. **71205104018, 71205104043** were examined by us in the project viva-voce examination held on.28/4/09

INTERNAL EXAMINAR

EXTERNAL EXAMINAR

ACKNOWLEDGEMENT

We would like to express our gratitude to **Prof.R.Annamalai, M.E.,** Vice Principal, Kumaraguru college of Technology, Coimbatore for helping us to complete this work successfully.

We are overwhelmingly grateful to **Dr.S.Thangasamy, Ph.D.,** Dean, Head of the Department of Computer Science and Engineering, Kumaraguru College of Technology, for his valuable guidance and useful suggestions.

We have immense pleasure in expressing our heartfelt thanks to our guide **Mrs.Amutha Venkatesh M.E.,** Senior Lecturer, Computer Science Department, Kumaraguru college of Technology, for her constant advice and support during the working of the project. We are grateful for her guidance.

We would like to thank our project co-ordinator **Mrs.P.Devaki, MS.,** Assistant Professor, Computer Science Department, Kumaraguru college of Technology, for her support during the course of the work.

We would like to express our sincere thanks to all the members of the faculty of the Department of Computer Science and Engineering for their support both directly and indirectly.

We also thank our parents for their blessing and support for the completion of our work successfully.

CONTENTS

ABSTRACT

ABSTRACT

Effective retrieval by content from database requires visual image properties to recover pictorial data. Shape plays an important role as a feature of image. This thesis proposes an approach for image similarity retrieval based on shape information. The shape information of the image is obtained by extracting the edge pixels. This edge pixels form the boundary of the image called shape. The shape then subjected to Discrete Fourier Transform. The lower frequency fourier transform coefficient values are used to guarantee the compactness and robustness to noise. The zeroth frequency coefficient is discriminated, because it only provides information about object position and says nothing about the object shape. The coefficients are modified in such a way to guarantee the translation, scaling, and starting point invariances. Then the shape of the query sketch image is compared with shape obtained from the database images. Finally images having similar shapes are retrieved from the database.

LIST OF ABBREVATIONS

JPEG-Joint Photographic Experts Group

GIF- Graphics Interchange Format

PNG- Portable Network Graphics Format

INTRODUCTION

1. INTRODUCTION

The visual information retrieval has become a major research area due to the ever-increasing rate at which images are generated in many application fields. Visual information retrieval systems support retrieval by visual content by directly addressing image perceptual features such as color, shape, texture and spatial relationship.

Shape plays an important role as feature of image. Shape similarity is obtained by computing the correlation between the query sketch and database edge images. Small shifts and distortions between the database images and the sketch are taken into account by shifting the local correlation between the two blocks of the database edge image and the user's sketch. This solution does not allow indexing. To filter out non-pertinent images, images in the database are aggregated according to a set of basic templates.

In fact, it has been verified that the feature vector model and Euclidean distance (more generally, metric distances) between feature points are not suited to model perceptual similarity between shapes.

1.1 PROBLEM DEFINITION

The main objective of this thesis is to retrieve digital images efficiently from large databases using shape similarity between the images. The strong intensity contrast pixels are called edge pixels of the image. These edge pixels form the shape.

1.2 PROPOSED SYSTEM

To measure the similarity between shapes, partitioning may be useful for comparison purpose as well as to identify the partial occlusions. Partial occlusions mean some of the partitions (Tokens) are identified to be similar in the compared shapes. While partitioning, the tokens have to be stored in the tree structures to avoid the unwanted comparisons. The predefining of kernel is needed to find the similar tokens in different images. So to avoid this, the entire shape is considered as a single token.

This work needs the identification of the image from the database that contains the similar shape. The database images may have similar shapes as that of query sketch, which may have a different scale and orientation. To guarantee this only the lower frequency Discrete Fourier Transform (DFT) coefficients are taken and the higher frequency coefficients are neglected. Zeroth coefficient is also discriminated because it contains information about object position not about the shape position.

OVERVIEW

2. OVERVIEW

Large image databases are increasingly used in many application areas like crime prevention, architectural and engineering design, fashion, medical diagnosis, journalism, advertising, and land analysis. This has motivated growing research interest on efficient and effective methods enabling the retrieval of images on the basis of their content from large databases. With respect to other features, like color and texture, shape is much more effective in semantically characterizing the content of an image.

However, properly extracting and representing shape information are still challenging tasks. In particular, even when accurate object boundaries are obtainable (this is the case when some domain knowledge is available or when images represent simple objects), the problem of representing them so as to allow the implementation of efficient and effective matching and retrieval methods is still not solved in a satisfactory way. The scenario is further complicated when invariance, with respect to a number of possible transformations, such as scaling, shifting, and rotation, is required.

Effective and efficient retrieval of images based on their shape content calls for a set of basic, often contrasting, requirements:

Compactness and simplicity of shape descriptors are necessary for minimizing the storage overhead and the extraction time; for an effective retrieval, the shape descriptors should be robust to noise and invariant to transformations (namely, translation, scaling, and rotation); finally, in order to avoid a sequential

scan of the whole (large) database, Shape descriptors should be indexable, e.g., by using metric trees, like the M-tree, that are already profitably applied in several other image and multimedia applications.

Among the different approaches that are available for the representation of shape information, those based on the Discrete Fourier Transform (DFT) describe the outside contour by means of a limited number of coefficients in the frequency domain [1].

Fourier-based approaches have two innovative characteristics:

1) The preservation of phase information
2) The use of the Dynamic Time Warping (DTW) distance to compare the shape descriptors.

Phase: The rationale for maintaining phase information is based on the observation that current Fourier-based techniques discard the phase of DFT coefficients with the purpose of achieving rotation invariance as well as independence from the starting point of the parameterization. This is a consequence of the fact that rotating an object boundary or changing the starting point introduces a phase shifting in the DFT coefficients. It resolve the apparent contradiction of preserving phase without giving up rotation and starting point invariances by deriving appropriate "compensation" terms that are added to the original phases, thus yielding a modified phase spectrum.

Image retrieval directly based on shape feature in DCT compressed domain depends on the edge feature of DCT compressed image, the rough object shape is extracted from DCT compressed-domain to form a binary edge map; then the similarity measurement is redefined by using the moments invariant to improve the retrieval performance.

Digital image databases have grown enormously in both size and number over these years. In order to save storage space and reduce transmission time, digital images are often compressed. Most of the compression standards, such as JPEG, MPEG 1/2, H.163, H.263, are based on DCT, which makes the research of images retrieval in DCT compressed domain significant. Compared to the traditional retrieval algorithm in spatial domain, it can avoid great computation of decompression by processing images directly in DCT compressed domain [2].

Although, some researches have been undertaken on images retrieval in compressed domain recently, most of these researches are based on the texture feature. Image retrieval based shape feature, however, plays an important part in CBIR (Content Based Image Retrieval), and consequently, it is meaningful to do some researches on shape feature based image retrieval in DCT compressed domain. It just implements entropy decoding to form the binary edge map in DCT compressed domain, then describes the shape of objects by moments invariant, and redefines the similarity measurement in order to retrieve.

Computer databases containing thousands or millions of images. Although many typical image database algorithms rely on a variety of shading, texture, and color attributes, there are significant opportunities for the use of shape as a discriminator, particularly for binary or high- contrast images. Fourier shape

descriptors have been studied extensively for shape comparison, however the descriptor phases have been mostly neglected either ignored entirely or treated Simplistically.

The image processing literature has had a long interest in the problem of shape matching; shape matching problem has been recently reignited by the fantastic proliferation of electronic imaging and images, particularly huge computer databases containing thousands or millions of images. Although many typical image database algorithms, such as finding two matching or cropped photographs, may rely on a variety of shading, texture, and color attributes, there are significant opportunities for the use of shape as a discriminator. Most trademarks are high-contrast binary images (foreground on background) from which a shape is readily acquired, and where shading, texture, and color play only a secondary role, or not at all.

Fourier shape descriptors have been studied extensively, however the descriptor phases have been mostly neglected. The phase leads to highly ambiguous matches [3], or treated simplistically in manner which is reasonable for complex, natural shapes, but which fails for simply structured shapes common among trademarks.

With the given huge number of binary images, they can be tested on the basis of shape, there are two, obvious criteria:

1. Comparison Speed, and
2. Compactness of Representation.

If two-dimensional shapes bounded by simple, closed curves, then it stands to reason that the one-dimensional boundary may be a much more compact description than the huge two-dimensional array of pixels making up the shape. Indeed, chain codes been developed for such a purpose to clockwise trace the outline of a shape.

The entire challenge, then, of shape matching is the interpretation and comparing of the Fourier descriptors. Because of the sensitivity of the phase (the complex angle component) of to image rotations and changes in the shape origin, often only the magnitude components are examined.

With the rapid growth of Internet and multimedia system, digital images have been enormously used in many applications now. The way of how to retrieve the image from a large image database correctly plays a more important role. Currently, most proposed methods for image retrieval can only be applied with images in the same representing domain.

- Spatial Images,
- DCT compressed Images,
- Wavelet-compressed images.

Most of the systems extract image features including color, texture, shape, and sketch to retrieve images in the spatial domain. However, these systems are not practical nowadays because many image compression methods are developed in different domains including JPEG and JPEG-2000.

If one still wants to use these spatial-based image retrieval systems to retrieve images, these compressed images should be first decompressed into uncompressed images.

Shneier and Abdel-Mottaleb proposed a method that can retrieve images directly on JPEG images without decompressing JPEG images [4]. However, the system cannot apply their method directly on uncompressed images because their source images are limited in compressed images. Meanwhile, the system cannot apply their method on different kind of compressed images such as JPEG-2000 images. Scale-invariant property can be then achieved by normalizing feature images to a predefined size. Then, the wavelet transform is performed for these normalized feature images and the energy of each high subband is used as the feature. Meanwhile, the proposed feature is further modified into a rotation-invariant feature. Image database may contain over 1000 images, which consist of raw image data, JPEG image, and wavelet-compressed image.

In order to achieve the purpose of directly retrieving the various types of images in their original domains, the images should be preprocessed in their own domains to extract feature images. For the same image, the feature images obtained in different domains will be the same or similar. To achieve this the images of different domain is preprocessed so that the pixel values of feature image is calculated.

Raw images divided into 8 by 8 blocks and the mean value of each block is assigned to be the pixel value of feature image. Because the feature image of raw images composed of the mean value of each 8 by 8, the mean value of each block in the JPEG image is directly extracted from its DC coefficient. For the reason that

a level shifting by -128 in JPEG images the real mean value of the block is (DC coefficient+128).For a wavelet compressed image, feature image is extracted from low-low band of the wavelet compressed the mean value of 4 by 4 block in the low-low sub image is assigned to the pixel value of the feature image.

The feature images will be the same if they are extracted from the raw image and the JPEG image of the same image. Moreover, the mean squared error (MSE) between feature images generated from the raw image and from the wavelet-compressed image is quite small. The feature images will be normalized to a predefined size. This normalized step ensures the scale-invariant property of the proposed method.

3. FILE FORMATS

3.1JPEG (Joint Photographic Experts Group)

JPEG is a standardized image compression mechanism. JPEG is designed for compressing either full-color (24 bit) or grey-scale digital images of "natural" (real-world) scenes. It works well on photographs, naturalistic artwork, and similar material; not so well on lettering, simple cartoons, or black-and-white line drawings (files come out very large). JPEG handles only still images, but there is a related standard called MPEG for motion pictures.

JPEG is "lossy", meaning that the image you get out of decompression isn't quite identical to what you originally put in. The algorithm achieves much of its compression by exploiting known limitation of the human eye, notably the fact that small color details aren't perceived as well as small details of light-and-dark. Thus, JPEG is intended for compressing images that will be looked at by humans. A lot of people are scared off by the term "lossy compression". But when it comes to representing real-world scenes, no digital image format can retain all the information that impinges on your eyeball. By comparison with the real-world scene, JPEG loses far less information than GIF.

3.2 QUALITY V COMPRESSION

A useful property of JPEG is that the degree of lossiness can be varied by adjusting compression parameters. This means that the image maker can trade off file size against output image quality. For good-quality, full-color source images, the default quality setting (Q 75) is very often the best choice. Except for experimental purposes, never go above about Q 95; using Q 100 will produce a file two or three times as large as Q 95, but of hardly any better quality. If you see a file made with Q 100, it's a pretty sure sign that the maker didn't know what he/she was doing. To get a very small file (say for preview or indexing purposes) and are prepared to tolerate large defects, a Q setting in the range of 5 to 10 is about right.

3.3 GIF (Graphics Interchange Format)

The Graphics Interchange Format was developed in 1987 at the request of Compuserve it needed a platform independent image format that was suitable for transfer across slow connections. It is a compressed (lossless) format (it uses the LZW compression) and compresses at a ratio of between 3:1 and 5:1

It is an 8-bit format which means the maximum number of colors supported by the format is 256. There are two GIF standards, 87a and 89a (developed in 1987 and 1989 respectively). The 89a standard has additional features such as improved interlacing, the ability to define one color to be transparent and the ability to store multiple images in one file to create a basic form of animation. Both Mosaic and Netscape will display 87a and 89a GIFs, but while both support transparency and interlacing, only Netscape supports animated GIFs.

3.4 PNG (Portable Network Graphics format)

In January 1995 Unisys, the company Compuserve contracted to create the GIF format, announced that they would be enforcing the patent on the LZW compression technique the GIF format uses. This means that commercial developers that include the GIF encoding or decoding algorithms have to pay a license fee to Compuserve. This does not concern users of GIFs or non-commercial developers. However, a number of people banded together and created a completely patent-free graphics format called PNG (pronounced "ping"), the Portable Network Graphics format. PNG is superior to GIF in that it has better compression and supports millions of colours. PNG files end in a .png suffix.

JPEG is not going to displace GIF entirely. For some types of images, GIF is superior in image quality, file size, or both. One of the first things to learn about JPEG is which kinds of images to apply it to. Generally speaking, JPEG is superior to GIF for storing full-color or grey-scale images of "realistic" scenes; that means scanned photographs and similar material. Any continuous variation in color, such as occurs in highlighted or shaded areas, will be represented more faithfully and in less space by JPEG than by GIF.

GIF does significantly better on images with only a few distinct colors, such as line drawings and simple cartoons. Not only is GIF lossless for such images, but it often compresses them more than JPEG can. For example, large areas of pixels that are all exactly the same color are compressed very efficiently indeed by GIF. JPEG can't squeeze such data as much as GIF does without introducing visible defects. (One implication of this is that large single-color borders are quite cheap in GIF files, while they are best avoided in JPEG files.)

3.5 JPEG Encoder

This encoder is an implementation of the baseline JPEG specification and saves files in the JFIF format.

In order to use it in Java applications, simply create a JpegEncoder object, specifying an image, the desired quality, and an OutputStream, and then call that object's Compress method:

...

```
JpegEncoder jpg = new JpegEncoder(image, quality, outStream);

jpg.Compress();
```

...

The JpegEncoder object has its own MediaTracker, so just create an image object and initialize it to get image. quality may be any integer from 0 to 100. 0 is maximum compression and 100 is minimum compression. 70 - 80 is an excellent range but 50 gives good results--the image degradation is just noticeable (and for the test image, the resulting jpeg file size was 3% of the original BMP file).

Any OutputStream can be sent to the JpegEncoder object. JpegEncoder creates a BufferedOutputStream using the outStream object. A command line interface that allows the user to create a JPEG file from any of the image formats that the Java Virtual Machine can read. Inspiration for this encoder was provided by Florian Raemy's JPEG encoder, the one that doesn't write a file. More importantly, a major portion of the code was based on the C code in the Independent JPEG Group's 6a release of their JPEG library. A few things were

done differently to take advantage of Java's capabilities. The result is a surprisingly fast JPEG encoder and JFIF writer.

Computer-drawn images (ray-traced scenes, for instance) usually fall between photographs and cartoons in terms of complexity. The more complex and subtly rendered the image, the more likely that JPEG will do well on it. The same goes for semi-realistic artwork (fantasy drawings and such).

JPEG has a hard time with very sharp edges: a row of pure-black pixels adjacent to a row of pure-white pixels, for example. Sharp edges tend to come out blurred unless you use a very high quality setting. Edges this sharp are rare in scanned photographs, but are fairly common in GIF files: borders, overlaid text, etc. The blurriness is particularly objectionable with text that's only a few pixels high. If you have a GIF with a lot of small-size overlaid text, don't JPEG it.

Plain black-and-white (two level) images should never be converted to JPEG; they violate all of the conditions given above. You need at least about 16 grey levels before JPEG is useful for grey-scale images. It should also be noted that GIF is lossless for grey-scale images of up to 256 levels, while JPEG is not.

3.6 GIF Encoder

Java lets programmers load images (and URLs) with a single method call. This is a very empowering feature, and has resulted in many wonderful and vivid applets. Unfortunately, there is no built in way to save an image to a file. Because of this, Java hasn't yet spawned a new generation of image editors and filters.

3.7 Class GIF Encoder

public class **GIF Encoder** extends Object

GIF Encoder is a class which takes an image and saves it to a stream using the GIF file format. A GIFEncoder is constructed with either an AWT Image (which must be fully loaded) or a set of RGB arrays. The image can be written out with a call to Write.

Three caveats:

- GIF Encoder will convert the image to indexed color upon construction. This will take some time, depending on the size of the image. Also, actually writing the image out (Write) will take time.
- The image cannot have more than 256 colors, since GIF is an 8 bit format.
- Since the image must be completely loaded into memory, GIFEncoder may have problems with large images. Attempting to encode an image which will not fit into memory will probably result in the java.awt.AWTException.

EDGE DETECTION

4. EDGE DETECTION

Edges characterize boundaries and are therefore a problem of fundamental importance in image processing. Edges in images are areas with strong intensity contrasts of a jump in intensity from one pixel to the next. Detecting edges in an image significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image.

- First and most obvious is low error rate. It is important that edges occurring in images should not be missed and that there be no responses to non-edges.

- The second criterion is that the edge points be well localized. In other words, the distance between the edge pixels as found by the detector and the actual edge is to be at a minimum.

- A third criterion is to have only one response to a single edge. This was implemented because the first 2 were not substantial enough to completely eliminate the possibility of multiple responses to an edge.

Based on these criteria, the edge detector first smoothen the image to eliminate and noise. It then finds the image gradient to highlight regions with high spatial derivatives. Then tracks along these regions and suppresses any pixel that is not at the maximum (nonmaximum suppression). The gradient array is now further reduced by hysteresis. Hysteresis is used to track along the remaining pixels that have not been suppressed. Hysteresis uses two thresholds and if the magnitude is

below the first threshold, it is set to zero (made a nonedge). If the magnitude is above the high threshold, it is made an edge. And if the magnitude is between the 2 thresholds, then it is set to zero unless there is a path from this pixel to a pixel with a gradient above high threshold.

In order to implement the edge detection, a series of steps are followed.

The first step is to filter out any noise in the original image before trying to locate and detect any edges. And because the Gaussian filter can be computed using a simple mask. Once a suitable mask has been calculated, the Gaussian smoothing can be performed using standard convolution methods. A convolution mask is usually much smaller than the actual image. As a result, the mask is slid over the image, manipulating a square of pixels at a time. The larger the width of the Gaussian mask, the lower is the detector's sensitivity to noise. The localization error in the detected edges also increases slightly as the Gaussian width is increased.

After smoothing the image and eliminating the noise, the next step is to find the edge strength by taking the gradient of the image. Then, the approximate absolute gradient magnitude (edge strength) at each point can be found. Usually it uses a pair of 3x3 convolution masks, one estimating the gradient in the x-direction (columns) and the other estimating the gradient in the y-direction (rows). They are shown below:

The magnitude, or Edge Strength, of the gradient is then approximated using the formula:

$|G| = |Gx| + |Gy|$

The next step involves in finding the edge direction is trivial once the gradient in the x and y directions are known. Whenever the gradient in the x direction is equal to zero, the edge direction has to be equal to 90 degrees or 0 degrees, depending on what the value of the gradient in the y-direction is equal to. If GY has a value of zero, the edge direction will equal 0 degrees. Otherwise the edge direction will equal 90 degrees.

The formula for finding the edge direction is just:

Theta = invtan (Gy / Gx)

Once the edge direction is known, the next step is to relate the edge direction to a direction that can be traced in an image. So if the pixels of a 5x5 image are aligned as follows:

x x x x x

x x x x x

x x a x x

x x x x x

x x x x x

Then, it can be seen by looking at pixel "a", there are only four possible directions when describing the surrounding pixels - 0 degrees (in the horizontal direction), 45 degrees (along the positive diagonal), 90 degrees (in the vertical direction), or 135 degrees (along the negative diagonal). So now the edge orientation has to be resolved into one of these four directions depending on which direction it is closest to (e.g. if the orientation angle is found to be 3 degrees, make it zero degrees). Therefore, any edge direction falling within the yellow range (0 to 22.5 & 157.5 to 180 degrees) is set to 0 degrees. Any edge direction falling in the green range (22.5 to 67.5 degrees) is set to 45 degrees. Any edge direction falling in the

blue range (67.5 to 112.5 degrees) is set to 90 degrees. And finally, any edge direction falling within the red range (112.5 to 157.5 degrees) is set to 135 degrees.

After the edge directions are known, nonmaximum suppression now has to be applied. Nonmaximum suppression is used to trace along the edge in the edge direction and suppress any pixel value (sets it equal to 0) that is not considered to be an edge. This will give a thin line in the output image.

Finally, hysteresis is used as a means of eliminating streaking. Streaking is the breaking up of an edge contour caused by the operator output fluctuating above and below the threshold. If a single threshold, T1 is applied to an image, and an edge has an average strength equal to T1, then due to noise, there will be instances where the edge dips below the threshold. Equally it will also extend above the threshold making an edge look like a dashed line. To avoid this, hysteresis uses 2 thresholds, a high and a low. Any pixel in the image that has a value greater than T1 is presumed to be an edge pixel, and is marked as such immediately. Then, any pixels that are connected to this edge pixel and that have a value greater than T2 are also selected as edge pixels. If you think of following an edge, you need a gradient of T2 to start but you don't stop till you hit a gradient below T1.

IMPLEMENTATION

5. IMPLEMENTATION

This thesis proposes retrieval by shape similarity for generic shapes using local features. This deals with cases in which the relevant information is concerned only with relative distances between objects rather than with their absolute positions in a multidimensional space.

Firstly the images are stored into the database for feature retrieval purpose which consist of the following tasks

1. Registering the oracle driver (using the *registerdriver* method of *DriverManager* object)
2. Connecting the oracle database (using the *getconnection* method of *DriverManager* object). This *DriverManger* object belongs to SQL package of java.

Connecting the oracle database requires IPAddress of the server where the database is actually resides to, username, password, Sid name of the database and the table name. The images are stored into the databases as blobs.

Edges are extracted from images to form the boundary called shape. This discriminates the unwanted information from the images while preserving the structural information.

While searching, the database is again registered and connected and the database images are retrieved in the order of their Imageid. Each images from the database is subjected to edge detection and the detected edge image is compared with the edged image of the query sketch.

The comparison requires 720 comparisons for each database image. That is the database edged image is subjected to 10 scaling and 72 rotations (for each five degree). So each scaled and rotated image is compared with the query shape to identify the similar image in the database. Even their shape is scaled or rotated the images are identified as similar images as that of query image. Finally the most similar image is identified and retrieved from the database.

This work is started by double clicking the *start.bat* file that contains the java command file to start *ImageFrame.java*. Upon clicking the load image button, a file open dialog box is opened to choose the required image file. The selected image file is subjected to edge detection and the edged image is considered as the shape of that image and is displayed.

To enable the search process the search button is activated. This makes the connection with the Oracle server. Now it is possible to retrieve all images one by one using BLOB Technology. The image is retrieved from the database in the order of their imageId and is temporarily stored in local computer. Then the image is converted into pixels which inturn the pixels are converted into integer values. Now the edge detection is applied to retrieve the strong intensity contrast pixels from the image and the non-edge pixels are discriminated. These edge pixels are stored in a pixel array.

java Affine Transformation utility is used to rotate the retrieved shape, from the temporarily stored image that is retrieved from the database for comparison purpose.

The *Transform* method of *AffineTransform* object is used to perform rotation for each five degree. The rotated shape is then compared with the shape of the local image.

The scaling on shape is performed using the *getScaledImage* method of the *image* object. The scaled image is then compared with the shape of the input local image

The comparison is performed continuously with 720 combinations of the scale and rotations.

Clean button is used to reset the images to original form. *TransferFilestoDatabase* button is activated to upload all the images inside the predefined folder to oracle database using BLOB Streaming Technology.

Another image is retrieved from the database in the sequential order of imageId and continues to perform the above steps for all images in the database. Finally the most closely matched image is retrieved and stored in a result folder.

To identify the images that are similar in their shape information, the images are subjected to edge detection. So the structural information is preserved while the others are discriminated. The edge-detected images are compared using their kernels.

The implementation is done using Java as the front end and Oracle 9i as the back end. This is designed with a user-friendly interface enabling the users to maneuver the menu's easily.

SOFTWARE REQUIREMENTS

6. SOFTWARE REQUIREMENTS

The hardware and software configurations used to develop this work are given below

6.1 HARDWARE CONFIGURATION

Processor : Pentium IV

Clock speed : 1.5GHz

Main memory : 256RAM

Hard disk : 20GB

6.2 SOTWARE CONFIGURATION

Operating system : Windows XP

Front end : Java

Back end : Oracle 9i

CONCLUSION

7. CONCLUSION

The shape information of the image is obtained by extracting the edge pixels. These edge pixels form the boundary of the image called shape. The shape is then subjected to Discrete Fourier Transformation. The lower frequency fourier transform coefficient values are used to guarantee the compactness and robustness to noise. The zeroth frequency coefficient is discriminated, because it provides information about object position and says nothing about the object shape. The coefficients are modified in such a way to guarantee the translation, scaling, and starting point invariances. The shape of the query sketch image is compared with shape obtained from the database images. Finally images having similar shapes are retrieved from the database.

The images are stored as blobs in the database. In future, the compressed images can be stored in the database as well as it is possible to develop an algorithm that directly works on the compressed image. This may reduce the storage requirements.

FUTURE ENHANCEMENT

This technique is developed in such a way that any enhancements may be made with ease. In future the proposed method can be improved by increasing the speed and also the number of images in the database.

APPENDIX

8. SCREEN SHOTS

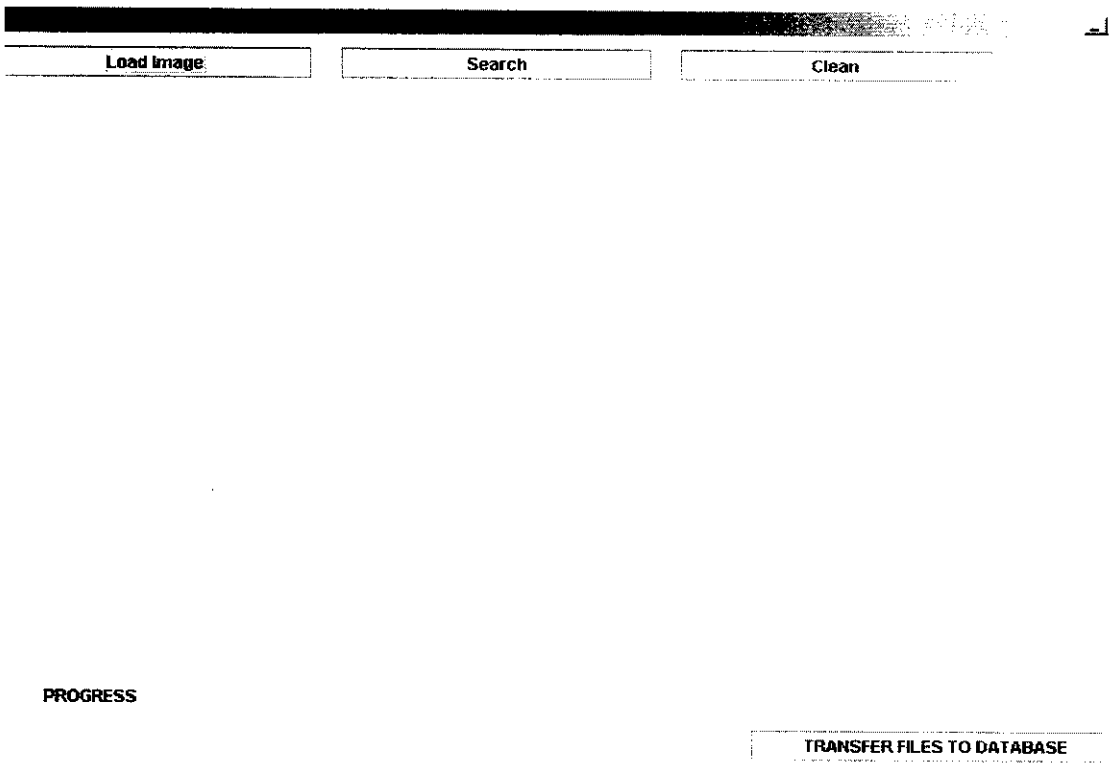

Fig 1. Form design

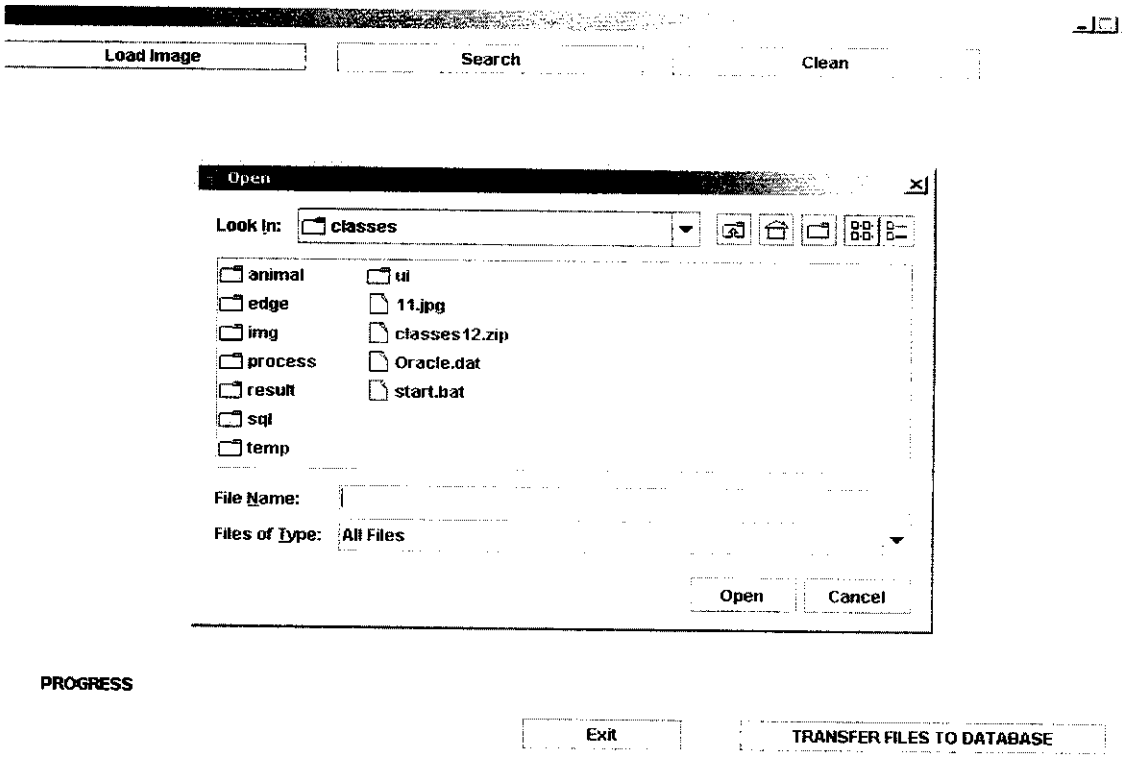

Fig 2. Open dialogue box

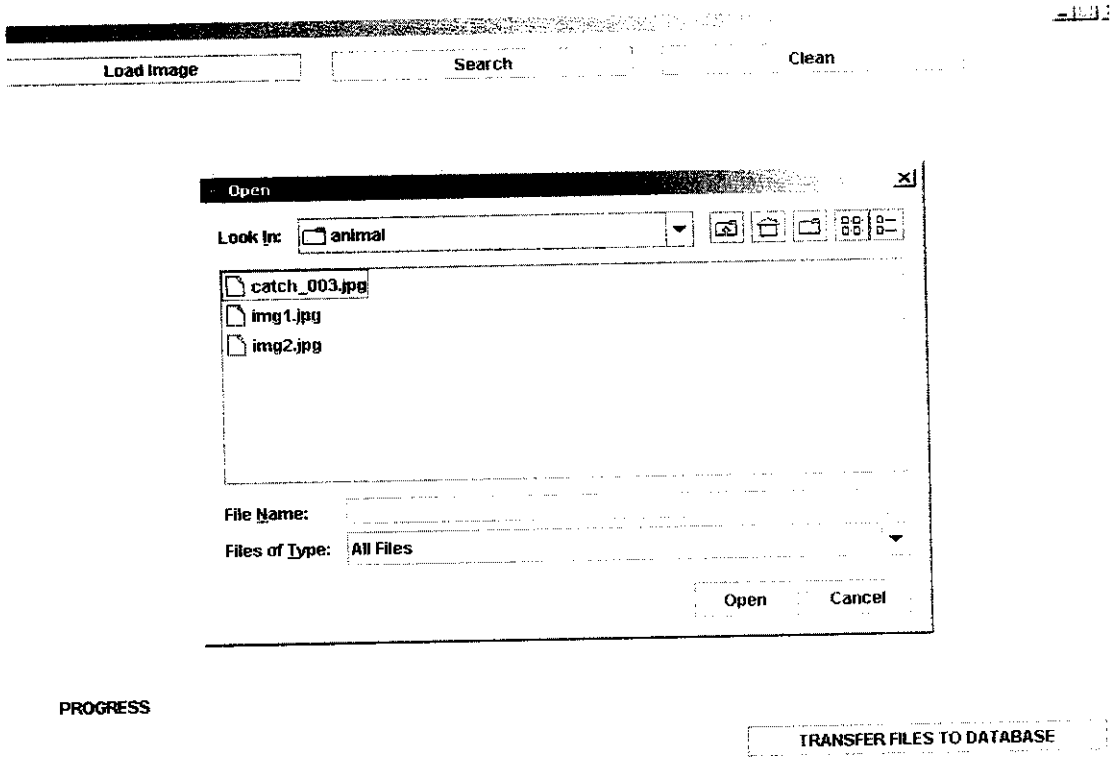

Fig 3. Open dialogue box with selected Query sketch

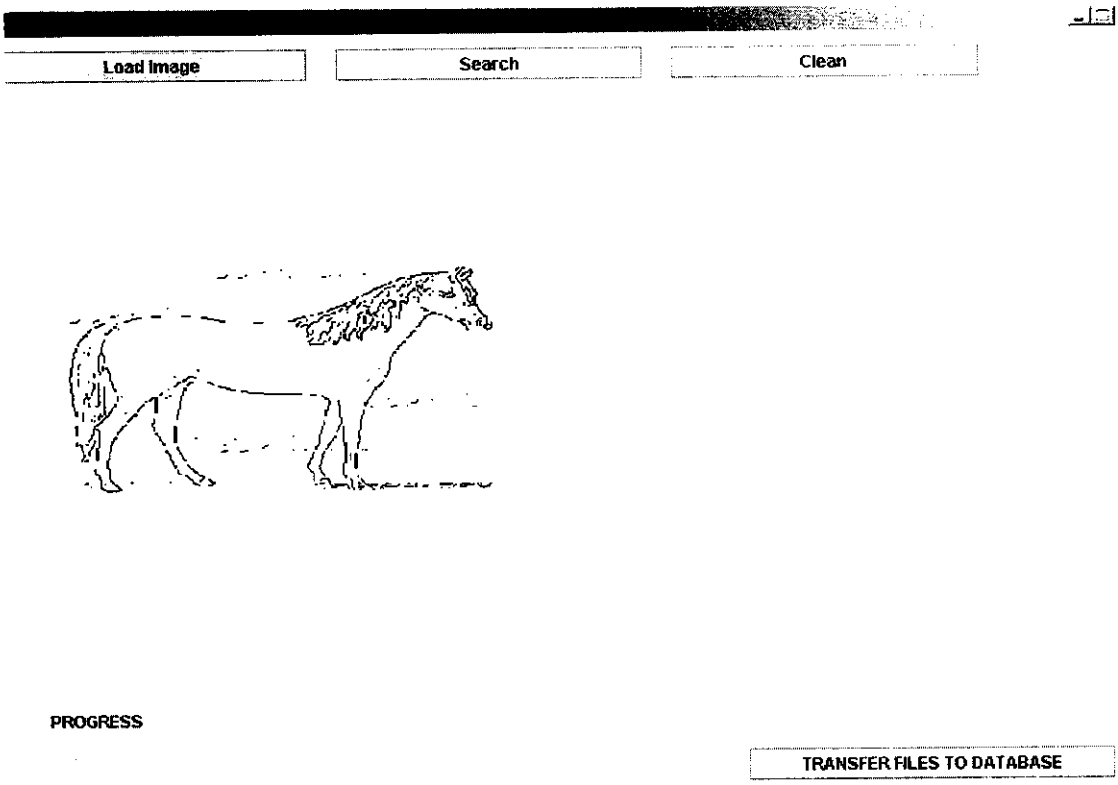

Fig 4. Shape of the selected Query sketch

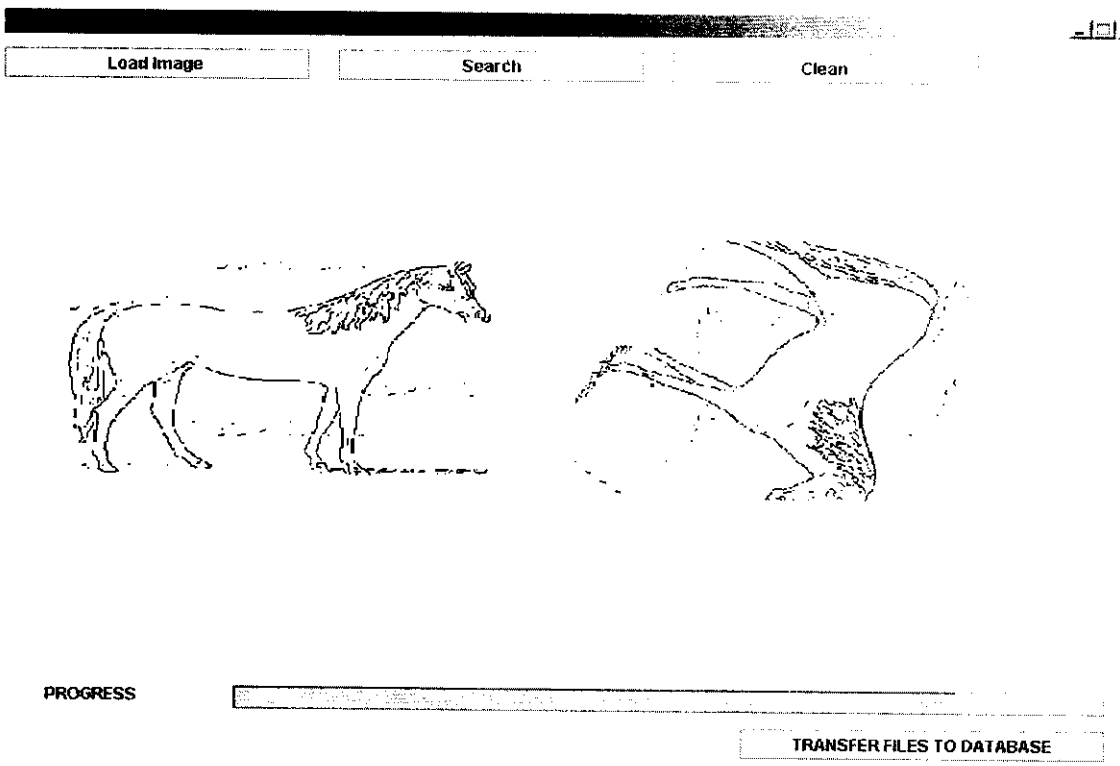

Fig 5. Comparison of Query shape with rotated shape of database image

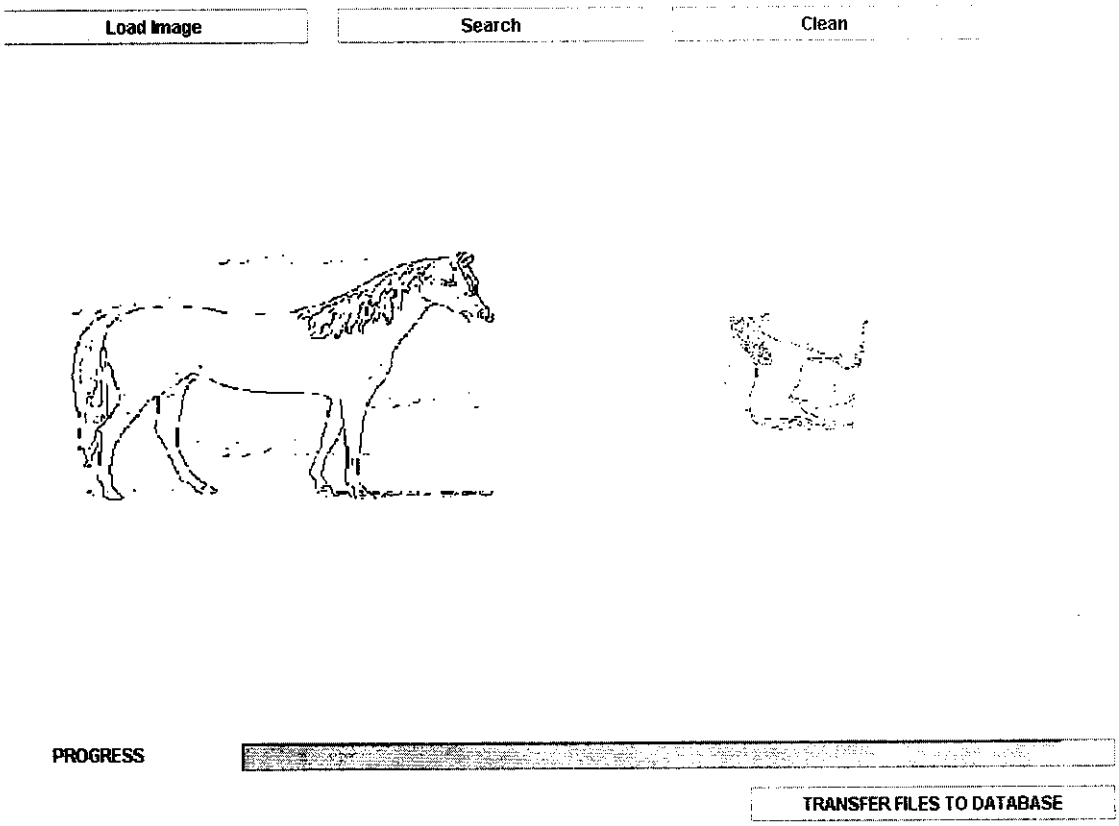

Fig 6. Comparison with scaled and rotated database image

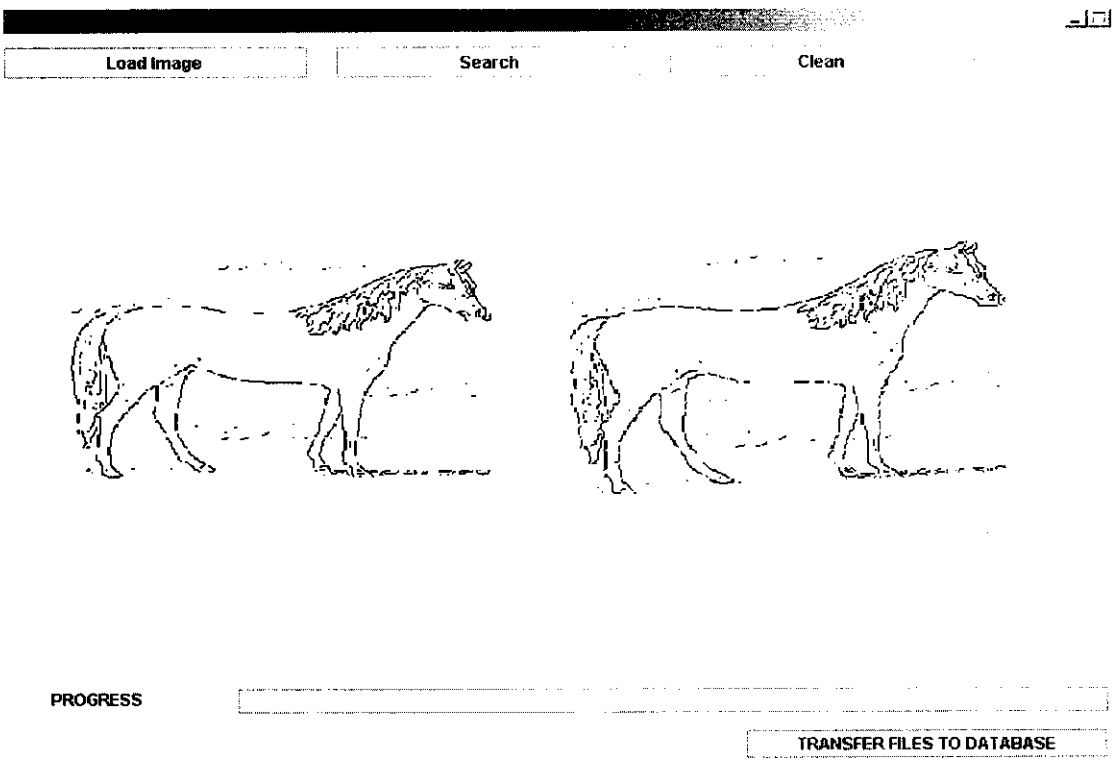

Fig 7. Retrieved image with similar shape.

REFERENCES

9.REFERENCES

1. Ilaria Bartolini, Paolo Ciaccia, and Marco Patella, "Accurate Retrieval of Shapes Using Phase of Fourier Descriptors and Time Warping Distance", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 1, January 2005.

2. Zhang Xihuang, Bian Guochun,Xu Wenbo, "A Shape Feature Based Image Retrieval in DCT Compressed-Domain" *International Conference on Computer and Information Technology*, IEEE 2005.

3. Paul Fieguth, Paul Bloore Adrian Domsa, " Phase-based Methods for Fourier Shape Matching", *IEEE* 2003.

4. Ruey-Feng Chang, Wen-Jia Kuo and Hung-Chi Tsai, "Image Retrieval on Uncompressed and Compressed Domains", *IEEE* 2000.

5. Alberto Del Bimbo and Pietro Pala, "Visual Image Retrieval by Elastic Matching of User Sketches", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, February 1997.

6. Fang Liu and Rosalind W. Picard, "Periodicity, directionality and randomness: Wold features for image modeling and retrieval", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 7, July 1996.

7. Simone Santini, and Ramesh Jain, "Similarity Measures", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, September 1999.

8. David A White, Ramesh Jain, "Similarity Indexing with the SS-tree", *IEEE* 1996.

9. G.M. Petrakis, Evangelos Miliost, "Efficient Retrieval by Shape Content", *IEEE* 1999.