CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Aim of the Project

The aim of this project is to review the current state of the art in content-based image retrieval (CBIR), a technique for retrieving images on the basis of automatically-derived features such as color, texture and shape. Our findings are based both on a review of the relevant literature and on discussions with researchers in the field.

The need to find a desired image from a collection is shared by many professional groups, including journalists, design engineers and art historians. While the requirements of image users can vary considerably, it can be useful to characterize image queries into three levels of abstraction: primitive features such as color or shape, logical features such as the identity of objects shown and abstract attributes such as the significance of the scenes depicted. While CBIR systems currently operate effectively only at the lowest of these levels, most users demand higher levels of retrieval.

1.2 Image retrieval

An image retrieval system returns a set of images from a collection of images in the data base to meet users demand with similarity evaluations such as image content similarity, edge pattern similarity etc. An image retrieval system offers an efficient way to access, browse and retrieve a set of similar images in the real time applications. Several approaches are been developed to capture the information of image contents by directly computing the image features from an image.

Content based image retrieval (CBIR) is the mainstay of current image retrieval systems. In general the purpose of CBIR is to present an image conceptually, with a set of low level visual features such as texture, shape etc

1.3 Shape features

Shape feature provides the most important semantic information about an image. Shape features are usually described using part or region of an image. The accuracy of shape features largely depends upon the segmentation scheme used to divide an image into meaningful objects. However, fast and robust segmentation is difficult to achieve. This limits the shape features only to those retrieval applications where objects or region of images are readily available. The shape descriptors are categorized into two classes: boundary based descriptor and region based descriptor. Some boundary based

representative shape description techniques are chain codes, polygonal approximations, Fourier descriptor and finite element model. On the other hand state of the art region based descriptors are statistical moment and area. A good shape feature should be invariant to translation, rotation and scaling. A detail review of shape matching techniques used in image retrieval application can be found in [Mingqiang et al.(2008)].

1.4 Similarity measure

The degree of similarity between query and target images is calculated based on the value of similarity measure. The images are ranked according to their similarity value and presented as output of CBIR system. Often, the choice of similarity measure affects the performance of retrieval system. Many similarity measures have been developed over the years based on the quantitative estimates of the distribution of features in the image. Some of the most commonly used similarity measures employed in CBIR are Euclidean distance, Minkowski - form distance.

1.5 Motivation of the work

In the last two decades, CBIR systems have been improved a lot. However, there still remain some problems which have not been answered satisfactorily. First and foremost problem is of semantic gap, which exist between low level feature representation of images and the actual visual perception of the image. Researchers all over the globe are working in the direction of narrowing down this semantic gap. Semantic gap is a big problem which can be seen as a collection of many small problems. In this work, we have identified such problems and tried to provide an effective solution to these problems.

1.6 General Introduction

1.6.1Content Based Image Retrieval

Content-based image retrieval (**CBIR**), also known as query by image content (QBIC) and content-based visual information retrieval (CBVIR) is the application of computer vision to the image retrieval problem, that is, the problem of searching for digital images in large databases.

"Content-based" means that the search will analyze the actual contents of the image. The term 'content' in this context might refer colors, shapes, textures, or any other information that can be derived from the image itself. Without the ability to examine

image content, searches must rely on metadata such as captions or keywords. Such metadata must be generated by a human and stored alongside each image in the database.
Problems with traditional methods of image indexing [Enser,1995] have led to the rise of interest in techniques for retrieving images on the basis of automatically-derived features such as color, texture and shape – a technology now generally referred to as Content-Based Image Retrieval (CBIR). However, the technology still lacks maturity, and is not yet being used on a significant scale. In the absence of hard evidence on the effectiveness of CBIR techniques in practice, opinion is still sharply divided about their usefulness in handling real-life queries in large and diverse image collections. The concepts which are presently used for CBIR system are all under research.

1.6.2 Images

Let us start with the word "image". The surrounding world is composed of images. Humans are using their eyes, containing 1.5x10^8sensors, to obtaining images from the surrounding world in the visible portion of the electromagnetic spectrum (wavelengths between 400 and 700 nanometers). The light changes on the retina are sent to image processor center in the cortex.
In the image database systems geographical maps, pictures, medical images, pictures in medical atlases, pictures obtaining by cameras, microscopes, telescopes, video cameras, paintings, drawings and architectures plans, drawings of industrial parts, space images are considered as images.

There are different models for color image representation. In the seventeen century Sir Isaac Newton showed that a beam of sunlight passing through a glass prism comes into view as a rainbow of colors. Therefore, he first understood that white light is composed of many colors. Typically, the computer screen can display 2^8 or 256 different shades of gray. For color images this makes 2^(3x8) = 16,777,216 different colors.

Clerk Maxwell showed in the late nineteen century that every color image cough be created using three images – Red, green and Blue image. A mix of these three images can produce every color. This model, named RGB model, is primarily used in image representation. The RGB image could be presented as a triple(R, G, B) where usually R, G, and B take values in the range [0, 255]. Another color model is YIQ model (lamination (Y) , phase (I), quadrature phase (Q)). It is the base for the color television standard. Images are presented in computers as a matrix of pixels. They have finite area. If we

decrease the pixel dimension the pixel brightness will become close to the real brightness.

1.6.3 Image Database systems

Set of images are collected, analyzed and stored in multimedia information systems, office systems, Geographical information systems(GIS), robotics systems, CAD/CAM systems, earth resources systems, medical databases, virtual reality systems, information retrieval systems, art gallery and museum catalogues, animal and plant atlases, sky star maps, meteorological maps, catalogues in shops and many other places.

There are sets of international organizations dealing with different aspects of image storage, analysis and retrieval. Some of them are: AIA (Automated Imaging/Machine vision), AIIM (Document imaging), ASPRES (Remote Sensing/ Protogram) etc.

There are also many international centers storing images such as : Advanced imaging, Scientific/Industrial Imaging, Microscopy imaging, Industrial Imaging etc. There are also different international work groups working in the field of image compression, TV images, office documents, medical images, industrial images, multimedia images, graphical images.

1.6.4 Logical Image Representation in Database Systems:

The logical image representation in image databases systems is based on different image data models. An image object is either an entire image or some other meaningful portion (consisting of a union of one or more disjoint regions) of an image. The logical image description includes: meta, semantic, color, texture, shape, and spatial attributes.

Color attributes could be represented as a histogram of intensity of the pixel colors. A histogram refinement technique is also used by partitioning histogram bins based on the spatial coherence of pixels. Statistical methods are also proposed to index an image by color correlograms, which is actually a table containing color pairs, where the k-th entry for <i, j> specifies the probability of locating a pixel of color j at a distance k from a pixel of color I in the image.

1.6.5 Classification and indexing schemes

Many picture libraries use keywords as their main form of retrieval – often using indexing schemes developed in-house, which reflect the special nature of their collections. A good example of this is the system developed by Getty Images to index their collection of

contemporary stock photographs. Their thesaurus comprises just over 10 000 keywords, divided into nine semantic groups, including geography, people, activities and concepts. Index terms are assigned to the whole image, the main objects depicted, and their setting. Retrieval software has been developed to allow users to submit and refine queries at a range of levels, from the broad (e.g. “freedom”) to the specific (e.g. “a child pushing a swing”).

Probably the best-known indexing scheme in the public domain is the Art and Architecture Thesaurus (AAT), originating at Rensselaer Polytechnic Institute in the early 1980s, and now used in art libraries across the world. AAT is maintained by the Getty Information Institute and consists of nearly 120,000 terms for describing objects, textural materials, images, architecture and other cultural heritage material. There are seven facets or categories which are further subdivided into 33 sub facets or hierarchies. The facets, which progress from the abstract to the concrete, are: associated concepts, physical attributes, styles and periods, agents, activities, materials, and objects. AAT is available at http://www.ahip.getty.edu/aat_browser/. Other tools from Getty include the Union List of Artist Names (ULAN) and the Getty Thesaurus of Geographic Names (TGN). Another popular source for providing subject access to visual material is the Library of Congress Thesaurus for Graphic Materials (LCTGM). Derived from the Library of Congress Subject Headings (LCSH), LCTGM is designed to assist with the indexing of historical image collections in the automated environment. Greenberg [1993] provides a useful comparison between AAT and LCTGM.

A number of indexing schemes use classification codes rather than keywords or subject descriptors to describe image content, as these can give a greater degree of language independence and show concept hierarchies more clearly. Examples of this genre include ICONCLASS from the University of Leiden [Gordon, 1990], and TELCLASS from the BBC [Evans, 1987]. Like AAT, ICONCLASS was designed for the classification of works of art, and to some extent duplicates its function; an example of its use is described by Franklin [1998]. TELCLASS was designed with TV and video programs in mind, and is hence rather more general in its outlook. The Social History and Industrial Classification, maintained by the Museum Documentation Association, is a subject classification for museum cataloguing. It is designed to make links between a wide variety of material including objects, photographs, archival material, tape recordings and information files.

A number of less widely-known schemes have been devised to classify images and drawings for specialist purposes. Examples include the Vienna classification for trademark images [World Intellectual Property Organization, 1998], used by registries Worldwide to identify potentially conflicting trademark applications, and the Opitz coding system for machined parts [Opitz et al, 1969], used to identify families of similar parts which can be manufactured together.

A survey of art librarians conducted for this report suggests that, despite the existence of specialist classification schemes for images, general classification schemes, such as Dewey Decimal Classification (DDC), Library of Congress (LC), BLISS and the Universal Decimal Classification (UDC), are still widely used in photographic, slide and video libraries. The former scheme is the most popular, which is not surprising when one considers the dominance of DDC in UK public and academic library sectors. ICONCLASS, AAT, LCTGM, SHIC are all in use in at least one or more of the institutions in the survey. However, many libraries and archives use in-house schemes for the description of the subject content. For example, nearly a third of all respondents have their own in-house scheme for indexing slides.

When discussing the indexing of images and videos, one needs to distinguish between systems which are geared to the formal description of the image and those concerned with subject indexing and retrieval. The former is comparable to the bibliographical description of a book. However, there is still no one standard in use for image description, although much effort is being expended in this area by a range of organizations such as the Museum Documentation Association, the Getty Information Institute, the Visual Resources Association the International Federation of Library Association/Art Libraries and the International Committee for Documentation (CIDOC) of the International Council of Museums (ICOM).

The descriptive cataloguing of photographs presents a number of special challenges. Photographs, for example, are not self-identifying. Unlike textual works that provide such essential cataloguing aids as title pages, abstracts and table of contents, photographs often contain no indication of author or photographer, names of persons or places depicted dates, or any textual information whatever. Cataloguing of images is more complex than that for text documents, since records should contain information about the standards used for image capture and how the data is stored as well as descriptive information, such as title, photographer (or painter, artist, etc). In addition, copies of certain types of images may involve many layers of intellectual property rights,

pertaining to the original work, its copy (e.g. a photograph), a digital image scanned from the photograph, and any subsequent digital image derived from that image.

Published reviews of traditional indexing practices for images and video include many writers discuss the difficulties of indexing images. The problems of managing a large image collection. He notes that, unlike books, images make no attempt to tell us what they are about and that often they may be used for purposes not anticipated by their originators. Images are rich in information and can be used by researchers from a broad range of disciplines. As Baser comments:

"A set of photographs of a busy street scene a century ago might be useful to historians wanting a 'snapshot' of the times, to architects looking at buildings, to urban planners looking at traffic patterns or building shadows, to cultural historians looking at changes in fashion, to medical researchers looking at female smoking habits, to sociologists looking at class distinctions, or to students looking at the use of certain photographic processes or techniques."

Svenonius [1994] discusses the question of whether it is possible to use words to express the "abruptness of a work in a wordless medium, like art. To get around the problem of the needs of different users groups, van der Starre [1995] advocates that indexers should "stick to 'plain and simple' indexing, using index terms accepted by the users, and using preferably a thesaurus with many lead-ins," thus placing the burden of further selection on the user. Shatford Layne (1994) suggests that, when indexing images, it may be necessary to determine which attributes provide useful groupings of images; which attributes provide information that is useful once the images are found; and which attributes may, or even should, be left to the searcher or researcher to identify. She also advocates further research into the ways images are sought and the reasons that they are useful in order to improve the indexing process. Constantopulos and Doer (1995) also support a user centered approach to the designing of effective image retrieval systems. They urge that attention needs to be paid to the intentions and goals of the users, since this will help define the desirable descriptive structures and retrieval mechanisms as well as understanding what is 'out of the scope' of an indexing system.

When it comes to describing the content of images, respondents in our own survey seem to include a wide range of descriptors including title, period, genre, subject headings, keywords, classification and captions (although there was some variation by format). Virtually all maintain some description of the subject content of their images. The majority of our respondents maintain manual collections of images, so it is

not surprising that they also maintain manual indexes. Some 11% of respondents included their photographs and slides in the online catalogues, whilst more than half added their videos to their online catalogues. Standard text retrieval or database management systems were in use in a number of libraries (with textual descriptions only for their images). Three respondents used specific image management systems: Index+, iBase and a bespoke in-house system. Unsurprisingly, none currently use CBIR software.

1.6.6 Research Trends in the Image Database Systems

Most image database systems are products of research, and therefore emphasize only one aspect of content-based retrieval. Sometimes this is the sketching capability in the user interface; sometimes it is a new indexing data structure, etc. Some systems are created as a research version and a commercial product. The commercial version is usually less advanced, and shows more standard searching capabilities. A number of systems provide user interface that allows more powerful query formulation than is useful in demo system. Most systems use color and texture features, few systems use shape features, and yet less use spatial features. The retrieval on color usually yield images that have similar colors. The larger the collection of images, the greater is the chance that it contains an image similar to the query image.

CHAPTER 2

LITERATURE SURVEY

2.1 The growth of digital imaging

The use of images in human communication is hardly new – our cave-dwelling ancestors painted pictures on the walls of their caves, and the use of maps and building plans to convey information almost certainly dates back to pre-Roman times. But the twentieth century has witnessed unparalleled growth in the number, availability and importance of images in all walks of life. Images now play a crucial role in fields as diverse as medicine, journalism, advertising, design, education and entertainment.

Technology, in the form of inventions such as photography and television, has played a major role in facilitating the capture and communication of image data. But the real engine of the imaging revolution has been the computer, bringing with it a range of techniques for digital image capture, processing, storage and transmission which would surely have startled even pioneers like John Logie Baird. The involvement of computers in imaging can be dated back to 1965, with Ivan Sutherland's Sketchpad project, which demonstrated the feasibility of computerized creation, manipulation and storage of images, though the high cost of hardware limited their use until the mid-1980s. Once computerized imaging became affordable (thanks largely to the development of a mass market for computer games), it soon penetrated into areas traditionally depending heavily on images for communication, such as engineering, architecture and medicine. Photograph libraries, art galleries and museums, too, began to see the advantages of making their collections available in electronic form. The creation of the World-Wide Web in the early 1990s, enabling users to access data in a variety of media from anywhere on the planet, has provided a further massive stimulus to the exploitation of digital images. The number of images available on the Web was recently estimated to be between 10 and 30 million [Sclaroff et al, 1997] – a figure which some observers consider to be a significant underestimate.

2.2 Current level 1 CBIR techniques

In contrast to the text-based approach of the systems, CBIR operates on a totally different principle, retrieving stored images from a collection by comparing features automatically extracted from the images themselves. The commonest features used are mathematical measures of color, texture or shape; hence virtually all current **CBIR systems**, whether commercial or experimental, operate at level1. A typical system allows users to formulate queries by submitting an example of the type of image being sought, though some offer alternatives such as selection from a palette or sketch input. The system then identifies those stored images whose feature values match those of the query most closely, and displays thumbnails of these images on the screen Error! Reference source not found.. Some of the more commonly used types of feature used for image retrieval are described below.

2.2.1 Color retrieval

Several methods for retrieving images on the basis of color similarity have been described in the literature, but most are variations on the same basic idea. Each image added to the collection is analyzed to compute a color histogram which shows the proportion of pixels of each color within the image. The color histogram for each image is then stored in the database. At search time, the user can either specify the desired proportion of each color (75% olive green and 25% red, for example), or submit an example image from which a color histogram is calculated. Either way, the matching process then retrieves those images whose color histograms match those of the query most closely. The matching technique most commonly used, histogram intersection, was first developed by Swain and Ballard [1991]. Variants of this technique are now used in a high proportion of current CBIR systems. Methods of improving on Swain and Ballard's original technique include the use of cumulative color histograms, combining histogram intersection with some element of spatial matching, and the use of region-based color querying. The results from some of these systems can look quite impressive.

2.2.2 Shape retrieval

The ability to retrieve by shape is perhaps the most obvious requirement at the primitive level. Unlike texture, shape is a fairly well-defined concept and there is considerable evidence that natural objects are primarily recognized by their shape. A number of features characteristic of object shape (but independent of size or orientation) are computed for every object identified within each stored image. Queries are then answered by computing the same set of features for the query image, and retrieving those stored images whose features most closely match those of the query. Two main types of shape feature are commonly used global features such as aspect ratio, circularity and moment invariants and local features such as sets of consecutive boundary segments. Alternative methods proposed for shape matching have included elastic deformation of templates, comparison of directional histograms of edges extracted from the image, and shocks, skeletal representations of object shape that can be compared using graph matching techniques. Queries to shape retrieval systems are formulated either by identifying an example image to act as the query, or as a user-drawn sketch.

Shape matching of three-dimensional objects is a more challenging task – particularly where only a single 2-D view of the object in question is available. While no general solution to this problem is possible, some useful inroads have been made into the problem of identifying at least some instances of a given object from different viewpoints. One approach has been to build up a set of plausible 3-D models from the available 2-D image, and match them with other models in the database. Another is to generate a series of alternative 2-D views of each database object, each of which is matched with the query image. Related research issues in this area include defining 3-D shape similarity measures, and providing a means for users to formulate 3-D shape queries.

2.2.3 Retrieval by other types of primitive feature

One of the oldest-established means of accessing pictorial data is retrieval by its position within an image. Accessing data by spatial location is an essential aspect of geographical information systems, and efficient methods to achieve this have been around for many years. Similar techniques have been applied to image collections, allowing users to search for images containing objects in defined spatial relationships with each other. Improved algorithms for spatial retrieval are still being proposed. Spatial indexing is seldom useful on its own, though it has proved effective in combination with other cues such as color and shape.

CHAPTER 3

EXISTING SYSTEM

Despite the shortcomings of current CBIR technology, several image retrieval systems are now available as commercial packages, with demonstration versions of many others available on the Web.

3.1 Commercial systems

IBM's QBIC system is probably the best-known of all image content retrieval systems. It is available commercially either in standalone form, or as part of other IBM products such as the DB2 Digital Library. It offers retrieval by any combination of color, texture or shape – as well as by text keyword. Image queries can be formulated by selection from a palette, specifying an example query image, or sketching a desired shape on the screen. The system extracts and stores color, shape and texture features from each image added to the database, and uses R*-tree indexes to improve search efficiency. At search time, the system matches appropriate features from query and stored images, calculates a similarity score between the query and each stored image examined, and displays the most similar images on the screen as thumbnails. The latest version of the system incorporates more efficient indexing techniques, an improved user interface, the ability to search grey-level images, and a video storyboarding facility.

3.2 Experimental systems

A large number of experimental systems have been developed, mainly by academic institutions, in order to demonstrate the feasibility of new techniques. Many of these are available as demonstration versions on the Web.

The Netra system uses colour texture, shape and spatial location information to provide region-based searching based on local image. An interesting feature is its use of sophisticated image segmentation techniques**.** An example of European CBIR technology is the Surf image system from INRIA, France. This has a similar philosophy to the MARS system, using multiple types of image feature which can be combined in different ways, and offering sophisticated relevance feedback facilities.

3.3 Automated System

In the automated system, we have adopted Principal Component Analysis (PCA) for achieving dimensionality reduction in the retrieval of gray map images by calculating the column wise mean of the image and making that as the index to represent that image in the feature database.

To improve the above method we have taken the mean of the principle diagonal pixels of the image and making this as the index. Here the computational time for calculating the mean is reduced but accuracy to retrieve exact image is reduced particularly when there is a huge set of images in the database. In this method we can store up to 2560000 different images with same resolution and size.

In the histogram analysis for retrieving the gray scale images, we have given the flexibility to change the no of bins while calculating the histogram counts.

In color component analysis for retrieving the similar images we have implemented Euclidean Distance formula to calculate the distance between the query image and the images present in the database. The images that fall within the specified threshold are retrieved. We can change the value of threshold according to our needs.

3.4 Objective

The need to find a desired image from a collection is shared by many professional groups, including journalists, design engineers and art historians. While the requirements of image users can vary considerably, it can be useful to characterize image queries into three levels of abstraction: *primitive* features such as color or shape, *logical* features such as the identity of objects shown and *abstract* attributes such as the significance of the scenes depicted. While CBIR systems currently operate effectively only at the lowest of these levels, most users demand higher levels of retrieval.

We have implemented the CBIR system which takes into consideration the low level features of image which is more comprehensive when compared to high level features and it also gives user a higher level of retrieval. We have divided an Image into two very basic categories of color and gray scale and used different features vector for similarity comparison and retrieval. We have used columnar mean, diagonal mean and histogram for gray scale and RGB values and Euclidean methods for color image. User always wants a friendly environment so that they can easily and effectively use the system without actually going into the finer details of the working. So, to create such a

user friendly platform for the system we have designed a Graphic User Interface where user can actually select the method which they want to be used for the image retrieval and that will give them an option of using different method if the result is not as per their requirement.

Users needing to retrieve images from a collection come from a variety of domains, including crime prevention, medicine, architecture, fashion and publishing. Remarkably little has yet been published on the way such users search for and use images, though attempts are being made to categorize users' behavior in the hope that this will enable their needs to be better met in the future. Attempts are also going on integrating the search for all kind of images and combining all above mentioned feature vectors for comparison and retrieval so as to achieve the best possible efficiency.

3.5 Demerits of existing system

The demerits of existing system are:

- Detailing within the concepts of those analysis of the extracted images is not done; the images had very lowest precise which could be generally that has the most of them within the overall concepts wherein within the concept.
- Precision of images which could be categorized is not achieved, and was time consuming as well to have feature extraction normally which could be done this was a major disadvantage in the major aspects.
- The modelling is never achieved and images is never extremely blur hence the enhancement of the images were not complete.
- There are multiple performance matrics measured within it in order to obtain the best suited of some of the most useful features to determine which is not in turn clear in the overall situation
- There is many problem of searching the images within the large database the content based image retrieval based means the search they analyse the content.

CHAPTER 4

IMPLEMENTATION

4.1 System Requirements

4.1.1 Hardware Requirements

- Pentium 4 2.66 Ghz/AMD Athlon 2800+ processor.
- 256MB RAM (Minimum)/512MB RAM (Recommended).
- 40 GB+ Hard Disk.
- Color Monitor.

4.1.2 Software Requirements

- MATLAB 7.0
- MySQL 2000

4.2 MATLAB Introduction

MATLAB is a software package for high-performance numerical computation and visualization. It provides an interactive environment with hundreds of built-in functions for technical computation, graphics and animation. It also provides easy extensibility with its own high level programming language. The name MATLAB stands for Matrix Laboratory.

MATLAB is an efficient program for vector and matrix data processing. It contains ready functions for matrix manipulations and image visualization and allows a program to have modular structure. Because of these facts MATLAB has been chosen as prototyping software.

MATLAB provides a suitable environment for image processing. Although MATLAB is slower than some languages (such as C), its built in functions and syntax makes it a more versatile and faster programming environment for image processing. Once an algorithm is finalized in MATLAB, the programmer can change it to C (or another faster language) to make the program run faster.

4.2.1 Overview of MATLAB

- MATLAB is built up by means of math works intended for fourth-generation programming language. A variety of process approved within MATLAB contains control concerning the matrix, purpose as well as plotting of data, execution regarding

algorithms, design of user interface, as well as integrating by means of programs formed within other languages like C, C++, and java. Despite mathematical calculation, MATLAB can be meant for representational calculation as well. MATLAB can be meant for embedded methods and by the guide regarding extra package known as Simulink. Specifically MATLAB permit intended for matrix estimation as well as thus can be intended for image processing. MATLAB is simple towards gaining knowledge of a variety of device boxes used for it; an illustration is image processing toolbox.

- MATLAB interfaces programming surroundings, calculation as well as mental picture. This contains integrated correcting, data compositions as well as object-oriented correcting devices. These integrated tasks create MATLAB appropriately used for education as well as to do research. To resolve scientific trouble MATLAB includes other benefits than usual programming language like C++ and java. MATLAB arrived into promotion in 1984 in addition to now it is employed globally. Additional graphical instructions are offered within MATLAB that builds the visual effects obtainable right away. A variety of device box contains signal processing, simulation, control theory as well as some former functions that are employed extensively in science and technology. The lone disadvantage regarding MATLAB is expenditure worry.

4.2.2 Features of Matlab

- Interactive background meant for aim investigation as well as resolving the difficulty.
- MATLAB is a sophisticated language intended for creating, calculating as well as building up a purpose.
- It contains numerical tasks such as figures, calculus, sorting out, developments, mathematical integration, as well as working out equations.
- Graphics integrated intended for visualization.
- Intended for generating traditional plot integrated equipments is accessible.

4.3 WORKING OF MATLAB

4.3.1 MATLAB TECHNOLOGY

The name MATLAB stands for Matrix Laboratory. MATLAB was written originally to provide easy access to matrix software developed by the LINPACK (linear system package) and EISPACK (Eigen system package) projects. MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming environment. Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming. These factors make MATLAB an excellent tool for teaching and research. MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems. MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. The software package has been commercially available since 1984 and is now considered as a standard tool at most universities and industries worldwide. It has powerful built-in routines that enable a very wide variety of computations. It also has easy to use graphics commands that make the visualization of results immediately available. Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

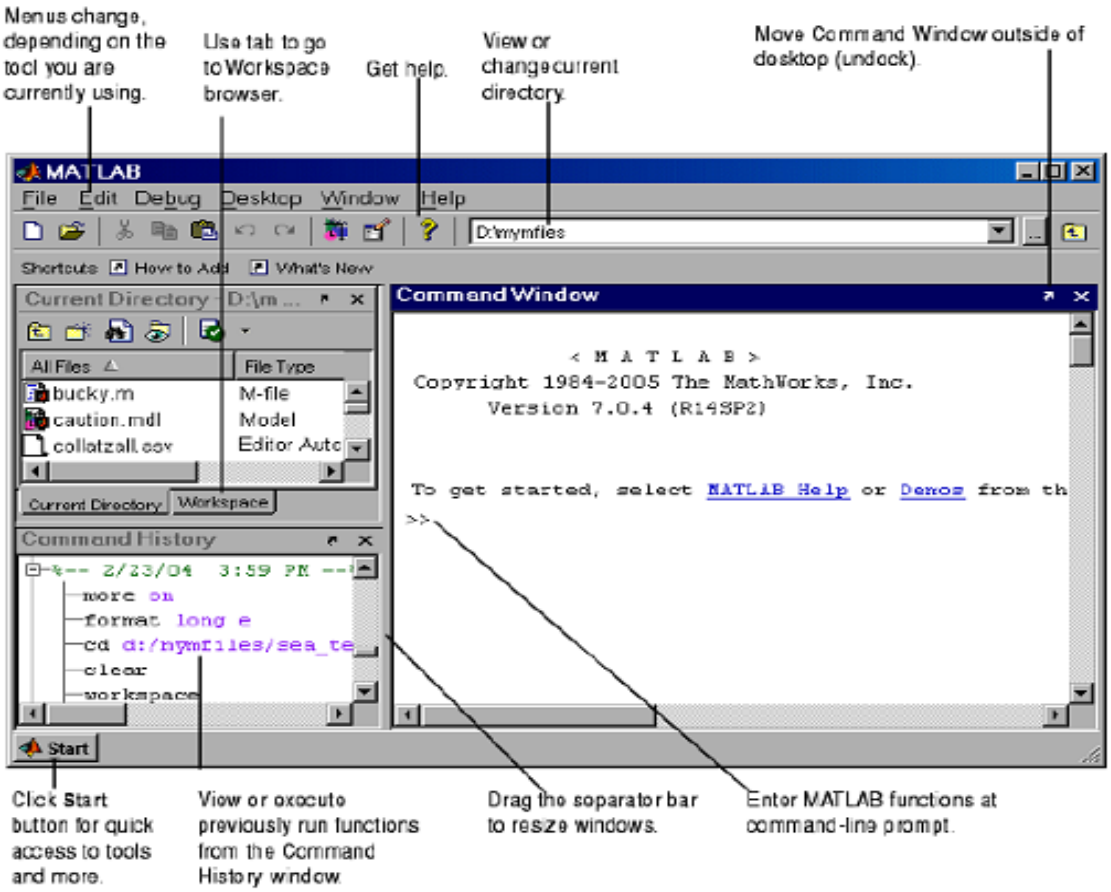

Figure 4.1- MATLAB Home screen

When MATLAB is started for the first time, the screen looks like the one that shown in the Figure 7.1. This illustration also shows the default configuration of the MATLAB desktop. You can customize the arrangement of tools and documents to suit your needs. Now, we are interested in doing some simple calculations. We will assume that you have sufficient understanding of your computer under which MATLAB is being run. You are now faced with the MATLAB desktop on your computer, which contains the prompt (>>) in the Command Window.

Usually, there are 2 types of prompt:

>>for full version

EDU> for educational version

Note: To simplify the notation, we will use this prompt, >>, as a standard prompt sign, though our MATLAB version is for educational purpose.

4.3.2 Quitting MATLAB

To end your MATLAB session, type quit in the Command Window, or select File —>Exit MATLAB in the desktop main menu.

4.3.3 MATLAB variables

MATLAB variables are created with an assignment statement. The syntax of variable assignment is

variable name = a value (or an expression)

Where expression is a combination of numerical values, mathematical operators, variables, and function calls. On other words, expression can involve:

- manual entry
- built-in functions
- user-defined functions

4.3.4 Miscellaneous commands

Here are few additional useful commands:

- To clear the Command Window, type clc
- To abort a MATLAB computation, type ctrl-c
- To continue a line, type . . .

Getting help

- To view the online documentation, select MATLAB Help from Help menu or MATLAB Help directly in the Command Window. The preferred method is to use the *Help Browser*. The Help Browser can be started by selecting the ?icon from the desktop toolbar.

  On the other hand, information about any command is available by typing

  >> help Command

- Another way to get help is to use the look for command. The look for command differs from the help command. The help command searches for an exact function name match, while the look for command searches the quick summary information in each function for a match. For example, suppose that we were looking for a function to take the inverse of a matrix. Since MATLAB does not have a function named inverse, the command help inverse will produce nothing. On the other hand, the command look for inverse will produce detailed information, which includes the function of interest, inv.

>> look for inverse

4.3.5 M-File functions

As mentioned earlier, functions are programs (or routines) that accept input arguments and return output arguments. Each Mfile function (or function or M file for short) has its own area of workspace, separated from the MATLAB base workspace

Anatomy of a M-File function

This simple function shows the basic parts of an M-file

function f = factorial (n) (1)

% FACTORIAL (N) returns the factorial of N. (2)

% Compute a factorial value. (3)

f = prod (1: n); (4)

The first line of a function M-file starts with the keyword function. It gives the function name and order of arguments. In the case of function factorial, there are up to one output argument and one input argument. Table 5.1 summarizes the M-File function.

As an example, for *n* = 5, the result is,

>> f = factorial (5)

f =120

Part No.	M-File element	Description
1	Function definition line	Define the function name, and the number and order of input and output arguments
2	H1 line	A one line summary description of the program, displayed when you request Help
3	Help text	A more detailed description of the program
4	Function body	Program code that performs the actual computations

Table 4.1- Anatomy of a M-File functions

Both functions and scripts can have all of these parts, except for the function definition line which applies to function only.

4.3.6 Output commands

As discussed before, MATLAB automatically generates a *display* when commands are executed. In addition to this automatic display, MATLAB has several commands that can be used to generate displays or outputs.

Two commands that are frequently used to generate output are: disp and fprintf. The main differences between these two commands can be summarized as follows:

- disp : Simple to use. Provide limited control over the appearance of output
- fprintf : Slightly more complicated than disp. Provide total control over the appearance of output

4.3.7 Debugging M-files

This section introduces general techniques for finding *errors* in M-files. *Debugging* is the process by which you isolate and fix *errors* in your program or code.
Debugging helps to correct two kind of errors:

Syntax errors - For example omitting a parenthesis or misspelling a function name.

Run-time errors - Run-time errors are usually apparent and difficult to track down. They produce unexpected results.

Debugging process

We can debug the M-files using the Editor/Debugger as well as using debugging functions
from the Command Window. The debugging process consists of

- Preparing for debugging
- Setting breakpoints
- Running an M-file with breakpoints
- Stepping through an M-file
- Examining values
- Correcting problems

- ➢ Ending debugging

Preparing for debugging

- ➢ Here we use the Editor/Debugger for debugging. Do the following to prepare for debugging:
- ➢ Open the file
- ➢ Save changes
- ➢ Be sure the file you run and any files it calls are in the directories that are on the search path.

Setting breakpoints

- ➢ Set breakpoints to pause execution of the function, so we can examine where the problem might be. There are three basic types of breakpoints:
    - A standard breakpoint, which stops at a specified line.
    - A conditional breakpoint, which stops at a specified line and under specified conditions.
    - An error breakpoint that stops when it produces the specified type of warning, error, NaN or infinite value.
- ➢ You cannot set breakpoints while MATLAB is busy, for example, running an M-file.

Running with breakpoints

- ➢ After setting breakpoints, run the M-file from the Editor/Debugger or from the Command Window.
- ➢ Running the M-file results in the following:
- ➢ The prompt in the Command Window changes to K>>indicating that MATLAB is in debug mode.
- ➢ The program pauses at the *first* breakpoint. This means that line will be executed when you continue. The pause is indicated by the green arrow.
- ➢ In breakpoint, we can examine variable, step through programs, and run other calling functions.

Examining values

- ➢ While the program is paused, we can view the value of any variable currently in the workspace.
- ➢ Examine values when we want to see whether a line of code has produced the expected result or not. If the result is as expected, step to the next line, and continue running.
- ➢ If the result is not as expected, then that line, or the previous line, contains an *error*. When we run a program, the current workspace is shown in the Stack field. Use who or whos to list the variables in the current workspace. Viewing values as datatipsfirst, we position the cursor to the left of a variable on that line. Its current value appears. This is called a *datatip*, which is like a *tooltip* for data.
- ➢ If you have trouble getting the datatip to appear, click in the line and then move the cursor next to the variable.

Correcting and ending debugging

- ➢ While debugging, we can change the value of a variable to see if the *new* value produces expected results.
- ➢ While the program is paused, assign a new value to the variable in the Command Window, Workspace browser, or Array Editor. Then continue running and stepping through the program.

Ending debugging

- ➢ After identifying a problem, end the debugging session. It is best to quit *debug mode* before editing an M-file.
- ➢ Otherwise, you can get unexpected results when you run the file. To end debugging, select Exit Debug Mode from the Debug menu.

4.3.8 Correcting an M-file

- ➢ To correct errors in an M-file, Quit debugging.
- ➢ Do not make changes to an M-file while MATLAB is in debug mode.
- ➢ Make changes to the M-file.

➢ Save the M-file.

➢ Clear breakpoints.

4.4 Image Representation in MATLAB

An image is stored as a matrix using standard Matlab matrix conventions. There are five basic types of images supported by Matlab:

1. Indexed images
2. Intensity images
3. Binary images
4. RGB images
5. 8-bit images

MATLAB handles images as matrices. This involves breaking each pixel of an image down into the elements of a matrix. MATLAB distinguishes between color and gray scale images and therefore their resulting image matrices differ slightly.

A color is a composite of some basic colors. MATLAB therefore breaks each individual pixel of a color image (termed 'true color') down into Red, Green and Blue values. What we get as a result, for the entire image, are 3 matrices, one representing each color. The three matrices are stacked next to each other creating a 3 dimensional m by n by 3 matrixes. For an image which has a height of 5 pixels and width of 10 pixels the resulting in MATLAB would be a 5 by 10 by 3 matrixes for a true color image.

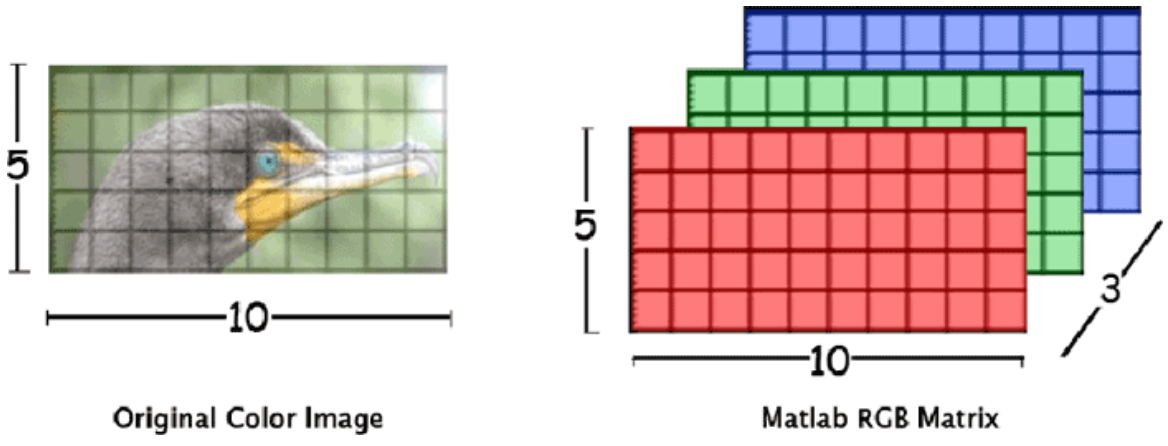

Fig: 4.2 Color image representation and RGB matrix

A gray scale image is a mixture of black and white colors. These colors, or as some may term as 'shades', are not composed of Red, Green or Blue colors. But instead they contain various increments of colors between white and black. Therefore to represent this one range, only one color channel is needed. Thus we only need a 2

dimensional matrix, m by n by 1. MATLAB terms this type of matrix as an Intensity Matrix, because the values of such a matrix represent intensities of one color.

For an image which as height of 5 pixels and width of 10 pixels the resulting matrix would be a 5 by 10 matrix for gray scale image.

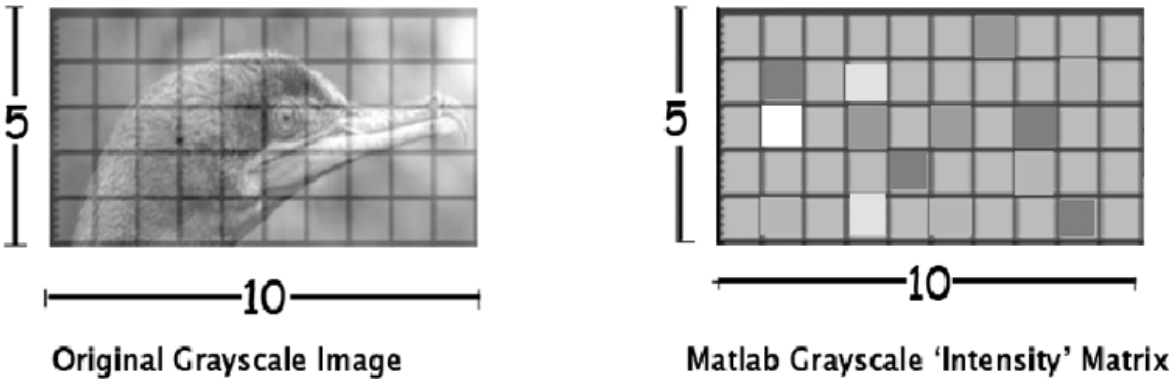

Fig: 4.3 Gray scale image representations.

4.4.1 The Advantages of MATLAB

MATLAB is a general purpose programming language. When it is used to process images one generally writes function files, or script files to perform the operations. These files form a formal record of the processing used and ensures that the final results can be tested and replicated by others should the need arise.

MATLAB provides many functions for image processing and other tasks. Most of these functions are written in the MATLAB language and are publicly readable as plain text files. Thus the implementation details of these functions are accessible and open to scrutiny. The defense can examine the processing used in complete detail, and any challenges raised can be responded to in an informed way by the prosecution. This makes MATLAB very different from applications, such as Photoshop.

It should be noted that some MATLAB functions cannot be viewed. These are generally lower level functions that are computationally expensive and are hence provided as 'built-in' functions running as native code. These functions are heavily used and tested and can be relied on with considerable confidence.

Another advantage of MATLAB is that it allows one to ensure maximal numerical precision in the final result.

In general, image files store data to 8 bit precision. This corresponds to a range of integer values from 0-255. A pixel in a color image may be represented by three 8 bit numbers, each representing the red, green and blue components as an integer value between 0 and 255. Typically this is ample precision for representing normal images.

However as soon as one reads this image data into memory and starts to process it it is very easy to generate values that lie outside the range 0-255. For example, to double the contrast of an image one multiplies the intensity values by 2. An image value of 200 will become 400 and numerical overflow will result. How this is dealt with will vary between image processing programs. Some may truncate the results to an integer in the range 0-255; others may perform the mathematical operations in floating point arithmetic and then rescale the final results to an integer in the range 0-255.

It is here that numerical precision, and hence image fidelity, may be lost. Some image processing algorithms result in some pixel values with very large magnitudes (positive or negative). Typically these large values occur at points in the image where intensity discontinuities occur, the edges of the image are common sources of this problem. When this image with widely varying values is rescaled to integers in the range 0-255 much of this range may be used just to represent the few pixels with the large values. The bulk of the image data may then have to be represented within a small range of integer values, say from 0-50. Clearly this represents a considerable loss of image information. If another process is then applied to this image the problems can then accumulate. Trying to establish the extent of this problem, if any, is hard if one is using proprietary software.

Being a general programming language it is possible to have complete control of the precision with which one represents data in MATLAB. An image can be read into memory and the data cast into double precision floating point values. All image processing steps can then be performed in double precision floating point arithmetic, and at no intermediate stage does one need to rescale the results to integers in the range 0-255. Only at the final point when the image is to be displayed and/or written to file does it need to be rescaled. Here one can use histogram truncation to eliminate extreme pixel values so that the bulk of the image data is properly represented.

MATLAB is a scientific programming language and provides strong mathematical and numerical support for the implementation of advanced algorithms. It is for this reason that MATLAB is widely used by the image processing and computer vision community. New algorithms are very likely to be implemented first in MATLAB, indeed they may only be available in MATLAB.

MATLAB may not be as user friendly as an application like Photoshop, however, being a general purpose programming language it provides many important advantages for forensic image processing.It ensures the image processing steps used are completely

documented, and hence can be replicated. In general, the source code for all image processing functions are accessible for scrutiny and test.

It allows one to ensure numerical precision is maintained all the way through the enhancement process. Image processing algorithms available under MATLAB are likely to be more advanced than those available from other image processing applications.

4.5 System Design

4.5.1 Content Based Image Retrieval System

The earliest use of the term content-based image retrieval in the literature seems to have been by, to describe his experiments into automatic retrieval of images from a database by color and shape feature. The term has since been widely used to describe the process of retrieving desired images from a large collection on the basis of features (such as color and shape) that can be automatically extracted from the images themselves. The features used for retrieval can be either primitive or semantic, but the extraction process must be predominantly automatic. Retrieval of images by manually-assigned keywords is definitely not CBIR as the term is generally understood even if the keywords describe image content.

CBIR differs from classical information retrieval in that image databases are essentially unstructured, since digitized images consist purely of arrays of pixel intensities, with no inherent meaning. One of the key issues with any kind of image processing is the need to extract useful information from the raw data (such as recognizing the presence of particular shapes or textures) before any kind of reasoning about the image's contents is possible. Image databases thus differ fundamentally from text databases, where the raw material (words stored as ASCII character strings) has already been logically structured by the author . There is no equivalent of level 1 retrieval in a text database.

CBIR draws many of its methods from the field of image processing and computer vision, and is regarded by some as a subset of that field. It differs from these fields principally through its emphasis on the retrieval of images with desired characteristics from a collection of significant size. Image processing covers a much wider field, including image enhancement, compression, transmission, and interpretation. While there are grey areas (such as object recognition by feature analysis), the distinction between mainstream image analysis and CBIR is usually fairly clear-cut. An example may make

this clear. Many police forces now use automatic face recognition systems. Such systems may be used in one of two ways. Firstly, the image in front of the camera may be compared with a single individual's database record to verify his or her identity. In this case, only two images are matched, a process few observers would call CBIR. Secondly, the entire database may be searched to find the most closely matching images. This is a genuine example of CBIR.

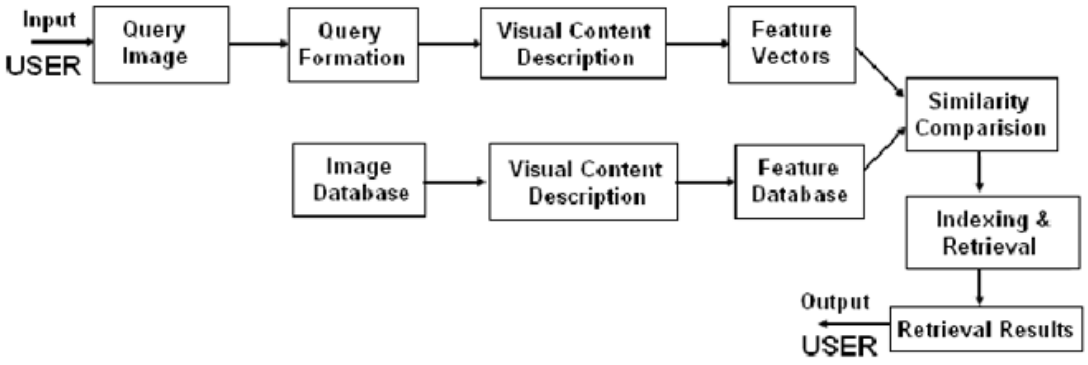

Fig: 4.4 Block Diagram of CBIR System.

The process of retrieving desired images from a large collection on the basis of features (such as color and shape) that can be automatically extracted from the images themselves. The features used for retrieval can be either primitive or semantic, but the extraction process must be predominantly automatic.

In typical Content-based image retrieval systems the visual contents of the images in the database are extracted and described by multi-dimensional feature vectors. The feature vectors of the images in the database form a feature database. To retrieve images, users provide the retrieval system with example images or sketched figures. The system then changes these examples into its internal representation of feature vectors. The similarities /distances between the feature vectors of the query example or sketch and those of the images in the database are then calculated and retrieval is performed with the aid of an indexing scheme. The indexing scheme provides an efficient way to search for the image database.

To modify the retrieval process in order to generate perceptually and semantically more meaningful retrieval results. In this chapter, we introduce these fundamental techniques for content-based image retrieval.

4.5.2 CBIR Definition

Content Based Image Retrieval is an application for retrieving the images from a huge set of image databases based on the image features such as color, texture and some other attributes. Here we take image feature as the index to that image and retrieve that particular image.

This project makes use of five methods to retrieve both Color and Gray scale images.

The methods used are as follows:

For Gray scale images-

- Columnar Mean.
- Diagonal Mean.
- Histogram Analysis.

For Color (RGB) images-

- RGB components.
- Retrieving similar images using Euclidean Distance.

Here we fix the dimension of the image to be 256X256 for the image analysis and feature extraction. If the input image is more than the specified dimension then we will resize it to 256X256. For Gray scale image analysis we have taken Post Gray Map (PGM) images and for color (RGB) image analysis we have taken JPG images.

The above mentioned methods are implemented in Matlab 7 and have been successfully run. In back end we use MySQL server.

Flow chart:

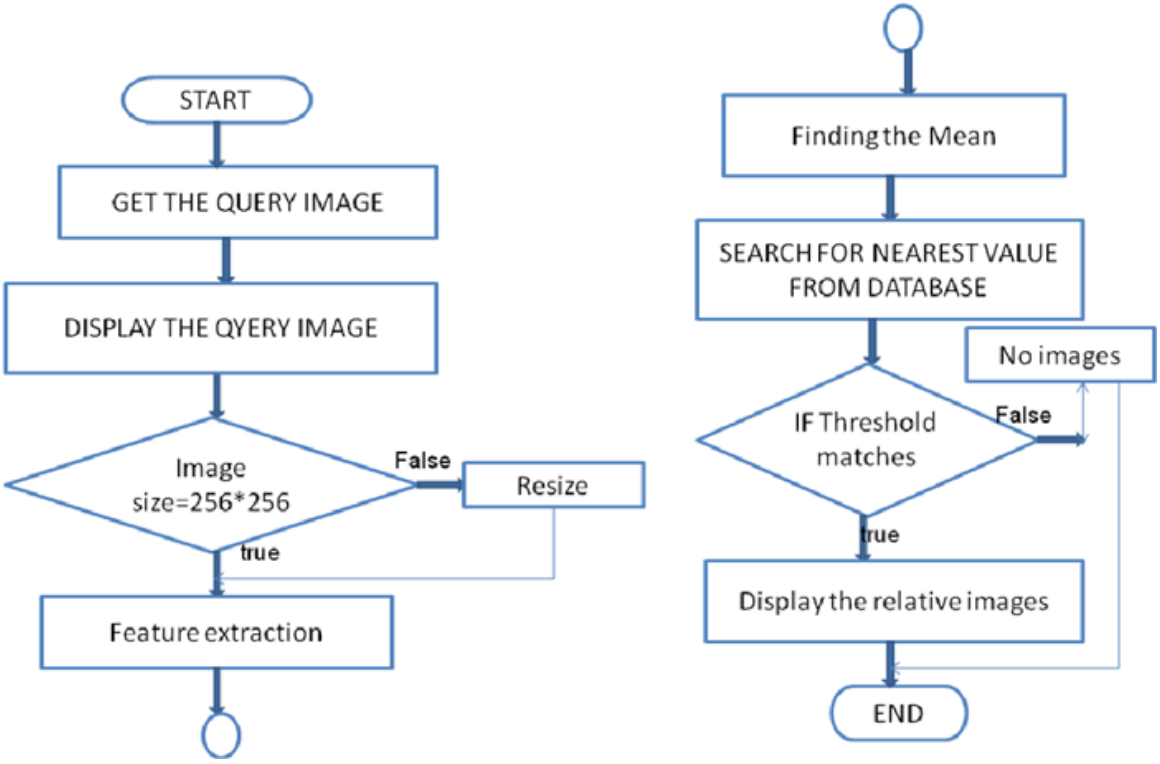

Fig: 4.5 Flow Chart for Image Retrieval

4.6 gray scale image analysis

4.6.1 Columnar Mean

Columnar Mean is one of the methodologies that we use in the CBIRS to retrieve gray map images. In this method we take only gray scale image for image analysis. In this approach we calculate the average (Empirical Mean or simply Mean) value of each column of the image (because image is stored as a matrix using standard Matlab matrix conventions) and make those values as the index for that image and are stored in the database. While retrieving the image from the database based on the input image, we calculate mean value of each column of the input image and will compare these values with that stored in the database, if there is a match then we will retrieve those images.

Empirical mean:

$$g = 1/MN \left(\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g_{mn} \right)$$

```
%Function to retrieve images using Columnar mean method

function retimgcol(path)
%opening a Database connection
conn = database('retsys','','');
%reading the input image
img=imread(path);
if((ndims(img)==3) | (size(img,3)==3))
 img=rgb2gray(img);
end;
dim=256;
re=imresize(img,[dim,dim],'bilinear');

%Calculating the Empirical Mean for each column
 for i=1:dim
 cmean(i)=mean(re(:,i));
 end;
```

4.6.2 Diagonal Mean

In this approach we calculate the Empirical Mean value of the pixels that lies on the principle diagonal of the image (because image is stored as a matrix using standard Matlab matrix conventions) and make that value as the index for that image and is stored in the database. While retrieving the image from the database based on the input image, we calculate mean value of diagonal elements of the input image and will compare these values with that stored in the database, if there is a match then those images are retrieved. The advantage of using this method is instead of taking mean value of all the 256 rows as the index, we take only one value as the index for that image. Hence the computational time is reduced as we need to match only one field in the database.

But the disadvantage of this approach is accuracy of the image retrieval is less and hence reducing the efficiency of the CBIRS.

```
%Function to retrieve images using diagonal mean method
 function retimgdia(path)
%read the input image
img=imread(path);
dim=256;
re=imresize(img,[dim,dim],'bilinear');
%calculate the mean value of the principle diagonal elements
 mean=mean2(me);
```

4.6.3 Histogram Analysis

The histogram is a summary graph showing a count of the data points falling in various ranges. The effect is a rough approximation of the frequency distribution of the data. The groups of data are called classes, and in the context of a histogram they are known as bins, because one can think of them as containers that accumulate data and "fill up" at a rate equal to the frequency of that data class. An image histogram is a chart that shows the distribution of intensities in an indexed or intensity image. The image histogram function **imhist** creates this plot by making n equally spaced bins, each representing a range of data values. It then calculates the number of pixels within each range.

Syntax for histogram

imhist(I, n) displays a histogram where n specifies the number of bins used in the histogram. n also specifies the length of the colorbar. If I is a binary image, n can only have the value 2.

[counts] = imhist(...) returns the histogram counts in counts and the bin locations in x.

The following example displays an image and a histogram based on 64 bins.

1. Read image and display it.

```
I = imread('flowers.pgm');
imshow(I);
```

2. Display histogram of image.

```
figure, imhist(I,64);
```

In this approach we calculate the histogram counts for the specified number of bins and

we index those count value to represent that image. Based on these count values we retrieve the images from the database.

The advantage of using histograms for image retrieval is, Histogram search characterizes an image by its color distribution, or histogram but the drawback of a global histogram representation is that information about object location, shape, and texture is discarded.

Fig: 4.6 Test image

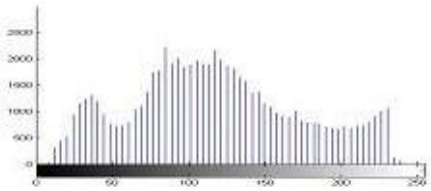

fig 4.7 Histogram bar

```
%Function to retrieve images using Histogram method
function retimghist(path,bins)
conn= database('retsys','','');
img=imread(path);
if((ndims(img)==3) | (size(img,3)==3))
img=rgb2gray(img);
end;
dim=256;
re=imresize(img,[dim,dim],'bilinear');
%calculating the Histogram counts in counts and the bin locations in x.
[counts,x]=imhist(re,bins);
%outputting the retrieved images
imshow(char(data(i)));
end;
end;
close(curs);
close(conn);
```

4.7 Color image analysis

4.7.1 RGB Components

An RGB image, sometimes referred to as a true color image, is stored in MATLAB as an m-by-n-by-3 data array that defines red, green, and blue color components for each individual pixel. RGB images do not use a palette. The color of each pixel is determined by the combination of the red, green, and blue intensities stored in each color plane at the pixel's location. Graphics file formats store RGB images as 24-bit images, where the red, green, and blue components are 8 bits each. This yields a potential of 16 million colors.
The precision with which a real-life image can be replicated has led to the commonly used term true color image.

To further illustrate the concept of the three separate color planes used in an RGB image, the code sample below creates a simple RGB image containing uninterrupted areas of red, green, and blue, and then creates one image for each of its separate color planes (red, green, and blue). It displays each color plane image separately, and also displays the original image.

- RGB=reshape(ones(64,1)*reshape(jet(64),1,192),[64,64,3]);
- R=RGB(:,:,1);
- G=RGB(:,:,2);
- B=RGB(:,:,3);
- imshow(R)

The following figure depicts an RGB image of class double.

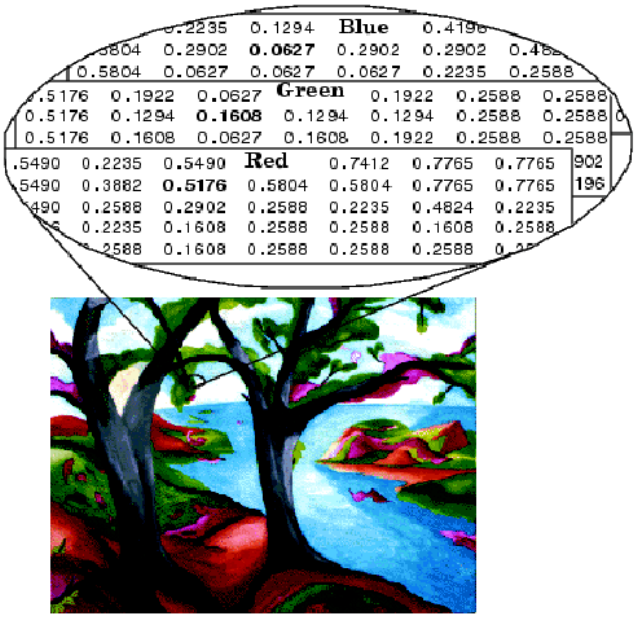

Fig: 4.8 RGB values of an Image

To further illustrate the concept of the three separate color planes used in an RGB image, the code sample below creates a simple RGB image containing uninterrupted areas of red, green, and blue, and then creates one image for each of its separate color planes (red, green, and blue). It displays each color plane image separately, and also displays the original image.

- RGB=reshape(ones(64,1)*reshape(jet(64),1,192),[64,64,3]);
- R=RGB(:,:,1);
- G=RGB(:,:,2);
- B=RGB(:,:,3);

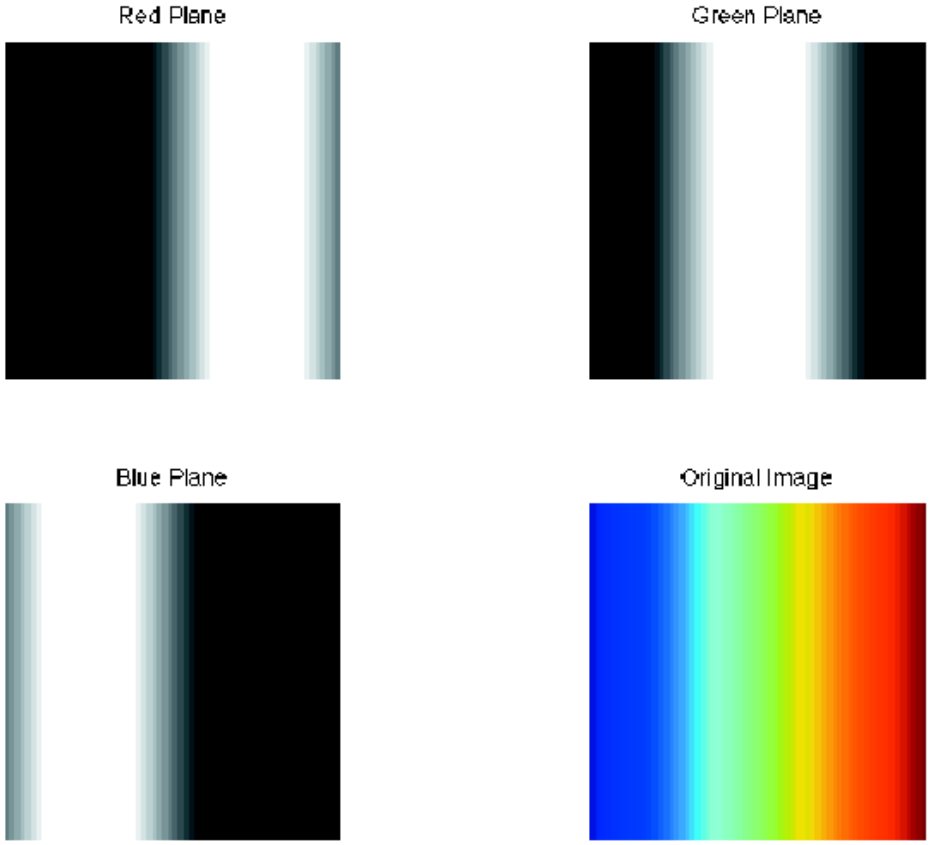

Fig: 4.9 Separate RGB plane

```
%Function to retrieve images using Histogram method
function retimgRGB(path)
conn=database('retsys','','');
img=imread(path);
if((ndims(img)~=3) | (size(img,3)~=3))
menu(' Input image must be RGB','OK');
```

```
else
re=imresize(img,[256,256],'bilinear');
%Extract Red, Green and Blue Components
red=re(:,:,1);
green=re(:,:,2);
blue=re(:,:,3);
redm=mean2(red);
greenm=mean2(green);
bluem=mean2(blue);
%Outputting the retrieved images.
imshow(path);
end;
end;
end;
close(conn);
```

4.7.2 Retrieving similar images using Euclidean Distance

Retrieval using global average RGB:

We use average RGB to calculate color similarity. Average RGB is to compute the average value in R, G, and B channel of each pixel in an image, and use this as a descriptor of an image for comparison purpose.

Notation:

Iàan image

wàwidth of an image.

hàheight of image.

I(x,y)àthe pixel of image I at row y and column x.

R(p), G(p), B(p)àthe red, green, and blue color component of pixel p.

ra, ga, baàthe average red, green and blue component of an image Ia.

d (Ia, Ib) àthe distance measure between Ia and Ib.

The average values of R, G, and B used for calculating the Euclidean distance is the same value which is used in the retrieval of images using RGB components for color images. Euclidean distance is a geometrical concept which takes into consideration the co-ordinate values of the pixel points between which the distance is to be found. This distance defines the position variance of two points in terms of pixel values which in case of image processing is the values of R, G, and B.

Three equations for computing the average R, G, B component of an image I

x=w, y=h

r= R(I(x,y)) / (w x h)

x=1, y=1

x=w, y=h

r= R(I(x,y)) / (w x h)

x=1, y=1

x=w, y=h

r= R(I(x,y)) / (w x h)

x=1, y=1

Here is the equation for distance measure of image I_a and I_b, we use the weighted Euclidean distance. The distance between two exact images will be 0 and the distance between two most dissimilar images (black and white) will be 1 or 255 depending on weather the range of RGB is from 0 to 1 or 0 to 255.

Formula used for calculating the Euclidean Distance is as follows:

$$d(I_a,I_b) = \left(\frac{(r_a-r_b)^2+(g_a-g_b)^2+(b_a-b_b)^2}{3}\right)^{1\backslash 2}$$

In this method we calculate the distance between the query image and candidate set images stored in the database, if the distance is within the already fixed threshold then we will retrieve those images as the similar images to the query images.

The advantage of this approach is that it is easy to implement and it also has a disadvantage that it consumes more computation time when the number of images in the database increases.

```
%function to retrieve images using euclidean distance formula
function retimgsegm(path,thres)
%reading the input image
img=imread(path);
%condition to test whether the input image is a color image(RGB)
if((ndims(img)~=3) | (size(img,3)~=3))
 menu(' Input image must be RGB','OK');
else
%calculating the red, green and blue components of an input image
ra=img(:,:,1);
ga=img(:,:,2);
```

```
ba=img(:,:,3);
%calculating the mean value of R G B components
ram=mean2(ra);
gam=mean2(ga);
bam=mean2(ba);
%displaying the retrieved images
imshow(char(curs.data(i,4)));
end;
i=i+1;
```

4.8 Database design

Database can be defined as a structured collection of records or data that is stored in a computer so that a program can consult it to answer queries. The records retrieved in answer to queries become information that can be used to make decisions. The computer program used to manage and query a database is known as a database management system (DBMS).

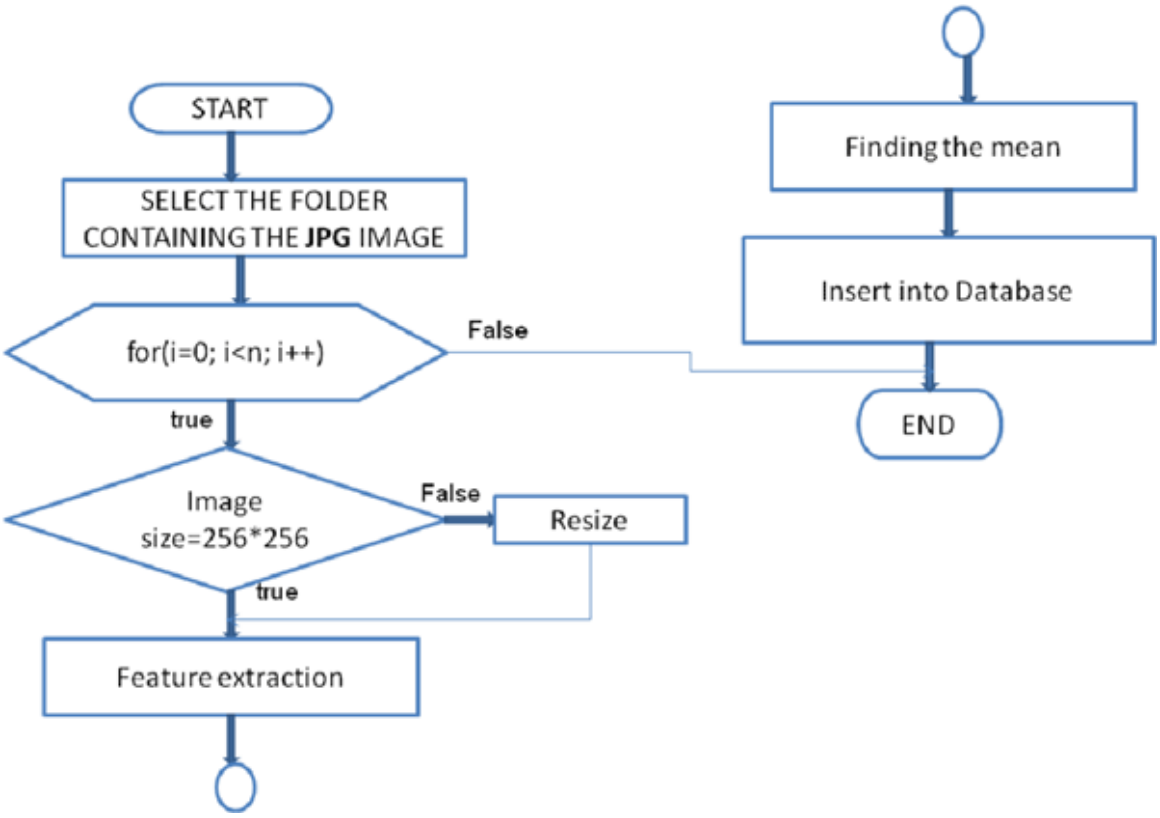

Fig: 4.10 Flowchart for inserting values into a Database.

While using CBIR system or any system which is designed such that comparison is to be

performed with an already existing database then database should be a standard database, so some of the standard databases used for CBIR in different field are as follows:

1. Database of Astronomical test images provided by IAPR technical committee 13.
2. C. A. Glaseby and G. W. Horgan: Image Analysis for the Biological Sciences (John Wiley, 1995) this database is provided from books
3. National Design Repository over 55,000 3D CAD and solid models of (mostly) mechanical/machined engineering designs. (Geometric & Intelligent Computing Laboratory / Drexel University)
4. UNC's 3D image database many images (Format: GIF)
5. AT&T: The Database of Faces (formerly 'The ORL Database of Faces') (Format: PGM)
6. Caltech Image Database 450 frontal face images of 27 or so unique people. (Format: GIF)
7. CMU PIE Database A database of 41,368 face images of 68 people captured under 13 poses, 43 illuminations conditions, and with 4 different expressions.
8. CMU VASC Image Database Images, sequences, stereo pairs (thousands of images) (Format: Sun Raster image)
9. CEDAR CDROM database of handwritten words, ZIP Codes, Digits and Alphabetic characters (Format: unknown)
10. NIST Fingerprint and handwriting datasets - thousands of images (Format: unknown)
11. El Salvador Atlas of Gastrointestinal Video Endoscopy Images and Videos of his-res of studies taken from Gastrointestinal Video endoscopy. (Format: jpg, mpg, gif)
12. The Mammographic Image Analysis Society (MIAS) mini-database.
13. Mammography Image Databases 100 or more images of mammograms with ground truth. Additional images available by request, and links to several other mammography databases are provided. (Format: homebrew)
14. Optic flow: the Barron and Fleet study.
15. Optical flow test image sequences 6 synthetic and 4 real image sequences (Format: Sun Raster image)
16. Particle image sequences real and synthetic image sequences used for testing a Particle Image Velocimetry application. These images may be used for the test of

optical flow and image matching algorithms. (Format: pgm (raw)) (LIMSI-CNRS/CHM/IMM/vision / LIMSI-CNRS)

17. Groningen Natural Image Database 4000+ 1536x1024 (16 bit) calibrated outdoor images (Format: homebrew)
18. IEN Image Library 1000+ images, mostly outdoor sequences (Format: raw, ppm)
19. University of Oulu wood and knots database: 1000+ color images, including classification. (Format: PPM)
20. In our project we have created a database by name "retsys", in that database we have created tables for each methods.

Syntax to create database:

create database retsys;

Example Query to create table in the database:

create table RGB(red dec(10,4), green dec(10,4),blue dec(10,4),path varchar(70));

RGB- Table name.

Red, green, blue & path- Fields in the Table RGB.

dec(10,4) & varchar(100)- Datatypes in MySql.

conn=database ('retsys','','');

Connects a MATLAB session to a database via an ODBC driver, returning the connection object to conn. The data source to which we are connecting is 'retsys'.

4.9 Graphic User Interface

It is an acronym for GUI. User always wants a friendly environment so that they can easily and effectively use the system without actually going into the finer details of the working. So, to create such a user friendly platform for the system we need Graphic User Interface where all the functionalities of the system is available for the user in graphics form and as we know user always find easier to understand and comprehend images and graphics rather than in textual form.

The graphical user interface (GUI) is an important part of software development. The designing of the GUI have to solve the following problems: learning time, speed of performance, rate of errors by users, retention over time, and subjective satisfaction .This software is, at the moment, intended to be used only for testing purposes. The most important property of this software is that the results of different test queries can be seen quickly and the results can be saved safely on a disk. Thus the visual layout is not as important as in case of a commercial software product.

In the GUI developed for our project first we ask user whether they want to search or insert into database as we have given option to user in which they can also add images and values to a database. Once they select anyone option they are directed to a new window where they can select anyone of the option for either grayscale or color from available 5 different options.

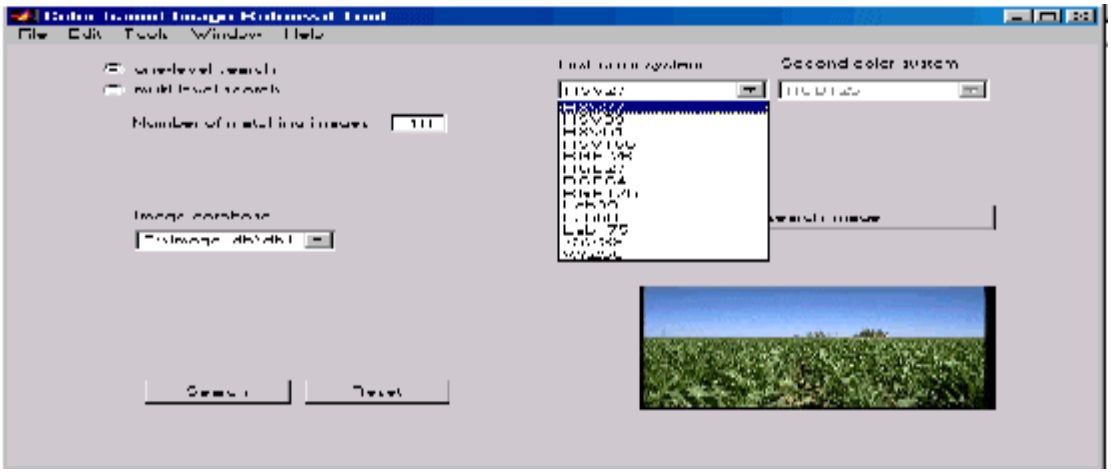

Fig: 4.11 A General GUI

CHAPTER 5

PSEUDOCODE

5.1 PSEUDOCODE FOR HSV HISTOGRAM

```
function hsvColorHistogram = hsvHistogram(image)
% input: image to be quantized in hsv color space into 8x2x2 equal bins
% output: 1x32 vector indicating the features extracted from hsv color
% space[rows, cols, numOfBands] = size(image);
% totalPixelsOfImage = rows*cols*numOfBands;
image = rgb2hsv(image);
% split image into h, s & v planes
h = image(:, :, 1);
s = image(:, :, 2);
v = image(:, :, 3);
% quantize each h,s,v equivalently to 8x2x2
% Specify the number of quantization levels.
thresholdForH = multithresh(h, 7);  % 7 thresholds result in 8 image levels
thresholdForS = multithresh(s, 1);  % Computing one threshold will quantize ..
% the image into three discrete levels
thresholdForV = multithresh(v, 1);  % 7 thresholds result in 8 image levels
seg_h = imquantize(h, thresholdForH); % apply the thresholds to obtain segmented image
seg_s = imquantize(s, thresholdForS); % apply the thresholds to obtain segmented image
seg_v = imquantize(v, thresholdForV); % apply the thresholds to obtain segmented image
% quantize each h,s,v to 8x2x2
% Specify the number of quantization levels.
numberOfLevelsForH = 8;
numberOfLevelsForS = 2;
numberOfLevelsForV = 2;
% Find the max.
maxValueForH = max(h(:));
maxValueForS = max(s(:));
maxValueForV = max(v(:));
```

```
% create final histogram matrix of size 8x2x2
hsvColorHistogram = zeros(8, 2, 2);
% create col vector of indexes for later reference
index = zeros(rows*cols, 3);
% Put all pixels into one of the "numberOfLevels" levels.
count = 1;
for row = 1:size(h, 1)
   for col = 1 : size(h, 2)
      quantizedValueForH(row, col) = ceil(numberOfLevelsForH *
h(row,col)/maxValueForH);
      quantizedValueForS(row, col) = ceil(numberOfLevelsForS *
s(row,col)/maxValueForS);
      quantizedValueForV(row, col) = ceil(numberOfLevelsForV *
v(row,  col)/maxValueForV);
 % keep indexes where 1 should be put in matrix hsvHist
      index(count, 1) = quantizedValueForH(row, col);
      index(count, 2) = quantizedValueForS(row, col);
      index(count, 3) = quantizedValueForV(row, col);
      count = count+1;
   end
end

% put each value of h,s,v to matrix 8x2x2
% (e.g. if h=7,s=2,v=1 then put 1 to matrix 8x2x2 in position 7,2,1)
for row = 1:size(index, 1)
   if (index(row, 1) == 0 || index(row, 2) == 0 || index(row, 3) == 0)
      continue;
   end
   hsvColorHistogram(index(row, 1), index(row, 2), index(row, 3)) = ...
      hsvColorHistogram(index(row, 1), index(row, 2), index(row, 3)) + 1;
end
% normalize hsvHist to unit sum
hsvColorHistogram = hsvColorHistogram(:)';
hsvColorHistogram = hsvColorHistogram/sum(hsvColorHistogram);
```

```
% clear workspace
% clear('row', 'col', 'count', 'numberOfLevelsForH', 'numberOfLevelsForS', ...
%     'numberOfLevelsForV', 'maxValueForH', 'maxValueForS', 'maxValueForV', ...
%     'index', 'rows', 'cols', 'h', 's', 'v', 'image', 'quantizedValueForH', ...
%     'quantizedValueForS', 'quantizedValueForV');
%%
% figure('Name', 'Quantized leves for H, S & V');
% subplot(2, 3, 1);
% imshow(seg_h, []);
% subplot(2, 3, 2);
% imshow(seg_s, []);
% title('Quatized H,S & V by matlab function imquantize');
% subplot(2, 3, 3);
% imshow(seg_v, []);
% subplot(2, 3, 4);
% imshow(quantizedValueForH, []);
% subplot(2, 3, 5);
% imshow(quantizedValueForS, []);
% title('Quatized H,S & V by my function');
% subplot(2, 3, 6);
% imshow(quantizedValueForV, []);
%%
End
```

5.2 PSEUDOCODE FOR L1 DISTANCE

```
function L1( queryImageFeatureVector, dataset,numOfReturnedImages)
% input:
%   numOfReturnedImages : num of images returned by query
%   queryImageFeatureVector: query image in the form of a feature vector
%   dataset: the whole dataset of images transformed in a matrix of
%   features
%
% output:
```

```
%   plot: plot images returned by query
% extract image fname from queryImage and dataset
query_image_name = queryImageFeatureVector(:, end);
dataset_image_names = dataset(:, end);
queryImageFeatureVector(:, end) = [];
dataset(:, end) = [];
% compute manhattan distance
manhattan = zeros(size(dataset, 1), 1);
for k = 1:size(dataset, 1)
%     manhattan(k) = sum( abs(dataset(k, :) - queryImageFeatureVector) );
    % ralative manhattan distance
    manhattan(k) = sum( abs(dataset(k, :) - queryImageFeatureVector) ./ ( 1 + dataset(k, :)
+ queryImageFeatureVector ) );
end

% add image fnames to manhattan
manhattan = [manhattan dataset_image_names];
% sort them according to smallest distance
[sortedDist indx] = sortrows(manhattan);
sortedImgs = sortedDist(:, 2);
 % mindist=find(sortedDist(:,1)<3);
% numOfReturnedImages=numel(mindist);
% clear axes
arrayfun(@cla, findall(0, 'type', 'axes'));

% display query image
str_name = int2str(query_image_name);
queryImage = imread( strcat('Images\0', str_name, '.jpg') );
subplot(3, 7, 1);
imshow(queryImage, []);
title('Query Image', 'Color', [1 0 0]);

% dispaly images returned by query
for m = 1:numOfReturnedImages
```

```
    img_name = sortedImgs(m);
    if img_name ~=sortedImgs(m+1)
    img_name = int2str(img_name);
    str_name = strcat('images\0', img_name, '.jpg');
    returnedImage = imread(str_name);
    subplot(3, 7, m+1);
    imshow(returnedImage, []);
    end
end

end
```

5.3 PSEUDOCODE FOR HARRIS

```
% HARRIS - Harris corner detector
%
% Usage:  [cim, r, c] = harris(im, sigma, thresh, radius, disp)
%
% Arguments:
%im     - image to be processed.
%sigma  - standard deviation of smoothing Gaussian. Typical
%values to use might be 1-3.
%thresh - threshold (optional). Try a value ~1000.
%radius - radius of region considered in non-maximal
%suppression (optional). Typical values to use might
%be 1-3.
%disp   - optional flag (0 or 1) indicating whether you want
%to display corners overlayed on the original
%image. This can be useful for parameter tuning.
%
% Returns:
%cim    - binary image marking corners.
%r      - row coordinates of corner points.
%c      - column coordinates of corner points.
%
```

```
% If thresh and radius are omitted from the argument list 'cim' is returned
% as a raw corner strength image and r and c are returned empty.


% Reference:
% C.G. Harris and M.J. Stephens. "A combined corner and edge detector",
% Proceedings Fourth Alvey Vision Conference, Manchester.
% pp 147-151, 1988.
%
% Author:
% Peter Kovesi
% Department of Computer Science & Software Engineering
% The University of Western Australia
% pk@cs.uwa.edu.au  www.cs.uwa.edu.au/~pk
%
% March 2002
function [cim, r, c] = harris(im, sigma, thresh, radius, disp)
   error(nargchk(2,5,nargin));
   dx = [-1 0 1; -1 0 1; -1 0 1]; % Derivative masks
   dy = dx'
   Ix = conv2(double(im), dx, 'same');    % Image derivatives
   Iy = conv2(double(im), dy, 'same');
   % Generate Gaussian filter of size 6*sigma (+/- 3sigma) and of
   % minimum size 1x1.
   g = fspecial('gaussian',max(1,fix(6*sigma)), sigma);
   Ix2 = conv2(Ix.^2, g, 'same'); % Smoothed squared image derivatives
   Iy2 = conv2(Iy.^2, g, 'same');
   Ixy = conv2(Ix.*Iy, g, 'same');
   cim = (Ix2.*Iy2 - Ixy.^2)./(Ix2 + Iy2 + eps); % Harris corner measure
   % Alternate Harris corner measure used by some.  Suggested that
   % k=0.04 - I find this a bit arbitrary and unsatisfactory.
%   cim = (Ix2.*Iy2 - Ixy.^2) - k*(Ix2 + Iy2).^2;
   if nargin > 2   % We should perform nonmaximal suppression and threshold
         % Extract local maxima by performing a grey scale morphological
         % dilation and then finding points in the corner strength image that
```

```
        % match the dilated image and are also greater than the threshold.
        sze = 2*radius+1;                   % Size of mask.
        mx = ordfilt2(cim,sze^2,ones(sze)); % Grey-scale dilate.
        cim = (cim==mx)&(cim>thresh);       % Find maxima.
        [r,c] = find(cim);                  % Find row,col coords.
        if nargin==5 & disp      % overlay corners on original image
           figure, imagesc(im), axis image, colormap(gray), hold on
           plot(c,r,'ys'), title('corners detected');
        end
    else  % leave cim as a corner strength image and make r and c empty.
        r = []; c = [];
    end
```

CHAPTER 6

RESULT AND DISCUSSION

This CBIR system is implemented in Matlab and SQL is used as a backend tool for database creation and management. When user starts using this system first, they are directed to GUI window where they get an option to select whether they want to search or they want to insert into database which can be seen as shown in fig 4.1:

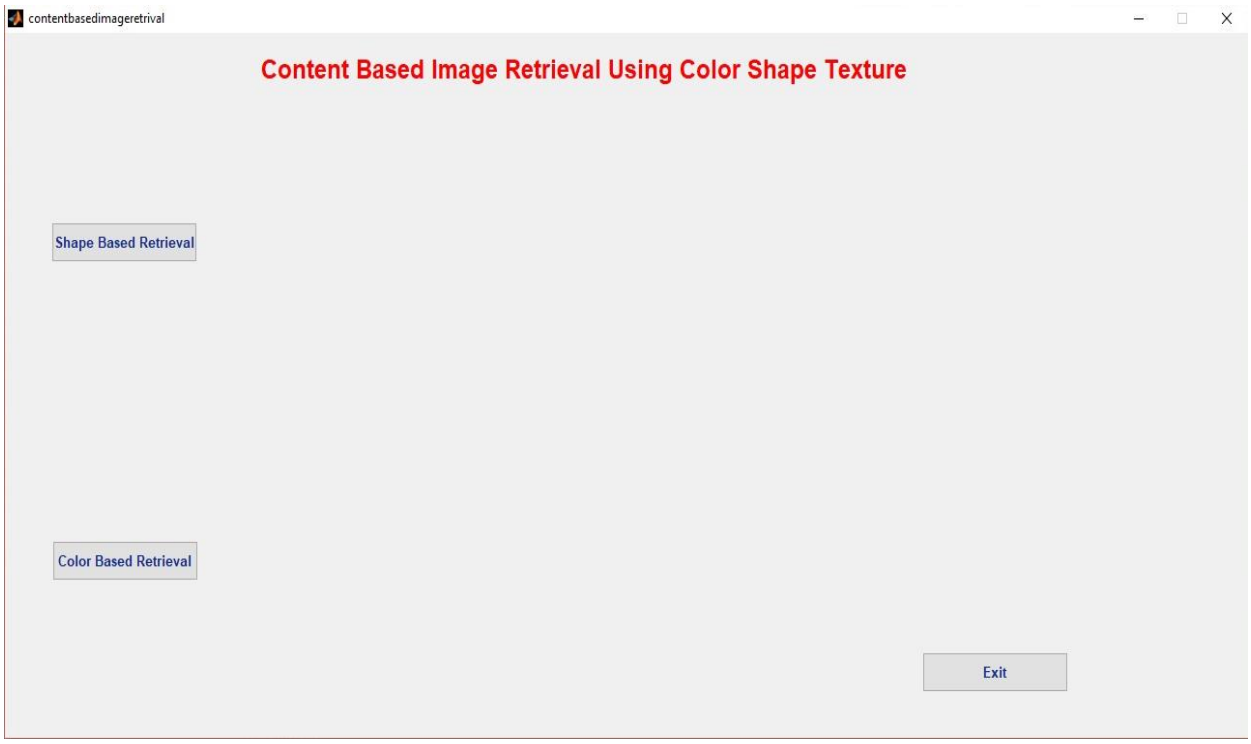

Fig: 6.1 Project 1st window .

Once user selects their option they will be directed to a new window according to their selection. If they select shape baased retrieval it moves to shape based window if they select color based retrieval it goes to color based window.

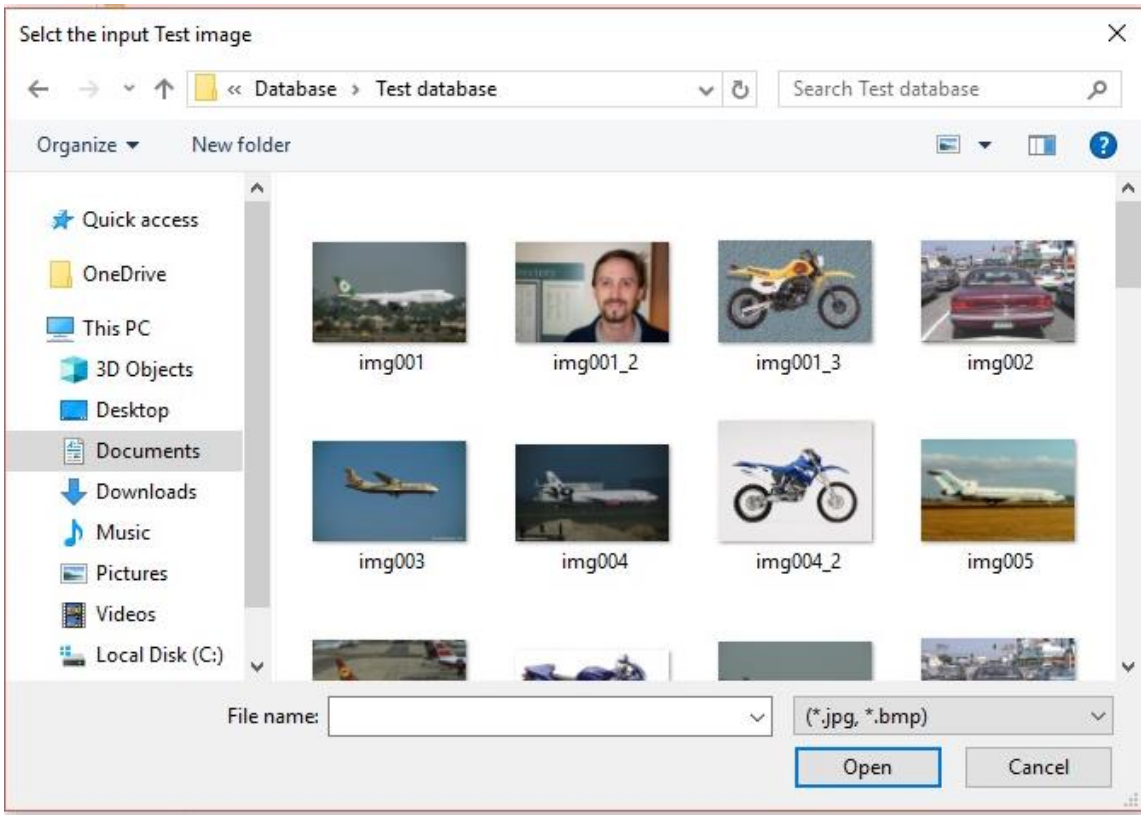

Fig: 6.2 selecting the image

In both the windows they will have the option of selecting the one of the implemented methods. In the search window they will have an option of browsing the image they want to search and that image will be displayed on the screen.

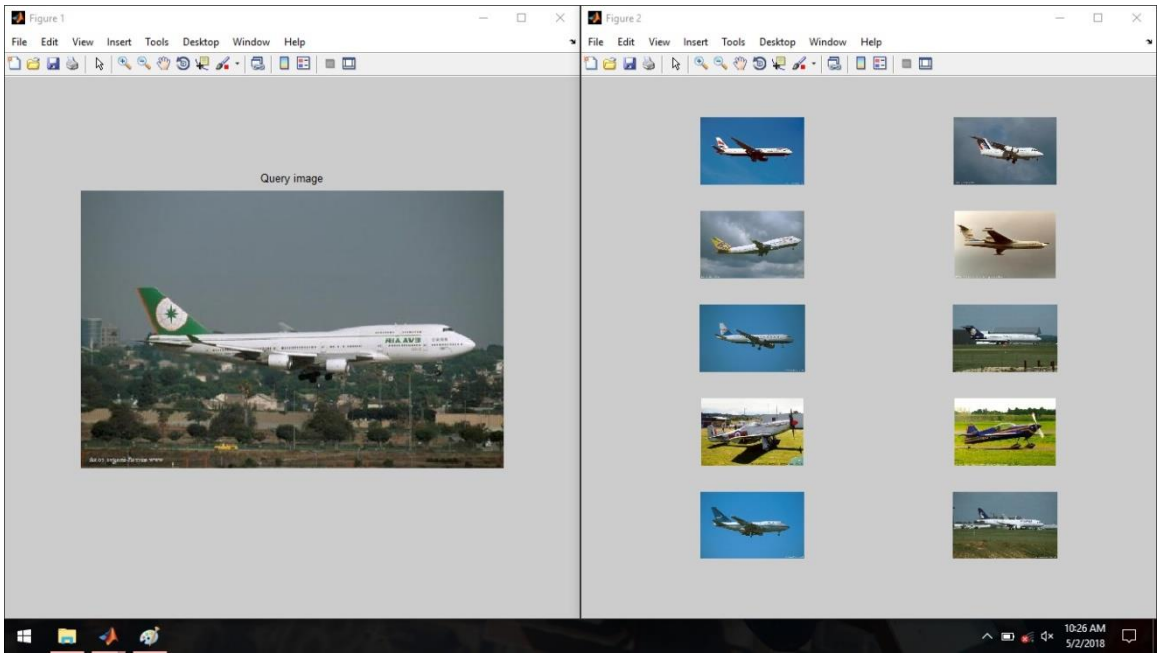

Fig: 6.3 Output image based on shape features

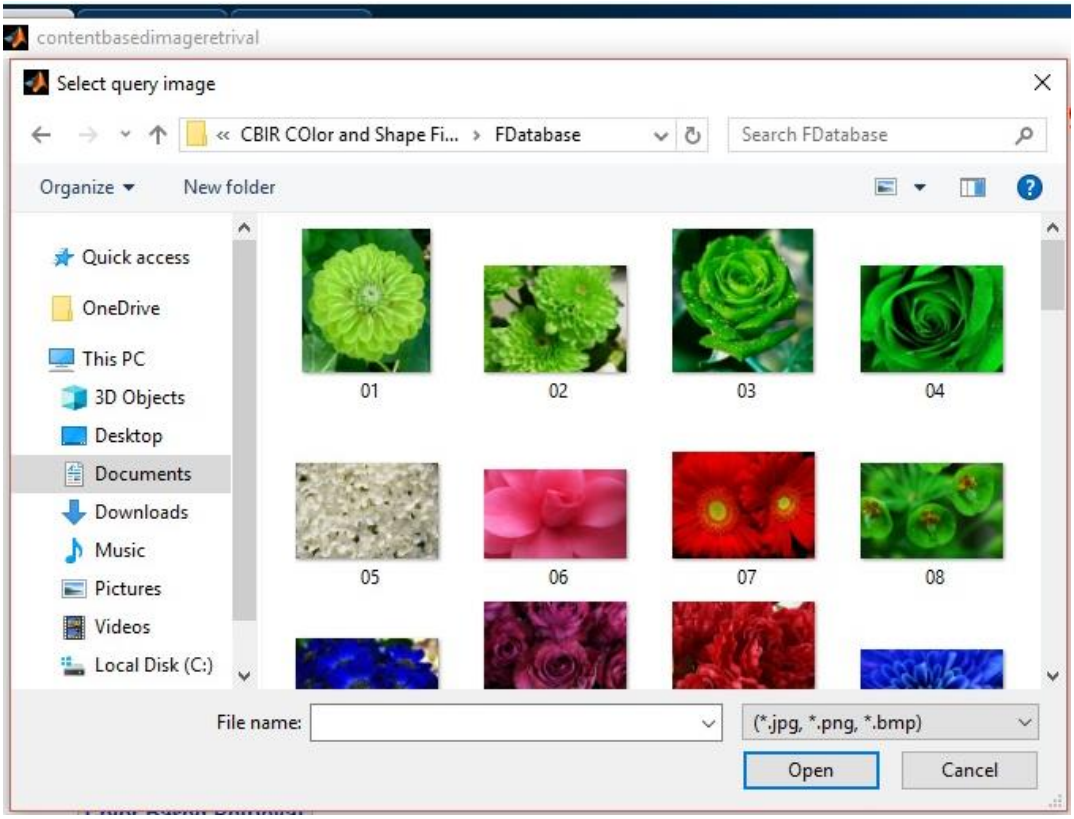

Fig: 6.4 Selecting the image based on color

Once user browses the image they will have to select one of the options for searching either for color or shape features. Once they select the option the relevant images will display.

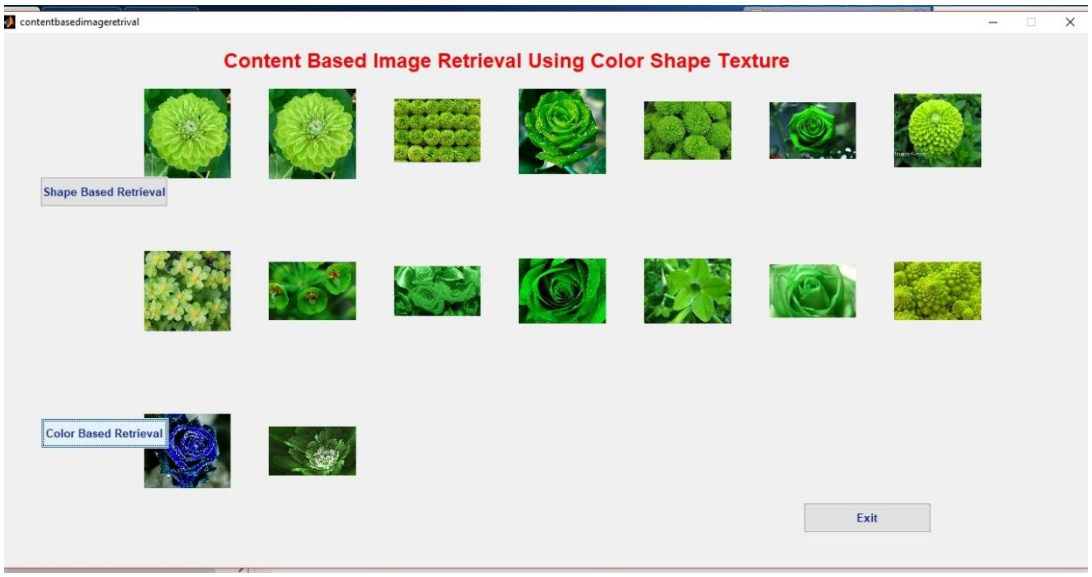

Fig: 6.5 Output image for color based retrieval.

We have created our own databases of the images taken from standard database of Caltech and found out the efficiency of different methods for the comparative analysis of all the methods employed. The comparison table is as shown:

METHODS USED	**EFFICIENCY ACHIEVED**			
	Data Base 1	Data Base 2	Data Base 3	Data Base 4
COLUMNAR MEAN	48%	44%	46%	41%
DIAGONAL MEAN	38%	34%	36%	31%
HISTOGRAM	36%	35%	38%	29%
RGB COMPONENTS	44%	42%	40%	42%
EUCLIDEAN DISTANCE	48%	44%	46%	41%

Table: 6.1 Comparison table (derived data bases).

METHODS USED	**EFFICIENCY ACHIEVED**			
	Caltech database	IAPR database	Groningen Image Database	AT&T database
COLUMNAR MEAN	18%	14%	16%	11%
DIAGONAL MEAN	16%	15%	12%	13%
HISTOGRAM	19%	11%	12%	12%
RGB COMPONENTS	16%	14%	18%	14%
EUCLIDEAN DISTANCE	17%	15%	19%	17%

Table: 6.2 Comparison table (standard databases).

The above table shows the comparative result of all the methodologies used in the project for standard databases. Caltech database is used for astronomical images, in the same way Groningen database is used for natural images where we have achieved better efficiency compared to aircraft database sets derived from AT&T database.

6.1 Applications of CBIR

6.1.1 Societal factors:

It is a truism to observe that images are currently used in all walks of life. The influence of television and video games in today's society is clear for all to see. The commonest single reason for storing, transmitting and displaying images is probably for recreational use, though this category includes a wide variety of different attitudes and interaction styles, from passively watching the latest episode of a soap opera to actively analyzing a tennis star's shots in the hope of improving one's own game. Images are increasingly used to convey information, in areas as diverse as map-making, weather forecasting and mail-order shopping, and to persuade or convey a mood, as in advertising. They can also be appreciated in their own right, as works of art.

A detailed sociological study of image use would be out of place in this report, particularly as there is currently little evidence for the existence of different user communities with different needs. Most individuals interact with images in different ways at different times, perhaps spending an hour in an art gallery one day, and watching a sports video the next. Trying to categorize such behavior by user type does not seem very useful.

6.1.2 Industrial factors:

In the realm of professional image use, the situation is rather different. While there are certainly differences in style between individual design engineers, for example, the nature of the design process imposes a number of inescapable constraints within which all engineers must work. Hence it is possible to generalize to some extent about the way images are used by different professions. Since this report is primarily concerned with image storage and retrieval, it makes sense to limit our discussion by concentrating on uses which involve stored collections of images in some way.

Some groups of people use images in their job on a daily basis, such as graphic designers and illustrators, whilst others may never be required to use them, such as bank managers and accountants. There is a wide range of professions lying between these two

extremes, including medicine and law. Other groups of workers, such as librarians and museum curators, may be required to find images on behalf of clients rather than for themselves. It is impossible to give a full picture here of the uses being made of visual information. The following examples should be interpreted as being merely a snapshot of the situation.

6.1.3 Crime prevention:

The police use visual information to identify people or to record the scenes of crime for evidence; over the course of time, these photographic records become a valuable archive. In the UK, it is common practice to photograph everyone who is arrested and to take their fingerprints. The photograph will be filed with the main record for the person concerned, which in a manual system is a paper-based file. In a computer-based system, the photograph will be digitized and linked to the corresponding textual records. Until convicted, access to photographic information is restricted and, if the accused is acquitted, all photographs and fingerprints are deleted. If convicted, the fingerprints are passed to the National Fingerprint Bureau. Currently, there is a national initiative investigating a networked Automated Fingerprint Recognition system involving BT and over thirty regional police forces. Other uses of images in law enforcement include face recognition, DNA matching, shoe sole impressions, and surveillance systems. The Metropolitan Police Force in London is involved with a project which is setting up an international database of the images of stolen objects

6.1.4 Medicine:

The medical and related health professions use and store visual information in the form of X-rays, ultrasound or other scanned images, for diagnosis and monitoring purposes. There are strict rules on confidentiality of such information. The images are kept with the patients' health records which are, in the main, manual files, stored by unique identifier (NI number). Visual information, provided it is rendered anonymous, may be used for research and teaching purposes. Much of the research effort related to images is undertaken in the medical physics area. Aspects of concern include effective image processing (e.g. boundary/feature detection) systems which aid the practitioner in detecting and diagnosing lesions and tumors and tracking progress/growth.

6.1.5 Fashion and graphic design:

Imagery is very important for graphic, fashion and industrial designers. Visualization seems to be part of the creative process. Whilst there will be individual differences in the way designers approach their task, many use images of previous designs in the form of pictures, photographs and graphics, as well as objects and other visual information from the real world, to provide inspiration and to visualize the end product. 2-D sketches, and, increasingly, 3-D geometric models are used to present ideas to clients and other colleagues. There is also a need to represent the way garments hang and flow.

Publishing and advertising. Photographs and pictures are used extensively in the publishing industry, to illustrate books and articles in newspapers and magazines. Many national and regional newspaper publishers maintain their own libraries of photographs, or will use those available from the Press Association, Reuters and other agencies. The photographic collections will be indexed and filed under, usually, broad subject headings (e.g. local scenes, buildings or personalities as well as pictures covering national and international themes). Increasingly, electronic methods of storage and access are appearing, alongside developments in automated methods of newspaper production, greatly improving the speed and accuracy of the retrieval process. Advertisements and advertising campaigns rely heavily on still and moving imagery to promote the products or services. The growth of commercial stock photograph libraries, such as Getty Images and Corbis, reflects the lucrative nature of the industry.

6.1.6 Architectural and engineering design:

Photographs are used in architecture to record finished projects, including interior and exterior shots of buildings as well particular features of the design. Traditionally these photographs will be stored as hardcopy or in slide format, accessible by, say, project number and perhaps name, and used for reference by the architects in making presentations to clients and for teaching purposes. Larger architects' practices with more ample resources, have introduced digital cameras and the electronic storage of photographs.

The images used in most branches of engineering include drawings, plans, machine parts, and so on. Computer Aided Design (CAD) is used extensively in the design process. A prime need in many applications is the need to make effective use of standard parts, in order to maintain competitive pricing [Bradley et al, 1994].

CONCLUSION

The extent to which CBIR technology is currently in routine use is clearly still very limited. In particular, CBIR technology has so far had little impact on the more general applications of image searching, such as journalism or home entertainment. Only in very specialist areas such as crime prevention has CBIR technology been adopted to any significant extent. This is no coincidence – while the problems of image retrieval in a general context have not yet been satisfactorily solved, the well-known artificial intelligence principle of exploiting natural constraints has been successfully adopted by system designers working within restricted domains where shape, color or texture features play an important part in retrieval.

CBIR at present is still very much a research topic. The technology is exciting but immature, and few operational image archives have yet shown any serious interest in adoption. The crucial question that this report attempts to answer is whether CBIR will turn out to be a flash in the pan, or the wave of the future. Our view is that CBIR is here to stay. It is not as effective as some of its more ardent enthusiasts claim – but it is a lot better than many of its critics allow, and its capabilities are improving all the time. Most current keyword-based image retrieval systems leave a great deal to be desired. In hard-nosed commercial terms, only one application of CBIR (video asset management) appears our view is that CBIR is here to stay. It is not as effective as some of its more ardent enthusiasts claim – but it is a lot better than many of its critics allow, and its capabilities are improving all the time. And as we argue in section **Error! Reference source not found.** above, most current keyword-based image retrieval systems leave a great deal to be desired. In hard-nosed commercial terms, only one application of CBIR (video asset management) appears to be cost-effective – but few conventional image management systems could pass the test of commercial viability either.

The process of designing of **CBIR** system has been successfully carried out and the expected outcome is achieved. The main functions that a **CBIRS** should perform are:

- Constructing feature vectors from the image based on its content and storing it in the database.
- Similarity comparison and segmentation.
- Retrieving the images based on the feature vectors.

The above mentioned functions are thoroughly checked using software called MATLAB 7.0. Also the design is very simple and easy to implement.

FUTURE ENHANCEMENTS

Developments and studies are going on for further improvements in design and performance of "CONTENT BASED IMAGE RETRIEVAL SYSTEMS".In our project we have only done color analysis and the information about object location, shape, and texture is discarded. Thus this project showed that images retrieved by using the above mentioned methods may not be semantically related even though they share similar color distribution in some results.

In the future enhancements we can implement:

- Shape and Texture analysis.
- Color Image histogram.
- Image ranking in Euclidean Distance method.

REFERENCES

[1] Y. A. Aslandogan and C. T. Yu, "Techniques and Systems for Image and Video Retrieval," IEEE Transactions on Knowledge and Data Engineering, Vol. 11, Issue 1, pp. 56-63, Jan/Feb 1999.

[2] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-Based Image Retrieval at the End of the Early Years," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, Issue 12, pp. 1349-1380, December 2000.

[3] M. J. Swain and D. H. Ballard, „„Color indexing"", International Journal of Computer Vision, vol. 7, Issue 1, pp. 11-32, 1991.

[4] Irena Valova, Boris Rachev and Michael Vassilakopoulos, "Optimization of the Algorithm for Image Retrieval by Color Features", International Conference on Computer Systems and Technologies- CompSysTech", pp 1-4, 2006.

[5] Sarfraz and M. Ridha "Content-Based Image Retrieval Using Multiple Shape Descriptors", IEEE/ACS International Conference On Computer Systems and Applications, pp. 730-737, 2007.

[6] G. Pass and R. Zabih, "Histogram Refinement for Content-Based Image Retrieval," 3rd IEEE Workshop on Applications of Computer Vision, pp. 96-102, 1996.

[7] A. Vellaikal and C. C. J. Kuo, „„Content Based Image Retrieval using Multiresolution Histogram Representation"", SPIE - Digital Image Storage and Archiving Systems, Vol. 2606, pp. 312-323, 1995.

[8] H. J. Zhang, Y. Gong, C. Y. Low and S. W. Smoliar, „„Image Retrieval Based on Color Feature: An Evaluation Study"", SPIE – Digital Image Storage and Archiving Systems, vol. 2606, pp. 212-220, 1995.

[9] Shamik Sural, Gang Qian and Sakti Pramanik, "Segmentation and Histogram Generation Using the HSV Color Space for Image Retrieval", International Conference on Image processing, Vol. 2, pp. 589-592, 2002.

[10] Xiaoqian Xu, Dah-Jye Lee, Sameer Antani, and L. Rodney Long, "A Spine X-Ray Image Retrieval System Using Partial Shape Matching", IEEE Transactions On Information Technology In Biomedicine, Vol. 12, Issue 1, pp. 100-108, January 2008.

[11] Amit Jain, R. Muthuganapathy, and Karthik Ramani, "Content-Based Image Retrieval Using Shape and Depth from an Engineering Database", Proceedings of the 3rd international conference on Advances in visual computing, Vol. Part II, pp. 255-264, 2007.

[12] Rafel C. Gonzalez and Richard E. Woods, “Digital Image Processing”, Second Edition, Pearson Education Asia, 2005.

[13] Aleksandra Mojsilovic, Jianying Hu and Emina Soljanin, “Extraction of Perceptually Important Colors and Similarity Measurement