CONTENT BASED IMAGE RETRIEVAL USING COLOR AND SPATIAL FEATURES

A DISSERTATION
SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF THE DEGREE OF

MASTER OF ENGINEERING
(COMPUTER TECHNOLOGY & APPLICATIONS)

BY
PRABHANJANA KUMAR NALLANI
College Roll No. 09/CTA/04
Delhi University Roll No. 8511

Under the guidance of
Mrs. RAJNI JINDAL

DEPARTMENT OF COMPUTER ENGINEERING
DELHI COLLEGE OF ENGINEERING
UNIVERSITY OF DELHI

JUNE-2006

CERTIFICATE

It is to certify that the work that is being presented in this dissertation entitled **"Content Based Image Retrieval Using Color and Spatial features"**, in partial fulfillment of the requirement for the award of the degree of **Master of Engineering** in **Computer Technology and Applications** submitted by **Prabhanjana Kumar Nallani** (09/CTA/04) to the Department of Computer Engineering, Delhi College of Engineering, is the record of the student's own work carried out under my supervision and guidance.

Mrs. Rajni Jindal
Dept. of Computer Engineering,
Delhi College of Engineering,
Delhi

ACKNOWLEDGMENTS

This dissertation, as we see today in its present shape is an outcome of persistent effort and a great deal of dedication and it has drawn intellectual and moral support from various people within the institution.

I am extremely indebted to my honorable guide **Mrs. Rajni Jindal** for her invaluable support and guidance throughout the development of this dissertation. It is a matter of great pride for me to have worked under her, which in itself was a source of inspiration for me to complete the dissertation with great enthusiasm and determination. I also give special thanks to her for dedicating her valuable time whenever I needed to discuss thesis related works without any delay.

I would like to thank **Dr. Goldie Gabrani**, Professor and Head, department of Computer Engineering, for providing facilities for this dissertation.

I am also thankful to **Prof. D. Roy Choudhury** and **Mr. Rajiv Kumar** who helped me during the course of this dissertation.

I take this opportunity to thank all members of Computer Engineering Department for their valuable help in this thesis.

I am also thankful to all my friends who continuously helped and motivated me during the course of this dissertation.

Prabhanjana Kumar Nallani

09/CTA/04

M.E.(CTA)

Abstract

Content Based Image Retrieval is an interesting and most emerging field in the area of ‘Image search’, finding the similar images for the given query image from the image database. Current approaches include the use of color, texture and shape information, considering these features in individual, most of the retrievals are poor in results and sometimes we are getting some non relevant images for the given query image.

So, this dissertation proposes an algorithm **C**olor-**S**patial based **I**mage **R**etrieval (CSIR), for image retrieval by considering both color and spatial features of the image together to improve the retrieval results in terms of its accuracy. CSIR considers both color clustering for getting color information and spatial clustering for getting spatial information of image. CSIR computes image features automatically from a given image and they can be used to archive and/or retrieve images. These features are based on color and spatial distribution information in an image.

We have tested our methods on two different real databases, viz. brodatz image database and natural image database. CSIR performs very well on both databases. The output based on our CSIR technique retrieves top ten similar images that are ranked in order of similarity. This method is applied on L*a* b* color space.

Contents

List of Figures

Chapter 1
INTRODUCTION

1.1 Motivation

In recent years, more and more attention is focused on Content-Based Image Retrieval (CBIR), which is a sub-problem of Content Based Retrieval (CBR). The tremendous growth of the numbers and sizes of digital image and video collections on Web is making it necessary to develop power tools for retrieving this unconstrained imagery. In addition, CBIR is also the key technology for improving the interface between user and computer.

Most conventional image databases are text-annotated [1]. As a result, image retrieval is based on keyword searching. Text-annotated images are simple and easy to manipulate. However, there are two major problems with this method. First, creating keywords for a large number of images is time consuming. Moreover, the keywords are inherently subjective and not unique. Due to these disadvantages, automatic indexing and retrieval based on image content becomes more desirable for developing large volume image retrieval applications.

Research on multimedia systems and content-based image retrieval has been given tremendous importance during the last decade. The reason behind this is the fact that multimedia databases deal with text, audio, video and image data which could provide us with enormous amount of information and which has affected our life style for the better. Content-based image retrieval (CBIR) is a bottleneck of the access of multimedia databases simply because there still exists vast differences in the perception capacity between a human and a computer. There are two basic problems that remain in the area, which are proving difficult to be resolved. The first one is the problem of efficient and meaningful image segmentation where we break-up a particular image into meaningful parts based on low level features like color, texture, shape and spatial locations. The second one is the vast gap existing for an image between low-level features mentioned earlier and high-level or semantic expressions contained in the image like the image of a car, a house, a table and so on.

One of the primary challenges in digital libraries is the problem of providing intelligent search mechanisms for multimedia collections. While there are good tools for searching text collections, images are much more difficult. If the images are annotated by hand, a textual search can be used; however, this approach is too labor-intensive to scale up with large digital libraries. Automated methods for searching large image collections are therefore necessary. This in turn requires simple and effective image features for comparing images based on their overall appearance. Color histograms are widely used. The histogram is easy to compute and is insensitive to small changes in viewing positions. A histogram is a coarse characterization of an image, however, and images with very different appearances can have similar histograms. When image databases are large, this problem is especially acute.

So to avoid these kinds of problems there is a need to consider spatial information along with the color information so that this feature is unique to each image and some what equal to the similar images to this image. This uniqueness among the similar images greatly helps the content based image retrieval systems to retrieve the similar images for the given query image. This project aims at considering this spatial color feature as image feature and is used in CBIR system to compute the similar images for the given query image.

1.2 Problem Definition

Color and color neighborhood statistics have been used extensively in image matching and retrieval. These techniques, however, do not consider absolute location and layout of colors in an image. In this project we consider a novel method for content based image retrieval that considers the both color and spatial location of that color in the image. First the image is clustered in to the different parts based on the color by using color clustering and later we consider the spatial location of these parts by using spatial clustering. These features are combined and are used in content based image retrieval system to compute the similar images for the given query image from the database.

1.3 Scope

The scope of this research is circumscribed to CBIR system based on Color and Spatial features of images so that to improve the efficiency of CBIR system. We have computed the image features described in Chapter 5 on Natural image database, and implemented color and spatial feature approach on that. Using color and spatial feature, we retrieved images from the database for the given query image

1.4 Organization of Thesis

This Thesis is organized as follows.

General description of the working of a conventional CBIR system is presented in Chapter 2. Overview of CBIR Techniques and Related work is discussed in Chapter 3. Basics of color representation and various color models are discussed in Chapter 4. Chapter 5 describes the color - spatial based image retrieval (CSIR) algorithm and computation of these features from the images are explained. Chapter 6 describes the experimental results of this project work and finally conclusion and future work is mentioned in chapter 7.

In Appendix A, we included three main programs; first program rbg2lab.c converts rgb color space to lab color space through CIE xyz color space which we discussed in chapter 4. Second program features.c extracts color and spatial features using our color - spatial based clustering algorithm which we discussed in chapter 5. Last Program retrieval.c evaluates Similarity Measures of query image and Images in Image database which we discussed in chapter 5.3. It retrieves top ten similar images for the given query image. In Appendix B, we included the instructions to run the project.

Chapter 2
Content Based Image Retrieval

2.1 Human vision versus Computer vision

Par excellence, humans can function well in complex environments, which provide them with a range of multi-modal cues. So far, entities that comprise artificial intelligence can not function properly if at all in such an environment. Therefore, human intelligence and, moreover, human cognitive capabilities are the baseline for the development of intelligent software applications, such as CBIR engines.

We know that the human visual system is powerful. It endows us with the ability to recover enormous details about our visual environment. Nevertheless, as a plethora of visual illusions demonstrates, it is far from accurate. Today we interpret many of these illusions not as an expression of the limits or failures of the visual system. Rather, they are the results of a highly developed and optimized representational process in which the visual system does not simply provide an internal one-to-one copy of the external visual world. Instead, the visual system is equipped with specific encoding mechanisms to optimize the use of precious processing resources by enhancing relevant features and providing only a sketchy representation of the less relevant aspects of our visual environment. In an ideal situation, computer vision should have similar characteristics.

From literature in neuro science and psychology, it is known that the human visual system utilizes features such as color, texture, and shape in recognizing objects, the environment, and photos or paintings. In addition, phenomena such as occlusion and subjective contours are topic of research. Research on these features and phenomena is merely done in controlled experimental environments. The advantage of control to a large extent is that the influence of one factor on the percept can be examined. The drawback is that the ecological validity of such research is often questionable.

Despite its limitations, especially on the long run, fundamental research is of great importance for application centered research. With such research, principles of human information processing can be unraveled step by step. Where in human vision research, this approach is the standard, for computer vision pragmatic considerations dominate. The latter is not surprising. The core aim of computer vision in general is not to model human vision but to be inspired by human vision: An efficient approach for practical problems within a limited domain.

Let us provide two examples which illustrate the strength of human vision:

(i) a painting can not be described with computer vision techniques like humans can, who can abstract from detail and 'feel' the expressed emotion through their senses in combination with associations derived from memory and (ii) Although computer vision techniques are utilized for the automatic classification of mammography images, the performance of the classifiers developed so far is far from good. Consequently, most medical imaging techniques are used as computer aided diagnosis instead of replacing the human diagnosis.

On the other hand, for certain tasks computer vision algorithms outperform human vision by far. When the exact percentage of a specific type of red (a R, G, B value) in an image has to be determined, a computer can provide an exact percentage, where a human can only estimate it. The latter is more complicated when a fuzzy description of a color (e.g., red) is provided and when the results are judged by humans. Then the question arises: what is red? Are certain pixels in all circumstances judged as red by humans or can they be judged as being black, for example, when the environment or the light source changes? The latter problems are denoted as the problem of color constancy. The computer can determine the color independent of environment or light source, which is impossible for the human visual system.

Above, we have introduced color as feature in visual information processing. Now, let us illustrate the influence of patterns or texture on the perceived color, where we define pixels as any of the small discrete elements that together constitute an image (as on a television or computer screen). For example, a square perceived as orange can exist without orange pixels. It can be generated by regularly alternating red and yellow pixels (so called dithering), which create the orange percept, as is know from computer graphics and human perception studies. In contrast, from a physics/engineering point of view, the R, G, and B channel of a CRT tube tell us that 66.6 percent of the square is red and 33.3 percent is green. Moreover, the pattern is a texture and will, in principle, be analyzed as such by image processing techniques. However, should this be preferred or should one conduct image processing at a higher level, abstracted from the pixel-level and, with that, perhaps mimicking the human percept. This simple example illustrates that it is eminent that the gap between the physical phenomena and the human percepts need to be bridged. This is especially of importance for computer vision and CBIR techniques.

In computer vision, color and texture features are also utilized for shape extraction. Hereby, the representation of both features is of paramount importance. Until now, the results are promising but far from good. In addition, shape extraction is computationally expensive and, therefore, not usable for real time image processing and computer vision tasks. Humans are able to detect shape and process it with a high accuracy. In most cases, the human visual processing system works perfectly. However, it can be tricked. For example, artists such as Escher were able to deceive the human visual system.

Let me further illustrate the beauty of the human visual system. In 1658, Pascal described that humans can see mirror symmetry at a glance. A broad range of fundamental research toward symmetry perception has been conducted. However, until now, no algorithm has been presented that completely describes the process of human symmetry recognition outside a controlled experimental setting; as Liu and Collins stated: choosing precisely which candidate is preferred by human perception is an open problem. As a consequence, the phenomenon of human symmetry recognition in natural object images (e.g., that contain a face), is hard if possible at all for computer vision.

In addition, humans can recognize objects that are occluded, without any problem. It is a fundamental issue in perception research. Thus far, the stimulus domain has been restricted to stylistic 2D line drawn stimuli. Only in the last five years attempts have been made to extend occlusion research to the domain of natural object images. The ease with which humans recognize occluded objects is in sharp contrast with the problems such tasks reveal for computer vision.

So, despite the rapid growth of computer vision techniques, human vision is still superior in most aspects. Intrigued by the efficiency of human vision on one hand, we aim at adopting principles of human vision for computer vision and for CBIR techniques. On the other hand, image processing and CBIR techniques can and should be improved. So both from a computer vision and human vision point of view, CBIR should be approached.

2.2 Content-Based Image Retrieval

In 1992, Kato introduced the term content-based image retrieval (CBIR), to describe his experiments on automatic retrieval of images from a database by color and shape features. Since then, CBIR arose as a new field of research.

CBIR is the application of computer vision to the image retrieval problem; i.e., the problem of searching for images in large image databases. Most image retrieval engines on the World Wide Web (WWW) make use of text based image retrieval, in which images are retrieved based on their labels, descriptions, and surrounding text. Although text-based image retrieval is fast and reliable, it fully depends on the textual annotations that accompany images. Consequently, it requires every image in the database or on the WWW to be well annotated or labeled. The major functions of the CBIR system are

1. Analyze the contents of the source information, and represent the contents of the analyzed sources in a way that will be suitable for matching user queries. This step is normally time consuming since it has to process sequentially all the source information (images) in the database. However, it has to be done only once and can be done offline.
2. Analyze user queries and represent them in a form that will be suitable for matching with the source database. Part of this step is similar to the previous step, but applied only to the query image.
3. Define a strategy to match the search queries with the information in the stored database. Retrieve the information that is relevant query.

The following figure is an abstract view of the content based retrieval system; the functions of the system are described later sections in this chapter.

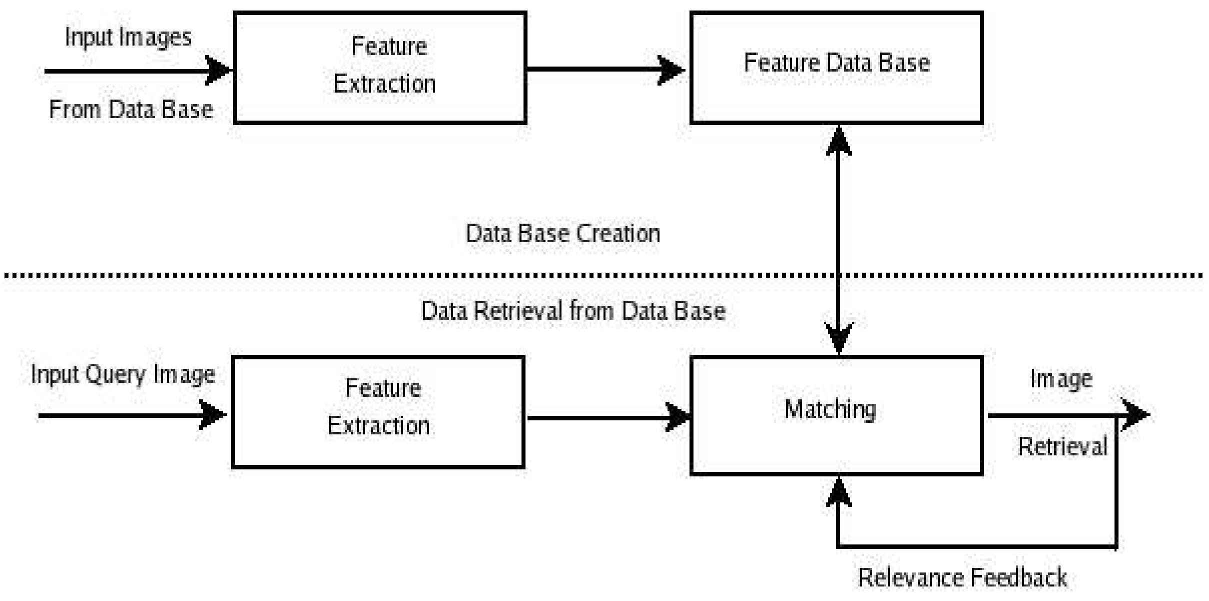


Fig 2.1: Abstract view of Content Based Image Retrieval System

2.3 Color Retrieval

Color is the first and most straightforward visual feature for retrieval of images. 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 submits an example image from which a color histogram is calculated. The matching process then retrieves those images whose color histograms match those of the query most closely. Swain and Ballard developed a matching technique which uses histogram intersection. Variants of this technique are now used in a high proportion of current CBIR systems. In out discussion we have concentrated in variants of the same technique.

Color Histogram: Statistically, a color histogram is a way to approximate the joint probability of the values of the three color channels. The most common form of the histogram is obtained by splitting the range of the data into equally sized bins. Then for each bin, the number of points from the data set (here the color of the pixels in an image) that fall into each bin are counted. There are different ways of describing the underlying color distributions.

The regions in the color space can be defined in a non-parameterized way by nonparametric clustering algorithms, or simply given by fixed borders in some color space. For example in RGB color space, if we divide each channel R,G and B into 8 equally intervals with a length of 32: 0-31, 32-63, ..., 224-225, we will have an 8 by 8 by 8 color histogram of 8X8X8=512 color bins. The color histogram is the most popular representation of color distributions since it is insensitive to small object distortions and is easy to compute.

2.4 Texture Retrieval

The ability to retrieve images on the basis of texture similarity may not seem very useful. But the ability to match on texture similarity can often be useful in distinguishing between areas of images with similar color (such as sky and sea, or leaves and grass). A variety of techniques has been used for measuring texture similarity. The best established rely on comparing values of what are known as second order statistics calculated from query and stored images. Essentially, these calculate the relative brightness of selected pairs of pixels from each image. From these it is possible to calculate measures of image texture such as the degree of contrast, coarseness, directionality and regularity, or periodicity, directionality and randomness.

Texture is widely used and intuitively obvious but has no precise definition due to its wide variability. One existing definition states that ”an image region has a constant texture if a set of its local properties in that region is constant, slowly changing, or approximately periodic”.

There are many ways to describe texture: Statistical methods often use special frequency, co-occurrence matrices, edge frequency, primitive length etc. From these many simple features such as energy, entropy, homogeneity, coarseness, contrast, correlation, cluster tendency, anisotropy, phase, roughness, directionality, stripes, repetitiveness, and granularity are derived. These texture description methods compute different texture properties and are suitable if texture primitive sizes are comparable with the pixel sizes.

Syntactic and hybrid (combination of statistical and syntactic) methods such as shape chain grammars, or graph grammars are more suitable for textures where primitives can easily be determined and their properties described. There are many review papers in this area.

2.5 Shape Retrieval

Defining the shape of an object is often very difficult. Shape is usually represented verbally or in figures, and people use terms such as elongated, rounded. Computer based processing of shape requires describing even very complicated shapes precisely and while many practical shape description methods exists, there is no generally accepted methodology of shape description. Two main types of shape features are commonly used: Boundary based and Region based features. The former uses only the outer boundary of the shape while the later uses the entire shape region. Examples of the rest type include chain codes, Fourier descriptors, simple geometric border representations (curvature, bending energy, boundary length, signature), and examples of the second include area, Euler number, eccentricity, elongated ness, and compactness.

2.6 Database Creation

Database creation can be done offline. On each image in the database of Natural images and brodatz images, compute the features described in the section 5.2. Along with the name of the image, store these image features. We have created databases for different color spaces namely RGB and Lab for the same brodatz images and natural images like flowers and sunsets databases.

2.7 Query Processing

In Query processing, if we have chosen a query image from natural database, compute the color-spatial features on the query image and from the computed database retrieve the images which are similar to the given query using a similarity metric like Euclidean distance. It will retrieve the similar images for the given query image using the color-spatial feature.

2.8 Fields of application

CBIR finds the following fields of applications

- crime prevention
- the military
- intellectual property
- architectural and engineering design

- fashion and interior design
- journalism and advertising
- medical diagnosis
- geographical information and remote sensing systems
- cultural heritage
- education and training
- home entertainment
- WWW searching

We will discuss two of these Fields of application:

(i) The WWW,

(ii) photo books, as an application for customers.

2.8.1 The World Wide Web

The contrast between nowadays Internet and the Internet as it was at its launch is enormous. More and more digital multi modal information sources are used; especially, images dominate the WWW with an average between 14.38 and 21.04 images per page. In principle, CBIR can be used to retrieve these images from the WWW. However, CBIR on the (unrestricted) WWW suffers from time, computational, and storage (or space) complexity. A substantial effort has to be made before these are tackled.

2.8.2 Photobook

More than a decade ago, one of the early CBIR systems was launched Photobook [28]. Its name illustrates its intended domain of application: photo-collections. Nowadays, a still increasing, vast amount of people has a digital photo/video-camera. The ease of making digital photo’s led to an explosion in digital image and video material. The exchange of these materials is facilitated through both Internet and mobile telephones. In addition, the costs for storing them have declined rapidly in the last years.

The exploded amount of digital image material is transported and stored on CDs, DVDs, and hard disks. In contrast, only a decade ago everybody used paper photo books to manage their photo-collection. The need emerged for digital photobooks and, subsequently, a range of them was developed and computers and their operating systems were adapted to facilitate in handling (e.g., processing and browsing) multimedia information. However, where in paper photo books people wrote small texts and placed dates accompanying photos, in their digital counterparts this effort is not made. Not in the last place, this will be due to the vast amount of digital images compared to analog ones, made in the recent past. The latter is due to the fact that (i) people tend to take an image without hesitating and (ii) there are no developing and publishing costs. As a consequence, the amount of digital photos in private photo collections is much larger than with their analog counterparts. Since the digital image collections are not or poorly annotated, text based image retrieval cannot be applied. Manual searching is a frequently used alternative but becomes less attractive with the increasing size of the private, digital photo collections.

The WWW and large professional databases (up to a lower extent) suffer from a computational burden, due to the large amount of (high quality) image material. No such problems are present with private image collections. Where private image collections can be too large for manual searching, they are small compared to most professional databases. So, CBIR systems can provide a substantial contribution in managing private photo collections.

Chapter 3
An Overview of CBIR Techniques

There were two approaches to content-based image retrieval initially [22]. The first one is based on attribute representation proposed by database researchers where image contents are defined as a set of attributes which are extracted manually and are maintained within the framework of conventional database management systems. Queries are specified using these attributes. This obviously involves high-level of image abstraction. The second approach which was presented by image interpretation researchers depends on an integrated feature-extraction / object-recognition subsystem to overcome the limitations of attribute-based retrieval. This system automates the feature-extraction and object recognition tasks that occur when an image is inserted into the database. These automated approaches to object recognition are computationally expensive, difficult and tend to be domain specific. Recent content-based image retrieval research tries to combine both of these above mentioned approaches and has developed efficient image representations and data models, query-processing algorithms, intelligent query interfaces and domain-independent system architecture.

There are two major categories of features. One is basic which is concerned with extracting boundaries of the image and the other one is logical which defines the image at various levels of details. Regardless of which approach is used, the retrieval in content-based image retrieval is done by color, texture, sketch, shape, volume, spatial constraints, browsing, objective attributes, subjective attributes, motion, text and domain concepts [23].

The developments in this field have been put forward in three levels [24].

Level one is the fundamental level where low-level features like color, texture, shape and spatial locations are applied to segment images in image database and then find symmetry based on these segmentations, with the input image. Quite a bit of research works were being done during the last decade.

Level two talks about bringing out semantic meanings of an image of the database. One of the best known works in this field is of Forsyth [25] by successfully identifying human beings within images and this technique had been applied for other objects like horses and trees.

Level three talks about retrievals with abstract attributes. This level of retrieval can be divided into two groups. One is a particular event like ‘Find pictures of a particular birthday celebration’. Second one could be ‘Find picture of a double-decker bus’. To interpret an image after segmentations and analyzing it efficiently require very complex logic. This would also require retrieval technique of level two to get semantic meanings of various objects. It is obvious this retrieval technique is far from being developed with modern technology available in the field.

Several systems have been proposed in recent years in the research community for content-based information retrieval:
(1) QBIC (Query by Image Content) by IBM,
(2) Virage by Virage, Inc.,
(3) Photobook by MIT Media Lab.,
(4) Visual SEEK by Columbia University,
(5) Retrieval Ware by Excalibur Technologies Corp.,
(6) NeTra by the University of California, Alexandria Digital Library,
(7) IRIS by German Software Development Laboratory of IBM and the AI group of the University of Bremen,
(8) CORE by the University of Singapore.

QBIC was the first commercial CBIR system developed by Flickner [26]. Its system framework and techniques had great effects on later image retrieval systems. The QBIC system allows queries on a large image and video database, based on color, shape, texture, example images and sketches. The color features used in QBIC are the average (*R*,*G*,*B*), (*Y*,*i*,*q*), (*L*,*a*,*b*) and MTM(Mathematical Transform to Munsell) coordinates. Query by Image Content (**QBIC**) system provides a comprehensive, operational image retrieval system based on the a priori 8 feature extraction approach. The features are extracted semi-automatically. The QBIC system allows queries on large image and video databases based on example images, user-constructed sketches and drawings, color and texture patterns, and camera and object motion.

The Virage [36] Search Engine supports querying of still image and video streams [27]. Similar to QBIC, it supports visual queries based on color, composition (color layout), texture and structure (object boundary information). It also supports arbitrary combinations of the above four atomic queries. The user can adjust the weights associated with the atomic features according to their own purposes. **Virage** was a system developed based on QBIC approach which made use of color, composition (color layout), texture and structure (object boundary details) and is being applied in face recognition and in retrieval of ophthalmologic images.

The Photobook system is a set of interactive tools for browsing and searching images [28]. The key idea behind this suite of tools is semantics preserving image compression, which reduces an image to a small set of perceptually significant coefficients. Photobook consists of three sub-books. There are the Appearance Photobook, Shape Photobook and Texture Photobook, which can extract the face, shape and texture, respectively. Users can query an image based on the corresponding features in each of the three sub books, or on a combination of different mechanisms with a text-based description. Other systems (i.e., 3, 6, 7 and 8 listed above) [29] [30] [31] [32] provide querying based on global color, color layout, shape, texture and semantic content Spatial relationship querying of image regions is the main Even though the above content-based image retrieval system provide many features for image querying, none of them combine global color, color region, color sensation, shape and qualitative spatial relation features to query an image.

Excalibur, a Visual Retrieval Ware system enables queries on gray shape, color, shape and texture using adaptive pattern recognition techniques [38]. Excalibur also provides data blades for Informix databases. An example data blade is a scene change detector for video. The data blade detects shots or scenes in video and produces a summary of the video by example frames from each shot.

There are few systems for the World Wide Web such as **Web Seek** [39] builds several indexes for images and videos based on visual features such as color as well as non-visual features such as keywords assigned subjects and image/video types. In order to classify images and videos into subject categories, a key word dictionary is built from selected terms appearing in a Uniform Resource Locator (URL), the address of a page on World Wide Web. The terms are selected based on their frequency of occurrence and whether they are meaningful subject terms. The latter judgment is made manually. After the key word dictionary is built, directory portion of the image and video URLs are parsed and analyzed. This analysis produced an initial set of categories of the images and videos which is then manually verified. Videos are summarized by picking one frame for every second of video and then packaging them as an animated GIF image.

Other Systems:

Systems such as **CVEPS** and **JACOB** [33] support automatic video segment decomposition, and video indexing based on key frames or objects. The users can employ image analysis and retrieval components to index and retrieve key frames or video objects based on their visual features or spatial layout. The CVEPS system also provides these functions as well as video editing in compressed domain. The JACOB system uses artificial neural networks for automatic shot detection. Among systems using captions or annotations for image retrieval, the caption-based image retrieval system of Dublin City University uses **WordNet,** an electronic dictionary for query expansion. Srihari [34] describes a system that uses a semantic model for interpreting captions in order to guide person recognition. The **SCORE** [35] system uses an extended Entity-Relationship model to represent image contents and WordNet to expand queries as well as database descriptions.

Informix data blades [37], formerly Illustra, enable user defined content processing and analysis routines to be stored in a multimedia database. Data blades for free text, images, sound and video are becoming available by Informix and third party suppliers.

Ogle and Stonebraker developed "**Chabot**: Retrieval from a relational database of images" [40]. This system design is based on the a priori feature extraction approach, with features extracted automatically. This system demonstrates how text based search criteria can combine with content-based criteria to achieve good retrieval results. In this version, content analysis is primarily based on color histograms, and query optimization techniques are used to minimize the number of histograms considered when a query is processed.

Chapter 4
Color Representation

In this chapter, general color concepts are introduced, which will be used in the remainder of thesis, for color analysis, spatial color analysis, and CBIR. First, the color histogram is described: a discrete function that quantizes the distribution of colors of an image. Next, several color spaces and their color quantization schemes are discussed.

Color is the sensation caused by light as it interacts with our eyes and brain. Color is the sensation caused by light as it interacts with our eyes and brain. Color is the sensation caused by light as it interacts with our eyes and brain. The perception of color is greatly influenced by nearby colors in the visual scene. The human eye contains two types of visual receptors: rods and cones. The rods are responsive to faint light and therefore, sensitive to small variations in luminance. The cones are more active in bright light and are responsible for color vision. Cones in the human eye can be divided in three categories, sensitive to long, middle, and short wave length stimuli. Roughly these divisions give use to the sensations of red, green, and blue. The perception of color is greatly influenced by nearby colors in the visual scene. The human eye contains two types of visual receptors: rods and cones. The rods are responsive to faint light and therefore, sensitive to small variations in luminance. The cones are more active in bright light and are responsible for color vision. Cones in the human eye can be divided in three categories, sensitive to long, middle, and short wave length stimuli. Roughly these divisions give use to the sensations of red, green, and blue. The perception of color is greatly influenced by nearby colors in the visual scene. The human eye contains two types of visual receptors: rods and cones. The rods are responsive to faint light and therefore, sensitive to small variations in luminance. The cones are more active in bright light and are responsible for color vision. Cones in the human eye can be divided in three categories, sensitive to long, middle, and short wave length stimuli. Roughly these divisions give use to the sensations of red, green, and blue.

The use of color in image processing is motivated by two principal factors. First, color is a powerful descriptor that facilitates object identification and extraction from a scene. Second, humans can discern thousands of color shades and intensities, compared to about only two dozen shades of gray.

In this chapter, general color concepts, as used in this thesis, will be introduced. We will start with a description of the color histogram. Next, color quantization will be explained, followed by the description of several color spaces and their quantization method.

4.1 Color Histogram

The color histogram is a method for describing the color content of an image. It counts the number of occurrences of each color in an image. The color histogram of an image is rotation, translation, and scale-invariant; therefore, it is very suitable for color-based CBIR: content-based image retrieval using solely global color features of images. However, the main drawback of using the color histogram for CBIR is that it only uses color information, texture and shape-properties are not taken into account. This may lead to unexpected errors; for example, a CBIR engine using the color histogram as a feature is not able to distinguish between a red cup, a red plate, and a red car.

4.2 Color quantization

In order to produce color histograms, color quantization has to be applied. Color quantization is the process of reducing the number of colors used to represent an image. A quantization scheme is determined by the color space and the segmentation (i.e., split up) of the color space used. A color space is the representation of color in a three dimensional space.

4.3 Color Spaces

A color space specifies colors as tuples of (typically three) numbers, according to certain specifications. Color spaces lend themselves to (in principle) reproducible representations of color, particularly in digital representations, such as digital printing or digital electronic display. The purpose of a color space is to facilitate the specification of colors in some standard, generally accepted way.

One can describe color spaces using the notion: perceptual uniformity. Perceptually uniform means that two colors that are equally distant in the color space are perceptually equally distant. Perceptual uniformity is a very important notion when a color space is quantized. When a color space is perceptually uniform, there is less chance that the difference in color value due to the quantization will be noticeable on a display or on a hard copy. In the remainder of this section, several color spaces described.

4.3.1 The RGB Color Space

The RGB color space is the most used color space for computer graphics. Note that R, G, and B stand here for intensities of the Red, Green, and Blue guns in a CRT, not for primaries as meant in the CIE RGB space. It is an additive color space: red, green, and blue light are combined to create other colors. It is not perceptually uniform. The RGB color space can be visualized as a cube.

The RGB color space is the most prevalent choice for digital images [1] because color-CRTs (computer display) use red, green, and blue phosphors to create the desired color. Also, it is easy for programmers to understand and program since this color space has been widely used for a number of years. However, a major drawback of the RGB space is that it is senseless. The user finds it difficult to understand or get a sense of what color $R = 100$, $G = 50$, and $B = 80$ is and the difference between R=100, $G = 50$ and $B = 50$, and $R = 100$, G=150 and $B = 150$.

Each color-axis (R, G, and B) is equally important. Therefore, each axis should be quantized with the same precision. So, when the RGB color space is quantized, the number of bins should always be a cube of an integer.

4.3.2 The HSx Color Space

The HSI, HSV, HSB, and HLS color spaces (conventionally called ‘HSx’) are more closely related to human color perception than the RGB color space, but are still not perceptually uniform.

The axes from the HSx color spaces represent hue, saturation, and lightness (also called value, brightness and intensity) color characteristics [1]. The difference between the different HSx color spaces is their transformation from the RGB color space. They are usually represented by different shapes (e.g., cone, cylinder). Hue describes the actual wavelength of a color by representing the color’s name, for example, green, red or blue. Saturation is a measure of the purity of a color. For instance, the color red is a 100% saturated color, but pink is a low saturation color due to the amount of white in it. Intensity indicates the lightness of a color. It ranges from black to white.

Hue is the color component of the HSx color spaces. Hue is an angle between a reference line and the color point in RGB space, the range of this value is between 0 and 360, for example blue is 240. According to the CIE (Commission Internationale de l Eclairage), hue is the attribute of a visual sensation according to which an area appears to be similar to one of the perceived colors, red, yellow, green, and blue, or a combination of two of them. In other words, hue is the color type, such as red or green. Also according to CIE, saturation is the colorfulness of an area judged in proportion to its brightness. In the cone, the saturation is the distance from the center of a circular cross-section of the cone, the ‘height’ where this cross-section is taken is determined by the Value, which is the distance from the pointed end of the cone. The value is the brightness or luminance of a color, this is defined by CIE as the attribute of a visual sensation according to which an area appears to emit more or less light. When Saturation is set to 0, Hue is undefined. The Value-axis represents the gray-scale image.

The HSV color space can easily be quantized, the hue is the most significant characteristic of color so this component gets the most part of quantization. In the hue circle, the primary colors red, green, and blue, are separated by 120. The secondary colors, yellow, magenta, and cyan, are also separated by 120 and are 60 away from the two nearest primary colors.

The most common quantization of the HSV color space is in 162 bins, where hue gets 18 bins and saturation and value both get 3 bins. When hue is divided in 18 bins, each primary color and secondary color is represented with three subdivisions.

4.3.3 YUV and YIQ Color Space

The YUV and YIQ color spaces are developed for television broadcasting. The YIQ color space is the same as the YUV color space, where the I-Q plane is a 33 rotation of the U-V plane [9]. The Y signal represents the luminance of a pixel and is the only channel used in black and white television. The U and V for YUV and I and Q for YIQ are the chromatic components.

The Y channel is defined by the weighted energy values of R(0:299),G(0:587), and B(0:144) . The YUV and YIQ color spaces are not perceptually uniform. When the YUV and YIQ color spaces are quantized, each axis is quantized with the same precision. The quantization schemes used for the YUV and YIQ color spaces in this thesis are: 8 (23), 27 (33), 64 (43), 125 (53), and 216 (63) bins.

To optimize color appearance the YUV color space is often sampled. The samplings used to construct the color correlogram are: 4:4:4, 4:2:2, and 4:1:1, where the numbers denote the relative amount of respectively Y on each row, U and V on each even-numbered row, and U and V on each odd numbered row in the image.

4.3.4 The CIE XYZ and LUV color spaces

The rgb color space developed by the CIE is the XYZ color space [9]. The Y component is the luminance component defined by the weighted sums of R(0:212671), G(0:715160), and B(0:072169). The X and Z are the chromatic components. The XYZ color space is perceptually not uniform. In quantizing the XYZ space, each axis is quantized with the same precision.

A color is identified by two coordinates, *x* and *y*, in the C.I.E. L*u*v* Color Space. Lightness *L** is based on a perceptual measure of brightness, and *u** and *v** are chromatic coordinates. Also, color differences in an arbitrary direction are approximately equal in this color space. Thus, the Euclidean Distance can be used to determine the relative distance between two colors. However, coordinate transformation to the RGB space is not linear.

The CIE LUV color space is a projective transformation of the XYZ color space that is perceptually uniform. The L-channel of the LUV color space is the luminance of the color. The U and V channels are the chromatic components. So, when U, and V are set to 0, the L-channel represents a gray-scale image.

In quantizing the LUV space, each axis is quantized with the same precision. For both the XYZ color space and the LUV color space, the following quantization schemes are used: 8 (23), 27 (33), 64 (43), 125 (53), and 216 (63) bins.

We used CIE L*a*b* color space to extract color features of an image. Since this color space is perceptually uniform, distances in this space are meaningful. Each pixel has a three dimensional color description in the L*a*b* color space, where Luminance is measured by L (0 to 100), while a (-100 to 100) and b (-100 to 100) represent the red-green (a, the smallest a yields green) and yellow-blue (b, the smallest b yields blue) components.

The equations for transformed L*a*b* are given below: Initially each component of RGB are gamma corrected. If red, green, blue are the values extracted from the image, then Gamma correction is done as follows.

R = red/255; G=green/255; B=blue/255;

If (R>0.04045) then R = $\left(\dfrac{R+0.055}{1.055}\right)^{2.4}$ else R = $\dfrac{R}{12.92}$

If (R>0.04045) then G = $\left(\dfrac{G+0.055}{1.055}\right)^{2.4}$ else G = $\dfrac{G}{12.92}$

If (R>0.04045) then B = $\left(\dfrac{R+0.055}{1.055}\right)^{2.4}$ else B = $\dfrac{B}{12.92}$

R = R *100; G = G * 100; B = B * 100.

Then R, G, B are converted into XYZ color space using the equations:

X = R *0.4124 + G * 0.3576 + B * 0.1805
Y = R *0.2126 + G *0.7152 + B *0.0722
Z = R * 0.0193 + G *0.1192 + B * 0.9505.

XYZ is converted into L*a*b* as follows:

$$L* = 116(\frac{Y}{Y_n})^{1/3} - 16, if\ \frac{Y}{Y_n} > 0.008856, L* = 903.3\left(\frac{Y}{Y_n}\right) otherwise.$$

$$a* = 500 * (f_n\left(\frac{X}{X_n}\right) - f_n\left(\frac{Y}{Y_n}\right))$$

$$b* = 500 * (f_n\left(\frac{Y}{Y_n}\right) - f_n\left(\frac{Z}{Z_n}\right))$$

Where

$f_n = t^{1/3}$ if t>0.008856, $f_n = 7.787 * t + \left(\dfrac{16}{116}\right) otherwise$

(X_n, Y_n, Z_n) defines an appropriately chosen reference white point.

So for every pixel in the query image, RGB values are taken and the 3 color features (L*,a*,b*) are extracted.

4.4 Techniques for Color Image Retrieval

As mentioned, color features are the most widely used in current CBIR systems. Many of these systems are based on image color histograms, in which distance between images is measured by calculating the distance between color histograms. Some histogram distance metrics, as well as some color-based Image Retrieval techniques, are discussed in this section.

4.4.1 Global Color Histogram

The Global Color Histogram (GCH) approach is the most popular color-based Image Retrieval technique and is often used as a benchmark for other techniques. The color histogram depicts color distribution using a set of bins. Using the GCH, an image will be encoded with its color histogram, and the distance between two images will be determined by the distance between their color histograms. To compute the distance between color histograms of two images A and B

$$d_{GCH}(A,B) = \sqrt{\sum_{j=1}^{n}\left(H_j^A - H_j^B\right)} \text{..........................(4.1)}$$

Where n is the number of bins, and H_j^A is the color histogram for j^{th} color in the image A.

The GCH is the traditional method for color-based image retrieval. However, it does not include information concerning the color distribution of the regions, so the distance between images sometimes cannot show the real difference between images. Moreover, in the case of a GCH, it is possible for two different images to have a very short distance between their color histograms. This is the main disadvantage of GCH's.

4.4.2 Local Color Histogram

This approach (referred to as LCH) includes information concerning the color distribution of regions. The first step is to segment the image into blocks and then to obtain a color histogram for each block. An image will then be represented by these histograms. When comparing two images, we calculate the distance, using their histograms, between a region in one image and a region in same location in the other image. The distance between the two images will be determined by the sum of all these distances. If we use the square root of Euclidean distance as the distance between color histograms, the distance metric between two images and used in the LCH will be defined as:

$$d_{LCH}(Q, I) = \sum_{k=1}^{M} \sqrt{\sum_{j=1}^{N} \left(H_Q^A[i] - H_I^k[i]\right)^2} \text{ ……………………(4.2)}$$

Where M is the number of segmented regions in the images, N is the number of bins in the color histograms, and is the value of bin i in color histogram , which represents the region 'k' in the image Q(I).

In some scenarios, using LCH's, can obtain better retrieval effectiveness than using GCH's. The above examples show that the LCH overcomes the main disadvantage of the GCH, and the new distances between images may be more reasonable than those obtained using the GCH. However, since the LCH only compares regions in the same location, when the image is translated or rotated, it does not work well.

4.4.3 IRM

In Integrated Region Matching similarity measure [7], we match regions in two images. Being aware that segmentation cannot be perfect, we match one region of an image with several regions of another image. A region-to-region match is obtained when the regions are relatively similar to each other in terms of the features extracted.

The principle of matching is that the closest pair is matched first. We call this matching scheme Integrated Region Matching (IRM) to stress the incorporation of regions in the retrieval process. After regions are matched, the similarity measure is computed as a weighted sum of the similarity between region pairs, with weights determined by the matching scheme.

Chapter 5

Color-Spatial based Image retrieval (CSIR)

The management of a large number of images in a multimedia database has received much attention in recent years. Most of the earlier works are largely focused on techniques to extract useful information (such as the color, texture or shape features) that represents the content of images. It is only recently that this extraction technology is more mature, that researchers are becoming more concerned about assisting structures or indexes that will help to retrieve the relevant images quickly. Rapid retrieval is becoming an important issue as image databases continue to grow in size and a slow system will no longer be acceptable to the user community.

Traditionally, an image is represented by its features such as the color, texture and shape of objects. It has been observed that a single feature is insufficient to fully represent the content of an image . For example, the effectiveness (in terms of recall and precision) of retrieving images based on color alone or shape alone is found to be much lower than if both the color and spatial features are integrated. Moreover, the set of relevant images for each feature can be very different if they are used separately.

In this thesis, we present an image retrieval system that supports color and spatial features. In our system, an image is represented by a set of clusters that are extracted from the image. For each cluster, we maintain information on three features: its color, size and spatial location within the image. During the retrieval process, the query image's clusters and features information are also extracted and compared against those stored in the database. The retrieved images are ranked and returned to the users.

We implemented CBIR system, and evaluated its performance on a collection of 1000 Natural images and Brodatz images. Our results show that it is able to retrieve images effectively and efficiently.

Before we proceed further, we shall briefly discuss the color representation that we adopt. Traditionally, the RGB color space is split into a large number of “bins”, each representing a distinct color. For a given image, the number of pixels that fall into each of the bins can be determined. Thus, the degree of similarity on a color between two images can be determined by matching the corresponding bins. However, because of the large number of bins, pixels with trivial differences (e.g., due to small changes in illumination conditions) may end up in different bins. As a result, the effectiveness of this method is limited, and fails to retrieve all the similar images. It has, however, been shown that considering the similarity/distance measure across different color bins rather than bin-to-bin comparison can improve the performance drastically.

Intuitively, colors that are perceptually similar should be grouped together into a bin (since humans would have perceived these similar colors as the same). In this way, each bin captures distinguishably different color perception. Such an approach is expected to be robust to changes in illumination, and has been shown to provide more accurate similarity measure between images. However, grouping colors is a difficult task under the RGB color space. This is because the RGB color space does not carry direct semantic information about the color. In other words, it is not possible to visualize a color given its (R,G,B) triplet. Moreover, equal geometric distances in the RGB color space do not generally correspond to equal perceptual changes in color.

One approach to overcome the problem is to make use of color spaces with uniform color difference, such as the CIE L*a*b*, CIE L*u*v*, HSV and Munsell color spaces. we employ the CIE L*a*b* color space. The linearity of this color space allows the perceptual differences between the colors to be determined by the Euclidean distance measure. In other words, two dierent pixels in an image (or from dierent images) are perceptually similar in terms of color if their colors are close to one another in the 3D-space.

5.1 Color and Spatial Features

The pixels of a color image can be regarded as points in the 3-D color space. Their coordinates in the 3-D color space are the color values of the pixels. Usually the number of objects in an image is small, hence the reflection characteristic of each object in natural light tends to be consistent and it results as clusters in the color space. If clustering is performed in the 3-D color space, a few clusters will be obtained; each cluster corresponds to one of the dominant colors in the image. A representative sample (e.g. the mean color) of all such clusters can be used to define a color feature representing the color information of the image. The spatial location (e.g. centroid of a color region in image coordinates) and the population of the clusters in the image define another feature, representing the spatial distribution information of the color image. These two features can thus be used to capture the color content of the image along with the spatial distribution of the colors in the image.

5.2 Color and Spatial Clustering in Lab Color Space

5.2.1 Color Clustering

The input images are represented as RGB primary color images. Color clustering can be used to find the color clusters and assign a representative color to each of these clusters. From the color clusters, spatial clustering can be used to separate each color cluster into a number of smaller spatial clusters.

The color clustering algorithm presented here is an unsupervised method that assumes each color cluster follows a normal distribution with one peak and there is sufficient distance between every two color clusters. The color clustering algorithm is as follows:

1. Obtain the RGB components of image.
2. Convert RGB values to corresponding L a b values
3. Find all color clusters.

i) Compute the color distance of each pixel from the existing color clusters. If no color clusters exist, then a new color cluster is created. The color distance is given by

$$\sqrt{(\Delta R)^2 + (\Delta G)^2 + (\Delta B)^2}$$

ii) If the minimum color distance is less than the minimum threshold value, then a match is found. Otherwise, a new color cluster is created. We have used a minimum threshold of 40 for a 200x200 image which is equal to 0.0100 percent of the total number of pixels. An order of magnitude variation of this parameter does not change the retrieval results.

iii) For each match, the RGB values and the population of the clusters are updated. The new representative color of the cluster is the weighted average of the original cluster and the current pixel's color.

4. Sort the clusters in a descending order, based on the cluster population.
5. Determine the number of clusters which do not have a very small population. We have considered a cluster with less than 1 % population as small. A variation of a factor of 5 in this threshold did not affect the retrieval results.
6. Merge the very small clusters to their nearest color clusters. The nearest cluster is computed based on the color distance metric. The representative for this color cluster is then the weighted mean of the two original clusters.
7. For each pixel, compute the color distance to different cluster. Assign the pixel to the cluster for which the color distance is minimum. Thus every pixel gets assigned to one cluster.

Thus, after this algorithm is applied on an input image, the color clusters will be obtained and each pixel of the image will be assigned to one of these clusters. Note that spatial correlation information can be used to improve the results of the clustering using methods based on Markov random fields. Moreover, alternate color clustering methods such as could also be used. A better clustering method will only improve the retrieval results.

5.2.2 Spatial Clustering

Using the output generated by the color clustering algorithm, the following steps are used to obtain the spatial clusters:

1. Split the image into different color layers. The number of color layers is equal to the number of color clusters that is determined by the color clustering algorithm.
2. Do a connected components labeling to separate the spatial clusters
3. For each color layer, sort the spatial clusters in a descending order, based on the cluster population.
4. Determine the number of clusters which do not have a very small population.
5. Clusters which have population less than the lower threshold is discarded. We used a lower threshold of 50 pixels (0.03125%)

6. For clusters which have population less than the upper threshold (we used an upper threshold of 1000 pixels 0.625%) but more than the lower threshold, a density function is computed as follows:

• Assume that points (x_1,y_1) and (x_2,y_2) are the corner points of a rectangle which bounds a particular spatial cluster. Find the maximum length I_{max} of the box.

$$I_{max} = \max (||x_2 - x_1||, ||y_2 - y_1||)$$

• The density function of the cluster is given as density,

$$\rho = \frac{(PopulationOfClustur)}{(I_{\max})^2}$$

7. If the density, $\rho \geq 10$ % then the cluster is retained. Otherwise the cluster is ignored. The advantage of this method is that it will eliminate those clusters which have a low density, e.g. a thin line that forms a cluster. This is particularly useful in eliminating noise which manifests in the form of short lines. This type of noise is introduced at object boundaries due to the image scanning process.

5.2.3 The color and spatial feature vector

Suppose an image I of size N pixels has m color clusters and n spatial clusters, with n >= m. Each color clusters $C_{c,I}$ is defined as follows:

$$C_{c,i} = \{R_i, G_i, B_i, \lambda_{c,i}, x_{c,i}, y_{c,i}\} \qquad \text{i=1.2……m.}$$

where (R_i,G_i,B_i) is the representative color of the color cluster (mean) and $\lambda_{c,i}$ is the fraction of the pixels in that color cluster as compared to the total number of pixels.

$$\lambda_{c,i} = \frac{NumberOfPixels \in C_{c,i}}{N}$$

$(x_{c,i}, y_{c,i})$ is the centroid (in image coordinates) of the ith color cluster. The color cluster feature of image I is defined as

$$fc = \{C_{c,i} \mid i = 1,2,3.....m\}$$

Each color-spatial cluster is defined as follows:

$$C_{c,i} = \{R_i, G_i, B_i, \lambda_{c,i}, x_{c,i}, y_{c,i}\} \qquad \text{i=1.2……m.}$$

Where (R_i,G_i,B_i) is the representative color of the color-spatial cluster and $\lambda_{c,i}$ is the fraction of the pixels in that color-spatial cluster as compared to the total number of pixels.

$$\lambda_{s,i} = \frac{NumberOfPixels \in C_{s,i}}{N}$$

($x_{s,i}$, $y_{s,i}$) is the centroid of the i^{th} color-spatial cluster. The color-spatial cluster feature of image I is defined as

$$fs = \{C_{s,i} \mid i = 1,2,3.....m\}$$

5.3 The Similarity Measures

In this section, we describe the computation of the similarity measure between a pair of images. The similarity measure presented here compares the color information of two images, as well as the spatial distribution of the colors. Since it is not always true that the two images to be compared have similar or the same number of color and spatial clusters, the measure will penalize any missing color or spatial clusters. This corresponds to the intuitive notion of image similarity. The similarity computation includes finding the closest cluster for each of the color and color-spatial clusters and then calculating the distance measure.

5.3.1 Finding the closest cluster

Assume that the query image Q has m color clusters and p spatial clusters and the database image I has n color clusters and q spatial clusters. The closest color cluster assignment function P_c maps every color cluster i of image Q to the closest color cluster $P_{c(i)}$ of image I. This is a function in the sense that it is a 1-to-1 assignment. The computation of the closest color cluster assignment function, P_c is as follows:

1. Form the distance matrix, $D_{m,n}$,

$$D_{m,n} = [d_{m,n}]_{m \times n}$$

where $d_{m,n}$ is the color distance between $C_m^Q and C_n^I$

2. Find the minimum entry (a, b) in matrix D_{mn} and note that P_c (a)= b.

3. Strike off row a and column b of the matrix D_{mn}. If the matrix is non-degenerate, go to step 2 else stop. A matrix is considered degenerate when the number of rows or columns which is not yet struck off is zero

For every matched color cluster, a spatial match analogous to the computation of P_c is done, using the spatial centroid for distance measure,

$$d = \sqrt{(x_{sp} - x_{sq})^2 + (y_{sp} - y_{sq})^2}$$

This will produce a closest color-spatial cluster assignment function, Ps, which maps every color spatial clusters of image Q to the closest one in I.

5.3.2 Similarity Measure for Color and Spatial Feature

Suppose that the query image Q has m color clusters and p spatial clusters and the database image I has n color clusters and q spatial clusters. Also the closest color cluster assignment function, Pc, maps k color clusters of Q to I and the closest color-spatial cluster assignment function, Ps, maps l color-spatial clusters of Q to I. For computing the similarity, the distance measure, D (Q, I) , is given by

$$D(Q,I) = \omega_1\Psi_1 + \omega_2\Psi_2 + \omega_3\Psi_3 + \omega_4\Psi_4 + \omega_5\Psi_5 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots(5.1)$$

Where

$$\psi_1 = \frac{\sum_{i=1}^{\max(m,n)} cdist(c_i^Q, c_{P_c(i)}^Q)}{k} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5.2)$$

$$\Psi_2 = m\sum_{i=1}^{\max(m,n)} \sqrt{(\lambda_{c,i}^Q - \lambda_{c,P_c(i)}^I)^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5.3)$$

$$\Psi_3 = \frac{\sum_{i=1}^{\max(m,n)} \sqrt{(x_{c,i}^Q - x_{c,P_c(i)}^I)^2 + (y_{c,i}^Q - y_{c,P_c(i)}^I)^2}}{k} \quad \ldots\ldots\ldots\ldots\ldots(5.4)$$

$$\Psi_4 = \frac{\sum_{i=1}^{\max(p,q)} \sqrt{(\lambda_{s,i}^Q - \lambda_{s,Ps(i)}^I)^2}}{m} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5.5)$$

$$\Psi_5 = \frac{\sqrt{(x_{s,i}^Q - x_{s,P_s(i)}^I)^2 + (y_{s,i}^Q - y_{s,P_s(i)}^I)^2}}{ml} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5.6)$$

Where $\omega 1$, $\omega 2$, $\omega 3$, $\omega 4$ and $\omega 5$ are the weight factors and

$$cdist(Ci, C_{P_c(i)}) = \begin{cases} coldis(C_i^Q, C_{p_c(i)}^I) & if \lambda_{c.i}^Q, \lambda_{c_c P_c(i)}^I \\ 0 & otherwise \end{cases} \quad \ldots\ldots\ldots(5.7)$$

1. ψ_1 is the color distance between the color clusters. This measures the proximity of the colors in the given pair of color images.

2. ψ_2 is the relative frequency of pixels of the corresponding color clusters. Given that two images have similar colors, then this factor takes into account the relative proportions of the colors in the two images for ranking. It also helps to rank appropriately when the database image and query image have dierent number of colors. It prefers the images in the database that have same number of colors as in the query image, and penalizes otherwise.

3. ψ_3 is the spatial distance between the color clusters. This allows those database images to come on top which have spatially closer clusters to the clusters in the query image and penalizes other database images which do not satisfy this criterion.

4. ψ_4 is the relative frequency of pixels of the corresponding color-spatial clusters. This factor takes into account same size and number of spatial clusters in both the query and database images. The images satisfying this criteria appear at the top end of the retrieved image and others are pushed down the list.

5. ψ_5 is the spatial distance between the color-spatial clusters. It helps to brings out all those images in the database that have spatial clusters close to the spatial clusters in the query image.

The five components, $\psi 1$, $\psi 2$, $\psi 3$, $\psi 4$ and $\psi 5$ in the similarity measure are weighted by factors, $\omega 1$, $\omega 2$, $\omega 3$, $\omega 4$ and $\omega 5$, respectively. Thus different aspects of the similarity measure can be emphasized by increasing the weight factors. For example, if the color similarity between a pair of corresponding color clusters is considered the most important criterion for matching, then the weight factor, $\omega 1$ can be increased accordingly to emphasize its importance. Similarly the weight factors, $\omega 3$ and $\omega 5$ can be increased accordingly to emphasize the importance of the location of the color and spatial clusters in the distance measure. $\omega 2$ and $\omega 4$ can also be increased to emphasize the importance of the relative population of the corresponding clusters. The appropriate settings of $\omega 1$, $\omega 2$, $\omega 3$, $\omega 4$ and $\omega 5$ depends on the overall importance of each of the aspects in determining image similarity for the application under consideration.

5.3.3 Integrated Region Matching

Integrated Region Matching is a similarity measure for region based image similarity comparison. The targeted image retrieval systems represent an image by a set of regions, roughly corresponding to objects, which are characterized by features reflecting color, texture, shape and location properties. The IRM measure for evaluating overall similarity between images incorporates properties of all the regions in the images by region-matching scheme. Compared with retrieval based on individual regions, the overall similarity approach reduces the influence of inaccurate segmentation. In Integrated Region Matching similarity measure, we match regions in two images. Being aware that segmentation cannot be perfect, we match one region of an image with several regions of another image. A region-to-region match is obtained when the regions are relatively similar to each other in terms of the features extracted.

The principle of matching is that the closest pair is matched first. We call this matching scheme *Integrated Region Matching* (IRM) to stress the incorporation of regions in the retrieval process. After regions are matched, the similarity measure is computed as a weighted sum of the similarity between region pairs, with weights determined by the matching scheme.

IRM Method

IRM is defined as similarity between two sets of regions. Assume that images 1 and 2 are represented by region sets $R_1 = \{ r_1, r_2, \ldots, r_m\}$ and $R_2 = \{r_1', r_2', \ldots\ldots, r_n'\}$ where r_i and r_j^1 is the descriptor of region i. The distance between region r_i and r_j^1 is $d(r_i, r_j^1)$ which is written as $d_{i,j}$ in short. To compute the similarity measure between region sets R_1 and R_2, $d(R_1, R_2)$, we first match all regions in the two images. Our IRM matching scheme aims at building correspondence between regions that is consistent with human perception. To increase the robustness against segmentation errors, we allow a region to be matched to several regions in another image. A matching between r_i and r_j^1 is assigned with a significance credit $s_{i,j}$, $s_{i,j} >= 0$. The significance credit indicates the importance of the matching for determining similarity between images. The matrix $S = \{s_{i,j}\}$, $1 \le i \le n, 1 \le j \le m$, is referred to as the significance matrix.

A graphical explanation of the integrated matching scheme shows that matching between images can be represented by an edge weighted graph in which every vertex in the graph corresponds to region. If two vertices are connected, the two regions are matched with a significance credit being the weight on the edge. To distinguish from matching, two sets of regions. We refer to the matching of two regions as they are linked. The length of the edge can be regarded as the distance between the two regions represented. If two vertices are not connected, the corresponding regions are either from the same image or the significance credit of matching them is zero. Every matching between images is characterized by links between regions and their significance credits. The matching used to compute the distance between two images is referred to as the *admissible matching*. The admissible matching is specified by two conditions on the significance matrix. If a graph represents an admissible matching, the distance between the two region sets is the summation of all the weighted edge lengths, i.e.,

$$d(R_1, R_2) = \sum_{i,j} s_{i,j} d_{i,j} \quad \text{……………………………………….(5.8)}$$

We call this distance the integrated region matching (IRM) distance.

The problem of defining distance between region sets is then converted to choosing the significance matrix S. A natural issue to raise is what constraints should be put in $S_{i,j}$ so that the admissible matching yields good similarity measure. In the other words, what properties do we expect an admissible matching to possess? The first property we want to enforce is the fulfillment of signicance. Assume that the significance of r_i in image 1 is p_i, and r_j^1 image 2 is p_j^1 we require that

$$\sum_{j=1}^{n} s_{i,j} = p_i, i = 1,2,\ldots.., n$$
$$\sum_{j=1}^{m} s_{i,j} = p_j', j = 1,2,\ldots.., n \quad \text{………………………………..(5.9)}$$

For normalization, we have $\sum_{i=1}^{m} p_i = \sum_{j=1}^{n} p_j' = 1$ the fulfillment of significance ensures that all the regions play a role for measuring similarity. We also require an admissible matching to link the most similar regions at the highest priority. For example, if two images are same, the admissible matching should link a region in image 1 only to the same region in image 2. With this matching, the distance between the two images equals zero, which coincides with our intuition.

Following the Most Similar Highest Priority (MSHP) principle, the IRM algorithm attempts to fulfill the significance credits of regions by assigning as much significance as possible to the region link with minimum distance. Initially, assume that $d_{i1,j0}$ is the minimum distance, we set $s_{i^0,j} = 0$, for j != j^1 since the link between region i^1 and j^1 has filled the significance of region i^1. The significance credit left for region j^1 is reduced to $p'_{j'} - p_{i'}$. The updated matching problem is then solving $s_{i,j}$, $i \neq i'$, by the MSHP rule under constraints:

$$\sum_{j=1}^{n} s_{i,j} = p_i, 1 \le i \le m, i \neq i'$$

$$\sum_{i:1\le i\le m, i\neq i'} s_{i,j} = p_j', 1 \le j \le n, j \neq j' \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots..(5.10)$$

$$\sum_{i:1\le i\le m, i\neq i'} s_{i,j}' = p_{j'}' = p_{i'}$$

$$s_{i,j} \ge 0, 1 \le i \le m, i \neq i'; 1 \le j \le n \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(5.11)$$

We apply the previous procedure to the updated problem. The iteration stops when all the significance credits p_i and p_j^1 have been assigned. Algorithm for updating significance matrix S is:

Step 1: Set L={}, denote M = {(i, j); i=1,……,m; j=1,……,n}

Step 2: Choose minimum $d_{i,j}$ for (i, j)$\in$M – L. Label the corresponding (i,j) as (i^1, j^1)

Step 3: $\min(p_{i'}, p_{j'}') \rightarrow s_{i',j'}$

Step 4: If $p_{i'}, p_{j'}'\ Set, s_{i'j'} = 0, j \neq j'; otherwise, set, s_{i,j'} = 0, i \neq i'$

Step 5: $p_i' - \min(p_{i'}, p_{j'}') \rightarrow p_i'$

Step 6: $p_j' - \min(p_{i'}, p_{j'}') \rightarrow p_j'$

Step 7: $L + \{(i', j')\} \rightarrow L$

Step 8: If $\sum_{i=1}^{m} p_i > 0, and \sum_{j=1}^{n} p_j' > 0,$ goto step 2: otherwise, stop.

We now come to the issue of choosing p_i. The value of p_i is chosen to reflect the significance of region i in the image. If we assume that every region is equally important, then $p_i = 1/m$, where m is the number of regions. Another choice of p_i is the percentage of the image covered by region i based on the view that important objects in an image tend to occupy larger areas. This is referred as *area percentage scheme*. This scheme is less sensitive to inaccurate segmentation than the uniform scheme. For computing $d_{i,j}$ we used simple Euclidean distance.

Chapter6

Experimental Results

This chapter deals with the results of methods which are discussed in chapter 5 of this thesis and retrieving images from the databases using these methods. These methods are first applied on database to compute the features from the database and later these features are used to compute similarity between the query image and database images. We have tested our methods on two different databases, brodatz image database and natural database which consists of different images like flowers, animals, food related, vehicles, sunrise and sunset images, maps, animated images, beaches etc.

Brodatz image database consists of more than 1000 Black and White images which are used as training set. More than 1000 images of different nature are taken from database and tested.

The first section of this chapter shows the results of color and spatial clustering described in chapter 5 of this thesis. Here we explained color and spatial clustering algorithm by taking one image from remote sensing database. The section 2 shows the retrieval results for different query images such as Sunset, Red Car, Beach etc. from natural database. The Section 3 shows the retrieval results for different query images from brodatz image database.

6.1 Results Of Color and Spatial Clustering

The following are the results obtained by using color and spatial clustering. The following image is the remote sensing image taken from the remote sensing database

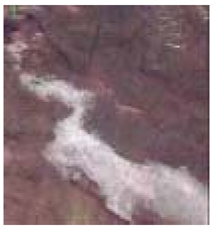


Figure 6.1: Input Image

Input image is processed at two levels, the first one is color clustering and second one is spatial clustering. The following figure is the image obtained after the color clustering process described in chapter 5 of this report.

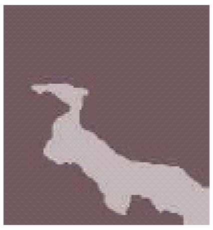
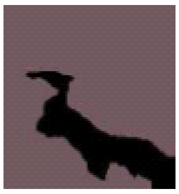


Figure 6.2: input image after color clustering

The above color clustered image contains two colors because all the pixels in the input image fall into two groups that’s why it contains only two colors after the color clustering process. These two colors are the mean values of the color components of pixels of the corresponding group. So the number of colors in the color clustered image represents number of pixel groups formed from the input image using color distance measure described in chapter 4 of this report.

After obtaining color clustered image from the input image it is further divided in to color-spatial clusters. The following are those color-spatial clusters obtained from the above color clustered image.



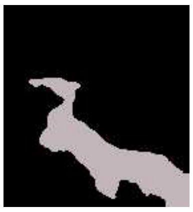


Figure 6.3: color spatial cluster 1 Figure 6.4: color spatial cluster 2

In the above images the black portion of images represents empty portion of image and the remaining color portion represents the one color-spatial cluster of the image. After the spatial clustering processed on the color clustered image figure: 6.2 we got the two color-spatial clusters because in the color clustered image only two connected components. The above figure is the one among the two color-spatial clusters obtained.

The black color in the above image represents empty portion of the image meaning that it is also one color-spatial cluster of color clustered image.

<u>Threshold Value Vs No. of Color clusters</u>

We calculated no. of color clusters for different threshold values for the image given in figure 6.5. It proves that, when threshold value is increasing, numbers of color clusters are decreasing.

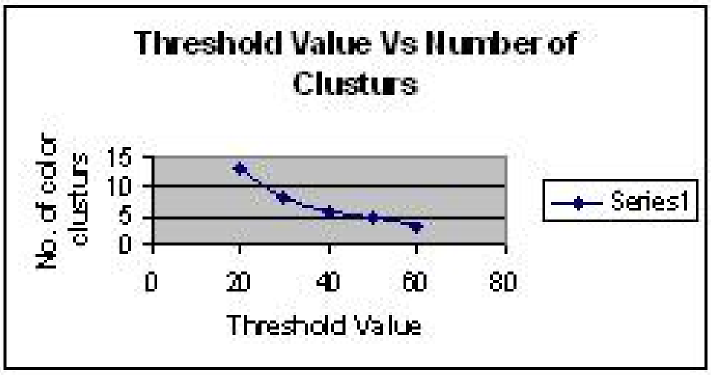
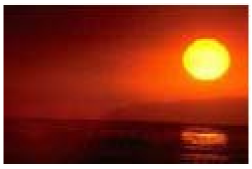


Figure 6.5: Graph for Threshold Value Vs No. of Color clusters

6.2 Retrieved Results from Mixed Natural Database

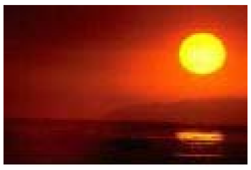
The following are the retrieval results using color and spatial features from mixed natural database consisting of around 1000 images of all types like flowers, sunsets, food, air show, clouds, Vehicles etc. The following result shows that it is also working well with mixed database along with the individual data sets as we discussed in previous section of this chapter.

Ex 1: Retrieved Results for Query Image "Sun Set"



Query Image

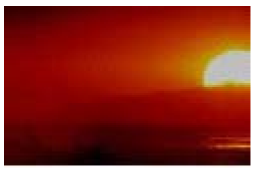
(i)



(ii)

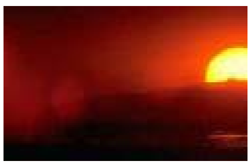
(iii)

(iv)

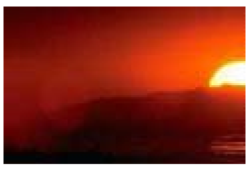


(v)

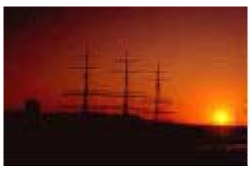
(vi)



(vii)



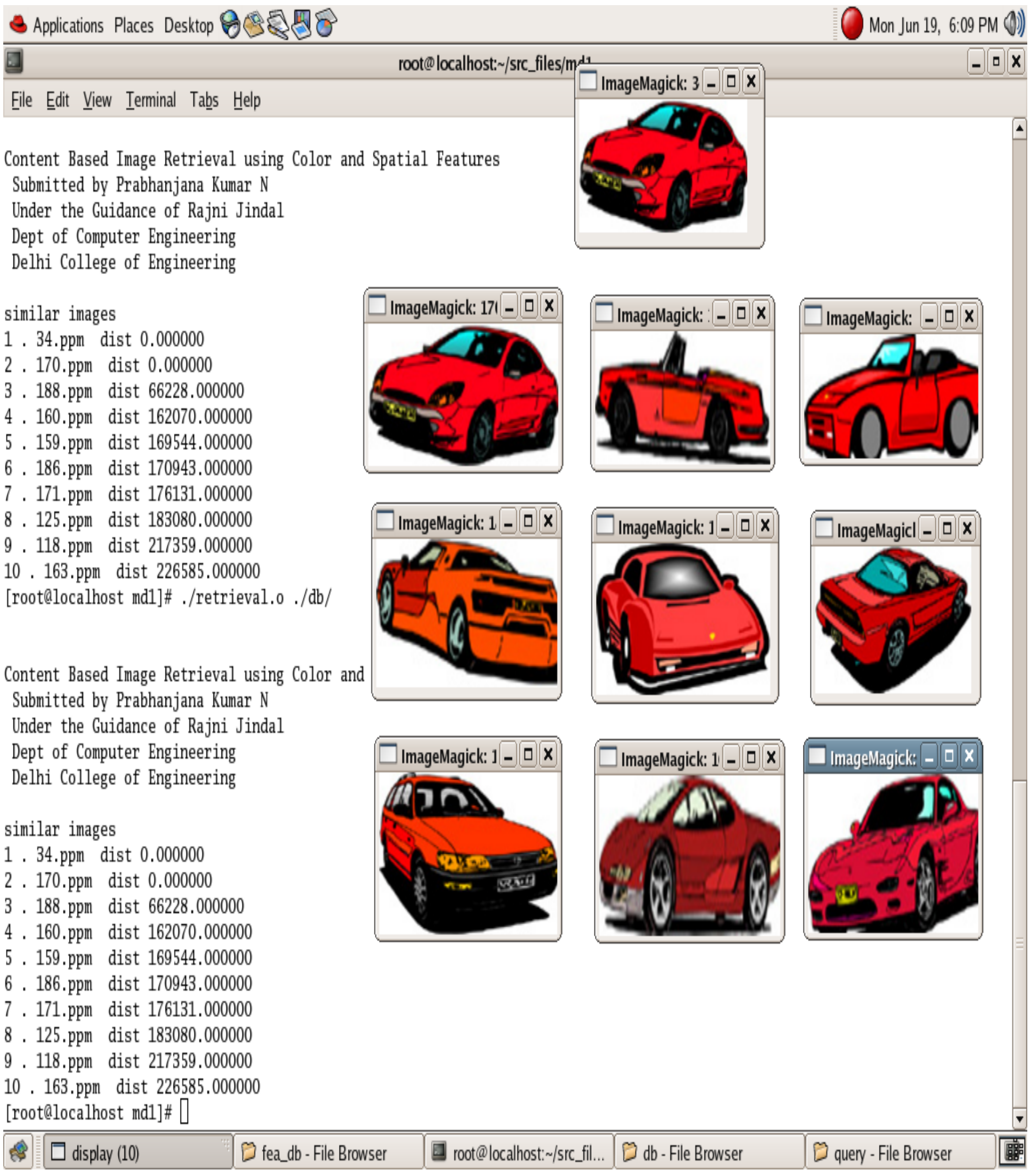
(viii)



(ix)

Figure 6.6: The query image and the retrieved images in the order of similarity from left to right and top to bottom using color and spatial feature

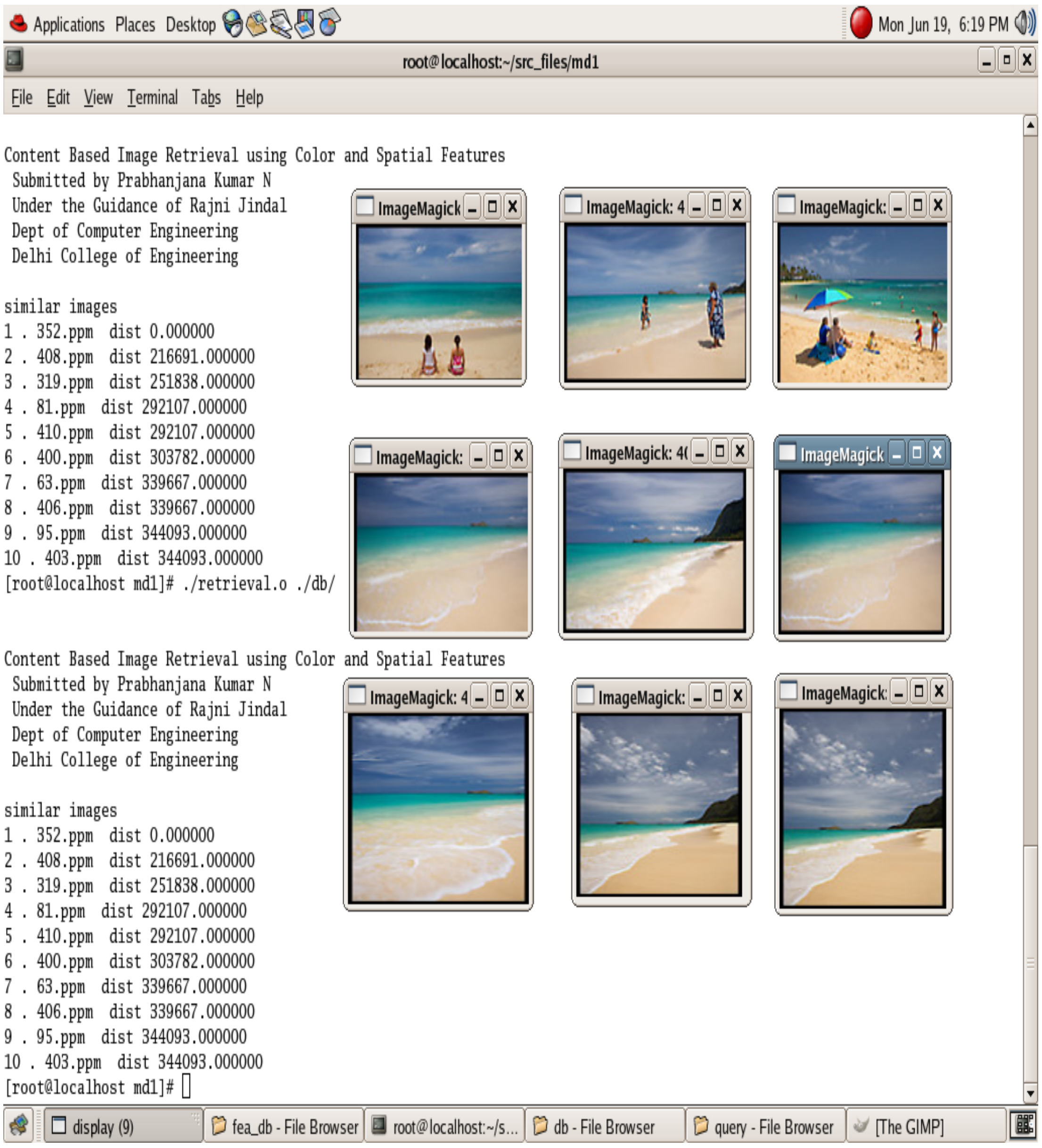
Ex 2: Retrieved Results for Query Image “Red Car”



```
Content Based Image Retrieval using Color and Spatial Features
 Submitted by Prabhanjana Kumar N
 Under the Guidance of Rajni Jindal
 Dept of Computer Engineering
 Delhi College of Engineering

similar images
1 . 34.ppm  dist 0.000000
2 . 170.ppm  dist 0.000000
3 . 188.ppm  dist 66228.000000
4 . 160.ppm  dist 162070.000000
5 . 159.ppm  dist 169544.000000
6 . 186.ppm  dist 170943.000000
7 . 171.ppm  dist 176131.000000
8 . 125.ppm  dist 183080.000000
9 . 118.ppm  dist 217359.000000
10 . 163.ppm  dist 226585.000000
[root@localhost md1]# ./retrieval.o ./db/


Content Based Image Retrieval using Color and
 Submitted by Prabhanjana Kumar N
 Under the Guidance of Rajni Jindal
 Dept of Computer Engineering
 Delhi College of Engineering

similar images
1 . 34.ppm  dist 0.000000
2 . 170.ppm  dist 0.000000
3 . 188.ppm  dist 66228.000000
4 . 160.ppm  dist 162070.000000
5 . 159.ppm  dist 169544.000000
6 . 186.ppm  dist 170943.000000
7 . 171.ppm  dist 176131.000000
8 . 125.ppm  dist 183080.000000
9 . 118.ppm  dist 217359.000000
10 . 163.ppm  dist 226585.000000
[root@localhost md1]# 
```

Figure 6.7: The query image and the retrieved images in the order of similarity from left to right and top to bottom using color and spatial feature

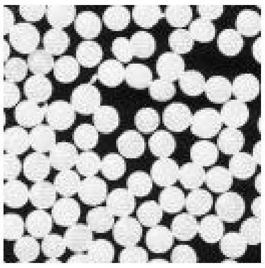
Ex 3: Retrieved Results for Query Image “Beach”



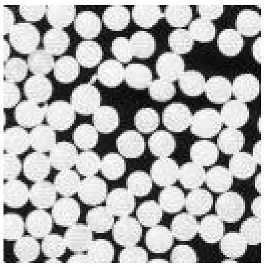
```
Content Based Image Retrieval using Color and Spatial Features
 Submitted by Prabhanjana Kumar N
 Under the Guidance of Rajni Jindal
 Dept of Computer Engineering
 Delhi College of Engineering

similar images
1 . 352.ppm  dist 0.000000
2 . 408.ppm  dist 216691.000000
3 . 319.ppm  dist 251838.000000
4 . 81.ppm  dist 292107.000000
5 . 410.ppm  dist 292107.000000
6 . 400.ppm  dist 303782.000000
7 . 63.ppm  dist 339667.000000
8 . 406.ppm  dist 339667.000000
9 . 95.ppm  dist 344093.000000
10 . 403.ppm  dist 344093.000000
[root@localhost md1]# ./retrieval.o ./db/

Content Based Image Retrieval using Color and Spatial Features
 Submitted by Prabhanjana Kumar N
 Under the Guidance of Rajni Jindal
 Dept of Computer Engineering
 Delhi College of Engineering

similar images
1 . 352.ppm  dist 0.000000
2 . 408.ppm  dist 216691.000000
3 . 319.ppm  dist 251838.000000
4 . 81.ppm  dist 292107.000000
5 . 410.ppm  dist 292107.000000
6 . 400.ppm  dist 303782.000000
7 . 63.ppm  dist 339667.000000
8 . 406.ppm  dist 339667.000000
9 . 95.ppm  dist 344093.000000
10 . 403.ppm  dist 344093.000000
[root@localhost md1]#
```

Figure 6.8: The query image and the retrieved images in the order of similarity from left to right and top to bottom using color and spatial feature

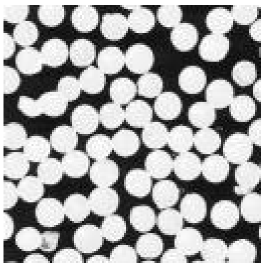
Ex 4: Retrieved Results from Brodatz database



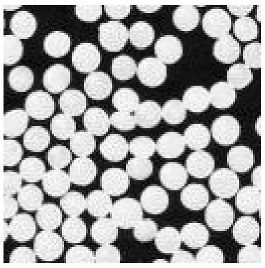
Query Image



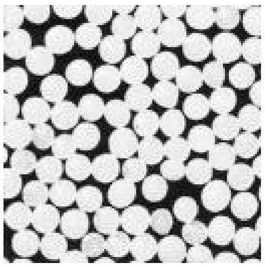
(i)



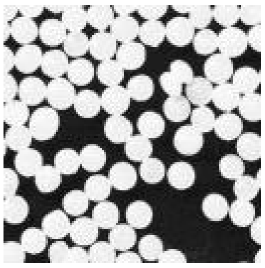
(ii)



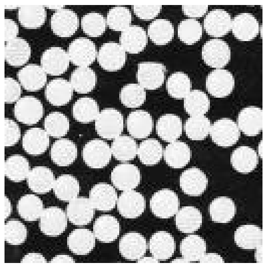
(iii)



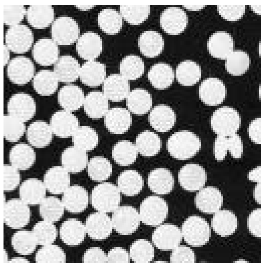
(iv)



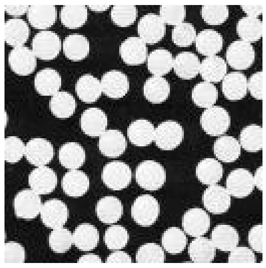
(v)



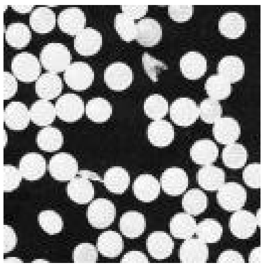
(vi)



(vii)



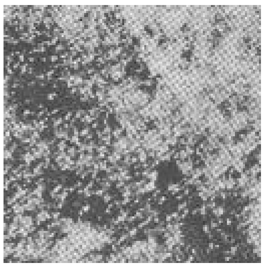
(viii)



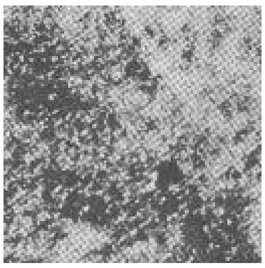
(ix)

Figure 6.9: The query image and the retrieved images in the order of similarity from left to right and top to bottom using color and spatial feature

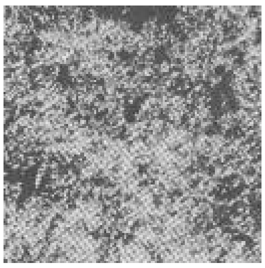
Ex 5: Retrieved Results from Brodatz database



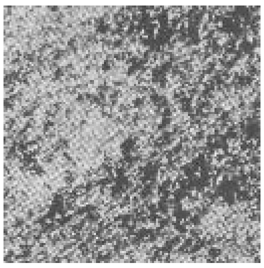
Query Image



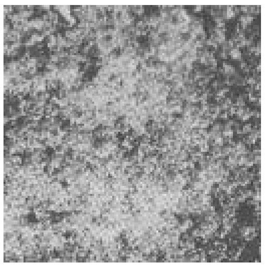
(i)



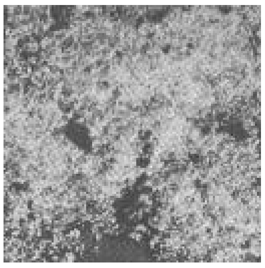
(ii)



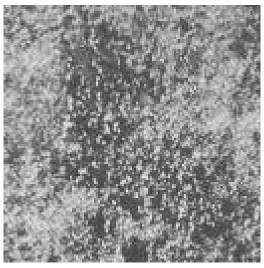
(iii)



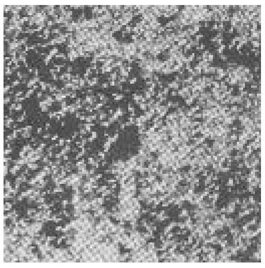
(iv)



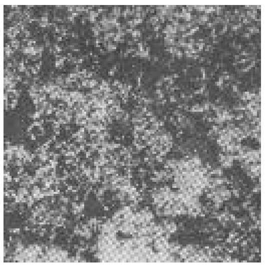
(v)



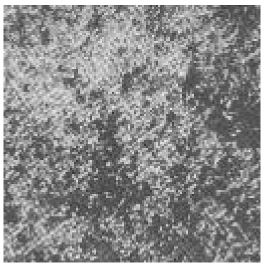
(vi)



(vii)



(viii)



(ix)

Figure 6.10: The query image and the retrieved images in the order of similarity from left to right and top to bottom using color and spatial feature

Chapter 7
Conclusion and Future work

7.1 Conclusion

We have described color - spatial based image retrieval (CSIR) technique for image retrieval by considering both color and spatial features of the image together to improve the retrieval results in terms of its accuracy. This method is applied on L*a* b* color space. This technique has been tested on two separate databases, one is natural database which consists of different images like flowers, animals, food related, vehicles, sunrise and sunset images, maps, animated images, beaches etc. and another one is brodatz image database consisting of more than 1000 images. The color and spatial feature technique performs very well on both brodatz and natural database images. The retrieval output based on this technique gives images that are ranked in order of similarity.

7.2 Future Work

The results of CBIR system presented in this thesis are based on color and spatial clustering, meaning that, first we will do color clustering based on the color and then spatial clustering to separate the colors according to their spatial location in the image and represent it as a feature for the image retrieval. The results may be improved if we consider shape along with the color at the time of clustering and also incorporating an indexing mechanism for efficient retrieval of images and avoid sequential matching of all images in the database.

APPENDIX A

rgb2lab.c

/* It converts rgb color space to lab color space through CIE xyz color space */

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
struct image
{
int r,g,b;
float L,A,B;
};
typedef struct image Pixel ;
void rgbtolab(Pixel **pixel,int height,int width)
{
int i,j;
float var_R,var_G,var_B,var_X,var_Y,var_Z;
float R,G,B,X,Y,Z;
/* RGB TO XYZ    */
for(i=0;i<height;i++)
{
 for(j=0;j<width;j++)
  {
          R=pixel[i][j].r; G=pixel[i][j].g; B=pixel[i][j].b;

          var_R = ( R / 255 );        //R = From 0 to 255
          var_G = ( G / 255 );        //G = From 0 to 255
          var_B = ( B / 255 );        //B = From 0 to 255
          if ( var_R > 0.04045 ) var_R = pow( ( ( var_R + 0.055 ) / 1.055 ) , 2.4);
          else                  var_R = var_R / 12.92;
          if ( var_G > 0.04045 ) var_G = pow( ( ( var_G + 0.055 ) / 1.055 ) , 2.4);
          else                  var_G = var_G / 12.92;
          if ( var_B > 0.04045 ) var_B =pow( ( ( var_B + 0.055 ) / 1.055 ) , 2.4);
          else                  var_B = var_B / 12.92;
          var_R = var_R * 100;
          var_G = var_G * 100;
          var_B = var_B * 100;       //Observer. = 2Â°, Illuminant = D65
          X = var_R * 0.4124 + var_G * 0.3576 + var_B * 0.1805;
          Y = var_R * 0.2126 + var_G * 0.7152 + var_B * 0.0722;
          Z = var_R * 0.0193 + var_G * 0.1192 + var_B * 0.9505;
          /* XYZ TO LAB */
          var_X = X /  95.047;         //Observer = 2Â°, Illuminant = D65;
          var_Y = Y / 100.000;
          var_Z = Z / 108.883;
          if ( var_X > 0.008856 ) var_X = pow( var_X , ( (float)1/3 ));
          else                  var_X = ( 7.787 * var_X ) + ( (float)16 / 116 );
          if ( var_Y > 0.008856 ) var_Y =pow(  var_Y , ( (float)1/3 ) );
          else                  var_Y = ( 7.787 * var_Y ) + ( (float)16 / 116 );
          if ( var_Z > 0.008856 ) var_Z = pow(var_Z , ( (float)1/3 ) );
          else                  var_Z = ( 7.787 * var_Z ) + ( (float)16 / 116 );
          pixel[i][j].L = ( 116 * var_Y ) - 16;
          pixel[i][j].A = 500 * ( var_X - var_Y );
          pixel[i][j].B = 200 * ( var_Y - var_Z );
  }
}
}//end of rgbtolab

void imagereadwrite(FILE *input,FILE *output)
{
char header[3],comment[200],comment1[200];
Pixel **pixel;
int width,height,intensity;
int i,j;
char c,c1;
fscanf(input,"%s",header);
while(c=getc(input)!='\n');
fscanf(input,"%c",&c1);
if(c1=='#')
 {
   fseek(input,-1,1);
   fscanf(input,"%[^\n]",comment);
 }
 else
  fseek(input,-1,1);
fscanf(input,"%d %d %d",&width,&height,&intensity);
pixel=(Pixel **)malloc(sizeof(Pixel*)*height);
  if(pixel==NULL)
    { printf("Error Allocating Memory(rows)for 2D array pixel"); exit(1);
    }
 for(i=0;i<height;i++)
  {
    pixel[i]=(Pixel *)malloc(width*sizeof(Pixel));
     if(pixel[i]==NULL)
     { printf("Error Allocating Memory(cols)for 2D array pixel");
       exit(1);
     }
  }
 for(i=0;i<height;i++)
    for(j=0;j<width;j++)
    {
          fscanf(input,"%d",&pixel[i][j].r);
          fscanf(input,"%d",&pixel[i][j].g);
          fscanf(input,"%d",&pixel[i][j].b);
    }
         rgbtolab(pixel,height,width);
if(c=='#')
 fprintf(output,"%s\n%s\n%d %d\n%d\n",header,comment,width,height,intensity);
else
 fprintf(output,"%s\n%d %d\n%d\n",header,width,height,intensity);
for(i=0;i<height;i++)
    for(j=0;j<width;j++)
    {
        // fprintf(output,"%.1f %.1f ",(float)i,(float)j );
         fprintf(output,"%.1f ",(float)pixel[i][j].L);
             fprintf(output,"%.1f ",(float)pixel[i][j].A);
             fprintf(output,"%.1f\n",(float)pixel[i][j].B);
   }
  for(i=0;i<height;i++)
    free(pixel[i]);
  free(pixel);
}
int main(int argc,char* argv[])
{
  FILE *input,*output;
  char dis[10][100],inputt[100],di[100];
 input=fopen(argv[1],"r");
  output=fopen(argv[2],"w");
imagereadwrite(input,output);
fclose(input);
fclose(output);
}
```

crdb.sh

/* This Program computes features of all images in the image database “db” and saves features into “fea_db” database */

```
#!/bin/bash
j=0

for i in ./db/*.ppm
do
./features.o $i
j=`expr $j + 1`
echo $j
done
```

features.c

/* It extracts color and spatial features using our color clustering algorithm and spatial clustering algorithm which we discussed in chapter 5. */

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
#include<string.h>
#include<dirent.h>
int meanR, meanG, meanB, meanx, meany, X, Y;
int clus_nofp; // no. of pixels in cluster
float calcdist(float r1,float g1,float b1,float r2,float g2,float b2)
{
        float dist;
        dist=(float)sqrt(pow((r2-r1),2)+pow((g2-g1),2)+pow((b2-b1),2));
        if(dist<0.0)
        {
                printf("DIST TURNED TO NEG ! ");
                exit(1);
        }
        return dist;
}
void itoa(int val,char *str,int base)
{
        int i,j=0;
        char tmp;

        while(val!=0)
        {
                str[j++]=(val%base)+48;
                val=val/base;
        }
        str[j]='\0';
        for(i=0,j=j-1;i<j;i++,j--)
        {
                tmp=str[i];
                str[i]=str[j];
                str[j]=tmp;
        }
}
FILE *create(FILE *hdr,int count)
{
        FILE *child;
        char *name,*vname,ch;
        int len=0,temp,dirlen;
        temp=count;
        dirlen=strlen("/root/src_files/md1/temp/");
        while(temp!=0)
        {
                len++;
                temp=temp/10;
        }
        vname=(char*)calloc(1,len*sizeof(char));
        if(vname==NULL)
        {
                printf("ERROR IN MEMORY ALLOCATION OF VNAME ! ");
                fgetc(stdin);       exit(1);
        }
        name=(char*)calloc(1,(dirlen+len+4)*sizeof(char));
        if(name==NULL)
        {
                printf("ERROR IN MEMORY ALLOCATION OF NAME ! ");
                fgetc(stdin);        exit(1);
        }
        name[0]='\0';
        strcat(name,"/root/src_files/md1/temp/");
        itoa(count,vname,10);
        strcat(name,vname);
        strcat(name,".ppm");
        child=fopen(name,"w");
        if(child==NULL)
        {
                printf("ERROR IN CREATING THE CHILD FILE ! ");
                fgetc(stdin);
                exit(1);
        }
        rewind(hdr);
        while(fscanf(hdr,"%c",&ch)!=EOF)
                fprintf(child,"%c",ch);
        free(vname);
        free(name);
        return(child);
}
void write(FILE *child,int r,int g,int b)
{
        fprintf(child,"%d ",r);
        fprintf(child,"%d ",g);
        fprintf(child,"%d ",b);
}
int main(int argc,char *argv[])
{
        FILE *src,*dst,*hdr,*childs,*temp,*labfp,*dst1;
        int lct,chct=0,ct,chr;
        int r1,g1,b1,r2,g2,b2;
        long int srcfppos,labfppos,fbase,count=0;
char ch,dstfile[10], hdrfile[10],touch_str[10], copy_str[4],srcrgbdumpfile[30],
labdumpfile[30], display_sourcefile[30], display_str[10], labfile_creation[40];
char display_sourceimg[20],clus_filename[6],display_clusimg[20];
float l1,la1,lb1,l2,la2,lb2,dist;
```

```
    strcpy(dstfile,"dst.ppm");
    strcpy(hdrfile,"hdr.ppm");
    strcpy(touch_str,"touch ");
    strcat(touch_str,dstfile);
    system(touch_str);
    strcpy(touch_str,"touch ");
    strcat(touch_str,hdrfile);
    system(touch_str);
    strcpy(srcrgbdumpfile,argv[1]);
    strcpy(copy_str,"cp ");
strcat(copy_str,srcrgbdumpfile);
    strcat(copy_str," ");
    strcat(copy_str,"srcrgbdumpfile.ppm");
    system(copy_str);
    strcpy(labfile_creation,"./rgb2lab.o  ");
    strcat(labfile_creation,"srcrgbdumpfile.ppm ");
    strcat(labfile_creation,"labdumpfile.ppm");
    system(labfile_creation);
    strcpy(display_sourcefile,argv[1]);
    strcpy(display_str,"display ");
    strcpy(display_sourceimg,display_str);
    strcat(display_sourceimg,display_sourcefile);
    strcat(display_sourceimg," & ");
    struct dirent **dirlist;
   int nf;
    nf=scandir("./temp/",&dirlist,0,alphasort);
    src=fopen("srcrgbdumpfile.ppm","r+");
    if(src==NULL)
    {
            printf("ERROR IN OPENING THE SOURCE FILE ! ");
            fgetc(stdin);        exit(1);
    }
    dst=fopen("dst.ppm","w+");
    if(dst==NULL)
    {
            printf("ERROR IN OPENING THE DEST FILE ! ");
            fgetc(stdin);         exit(1);
    }
    hdr=fopen("hdr.ppm","w+");
    if(hdr==NULL)
    {
            printf("ERROR IN OPENING THE HEADER FILE ! ");
            fgetc(stdin);     exit(1);
    }
    labfp=fopen("labdumpfile.ppm","r+");
    if(labfp==NULL)
    {
            printf("ERROR IN OPENING THE LABFP FILE ! ");
            fgetc(stdin);      exit(1);
    }
```

```
int loop_repeat=3;
for(lct=0;lct<loop_repeat;lct++)
{
        if(lct==1)
        {
          ch=getc(labfp);
          if(ch=='#')
            loop_repeat=4;
        }
        while((ch=getc(labfp))!='\n')
        {
        }
        while((ch=getc(src))!='\n')
                fprintf(hdr,"%c",ch);
        fprintf(hdr,"\n");
}
while(fscanf(src,"%d",&r1)!=EOF)
{
        fscanf(src,"%d",&g1);
        fscanf(src,"%d",&b1);
        fscanf(labfp,"%f",&l1);
        fscanf(labfp,"%f",&la1);
        fscanf(labfp,"%f",&lb1);
        count++;
        write(dst,256,256,256);
        if(r1==256)
                continue;
        srcfppos=ftell(dst);
        labfppos=ftell(labfp);
        childs=create(hdr,++chct);
        for(ct=0;ct<count-1;ct++)
                write(childs,256,256,256);
        write(childs,r1,g1,b1);
        while(fscanf(src,"%d",&r2)!=EOF)
        {
                fscanf(src,"%d",&g2);
                fscanf(src,"%d",&b2);
                fscanf(labfp,"%f",&l2);
                fscanf(labfp,"%f",&la2);
                fscanf(labfp,"%f",&lb2);
                if(r2==256)
                {
                        write(childs,256,256,256);
                        write(dst,256,256,256);
                        continue;
                }
                dist=calcdist(l1,la1,lb1,l2,la2,lb2);
                if(dist<=40.0)
                {
                        write(childs,r2,g2,b2);
                        write(dst,256,256,256);
                }
                else
                {
                        write(childs,256,256,256);
                        write(dst,r2,g2,b2);
                }
            }
            fclose(childs);
            rewind(dst);
            rewind(src);
            rewind(labfp);
            temp=src;
            src=dst;
            dst=temp;
            fseek(src,srcfppos,0);
            fseek(labfp,labfppos,0);
            for(ct=0;ct<count;ct++)
                write(dst,256,256,256);
    }
    fclose(src);
    fclose(labfp);
    fclose(dst);
    fclose(hdr);
    /* features calculation */
    char fea_file[30];
    char base_file[30];
    char base_file1[30];
    char remove_clusimg[40];
    FILE *p,*p1;
    system("touch ex.txt");
    strcpy(base_file,"basename ");
    strcat(base_file,srcrgbdumpfile);
    strcat(base_file,"  .ppm >> ex.txt ");
    system(base_file);
    p=fopen("ex.txt","r");
    if(p==NULL)
     {
      printf("\n error in openinig ex.txt file \n");
       fgetc(stdin);
       exit(1);
  }
    fscanf(p,"%s",base_file1);
    strcpy(fea_file,"touch ");
    strcat(fea_file,"./fea_db/");
strcat(fea_file,base_file1);
    strcat(fea_file,".fs");
    system(fea_file);
    strcpy(fea_file,"./fea_db/");
strcat(fea_file,base_file1);
```

```
        strcat(fea_file,".fs");
        p1=fopen(fea_file,"w");
       if(p1==NULL)
          {     printf(" error in opening features file \n");
           fgetc(stdin);
           exit(1);      }
        // cal mean and nofp
        char mean_str[20];
        nf=scandir("./temp/",&dirlist,0,alphasort);
        fprintf(p1,"%d \n",nf-2);
        fprintf(p1,"%s\n","Features");
        if(nf<0)
        printf("\n error in scaning dir \n");
        else
                {
                        while(nf!=2)
                                {
                                        nf--;
                                        strcpy(clus_filename,dirlist[nf]->d_name);
                                        strcpy(mean_str,"");
                                        strcat(mean_str,"./temp/");
                                        strcat(mean_str,clus_filename);
                                        mean_cal(mean_str);
        fprintf(p1,"cluster%d\n",nf-2);
fprintf(p1,"%d %d %d %d %d %d \n", meanx,meany,meanR,meanG,meanB,
clus_nofp);
                           }
                }
        nf=scandir("./temp/",&dirlist,0,alphasort);
        if(nf<0)
        printf("\n error in scaning dir \n");
        else
                {
                        while(nf!=2)
                                {
                                        nf--;
                                        strcpy(clus_filename,dirlist[nf]->d_name);
                                        strcpy(remove_clusimg,"rm ");
                                        strcat(remove_clusimg,"./temp/");
                                        strcat(remove_clusimg,clus_filename);
                                      system(remove_clusimg);
                           }
                }
        fclose(p);
        fclose(p1);
        system(" rm dst.ppm hdr.ppm srcrgbdumpfile.ppm labdumpfile.ppm ex.txt ");
        return 0;
}
mean_cal(char fname[])
{
```

```
FILE *fp;
char header[3],comment[200];
int width,height,intensity;
int r,g,b,pcount,mx,my;
long int mr,mg,mb,wxh;
X=Y=0;
meanx=meany=meanR=meanG=meanB=0;
clus_nofp=0;
mr=mg=mb=mx=my=pcount=0;
fp=fopen(fname,"r");
if(fp==NULL)
            {
           printf("\n error in openinig input file \n");
        }

fscanf(fp,"%s %[^\n] %d %d %d",header,comment,&width,&height,&intensity);
wxh=width*height;
while(!feof(fp))
{
 fscanf(fp,"%d",&r);
 fscanf(fp,"%d",&g);
 fscanf(fp,"%d",&b);
 if(r==256 && g==256 && b==256)
  {   if(Y>width)
       { Y=0;        X++;}
      else        Y++;
      continue;
   }
  else
    {
        if(Y>width)
            {          Y=0;    X++;      }
            else        Y++;
            pcount++;
        mx+=X;
            my+=Y;
            mr+=r;
            mg+=g;
            mb+=b;
    }
}
mr/=pcount;     mg/=pcount;
mb/=pcount;     my/=pcount;
mx/=pcount;     meanR=mr;
meanG=mg;      meanB=mb;
meanx=mx;       meany=my;
clus_nofp=pcount;
  fclose(fp);
}
```

Retrieval.c

/* This Program evaluates Similarity Measures of query image and Images in Image database. It retrieves top ten similar images for the given query image */

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
#include <dirent.h>
struct Features
{
  double H;
  double V;
  double MEAN_r;
  double MEAN_g;
  double MEAN_b;
  double SIZE;
  int status;
};
struct DISTANCE
{
  double  dist;
  int i;
  int j;
};
struct image
{
 char name[100];
 char path[100];
 double distance;
};
void imagereadwrite(char *);
void Retrieve_Images(char *path);
double CALC_DISTANCE(char *);
void SORT_DISTANCES(struct DISTANCE* ,int);
void sort_dist(struct DISTANCE*,int,int);
void swap_dist(struct DISTANCE *,int,int);
void SORT_IMAGES(struct image*,int);
void swap_images(struct image*,int,int);
void imagereadwrite(char *path)
{
        char cmd[1000];
            Retrieve_Images(path);
}
int main(int argc,char *argv[])
{
  if(argc<2)
  {
   printf(" USAGE: ./a.out query Directorycontaining database\n");
    exit(1);
  }
  imagereadwrite(argv[1]);
}
void Retrieve_Images(char *path)
{
   int DATA_SET_SIZE;
   int i,j,nf;
   char path1[1000],path2[1000],cmd[1000];
   struct image *img;
   struct dirent **dirlist;
   strcpy(path1,"/root/src_files/md1/fea_db/");
   nf=scandir(path1,&dirlist,0,alphasort);
   if(nf<2)
    perror("no files in the directory");
   DATA_SET_SIZE=nf-2;
   img=(struct image *)malloc(sizeof(struct image)*DATA_SET_SIZE);
   for(i=0;i<DATA_SET_SIZE;i++)
   {
     img[i].distance=0.0;
     strcpy(img[i].name,"");
     strcpy(img[i].path,"");
   }
   for(i=0;i<DATA_SET_SIZE;i++)
  {
      nf--;
     strcpy(path2,path1);
     strcat(path2,dirlist[nf]->d_name);
     img[i].distance=CALC_DISTANCE(path2);
  }
   nf=scandir(path,&dirlist,0,alphasort);
  if(nf<2)
    perror("no files in directory");
  for(i=0;i<DATA_SET_SIZE;i++)
  {
    nf--;
    strcpy(path2,path);
    strcat(path2,dirlist[nf]->d_name);
     strcpy(img[i].name,dirlist[nf]->d_name);
    strcpy(img[i].path,path2);
  }
  SORT_IMAGES(img,DATA_SET_SIZE);
  printf("\n\nContent Based Image Retrieval using Color and Spatial Features \n
Submitted by Prabhanjana Kumar N \n Under the Guidance of Rajni Jindal \n Dept of
Computer Engineering \n Delhi College of Engineering \n\nsimilar images\n");
  for(i=0;i<10;i++)
  {
    printf("%d . %s  dist %lf \n",i+1,img[i].name,img[i].distance);
    strcpy(cmd,"display  ");
    strcat(cmd,img[i].path);
    strcat(cmd," &");
    system(cmd);
  }
  free(img);
}
double  CALC_DISTANCE(char *path)
{
  int i,j,k,cnt;
  int IMAGE_CLUSTERS,QUERY_CLUSTERS;
  char h1[200];
  double ALPHA=0.6,BEETA;
  int stat1,stat2;
  double d1,d2,d3,d4,w;
  double d11,d12,d13;
  FILE *ip,*query;
  struct Features *Img_Features;
  struct Features *Query_Features;
  struct DISTANCE *D;
  ip=fopen(path,"r");
  query=fopen("/root/src_files/md1/query/query.fs","r");
  fscanf(ip,"%d",&IMAGE_CLUSTERS);
  *(h1+0) ='\0';
  fgets(h1, 80, ip);
  *(h1+0) ='\0';
  fgets(h1, 80, ip);
  Img_Features=(struct Features*)malloc(sizeof(struct
Features)*IMAGE_CLUSTERS);
  fscanf(query,"%d",&QUERY_CLUSTERS);
   *(h1+0) ='\0';
   fgets(h1, 80,query);
   *(h1+0) ='\0';
   fgets(h1, 80,query);
   Query_Features=(struct Features*)malloc(sizeof(struct
Features)*QUERY_CLUSTERS);
  D=(struct DISTANCE *)malloc(sizeof(struct DISTANCE
)*(IMAGE_CLUSTERS*QUERY_CLUSTERS));
  for(i=0;i<IMAGE_CLUSTERS;i++)
  {
    Img_Features[i].MEAN_r=0.0;
    Img_Features[i].MEAN_g=0.0;
    Img_Features[i].MEAN_b=0.0;
    Img_Features[i].SIZE=0.0;
    Img_Features[i].H=0.0;
    Img_Features[i].V=0.0;
    Img_Features[i].status=0;
  }
  for(i=0;i<QUERY_CLUSTERS;i++)
  {
    Query_Features[i].MEAN_r=0.0;
    Query_Features[i].MEAN_g=0.0;
    Query_Features[i].MEAN_b=0.0;
    Query_Features[i].SIZE=0.0;
    Query_Features[i].H=0.0;
    Query_Features[i].V=0.0;
    Query_Features[i].status=0;
  }
  for(i=0;i<QUERY_CLUSTERS*IMAGE_CLUSTERS;i++)
  {
   D[i].dist=0.0;
   D[i].i=0;
   D[i].j=0;
  }
  for(i=0;i<IMAGE_CLUSTERS;i++)
   {
      *(h1+0) ='\0';
      fgets(h1, 80,ip);
      fscanf(ip,"%lf",&Img_Features[i].H);
      fscanf(ip,"%lf",&Img_Features[i].V);
      fscanf(ip,"%lf",&Img_Features[i].MEAN_r);
      fscanf(ip,"%lf",&Img_Features[i].MEAN_g);
      fscanf(ip,"%lf",&Img_Features[i].MEAN_b);
      fscanf(ip,"%lf",&Img_Features[i].SIZE);
      *(h1+0) ='\0';
      fgets(h1, 80,ip);
   }
   for(i=0;i<QUERY_CLUSTERS;i++)
   {
      *(h1+0) ='\0';
      fgets(h1, 80,query);

      fscanf(query,"%lf",&Query_Features[i].H);
      fscanf(query,"%lf",&Query_Features[i].V);
      fscanf(query,"%lf",&Query_Features[i].MEAN_r);
      fscanf(query,"%lf",&Query_Features[i].MEAN_g);
      fscanf(query,"%lf",&Query_Features[i].MEAN_b);
      fscanf(query,"%lf",&Query_Features[i].SIZE);
      *(h1+0) ='\0';
      fgets(h1, 80,query);
   }
  fclose(ip);
  fclose(query);
  cnt=0;
    for(i=0;i<IMAGE_CLUSTERS;i++)
    {
      for(j=0;j<QUERY_CLUSTERS;j++)
      {
        Img_Features[i].status=0;
            Query_Features[j].status=0;
            d11=(double)abs(Img_Features[i].MEAN_r-Query_Features[j].MEAN_r);
            d12=(double)abs(Img_Features[i].MEAN_g-Query_Features[j].MEAN_g);
            d13=(double)abs(Img_Features[i].MEAN_b-Query_Features[j].MEAN_b);
```

```
            d1=sqrt(d11*d11+d12*d12+d13*d13);
            d2=(double)abs(Img_Features[i].H-Query_Features[j].H);
            d2=d2*d2;
            d3=(double)abs(Img_Features[i].V-Query_Features[j].V);
            d3=d3*d3;
            d2=d2+d3;
            d4=sqrt(d2);
            D[cnt].dist=(double)abs((ALPHA*d1+(1-ALPHA)*d4));
            D[cnt].i=i;
            D[cnt++].j=j;
     }
    }
    SORT_DISTANCES(D,cnt);
    BEETA=0.0;
    for(k=0;k<cnt;k++)
    {
      i=D[k].i;
      j=D[k].j;
      stat1=Img_Features[i].status;
      stat2=Query_Features[j].status;
      if(stat1==0 && stat2==0)
      {
            if(Img_Features[i].SIZE<Query_Features[j].SIZE)
            {
              w=Img_Features[i].SIZE;
              Query_Features[j].SIZE=Query_Features[j].SIZE-w;
              Img_Features[i].status=1;
            }
            else
            {
             w=Query_Features[j].SIZE;
             Img_Features[i].SIZE=Img_Features[i].SIZE-w;
              Query_Features[j].status=1;
             if(Img_Features[i].SIZE==0)
                Img_Features[i].status=1;
            }
        BEETA=BEETA+w*D[k].dist;
      }
    }
    free(Img_Features);
    free(Query_Features);
    free(D);
    return BEETA;
}
void SORT_DISTANCES(struct DISTANCE *D,int count)
{
  int i,j;
  int boolean;
  for(i=0;i<count;i++)
  {
    boolean=0;
    for(j=0;j<count-1;j++)
    {
      if(D[j].dist>D[j+1].dist)
          {       boolean++;
              swap_dist(D,j,j+1);           }
    }
    if(boolean==0)
    break;
  }
}
void swap_dist(struct DISTANCE *D,int i,int j)
{
  double temp;
  int tmp;
  temp=D[i].dist;       D[i].dist=D[j].dist;      D[j].dist=temp;
  tmp=D[i].i;           D[i].i=D[j].i;                  D[j].i=tmp;
   tmp=D[i].j;          D[i].j=D[j].j;            D[j].j=tmp;
}
void SORT_IMAGES(struct image *img,int count)
{
  int i,j;
  int boolean;
  for(i=0;i<count;i++)
  {
    boolean=0;
    for(j=0;j<count-1;j++)
    {
      if(img[j].distance>img[j+1].distance)
            {      boolean++;
                  swap_images(img,j,j+1);        }
      }
   if(boolean==0)
   break;
  }
}
void swap_images(struct image *img,int i,int j)
{
  char  temp[100];
  double tmp;
  tmp=img[i].distance;
  img[i].distance=img[j].distance;
  img[j].distance=tmp;
  strcpy(temp,img[i].name);
  strcpy(img[i].name,img[j].name);
  strcpy(img[j].name,temp);
  strcpy(temp,img[i].path);
  strcpy(img[i].path,img[j].path);
  strcpy(img[j].path,temp);
}
```

Appendix B

The following are instructions to run the project

Make sure that you are in LINUX operating system before running programs.

1) Enter into root's home directory, create one folder src_files
2) In src_files directory, create one folder with name md1
3) Put rgb2lab.c, features.c,retrieval.c and crdb.sh files in the md1 directory.

procedure:

1. please create a temporary directory named **"temp"** in the current directory i.e in the **"md1"** directory.
2. Open the program named **"features.c"** after that search for the pattern **"username"** in that program , please replace all the occurrences of **"username"** with your currently Linux user name (for example "**prabhanjan"** is your user name then replace **"username"** with **"prabhanjan"** in the program**).** I think you mostly find two occurrences of **"username"** in the function named **"create".** So please change properly those occurrences. actually, it is /home/**username**/src_files/md1/temp/
if your Linux user name is "**prabhanjan**" ,please replace it as a
/home/prabhanjan/src_files/md1/temp/ thats all.
3. now, compile the file **"rgb2lab.c"**

gcc rgb2lab.c -lm -o rgb2lab.o

gcc features.c -lm -o features.o

4. Now, you need to create directory named **"fea_db"** to keep features data.
5. Put 1.ppm image(take one jpeg image and save it as 1.ppm (ASCII mode)) in md1 directory.
6. To check, program is working or not, try at the command prompt

./features.o 1.ppm

if everything is ok it will create a file named “**1.fs”** in the directory **“fea_db”**. please check the file **“1.fs”**

vi ./fea_db/1.fs

there will be a some data in that file if everything is ok ,otherwise please look back whether you done it correctly or not.

7. Now, please keep your all color images in **ppm**(one of the image format) in the directory named **“db”**

8. run the following shell file to create features database

chmod a+x crdb.sh

./crdb.sh

It will take lot of time depends on number of images in the database i.e in the directory **“db”**

9. create a directory named **“query”**

10. After successful completion of feature database, for image retrieval please do in the following way

a) please replace **“username”** with your user name as explained in **step2** in the file **“retrieval.c”**

b) gcc retrieval.c -lm -o retrieval.o

11. copy one image feature to directory **“query”**

cp ./fea_db/bird.fs ./query/query.fs

here **bird.fs** is an example file

12. For retrieval ,

./retrieval.o ./db/

It will give top ten images similar to query image.

References:

[1] TIMOTHY K. SHIH, JIUNG-YAO HUANG, CHING-SHENG WANG, JASON C. HUNG, AND CHUAN-HO KAO "An Intelligent Content-based Image Retrieval System Based on Color, Shape and Spatial Relations" *Proc. Natl. Sci. Counc. ROC(A)* Vol. 25, No. 4, 2001. pp. 232-243

[2] Chung-Lin Huang, Dia-Hwa. huang, "A content based image retrieval system", *Image and Vision Computing*, vol. 16, pp. 149-163, 1998.

[3] W.Niblack et al, "The QBIC project: querying images by content using color, texture and shape", *Proc. SPIE Int. Soc. Opt. Eng.*, in *Storage and Retrieval* for *Image and Video Databases*, vol. 1908, pp. 173-187, 1993.

[4] A.D. Bimbo, *Visual information retrieval*, Morgan Kaufmann Publishers, San Franscisco, California, 1999.

[5] J. Smith, and S.F. Chang, "VisualSEEK: a fully automated content-based image query system", in *Proceedings of the 4th ACM Multimedia Conference,* Boston, MA, pp. 87-98, 1996.

[6] Badrinarayan Raghunathan and Scott T. Acton, "Content based Retrieval for remotely sensed imagery", The Oklahoma Imaging Laboratory, work supported by NASA EPSCoR.

[7] Jia Li, James Z. Wang and Gio Wiederhold, "IRM: integrated region matching for image retrieval", in *Proceedings of the 8th ACM International Conference on Multimedia*, Los Angles, California, USA, pp. 147-156, October 2000.

[8] *Color-based retrival* "IRM: integrated region matching for image retrieval", in *Proceedings of the 8th ACM International Conference on Multimedia*, Los Angles, California, USA, pp. 147-156, October 2000.

[9] Y.Deng, B .Manjunath, C.Kenney, M.Moore, and Y.Shin, *An efficient color representation for image retrieval.* IEEE Trans. Image Processing, 10(1):140-147, Jan. 2001.

[10] Mohan S.Kankanhali ,Babu M. Mehtre, Hock Yiung Huang *color and spatial feature for content based image retrival*, Pattern Recognition Letters 20 (1999) 109-108.

[11] Renato O.Stehling, Mario A. Nascimento and Alexandre X.Falco *An Adaptive and Efficient Clustering-based Approach for content based image retrieval*, IEEE transactions on image processing 2001.

[12] H.Lu, B.c.oi, and K.L Tan *Efficient image retrieval by color contents*. In Proc. of the 1994 Intl. Conf. on Applications of databases,pages 95-108,1994.

[13] F.Rabitti and P.Savino. *An information retrieval approach for image databases*. In proc. of the 18th INt. Conf. on Very Large Databases,pages 574-584,1992.

[14] C. Faloustsos et al. *Efficient and effective querying by image content*. Journal of Intelligent Information systems 3(3/4):231-262.1994.

[15] Markus Koskela, Jorma Laaksonen, and Erkki Oja *Comparison of Techniques for Content-Based Image Retrieval*. Proceedings of SCIA 2001,Bergen, Norway, June 2001.

[16] Vito Di Gesu, Valery Starovoitov *Distance-based functions for image comarision*. Pattern Recognition Letters 20 (199) 207-214 september 1998.

[17] Yung-Kuan Chan , Chih -Ya Chen *Image rettrieval system based on color-complexity and color-spatial features*. The journal of Systems and Software 71 (2004) 65-70, 2 october 2002.

[18] C. Carson, M. Thomas, S. Belongie, J. Hellerstein, and J. Malik, “Blobworld: a system for region-based image indexing and retrieval”, in *Proceedings of the 3rd International Conference on Visual Information systems*, Amsterdam, the Netherlands, pp. 509-516,1999.

[19] B.M. Mehtre, M. Kankanhalli and W.F. Lee, “Shape measures for content based image retrieval: A comparision, Information Processing and Management”, 33:319-337, 1997.

[20]. R.C. Gonzalez and R.E. Woods, *Digital image processing*, Addison Wesley Publishing Company, 1992.

[21] Stricker, M, orengo , M, 1995 *Similarity of color images*. In Proc .Storage and retrieval for image and video Databases, Vol. III,vol 2420,pp. 381-3

[22] Sagarmay Deb, Yanchun Zhang, "An Overview of Content-based Image retrieval Techniques"

[23] Gudivada, V.N. and Raghavan, V.V. (1995). Content-Based Image Retrieval Systems, *IEEE*,

[24] Eakins, J.P. and Graham, M.E. (1999). Content-based image Retrieval: A report to the JISC Technology Application Programme, Institute for Image Data Research, University of Northumbria at Newcastle,U.K.

[25] Forsyth.D.A and others.(1996). Finding pictures of objects in large collections of images, *Digital Image Access and Retrieval: 1996 Clinic on Library Applications of Data Processing*(Heidon,P.B and Sandore,B,eds),118-139, graduate School of Library and Information Science, University of Illinois at Urbana-Champaign.

[26] Flicker, M., H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker (1995) Query by image and video content: the QBIC system. *IEEE Computer Magazine*, **28**(9), 23-32.

[27] Bach, J. R., C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, and R. Jain (1996) The Virage image search engine: An open framework for image management. *SPIE Storage and Retrieval for Still Image and Video Databases*, 76-87.

[28] Pentland, A., R. Picard, and S. Sclaroff (1996) Photobook: content-based manipulation of image databases. *International Journal of Computer Vision*, **18**(3), 233-254.

[29] Alshuth, P., Termes, C. Klauck, J. Kreiss, and M. Roper (1996) IRIS image retrieval for images and video. *First Int'l. Workshop on Image Database and Multimedia Search*, pp. 170-179, Amsterdam, Netherlands.

[30] Dowe, J. (1993) Content-based retrieval in multimedia imaging. *SPIE Conference on Visual Communication and Image Processing*, Boston, MA, U.S.A.

[31] Ma, W. Y. and B. S. Manjunath (1997) Netra: a toolbox for navigating large image databases. *ICIP'97*, pp. 568-571, Santa Barbara, CA, U.S.A.

[32] Wu, J. K., A. D. Narashihalu, B. M. Mehtre, C. P. Lam, and Y. J. Gau (1995) CORE: a content-based retrieval engine for multimedia information systems. *Multimedia Systems*, **3**, 25-41.

[33]. M. La Cascia and E. Ardizzone, "JACOB: Just A content-based query system for video databases", *Proc. ICASS,* Atlanta, 1996.

[34]. R.K.Srihari, "Automatic indexing and content-based retrieval of captioned images", *Computer,* Vol.28, no.9, Sept. 1995.

[35].Y.A. Aslandogan, C. Their, T.C. Yu, C. Liu and K.Nair, "Design, implementation, and evaluation of SCORE (A System for COntent-based REtrieval of Picture)", *In Proc. IEEE ICDE*, Pages 280-287, March 1995.

[36]. The Virage System, available online at http://www.virage.com

[37]. The Informix System, available online at http://www.informix.com

[38]. The Excalibur System, available online at http://www.excalib.com

[39]. The WebSeek System, available online at http://www.webseek.com

[40]. V.E.Ogle and M.Stonebraker, "Chabot: Retrieval from a relational database of images", *Computer*, Volume 28, no.9, Pages 40-48, Sept. 1995.