IMAGE MINING METHODOLOGIES FOR CONTENT BASED RETRIEVAL

A Thesis
Presented to
The Academic Faculty

by

Prajakta Kalmegh

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
College of Computing

Georgia Institute of Technology
May 2010

IMAGE MINING METHODOLOGIES FOR CONTENT BASED RETRIEVAL

To my Father for being my greatest motivation

and

my Mother for her unconditional love

ACKNOWLEDGEMENTS

I would like to thank Dr.Shamkant Navathe for his guidance and support in the last two years of my association with Georgia Institute of Technology. I am deeply gratified to him for showing me new research directions and guiding along the way.

I would also like to thank Dr.Edward Omiecinski and Dr.Vijay Madisetti for their technical insights in the research, valuable feedback in improving the work and agreeing to be on the committee.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

This thesis presents a system for content based image retrieval and mining. The research presents a design of a scalable solution for efficient retrieval of images from large image databases using image features such as color, shape and texture. A framework for automatic labeling of images and clustering of meta data in database based on the dominant shapes, textures and colors in the image is proposed. The thesis also presents a new image tagging methodology to annotate the dominant image features to the image as meta data.

The users of this system can input a query image and select similar image retrieval criteria by selecting a feature type from amongst color, texture or shape. The system retrieves images from the database that match the specified pattern and displays them by relevance. The user can enter a set of keywords or a combination of keywords that form the input text query. Images in the database that match the input text query are fetched and displayed. This ensures content based similar image search even for text based search. An efficient clustering algorithm is shown to improve the image retrieval by an order of magnitude.

CHAPTER I

INTRODUCTION

Content based image retrieval (CBIR) is an automatic process for searching relevant images based on image features and user inputs. Content based image mining (CBIM) is the process of extracting patterns from image databases. To extract patterns from image databases CBIR techniques can be used. For CBIM systems, organized image databases and efficient browsing, storing, and retrieval algorithms are very important.

In recent years the focus of image retrieval has shifted from text-based to visual-based retrieval. Text-based image retrieval became popular since 1970s. It involved annotating the image with keywords and using text-based database management systems (DBMS) to retrieve the images. In text-based image retrieval system, keywords of semantic information were attached to the images. In the 1990s, due to rapid developments in processor, memory and storage technologies, the size of image databases began to grow dramatically. As the size of image databases grew, the demand of labor for annotating the images also rose dramatically. Indexing and searching a large image database using keywords became time-consuming and inefficient. Content based image retrieval automated the complex process of retrieving images that are similar to the reference image or descriptions given. CBIR widely used to describe the process of retrieving desired images from a large collection on the basis of features (such as color, texture and shape) that can be automatically extracted from the images themselves. The features used for retrieval can be either primitive or semantic, but the extraction process must be predominantly automatic.

The content based image retrieval systems try to mitigate the sensory and semantic gap between the user queries and the images in the database by extracting image features and finding patterns in the images. This research mitigates the sensory gap - the gap between the real world objects and their representation - by identifying the key features that uniquely characterize the images. For example, the present day similar image search engines for large image database, which rely on the text tagging, essentially lose the semantic description by not considering the actual content of the images. Searching similar images by individual features or as a combination is the key of our research that reduces the semantic gap in interpretation of information by the user and its actual representation in the database.

In 2001, Motion Picture Expert Group (MPEG) released the MPEG-7 standard, which is formally called Multimedia Content Description Interface. MPEG-7 specifies a standard set of descriptors that can be used to describe various types of multimedia information. The major purpose of MPEG-7 is to provide a description of multimedia contents for effective searching, identifying, filtering, and browsing of multimedia such as images, videos, audios, and other digital or even analogue materials. In most cases, the MPEG-7 standard only partly describes how to extract these features, to leave a maximum of flexibility to various applications. In addition, how the MPEG-7 descriptors are being used for further processing is not specified by MPEG-7. In particular, the similarity between images, audio or video is left to the specific application developers based on the application requirements.

Moreover, the focus of image formats till now has been on compression and quality. Extracting features from images is a time consuming and computationally intensive process and the extracted features are stored in separate databases which are not accessible to other search engines. For image search based on content, different types of color, shape and texture features have been used by different content based image search engines. This leads to compatibility problems between different search engines as no standard has been adopted to completely define the features of images.

The work presented here proposes a new image tagging format called Search Friendly Image Format or SFIF in short. The SFIF is based on the MPEG-7 [2] multimedia content description format. The SFIF includes meta information in images so as to define a scalable and inter-operable image format that represents a unique standard to define image content. This will ensure faster image mining for applications using the SFIF meta information embedded within the image as tags.

In addition to this, the research presents a set of technologies that facilitates efficient image data storage, faster retrieval of image characteristics and portable search of similar images using the meta data information in SFIF. The thesis presents the creation of an object dictionary and a feature dictionary that serve as a reference lookup for feature descriptors and object shape descriptors for all images in the database. The thesis also describes an implementation of object and feature extraction engines that extract MPEG-7 feature descriptors and object descriptors from an image. Finally we describe the similar image retrieval module that fetches images similar to the input image by an efficient lookup mechanism.

The thesis is structured as follows. Chapter 2 compares and contrasts our research with some of the leading image search engines. Chapter 3 gives a brief overview of the design approach from the database and user's perspective. Chapter 4 explains the architecture of the system which includes the following: the implementation details of the feature and shape extraction engines, construction of the shape and feature dictionary, search tag clustering algorithm, and SFIF image tagging. Chapter 5 presents the Web Based User interface implementation details. We explain the use of Oracle Multimedia database and ORDImage API in Chapter 6. Chapter 7 lists a set of probable applications of our system in commercial and academic areas. Next we will see the evaluation of the system performance followed by a summary of results for a sample set of 10000 images in Chapter 8. The thesis concludes with a summary of the key features of the project.

CHAPTER II

PREVIOUS WORK

A large amount of unstructured image data that resides in distributed multimedia repositories has lured researchers to come up with efficient image querying and retrieval mechanisms. The work in image retrieval can be broadly classified into the following sections.

2.1 Free browsing through the data

Users generally browse the repositories manually to find the desired data. Before the research in image retrieval most of the systems allowed the users to scan through the repositories manually. This is a time-consuming tedious process and is not suitable for large image databases.

2.2 Text Based retrieval

This is based on the text based tagging or annotation of images. A user query translates into a text search on this tagged information. Most of the commercial image retrieval engines used to rely on automatic or manual tagging of images. Research in MMDBs (Multi Media Database Systems) in the early 1980s was motivated with the text based tagging and retrieval methods. However, the approach fails where the user wishes to find a pattern in the images. Also, the user has to essentially rely on the credibility of the tagging process. [12] and [13] are an excellent example of the research motivated by the text based retrieval. [15] is a survey that highlights the gap in user's semantic queries and the shortcomings of text based retrieval systems in meeting these requirements.

2.3 Content Based Retrieval

To overcome the drawbacks of the above two approaches, researchers in the fields of image processing and digitization came up with alternatives to search images based on its content. This search is usually based on querying through the image semantics represented by image features like color, shape, texture etc. The tagging of images is done by multi-dimensional indexing of these image features. The user query specified by text, image, or hand-drawn sketch is translated into a database query using a feature extraction engine.

2.3.1 QBIC

The pioneering work in the field of content based image retrieval (CBIR) was done in early nineties by Kato[7]. They developed an automated system for image retrieval using color and shape features. IBMs QBIC[8] (Query By Image Content) was the first commercial CBIR system. It allowed the user to upload an image query, sketches and drawings, and some color and texture patterns, etc.

2.3.2 Virage

Virage [16] is a content-based image search engine developed at Virage Inc. Virage goes one step further than QBIC and supports combinations of visual queries based on color, composition, texture, and structure.

2.3.3 Photobook

MIT Media Labs Photobook[9] presents a set of interactive tools for browsing and searching images. It consists of three sub-books from which shape, texture, and face features are extracted, respectively. The system allows the users to query based on the corresponding features in each of the three sub-books.

2.3.4 MARS

Multimedia analysis and retrieval system [10], is the integration of database management system, and information retrieval (exact match with ranked retrieval), indexing and retrieval. The system focused on how to organize various visual features into a meaningful retrieval architecture which can dynamically adapt to different applications.

2.3.5 SIMPLIcity

Semantics-sensitive Integrated Matching for Picture Libraries [11] classified images into global semantic classes, such as textured or non-textured, graph or photograph. Using these classes as a reference, the system retrieved similar images from the database. The system required to tailor the feature extraction scheme to best suit each class.

2.3.6 Other Systems

RetrievalWare is a content-based image retrieval engine developed by Excalibur Technologies Corp.[17, 18]. VisualSEEk [19] is a visual feature search engine which introduced the idea of a feature extraction engine[20]. WebSEEk [21] is an internet based text/image search engine that explores the spatial relationship query of image regions. Netra[32] uses color, texture, shape, and spatial location information in the segmented image regions to search and retrieve similar regions from the database.

2.3.7 Our Research

A high volume of research has been done in the areas of image feature extraction and data mining [3] of images individually. The research in data mining has given many methods which can benefit the CBIR systems. The research presented here is an attempt to apply such methods to successful content based retrieval of similar images from a large image database. Our system uses 'pattern based high dimensional clustering' and 'image indexing for similarity based search' [3] as its key components of image retrieval.

Our system gives flexibility to the application developers to chose the implementation method used for extraction of MPEG-7 feature descriptors. The implementation of image shape and feature extraction engines can vary. The system associates the feature descriptors with the image records in the database during the pre-processing of image sets. The similarity measure computation used to identify similar images can also vary based on application requirements. The focus of our system is to define an efficient search tag clustering algorithm to facilitate fast retrieval of similar images from the database.

Text based search engines that facilitate the user to refine the search by individual features (e.g. color based search in Google Similar Images Search Engine) still rely on the image tags rather than the content to display the results. The mechanism used to define a Search Friendly Image Format (SFIF) makes this set of images interoperable with other systems which rely on crawling the image headers for image feature extraction.

CHAPTER III

OVERALL APPROACH TO CONTENT BASED IMAGE MINING

The focus of this system is efficient retrieval of image information rather than image data from large image databases. This research is an attempt to integrate the information visualization of images (creation of a database based on image contents) and the data mining principles in finding a pattern using multi-dimensional clustering and image indexes (retrieval of images based on its semantics).

Figure 1 shows the main components in system design. The system architecture has three levels in design - the database at the bottom, the middle level of image content extraction engines and the user interaction with the system at the top.

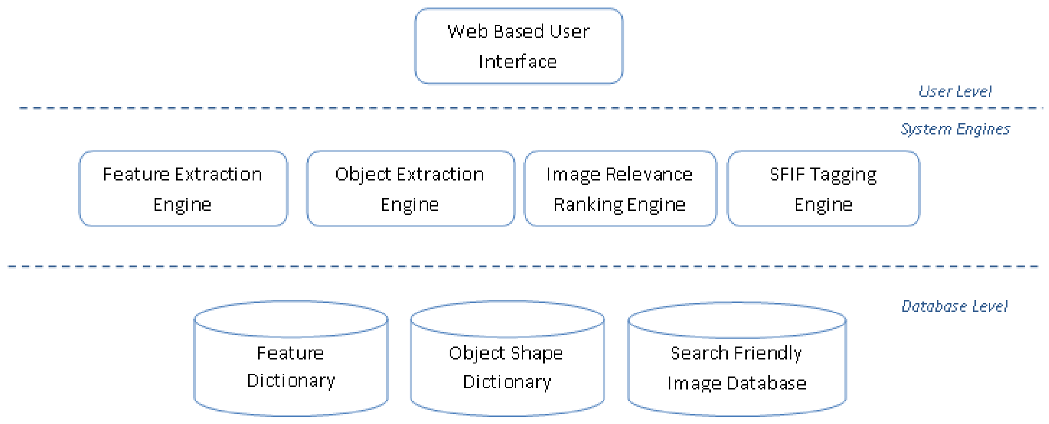


Figure 1: System Components

The database is designed in a way so that the system is scalable to accommodate a diversity of image features. The object dictionary, which consists of a reference lookup for an exhaustive list of shapes represented by the system, can be extended with ease to support a variety of shape boundaries. Similarly, the feature dictionary can also be extended to include a number of sample colors and textures which exhaustively represent the kind of images expected to be supported by the system. A higher count of sample sets in object and feature dictionaries will give a precise resultset to the end user. In short, the system design has taken into account the inter-dependence of these modules and hence is proposed with scalability and flexibility of implementation as primary goals. The following sections explain the system design from the database designer's perspective and from the user's perspective.

3.1 Database Design Strategy

The system uses MPEG-7 descriptors [1] to extract significant image features of color, texture and shape. These features uniquely identify the images semantically. The second most important design decision was the choice of a multimedia database. The system uses Oracle Multimedia 11g R1 as its backbone and Oracle multimedia API packages to represent image objects in the database. The results demonstrate the efficiency of using Oracle Multimedia as against storing images as BLOBs or flat file references in database. Also, the use of Oracle Multimedia as backbone made possible the annotation of search friendly image information with the image headers as meta data which makes them inter-operable with other systems. Finally, the design focuses on efficient management of data within the database. This design facilitates efficient image mining in our system. In fact, the overall design approach is centered around achieving high performance by implementing a clustering algorithm that is used to group image features in the database. The following sections give a brief approach towards our design strategy.

3.1.1 Image Feature Extraction

Considerable research has been reported in the high level image feature extraction domain. Image features include both text based features (image format, orientation, size, text annotations, keywords etc) and visual features (color, shape, texture, intensity, saturation etc). For this study, we are not considering domain specific features which classify images based on image object characteristics (eg. face detection, image background landscape, images containing crowd etc).

The text based features are computed while creating the image feature dictionary. For computing the visual features, there were a number of implementations available from various image visualization and segmentation groups. For dominant color histogram extraction, the possible choices were Swain and Ballard's L1 Metric [22], Niblack et al. [23] 's L2-metric, and Stricker and Orengo's [24] cumulative color histogram approach. Tamura et al.[25] proposed a visually meaningful texture metrics in terms of coarseness, contrast, directionality, line likeness, regularity, and roughness which was used as a basis in the implementation of the QBIC and MARS systems. In [26], Ma and Manjunath evaluated the texture image annotation by various wavelet transform representations. Finally for shape representation, the available choices were Rui et al.'s [27] modified Fourier descriptor, and Gross and Latecki's [28] approach to preserve qualitative differential geometry of the object boundary.

However, since the focus of the research is not to define an image feature extraction methodology, we are using open source implementation of MPEG-7 multimedia content description format [29] for image feature extraction. MPEG-7 has defined a set of standard descriptors for low level features such as color, shape and texture [2]. The Feature Extraction Engine, which uses the open source Java implementation [29], extracts the Scalable color descriptors (SCD) for dominant color representation.

We extract the dominant color in every image based on its highest percentage. The extraction engine extracts the edge histogram descriptor (EHD) for image texture representation. The object extraction engine extracts the Contour Shape descriptor (CSD) for object shape description. The engine stores these descriptors in a single dimensional array string representation in the database. The choice of MPEG-7 as visual descriptors for image features saved a considerable time in implementation and also led to dimensionality reduction in image feature representation in the database storage space.

3.1.2 Image Database Creation

The image database creation consists of a process that scans through all the images off-line and extracts its features using the image feature extraction engine and object extraction engine. Once the feature descriptors are extracted, the descriptors are stored in the database along with preliminary information about images and text tag annotation.

Next, based on the similarity measure of image features (for example, similarity by color, texture, orientation, size, object shape etc) the image records are clustered together by the clustering algorithm explained in the following chapter. After this off-line pre-processing of image database completes, the indexes on image feature search tags are re-constructed to include the newly added set of image records.

Object Dictionary

The object shape dictionary consists of a sample set of image files, image titles that describe dominant shapes in the images, and a set of object shape descriptors. The thesaurus is compiled as and when new shape descriptors are added to the database. It serves as our reference index for identifying the object's representation in the database once we identify its MPEG-7 contour shape descriptors.

Feature Dictionary

The feature dictionary serves as our color and texture reference index which consists of an exhaustive set of image colors and textures to be referenced. It also contains image titles which uniquely describe the dominant color in each image file. The dictionary also stores the MPEG-7 color SCD and texture EHD for the sample set.

3.1.3 Image Mining

Image mining is facilitated by two new efficiency measures as described below.

Search Friendly Image Format (SFIF)

The system uses ORDImage object, a Java class defined in Oracle Multimedia API, to embed SFIF tags in XMP (eXtensible Markup Platform) format to each image object. SFIF includes meta information in images with feature descriptors so as to define a scalable and inter-operable image format that represents a unique standard to define image content. This will ensure faster image mining for systems who wish to use the meta information in their custom database. A targeted set of probable applications, explained in Chapter 7, will elaborate on how other systems can use this meta information tagged with images in our SFIF.

Image Ranking by relevance

This is facilitated by storing the similarity distances of image features from the reference set of sample values for each feature. Once the retrieval engine determines the set of images similar to the user query image, the ranking engine ranks them using these distance values and displays the images by relevance to the user.

3.2 User Interface Design Strategy

The user will rank a image search system high in usability if it provides the user with three basic operations of flexible querying mechanisms, efficient browsing techniques and refine-search options of different granularity. The GUI is designed with these three design components in mind.

3.2.1 Querying Mechanisms

The querying options are simple and intuitive. Since this system is targeted towards average users who have a basic set of query options unlike technical professionals who need more research oriented options, the system provides a comprehensive GUI. With this in focus, the system provides the user with the following querying mechanisms.

Query based on input text

This relies on keyword text annotations. The system will search images with text tags that match with the text query input by the user.

Query based on input image

The user can upload an image and specify the search criteria (object based search, color based search, texture based search or a combination of any/all of these). This classification will enable the system to retrieve images with a focused similarity search using object dictionary or a specific feature dictionary thus avoiding scanning through a large set of possible results.

3.2.2 Browsing Technique

The interactive browsing is facilitated by the use of a ranking engine to display query results based on relevance. The relevance is determined based on the type of input query, that is, whether the user insists on searching images that contain similar shapes or searching images that contain similar color and texture features.

3.2.3 Refine-Search Options

The system provides the user with the following options to refine the search.

1. Image Size
2. Image Orientation
3. Dominant Colors in Images out of a sample set
4. Dominant Textures in Images out of a sample set

The clustering techniques used to create the database come as a useful resource to display the results of refined search criteria.

CHAPTER IV

SYSTEM ARCHITECTURE

This research presents a system that integrates efficient image feature extraction mechanisms and data mining principles. Image feature data contains valuable information like the dominant colors, textures, shapes, etc. Each feature type gives us a possible set of values that can be used to group images in the database. We can define clusters, which uniquely represent the content of images in the database, containing a combination of these reference sets. Each image in the database may fall in one of these clusters and all images in the same cluster will have similar characteristics. For example, if we have a 10000 image database and 12 color clusters, each image in the database will belong to one of these 12 groups. Also, all images in the group 'red' will have red as their dominant color. This gives us a high dimensionality clustering framework to create our database. Our system is designed on this concept and defines a content based retrieval mechanism for a large image database. The system extracts image features and creates object shape and feature clusters which serve as our key lookup mechanism for similar image search.

We create Oracle indexes on image features which links image records together semantically. For example, the index on color descriptors enables retrieving images based on specific color values thus eliminating searching all other records that contain other descriptive information on rest of the image features. The system also utilizes multi-dimensional indexing for a combinatorial search of similar images.

We have defined a clustering algorithm that clusters image records based on the distance of the feature descriptor vector for each image from the feature descriptor vectors of a sample set of images. The set of feature descriptors corresponding to a sample set of images is stored in the Object Dictionary and Feature Dictionary along with their respective pre-computed descriptors. This approach enables a segmented search of images thus eliminating the need to scan through a large set of non-relevant clusters within the database. The results in Chapter 8 demonstrate an order of magnitude performance enhancement for large number of images in the database.

The system architecture is presented in detail in the following four sections. The Image Content Extraction engines consists of the image feature extraction engine and the image object shape extraction engine. This section elaborates on the how the extraction engines store the MPEG-7 descriptors with each image record. Section 4.2 explains the creation of Object and Feature dictionary and also the indexing and the clustering mechanism used. Image Mining is explained by using the MPEG-7 based Search Friendly Image Retrieval and the image ranking mechanism. Section 4.4 explains the procedure to tag images with text that enables image retrieval based on content rather than user supplied text; a mechanism which makes this research distinct compared to most of the commercial systems of today.

4.1 Image Feature Extraction

We pre-process a set of images off-line to create our image database that contains MPEG-7 feature descriptors for color, texture and shape. During this off-line phase, the image feature extraction engine extracts feature descriptors for these images and stores them in the database. The descriptors are later used to construct clusters and search tags for images. Image feature extraction engine also stores feature descriptors off-line for the sample set of images used to build object and feature dictionaries.

The engine is also used during the run-time for querying images similar to an user-input image. Depending on the type of search specified (shape based/color based/texture based), the MPEG-7 descriptor for the query image is extracted, its distance is calculated by comparing it with each entry in the corresponding dictionary (object or feature) and a search tag is established. The database is queried for the cluster of images which match with this tag to fetch similar images. For example, if the user inputs a query image with blue as the dominant color and queries for similar images based on matching colors then the system computes the distances of the Scalable Color Descriptors (SCD) for this image from the color SCDs of all images in the color feature dictionary. The description of color with the smallest distance is used as a search tag for the query image and all images in database with the same search tag are fetched and displayed by relevance.

4.1.1 Why different engines for Object extraction and Other Feature Extraction?

The system presented here is meant for large image databases. When the size of a database grows, it becomes important to classify the information based on the application of the database. For example, if this database is used to organize an agricultural pattern GIS image data sets which has more than a million images taken by satellite then the user of such a system might retrieve similar images based on color and texture rather than shape boundaries. Intuitively, color and texture are the likely search criteria which need to be coupled together in image search. Also, as the size and diversity of images in the database grows, it is less likely to find many images with a matching shape boundary or even a definite shape boundary. By segregating the two engines, we are allowing this system to be portable with applications that want to make use of its color and texture management capability and also with systems that have image sets with some definite object boundary extraction requirements so as to make use of the overall system capabilities.

4.1.2 Feature extraction Engine

The feature extraction engine extracts the image color and texture descriptors and stores them in the feature dictionary. A few other image features like its orientation, format, size, height, width, etc. are also extracted off-line using the ORDImage functions, a Java based Oracle Multimedia API, and are stored along with the image record.

Color

Salembier et.al. [35] have defined the implementation of the MPEG-7 Scalable Color Descriptor (SCD) for image retrieval based on color. The scalable color descriptor is a color histogram extracted in HSV color space which is encoded by a Haar transform for storage efficiency. The first step in the SCD extraction is computing the color histogram with 256 bins in the HSV color space with hue component quantized to 16 bins, and saturation and value quantized to 4 bins each. We represent a bin as a single dimensional vector and is stored as a character string in the database. A series of 1-D Haar transforms are applied across the histogram bins. The result is a set of 16 low-pass coefficients and 240 high-pass coefficients. SCD exploits the impurity of colors images which is the slight variation of colors caused by varying amounts of illumination and shadows in images. Due to impurity of colors, the histograms of natural images tend to have high redundancy between adjacent histogram bins.The high-pass (difference) coefficients of the Haar transform expresses the information contained in finer-resolution levels (with higher number of bins). Due to the redundancy of the original histogram, the high-pass coefficients tend to have low (positive and negative) values. The binary representation of SCD is scalable in terms of bin numbers and bit representation accuracy over a broad range of data rates. SCD allows retrieving similar images based on color. The accuracy of image retrieval increases as the number of bits used in representation is increased.

Texture

Salembier et.al. [35] have defined the implementation of the MPEG-7 edge histogram descriptor (EHD) for retrieving similar images based on the image texture.The EHD exploits the fact that the spatial distribution of edges in an image is an useful description of the image texture. The EHD represents local-edge distribution in the image. The first step in EHD extraction is to divide the image space into 4 x 4 sub-images. The local-edge distribution for each sub-image is represented by a histogram. To generate the histogram, edges in the sub-images are categorized into five types; vertical, horizontal, 45 diagonal, 135 diagonal and non-directional edges. Since there are 16 sub-images, a total of 5 x 16 = 80 histogram bins are required.

4.1.3 Object extraction engine

We have an object extraction engine that identifies individual shapes within each image. It identifies the shape boundaries using the contour shape and edge attributes of the objects within an image. The advantage of using the object dictionary is that the descriptor values are classified and stored in the database for each image and thus saves the computational run time making the processing faster. Once object shape extraction is done, the system creates an internal object and word cloud index to enable content based search even for text search options. The following text describes how we extract the MPEG-7 shape descriptor for images in our engine.

Shape

For retrieving similar images based on shape, the system uses the MPEG-7 Contour Shape descriptor (CSD) defined by Salembier et.al.'s [35] work on MPEG-7 .This CSD is based on the Curvature Scale-Space representation and it captures characteristic shape features of an object in images based on the object contours. This descriptor exploits the fact that the humans tend to decompose shape contours into concave and convex sections while comparing different shapes.Thus similarity based on shape is assessed in terms of the concave and convex sections, their length relative to the contour length and their position and order on the contour. CSD emulates well the shape similarity perception of the human visual system and is robust to shape deformities. The contour of the object is captured in terms of the eccentricity and circularity values of the original and filtered contours.

4.1.4 Object and Feature Similarity Computation

We have used open source Java implementation [29] for defining our feature extraction engine. The input to the engine is the number of colors or textures to be used for quantization. We have modified the implementation and it gives us a 256-bin scalable color descriptor which contains low pass and high pass co-efficient values. We get a 80 bin texture descriptor for the edge histogram value of texture. We store these descriptors as string values in the database and associate them with the image records in the database. We use the C++ open source implementation [30] for shape extraction. We have modified it to return a shape descriptor; uniquely defined using the number of peaks in the shape, its global curvature, its prototype curvature, and x-y co-ordinates for all peaks. We have implemented our own similarity matching mechanism for both feature matching and shape matching. We use L1-norm [31] calculation to compute color and texture distance between two images. We have implemented our own shape matching method which calculates the graph distance between the highest peaks in the shape boundaries between two shape images as cumulative distance.

4.2 Image Database Creation

Once the feature descriptors are extracted for each image in the database , the descriptors are stored in the database. Next comes the process of object dictionary and feature dictionary creation. We have taken a small sample set for each feature type as our reference lookup categories. The characteristic of this dictionary creation process is that it can be extended with a large sample set to suit your application requirements. For example, say if Macy's decides to use our system, it can add a large sample set for dress contours, all possible color sets and dress textures. The results will be precise for a larger set of the object and feature dictionaries. We have used a small set of images so as to demonstrate the usability of the system.

Figure 2 below explains the creation process of object and feature dictionaries. We have used 12 color images which are plain color images representing the 12 primary colors which are likely to occur dominantly in images either individually or in combination. The texture sample set chosen is representative of the 5 texture categories which include 3 regular patterns, 6 near-regular patterns, 6 irregular patterns, 6 non-stochastic and 3 stochastic patterns. For object shape representation, we have chosen 9 generic shapes which contain varying number of peaks. Applications that will use our system in future can include a large set of shape samples to define the object dictionary of richer representations.

4.2.1 Object Dictionary

The object dictionary is a list of different object shapes with definite boundaries. The system administrator specifies a directory where the sample set of object images

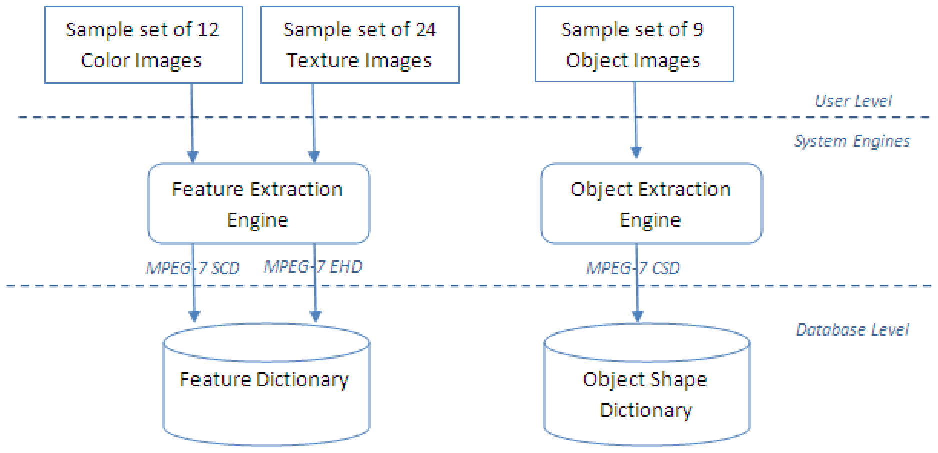


Figure 2: Creation of Object Dictionary and Feature Dictionary

reside. The system creates unique records for these object shapes along with their assigned text descriptions which effectively describe the shape. MPEG-7 contour shape descriptors are calculated for this sample set and stored along with the image information. The dictionary is again used during the run-time process and also during the off-line process.

Each object in the object dictionary is associated with a shape descriptor representation. It enables a run-time comparison with the uploaded query image object. We compare the shape descriptor values of query image with the shape descriptor values of all images in the object dictionary. We determine the image in the object dictionary which gives the minimum distance from the shape descriptor of input query image. The search tag is defined to be the description of this image record. This way we determine object similarity. This search tag is later used to fetch similar images.

During the off-line process, this dictionary serves as the lookup for the system to determine the cluster to which the newly inserted database images are allocated.

4.2.2 Feature Dictionary

The feature dictionary is created on similar lines to the object dictionary with a difference that we store the descriptor records with unique feature ids assigned to them for color and texture sample sets. It is also used during both the runtime and off-line processes as explained above.

For our experimental purpose, we have used 12 sample color sets. The images were chosen to include a diverset set of RGB values.

We have used 24 texture sets, 3 of regular pattern type, 6 of near-regular pattern types, 6 completely irregular patterns, 6 non-stochastic and 3 stochastic. A large amount of work is done in the texture synthesis domain. The structural methods [34] define variations of regular patterns as regular, near-regular and irregular. Examples of such patterns could be bricks, berries, criss-cross knit fabric, etc. depending on the nature of images. The statistical methods [34] work on stochastic patterns that lack regularity. Sand, crop, water, flame, etc. are some of the examples that we have chosen for stochastic and non-stochastic the texture pattern representation. The categories we have chosen are inferred from the probabilistic and hierarchical texture synthesis models defined in Popat's [34] work on texture synthesis, classification and compression.

Figure 3 below explains the process of creating our Search Friendly Image Database (SFID). This figure depicts how we insert image content information which later will be used for creating the search friendly image meta data representation.

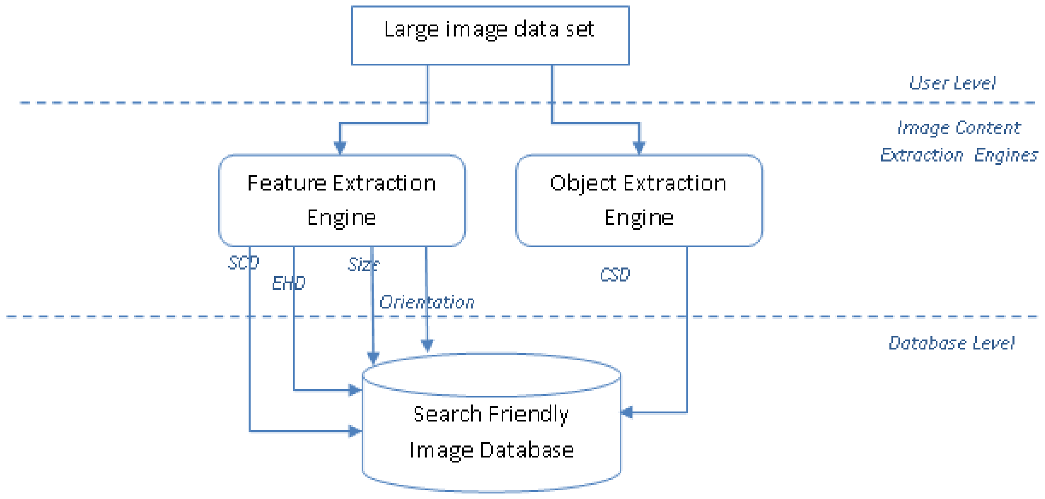


Figure 3: Creation of Search Friendly Image Database

4.2.3 Indexes and Clustering

We have defined a clustering algorithm which clusters images based on the similarity measure of image features (for example, similarity by color, texture, orientation, size, object shape etc). The algorithm identifies groups of images based on all possible ways the user is able to query the database. Our algorithm appends the images with tags based on color, size, orientation, texture, object alignment, etc. This also serves as text tags for text based queries. This clustering algorithm facilitates pattern finding mechanism by use of clusters in conjunction with multi-dimensional indexes.

Clustering Algorithm for all new images inserted in database

1. Create color based clusters

a. calculate the distance of the new image's SCD (scalable color descriptor) with the SCDs of 12 sample sets

b. rank their distances (called as norms) in ascending order; this is later used by the ranking engine

c. assign a dominant color tag to the image as the closest identified color with minimum distance

d. append this search tag to image as XMP (eXtensible Markup Platform) meta data description tag

2. Create texture based clusters

a. calculate the distance of the new image's EHD (edge histogram descriptor) with the EHDs of 24 sample sets

b. rank their distances (called as norms) in ascending order; this is later used by the ranking engine

c. assign a dominant texture tag to the image as the closest identified texture with minimum distance

d. append this search tag to image as XMP meta data description tag

3. Create shape based clusters

a. calculate the distance of the new image's CSD (contour shape descriptor) with the CSDs of 9 sample sets

b. rank their distances (called as norms) in ascending order; this is later used by the ranking engine

c. assign a dominant shape tag to the image as the closest proximity shape contour identified if this distance is outside an acceptable threshold, there is no identified object shape pattern for the image and classify this image in irregular shape cluster

d. append this search tag to image as XMP meta data description tag

4. Create clusters based on refine-search options

a. orientation (landscape,portrait) use ORDImage API to determine the height and width of image and compute its orientation

b. size (small, medium, large) use ORDImage API to determine the height and width of image and compute the size within standard thresholds to classify them as small, medium and large

c. style (color or black/white)

d. append this search tag to image as XMP meta data description tag

Figure 4 below gives a step-by-step procedure of creating searchable tags in database. The process makes use of the sample set feature information from the object and feature dictionaries. It shows how these tags are used as Search Friendly Image Format meta data representation.

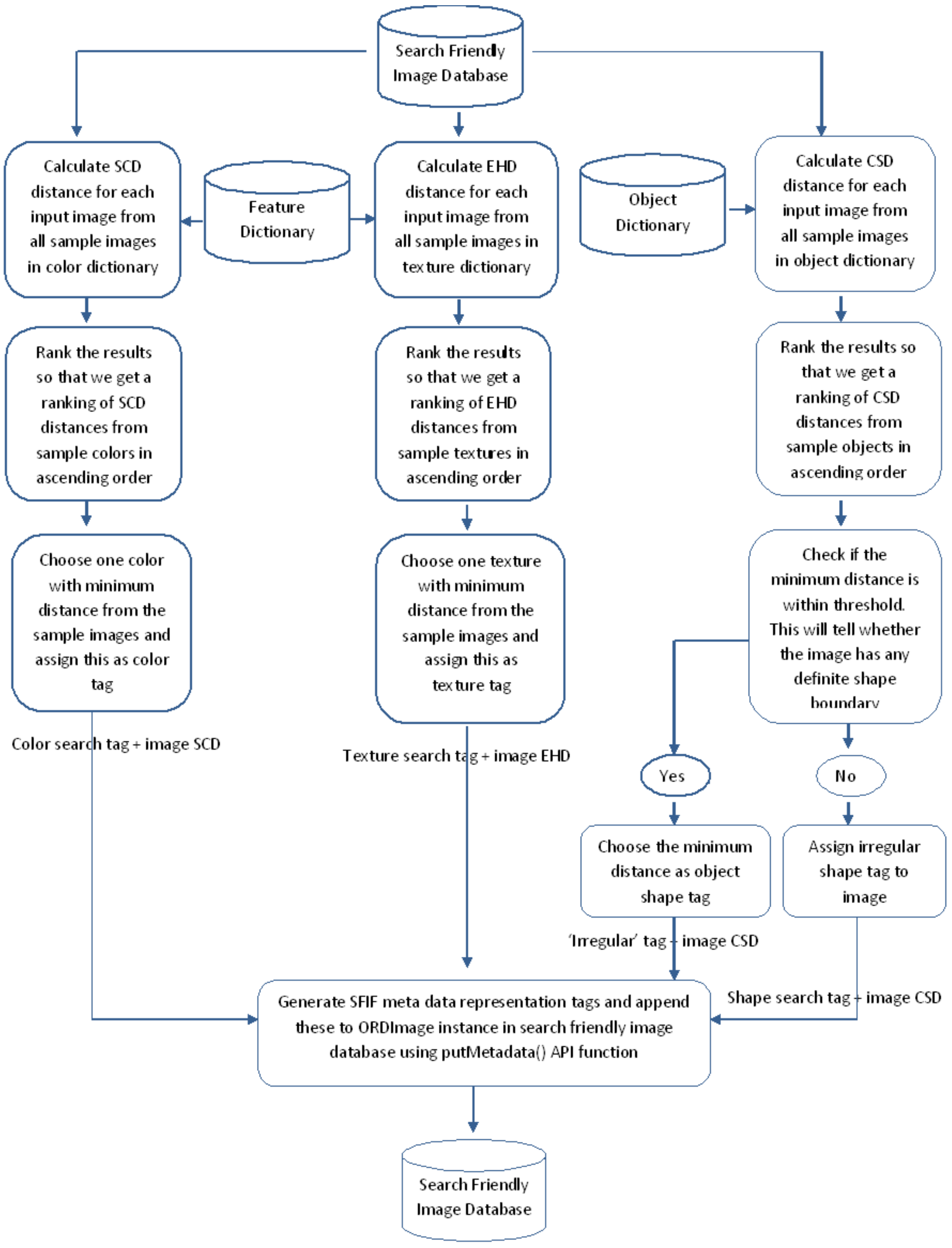

Search Friendly Image Database
Calculate SCD distance for each input image from all sample images in color dictionary
Feature Dictionary
Calculate EHD distance for each input image from all sample images in texture dictionary
Object Dictionary
Calculate CSD distance for each input image from all sample images in object dictionary
Rank the results so that we get a ranking of SCD distances from sample colors in ascending order
Rank the results so that we get a ranking of EHD distances from sample textures in ascending order
Rank the results so that we get a ranking of CSD distances from sample objects in ascending order
Choose one color with minimum distance from the sample images and assign this as color tag
Choose one texture with minimum distance from the sample images and assign this as texture tag
Check if the minimum distance is within threshold. This will tell whether the image has any definite shape boundary
Color search tag + image SCD
Texture search tag + image EHD
Yes
No
Choose the minimum distance as object shape tag
Assign irregular shape tag to image
'Irregular' tag + image CSD
Shape search tag + image CSD
Generate SFIF meta data representation tags and append these to ORDImage instance in search friendly image database using putMetadata() API function
Search Friendly Image Database

Figure 4: Tagging of SFIF meta data to images

Once this process is complete, we need to create/re-build indexes on the image feature columns. The multi-dimensional indexes are re-constructed once the off-line feature extraction process updates/inserts new records in the database. The code is defined in a database SQL procedure which runs a trigger on the tables after an update/insert. We have created several indexes on individual query-able fields and also on possible combinations like color tags and texture tags.

This application falls in the category of Online Analytical Processing (OLAP) of a database. We have proposed an architecture which facilitates fast retrieval of information by finding pre-computed patterns from the database. The database is updated only during the pre-processing off-line phase. During similar image retrieval process, information is only read from the database. Since this is an OLAP application, the definition of indexes plays an important role in querying the database. The database indexes defined on the search tag columns keeps an indexed list of all image records that match a particular search tag. We rebuild all the indexes when new images are added to the database thus keeping this information up to date. Hence, when any user queries for a dominant feature, the indexes help to retrieve all images by relevance efficiently. These indexes help speed up the retrieval process as all images that match the search combinations are already indexed in the database. This saves scanning through all image records and identifying the matching records one by one.

4.3 Image Mining

Image Mining is the process of finding patterns in the image database which do not have this information stored explicitly. The definition could be explicitly stated as automated discovery of previously unknown, nontrivial, and potentially useful information from databases. As seen in the earlier sections, the image content extraction engines give us the respective descriptors. The key to our approach to image mining is use of Search Friendly Image Format (SFIF) search tags, stored with images, for image retrieval. For each image and for each value of the feature, we assign a normalized score to the image with respect to a pre-defined set of images. This score is used to rank images by relevance. Unless the image relevance ranking in the database changes, this information from the extensible tags is used to search similar images at a much faster rate. The ranking will change only when new images are added off-line to the database during which the SFIF tag information will be updated as well keeping it in synch with relevance ranking engine operation.

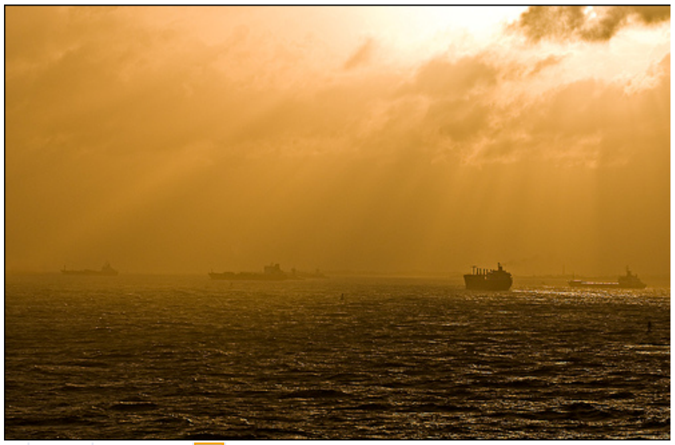
4.3.1 Development of a Search Friendly Image Format

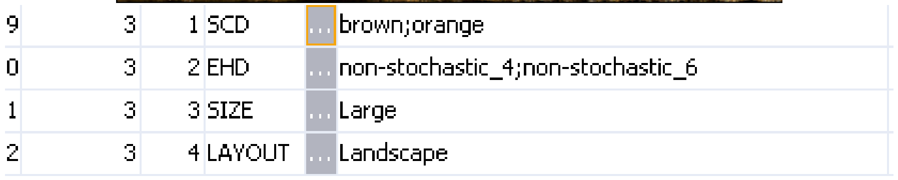
We have developed a new image meta data representation format called SFIF: Search Friendly Image Format. The SFIF is based on the MPEG-7 multimedia content description format [27]. MPEG-7 has provided standardized descriptors for description of multimedia content, effective searching, identifying, filtering, and browsing on multimedia contents such as images, videos, audios, and other digital or even analogue materials. MPEG-7 has defined a set of standard descriptors for low level features such as color, shape and texture [26]. SFIF has a subset of these features that are extracted and embedded in the image header along with searchable tags.

The advantage of including the low level features in the image itself is that it saves the computationally intensive process of feature extraction, every time a new search engine tries to index the image files. For example, Exchangeable image file format (EXIF) is a good example for jpeg image tagging. Exif data is embedded within the image file itself and is stored in one of JPEG's defined utility Application Segments as meta data. SFIF further extends this approach of including meta information in images to low level feature descriptors. The output of object and feature extraction engine is stored as tags in the SFIF image as image meta data along with the feature searchable text tags. The SFIF includes meta information in images along with feature descriptors so as to define a scalable and inter-operable image format that represents a unique standard to define image content. This has ensured faster image mining.

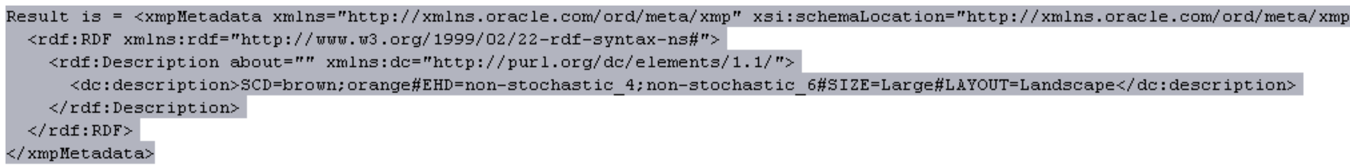
The clustering algorithm explained above shows the step where we append the search tag information to image meta data. We are using XMP (Extensible Markup Platform) which is a labeling technology that allows us to embed data about a file, known as meta data, into the file itself. Once we extract the color, shape and texture search tags for the image along with other textual image information, we form a hash separated description string and append it to the image XMP header. This is very similar to appending EXIF information to the images. Image EXIF meta data usually consists of camera specifications and vendor and/or manufacturer custom information. Hence, we have chosen to embed the image feature meta data information in the XMP header using the putMetadata() method of the ORDImage object, a java API class defined by Oracle Multimedia. Table 1 elaborates on its usage in the implementation.

Figure 5 below shows a sample image on which we extracted the meta data information and created a search tag to be embedded in XMP header with the image itself. The next picture is a sample snapshot of the database which shows the SCD (color tags), EHD (texture tags), size and layout information for this image. The third graphic is the SFIF meta data representation as it is extracted from the image using an XMP meta data viewer.





9	3	1	SCD	...	brown;orange
0	3	2	EHD	...	non-stochastic_4;non-stochastic_6
1	3	3	SIZE	...	Large
2	3	4	LAYOUT	...	Landscape



```
Result is = <xmpMetadata xmlns="http://xmlns.oracle.com/ord/meta/xmp" xsi:schemaLocation="http://xmlns.oracle.com/ord/meta/xmp
  <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <rdf:Description about="" xmlns:dc="http://purl.org/dc/elements/1.1/">
      <dc:description>SCD=brown;orange#EHD=non-stochastic_4;non-stochastic_6#SIZE=Large#LAYOUT=Landscape</dc:description>
    </rdf:Description>
  </rdf:RDF>
</xmpMetadata>
```

Figure 5: SFIF Image tagging example

4.3.2 Portability of system using SFIF

We use ORDImage object from Oracle Multimedia to embed SFIF tags in XMP format to each image object. The functions used are putMetaData() which accepts a schema-valid XML document and embeds it in the target image file format. We use the getMetaData() to retrieve the schema-valid embedded XML documents for EXIF, XMP, SFIF etc meta data information [33]. The ORDImage API ensures that this meta information is tagged to image headers. Other systems (eg. Google and Bing) can exploit only the image header information and use the SFIF tags in the images created in our system. Alternatively, an enterprise based system can use ORDImage object to extract the dominant colors, textures, shape, orientation, size etc. from our SFIF meta data representation and build its own custom application logic for image retrieval. An important point to consider here is that this will be a CBIR system as against a CBIM as the search will just boil down to fetching information from image headers as against finding patterns in data and displaying by relevance (as is done with our system).

4.3.3 Image Relevance Ranking

Relevance ranking is a dynamic process governed by the input query. We store a normalized score for images for each feature type. This score is used to rank images by relevance for every feature of the image. For example, we have a normalized score for color SCD, texture EHD and shape CSD. When the user inputs an image to retrieve similar images, the ranking of the result is based on the type of search query. It varies the relevance of images for the type of query user has input. For example, an image with high ranking for its color similarity may have low raking to the same input query image based on its texture similarity. This is taken care of by the relative distance values (the normalized score) for each type of descriptor stored as part of the clustering algorithm.

4.4 Query Expansion and Text Search Engine

We have developed a search engine that allows searching images using an input text query as well. The text annotations are added to images during the object dictionary and feature dictionary creation. Once we get the MPEG-7 descriptors for the images or image objects from the respective extraction engines, we find their distance measure from all the of the sample set of dictionary records. For example, to find the text annotations for dominant color descriptors, once we get the SCD, we compare it with the SCD values of all records in color feature dictionary and annotate the image records with the color name which has minimum distance from this SCD. Similarly, we calculate text annotations for texture of images and for image objects as well. By taking this approach we will be searching for images on the basis of their actual content, rather than relying on the label/text attached with images.

Once the search tag is computed, we build a word cloud for it by querying for keywords using the Google Search engine. The keyword cloud is used to fetch search results when user inputs text query. The user can chose to fetch image results that match 'any' of the input keywords or for results that match 'all' images that meet the criteria.

CHAPTER V

WEB BASED USER INTERFACE

We have so far seen the system architecture from the database designer's perspective. In this section we will see an elaborate user interaction work-flow with the system. The user interface should be intuitive. The user should be able to navigate through the system with minimal intervention and at desirable speed.

Figure 6 shows the various ways in which the user can interact with the system. The input query can be either an image or text. To search similar images by giving an input image as a query, the user has to upload an image to the system and select the search criteria. Depending on the search criteria selection, the corresponding extraction engine extracts all the dominant features in the image and retrieves similar images from the database. The search results can be refined based on other image features.

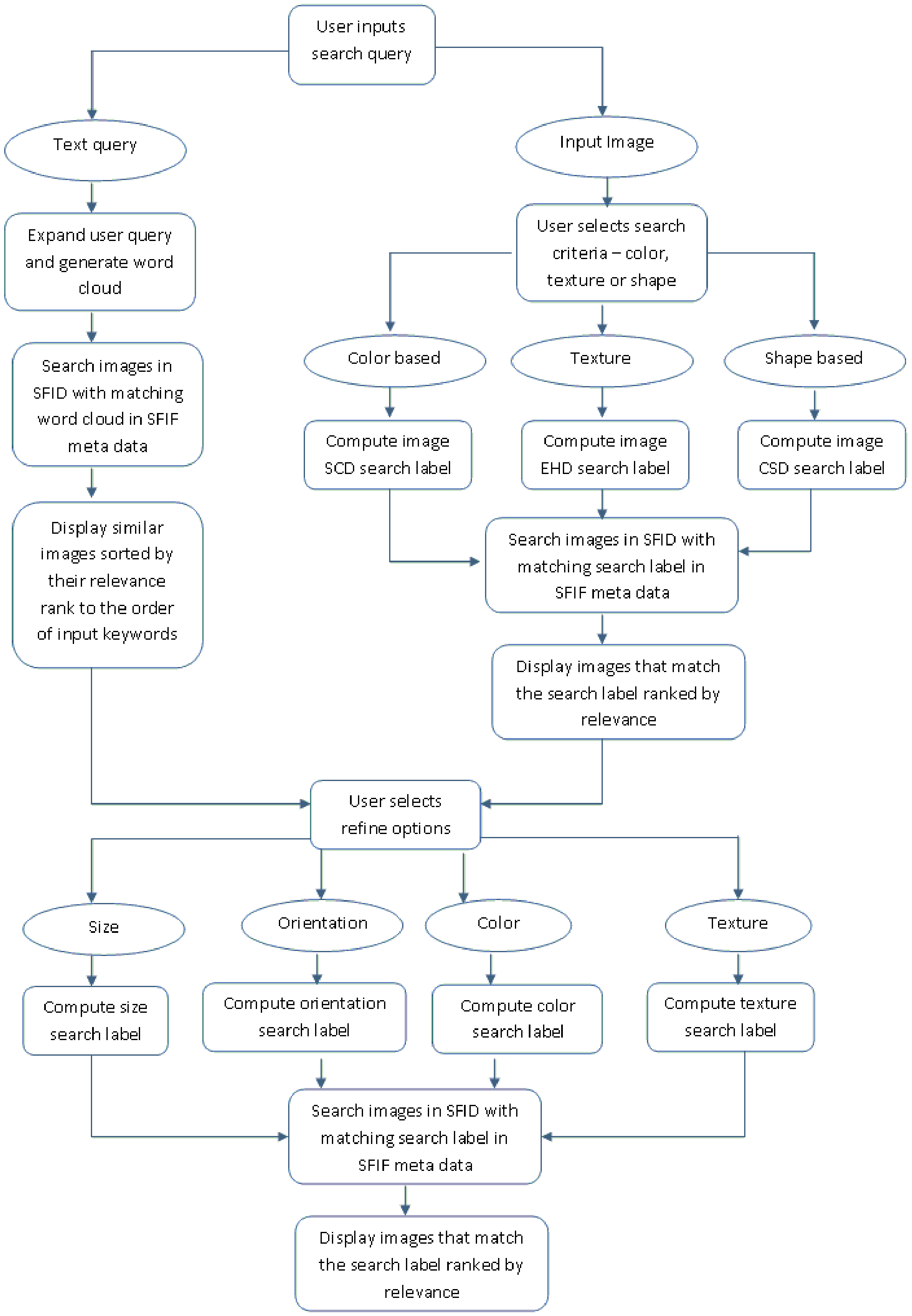

User inputs search query
Text query
Input Image
Expand user query and generate word cloud
User selects search criteria – color, texture or shape
Search images in SFID with matching word cloud in SFIF meta data
Color based
Texture
Shape based
Compute image SCD search label
Compute image EHD search label
Compute image CSD search label
Display similar images sorted by their relevance rank to the order of input keywords
Search images in SFID with matching search label in SFIF meta data
Display images that match the search label ranked by relevance
User selects refine options
Size
Orientation
Color
Texture
Compute size search label
Compute orientation search label
Compute color search label
Compute texture search label
Search images in SFID with matching search label in SFIF meta data
Display images that match the search label ranked by relevance

Figure 6: Web Based User Interface

The following sections describe these two flows separately.

5.1 *Text Keywords as query*

The user can input text to search for images with their content matching to the semantics of the input text. The ideal goal of the system is to fully understand the semantics of the query based on input text. We build a word cloud (or weighted list in visual design) for the entered keywords. We give the input query a rich semantic context to extract similar images from the database by expanding the query based on input text.

The next important thing to note is the interpretation of articles 'and' and 'or' in the input query. There might be cases where a user wishes to fetch results based on a combination of colors and texture patterns. For example, if the user wants to retrieve images with a criss-cross textured red color rectangular purse from the database. If the user does not have a sample image to upload, the user might as well enter 'red and rectangular and near-regular' as search option. The use of 'or' fetches all results that match any of the entered keyword. The default behavior supported is 'or'.

5.2 *Input Image as query*

The user can upload an input image to query for similar images. The user needs to select a criteria for search from amongst color, texture, shape. The reason the search results are not displayed with a combined search on these features is to enable the user have a richer and flexible interface experience. Similar image search based on a combination of options restricts the number of results. Hence, for an effective browsing technique, the user is provided with search criteria - color based, texture based or shape based. The user can upload an image chosing one of the search criteria. Once the results are displayed, the user can refine the results using the options displayed on the screen for a precise search result.

For similar image search using shape as the criterion, if no definite shape boundary is identified in the input image, all images in the database tagged as irregular shape images are displayed to the user. If a definite shape boundary is identified, only those images which match the shape search label are fetched from the database.

For similar image search using color or texture as the criteria, all images which match the respective feature are retrieved. All images are internally ranked by the ranking engine for these search labels. This saves time to process the relevance ranking and hence gives a better user interaction experience.

5.3 Search Results Refine Options

Once the results are displayed for text based or input image based search, the user can refine the results based on the image size (small, medium or large), image orientation (portrait or landscape), image colors (12 sample colors displayed in search options) and image texture (24 sample textures displayed in search options). The system currently supports only a single level of refining; that is, each time any refine option is selected, the original image results are filtered rather than using a hierarchical filtering. This suffices for the demonstration purpose but could be modified later for applications that need a refining for more than one level.

CHAPTER VI

USE OF ORACLE MULTIMEDIA FOR IMAGE RETRIEVAL

We use Oracle Multimedia on Oracle 11g Release 2 platform [33] as our backend support. Oracle Multimedia provides capabilities for storage, retrieval and management of digitized images in a database. Oracle Multimedia exposes an API for handling image objects (ORDImage). It greatly improves the speed of retrieval of images from the database. Our focus is on evaluating performance for realistic repositories involving hundreds of thousands of images. Searching from a distributed collection of databases and use of parallel processing into image search are some additional topics of future research.

We use ORDImage object from Oracle Multimedia to embed SFIF tags in XML format to each image object. The functions used are putMetaData() which accepts a schema-valid XML document and embeds it in the target image file format. We use the getMetaData() to retrieve the schema-valid embedded XML documents for EXIF, XMP, SFIF etc meta data information [33].

We have benefited from the use of Oracle Multimedia in our work in many ways. It has helped us to improve the performance of the system, define inter-operable SFIF tagging mechanism, extraction of a few image features, etc as listed below.

1. Retrieving and managing image files from various data sources distributed databases as well as web urls wherever applicable
2. Extracting and embedding meta data information in SFIF XML based format for

faster retrieval of information

3. Manipulation of image objects for generating thumbnails to be displayed on screen

4. Reducing database storage by using ORDImage objects instead of storing images as BLOBs which do not give the flexibility to manipulate image meta data and also increase CPU processing overhead

5. Reducing I/O overhead by querying ORDImage objects rather than storing images as flat files on physical disks

6. Use of various Oracle multimedia functions common to all supported multimedia data types (ORDAudio, ORDVideo, ORDImage, ORDDoc) and also ORDImage specific functions

7. Error Handling and tracking facility provided by ORDImageExceptions API

8. Oracle defines a SQL/MM Still Image API which supports functions to extract valuable image feature information as described in Table 1. Some of the Still Image feature extraction functions can be used to complement the MPEG-7 feature extraction methodology. Table 1 lists the Still Image methods that can be used for image feature extraction.

The following table lists the functions that are used as part of the implementation. All functions except those belonging to the Still Image package are presently used in the code. The Still Image functions could be used in future implementations and are mentioned here for records.

Table 1: Use of Oracle Multimedia Functions for CBIM

Project Module	Oracle Multimedia Function	Usage Description
Creating CBIM data repository	getSource( ), getSourceLocation( ), getSourceName( ), getSourceType( )	To determine source of image from where to import image objects in our database
Development of a Search Friendly Image Format	putMetadata(), getMetadata()	To extract and embed image object search friendly feature descriptors
Object extraction engine	process(), processCopy()	To crop image into regions and facilitate image segmentation for dominant object recognition
Object Dictionary	putMetadata()	To copy and store meta data in individual new image objects created as a result of Object Extraction Engine Module
Feature extraction Engine	getCompressionFormat(), getFileFormat(), getHeight(), getWidth()	To retrieve basic image properties which are used to refine similar image search
	SI AverageColor Method, SI ColorHistogram Methods, SI FeatureList Methods, SI PositionalColor Method, SI Texture Method	To retrieve image feature characteristics to facilitate comparing images based on basic feature set. This will complement or can replace the search of similar images based on our proposed MPEG-7 feature descriptor methodology
Query Expansion and Text Search Engine	putMetadata(), getMetadata()	This will help generate text tags and embed them with other image meta data information in SFIF format

CHAPTER VII

APPLICATIONS OF THE SYSTEM

The system with its scalable architecture can find its applications in many domains. The following sections list some probable application areas.

7.1 Commercial Applications

Retail stores with a wide variety of products can use this database to categorize their product catalog in terms of product colors, textures and shapes by exporting their image sets in our system and using the web interface to fetch a product line based on user input. An example is stores that sale fabrics, tiles, home furnishings, shoes, books, purses, handcraft papers, electronics, etc.

7.2 Academic Applications

(i) Research groups that are interested in exploring the scalability of the system by using our implementation of Search Friendly Image Database and SFIF meta information support, and defining their own implementation of image feature extraction instead of MPEG-7 descriptors can take advantage of the modular design and plug their own implementations to our system. These applications can explore the basic image meta information supported or can use our system to extend the meta information support by using the SFIF meta tags from ORDImage Objects and adding their own information to the images in XMP format.

(ii) Researchers can build their own interface to the database by exploiting our modular architecture to support custom application needs. For example, any of the object extraction or feature extraction engines can be eliminated with a custom GUI.

(iii) Researchers can define refining options to more than one level. A hierarchical refining can be used to filter search results with more than one filter criteria.

(iv) More of the image information from ORDImage API or Still Image API can be extracted and supported as part of the user interface (example image saturation and intensity).

7.3 Image Domain Oriented applications

(i) GIS applications - Consider a scenario where the GIS server takes thousands of images in a day and we need to categorize these images for applications that intend to use them; for example, a set of riverbed images for River Water Quality Control System, Images of agricultural farms for Nationwide agricultural land monitoring system etc. Our system can effectively sort the images based on river water content and agricultural land as the texture and color representations for these domains are different. We can input a sample set of images for each category and retrieve similar images based on color/texture based search.

(ii) Traffic Signs monitoring system - The Manual on Uniform Traffic Control Devices, or MUTCD defines the standards used to install and maintain traffic control signs and devices nationwide. In order to have an up-to-date information on the signs, local agencies collect road sign information by sending vehicles across the counties. We need systems to monitor the images collected from all agencies. These systems have millions of images taken by mobile vehicle cameras and they need to detect images with traffic signs in it. Such applications can define the object dictionary to contain only the MUTCD defined sign shapes, and feature dictionary to support only sign colors. The GUI can retrieve images similar to a sign type by uploading a sample input sign or by entering text keyword in terms of colors and geometrical shapes of signs.

(iii) Image Mining using input video - A video can be used to search for images in a database that match the frames in the video. In such a scenario, we need to extract the input video frame and for each frame as input query image. We can use this system to take a representative frames in a video at certain interval as input images. We can fetch images similar to this input query image from the database.

(iv) Medical and Biotechnology applications - These fields have a large number of datasets and need effective image mining mechanisms to sort images in categories; for example, if we have millions of images of human anatomical structure and we need to sort out images containing a tumor-like structure, we can use the object dictionary and feature dictionary to define our reference set of tumor types and find closest matching images to this reference set. Our system can also be used to mine images containing other general forms of biological objects.

7.4 Other Applications

With the scalability of the database design and the modular architecture, the system finds its uses in many application areas. Any application that needs an OLAP (Online Analytical Processing) oriented database for fast retrieval of image content information can use this system effectively. The application only needs to define their object and feature dictionaries for representing its image features like unique color set, texture patterns and exhaustive set of shapes.

CHAPTER VIII

RESULTS AND EVALUATION

The database consists of a set of 10000 images. We have used a sample set which gives us good, average and worst case results to help us identify the robustness of the architecture. There are various new methodologies proposed with the research. We needed a demonstrative and yet exhaustive reference set of images to compile our object and feature dictionaries. For this purpose we have taken 12 sample color images as reference using which we determine the SCD search tag for each of the 10k images in our database. Similarly, we have 24 texture images (for EHD search tag computation) and 9 object shape images (for CSD search tag computation) as a reference in the dictionaries. The following sections show the web based results from the system, the performance metrics and a comparison with other existing systems. All the results displayed are displayed by relevance and ranked horizontally.

8.1 System Results

Figure 7 shows the system home page where the user can upload an image as an input and select a search criterion, or input a text query.

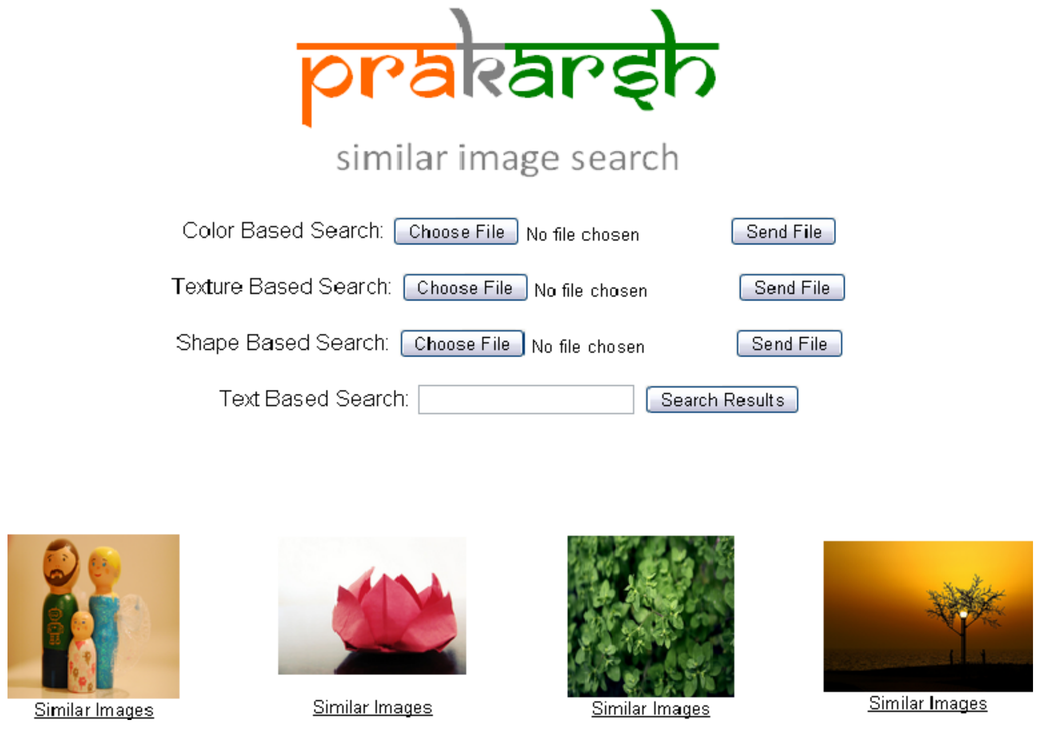

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Similar Images
Similar Images
Similar Images
Similar Images

Figure 7: Home Page

Figure 8 shows the options that can be used to refine the results. The user can refine the search based on the size, layout, colors, shape and textures. For any search, choosing other options from the same group will show no results. For example, for color based search, if we have blue as the dominant color as in the above case, we will not get any results if we try to refine the search based on any other 11 colors in the group.

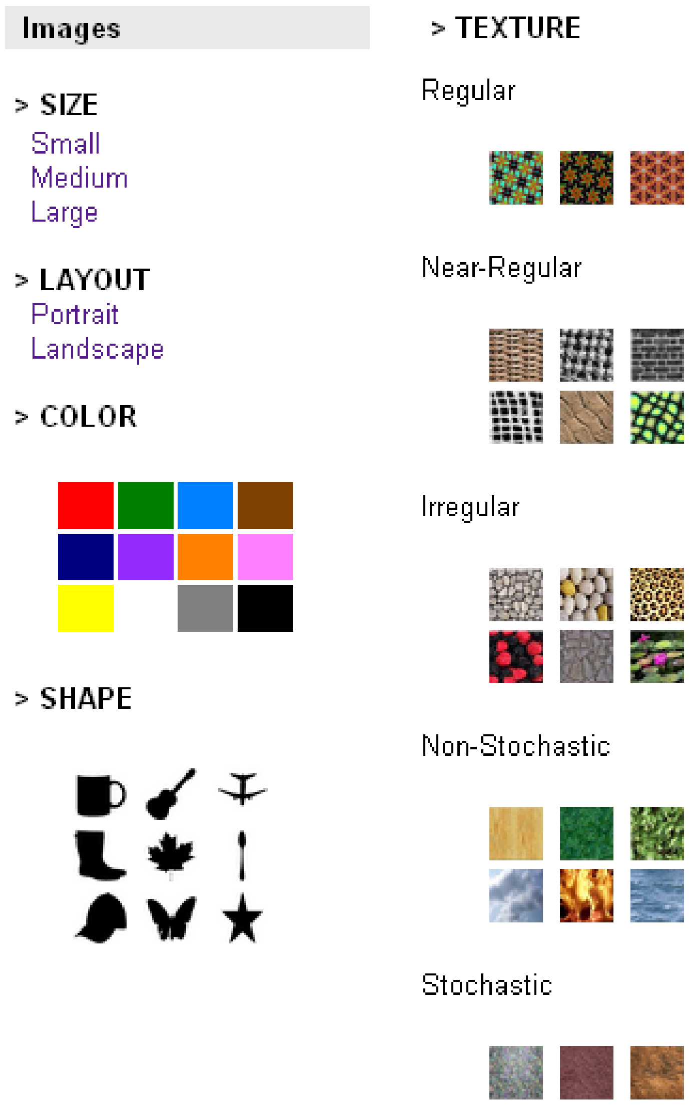


Figure 8: Refine Search Options

Figure 9 shows the results page which displays images by relevance based on the search criteria. For this demo, we have input a query image which appears on top left corner of the page and chose ”Color Based Search” as our selection criterion. The system computes the 2 most dominant colors in the input image and fetches results ranked by relevance to these colors.

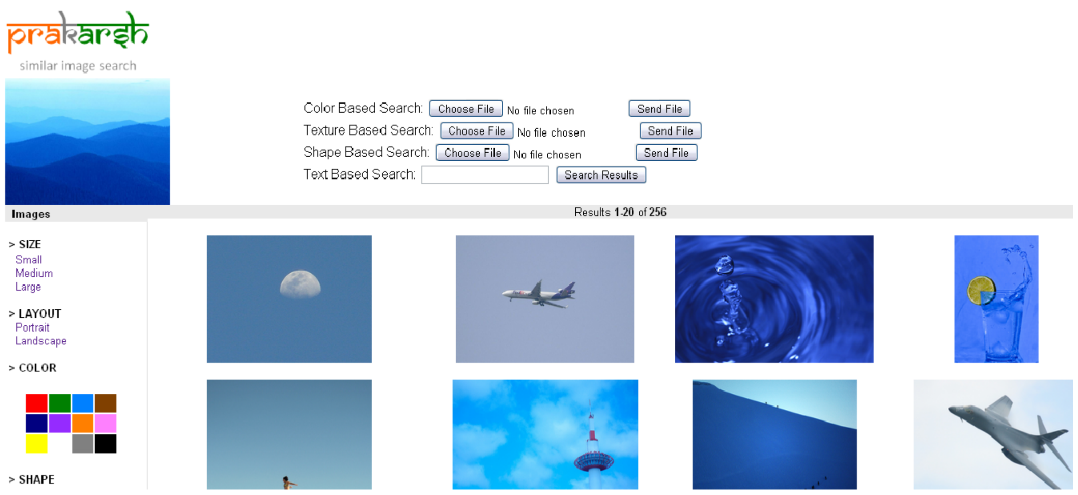

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-20 of 256
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE

Figure 9: Search Results for color based search

Figure 10 displays the search results when we filter the results shown in Figure 9 based on texture. We selected water to refine the search results. Water texture belongs to the near-stochastic group of textures. In [34], it is defined to belong to the statistical methods of texture synthesis which deal with image textures that lack regularity. The results are displayed based on rank of relevance with images, the ones with the closest distance displayed first.

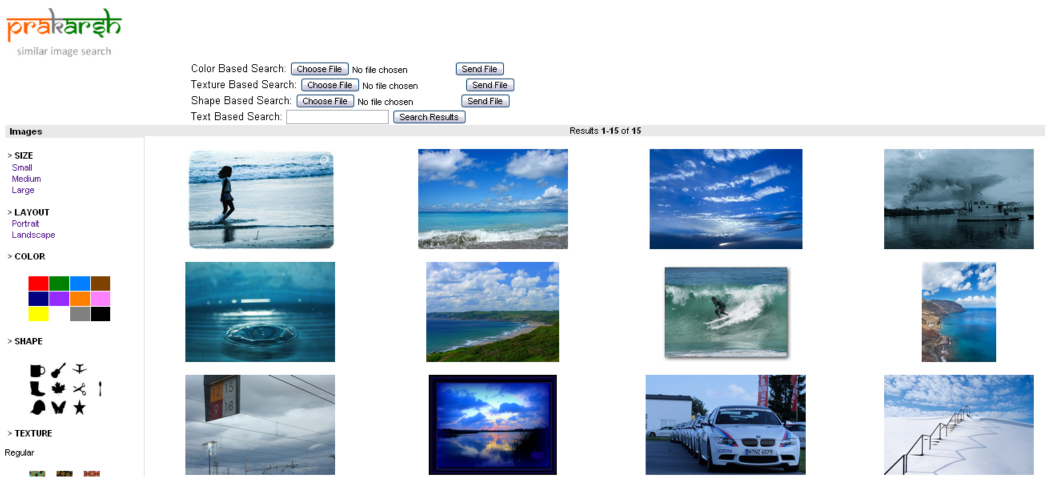
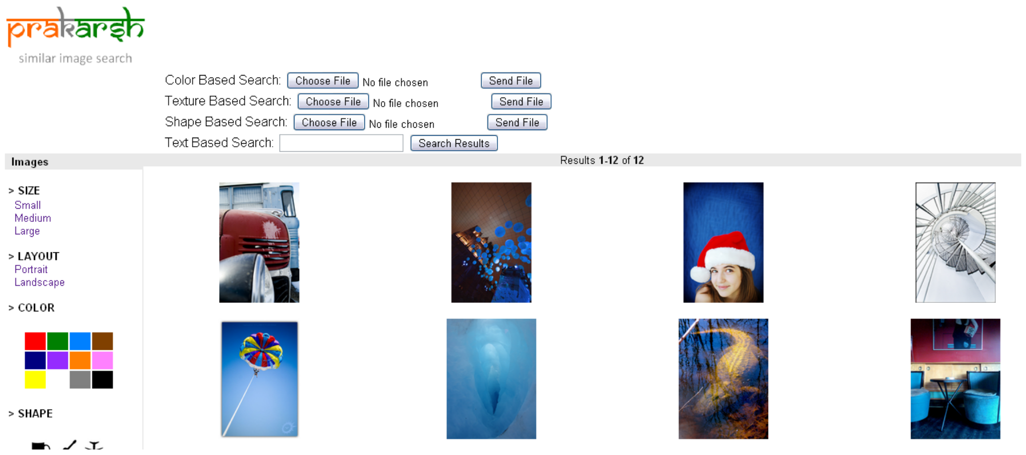

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-15 of 15
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE
> TEXTURE
Regular

Figure 10: Refining of Search Results using Texture

Figure 11 displays the search results when we filter the results shown in Figure 8 based on image layout by selecting "Portrait" images. Here the results are displayed as they are stored in the database, and not by any ranking, as portrait images do not have any ranking relevance. Every time we refine the results, we refine the original result set rather than refining the last displayed results. This is to enable the user to refine original results based on different criteria at the same time without losing the original result set.



prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-12 of 12
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE

Figure 11: Refined Search Results for Portrait Layout

Figure 12 shows the results page which displays images by relevance based on the texture based search criteria when we input a query image. You can further refine the options as depicted in figure 9.

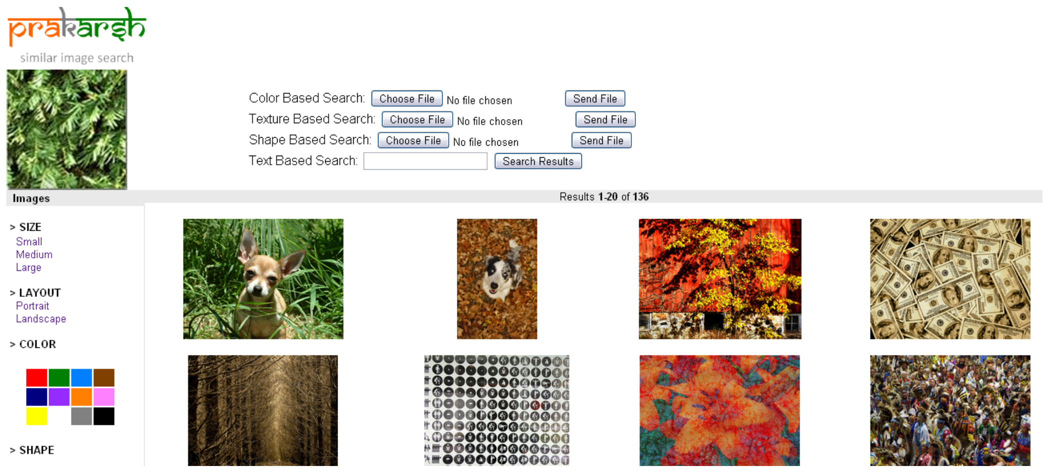

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-20 of 136
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE

Figure 12: Texture Based Search on Input Image

Figure 13 and 14 show the results page which displays images by relevance based on the shape based search criteria when we input a query image. Here for demo we have input an image which contains a cup in Figure 13 and a butterfly in Figure 14 as their dominant shapes.

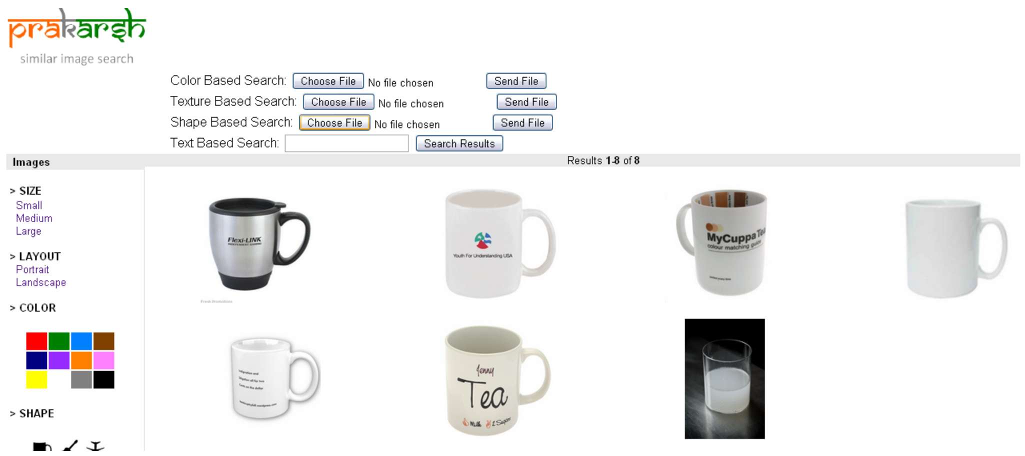

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-8 of 8
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE

Figure 13: Shape Based Search on Input Image

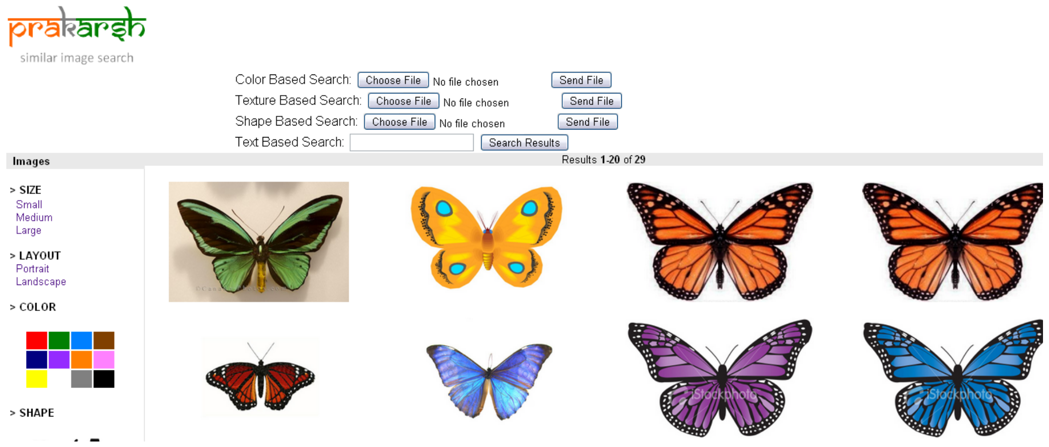

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-20 of 29
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE

Figure 14: Shape Based Search on Input Image

The user needs a flexibility to search for a combination of the images in the database without uploading any query image. Figure 15 shows the results page which displays images by relevance based on the text query entered by the user. We have restricted our search to only those keywords which are listed in our object or feature dictionaries. For this demo, we have entered red near-regular as our text query for listing all images with a red dominant color and near-regular texture. You can further refine the search using one of the refining criteria.

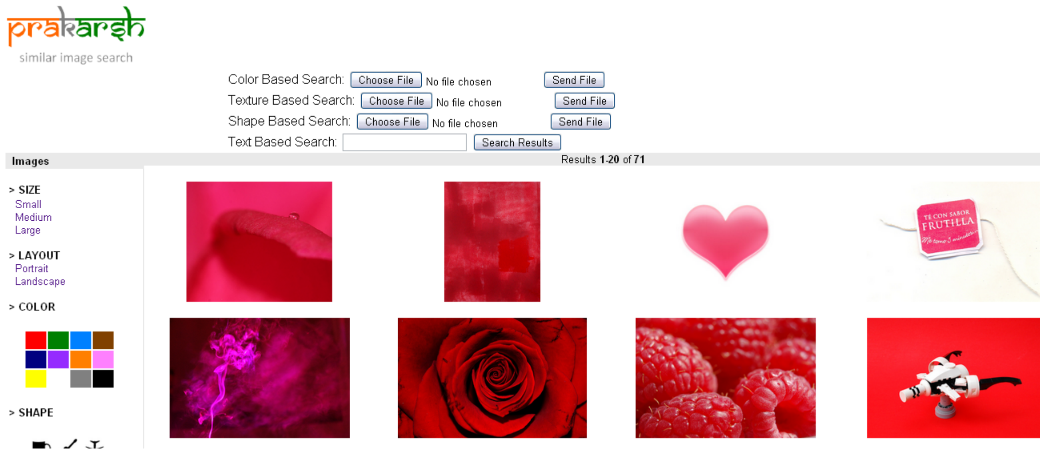

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-20 of 71
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE

Figure 15: Text Based Search Results

The user should be able to browse images in the database which belong to a particular group. For example, all landscape images, all images with green dominant color, all images with regular pattern, or all images with a butterfly in them etc. For this reason, we have provided a flexible GUI where the user can select a search criterion from the left side panel without entering any input image or text query. The limitation with this search is that the user will not be able to refine these search results, which can be removed with future implementations. Figure 16 shows search results for all images with a brick-like near-regular texture pattern in them.

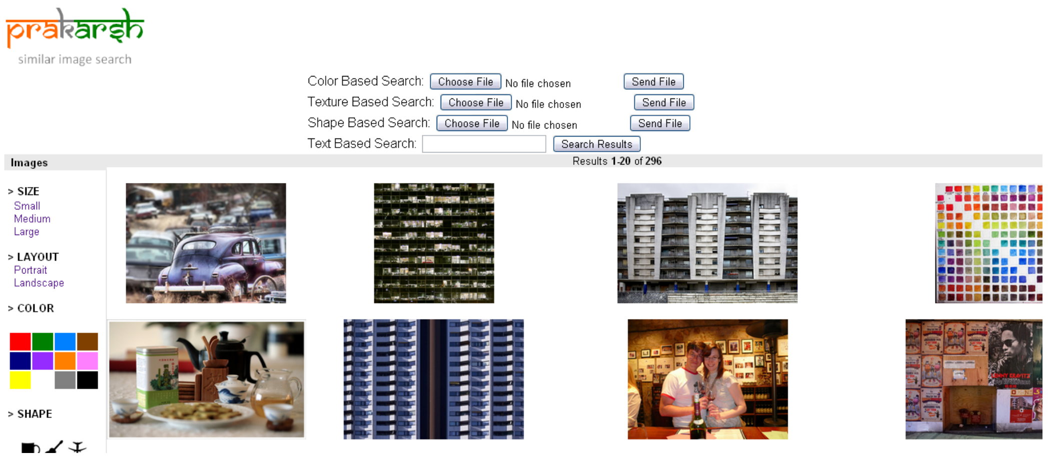

prakarsh
similar image search
Color Based Search: Choose File No file chosen Send File
Texture Based Search: Choose File No file chosen Send File
Shape Based Search: Choose File No file chosen Send File
Text Based Search: Search Results
Images
Results 1-20 of 296
> SIZE
Small
Medium
Large
> LAYOUT
Portrait
Landscape
> COLOR
> SHAPE

Figure 16: Search Results for crop texture group search

8.2 Performance Evaluation

For performance evaluation, we have conducted three sets of experiments.

8.2.1 Robustness

For the first experiment, we analyze the performance times for each type of feature and shape based similar image search query. We pre-process a set of 10000 images and build our database, enhanced with SFIF meta data, as described in chapter 4. We have queried for similar images by uploading a range of images, the results for which are shown in table 2. The table lists the time in milliseconds for each type of query. The results show a significantly fast retrieval even for large number of results.

Table 2: System Performance with and without Clustering

Search Type	**Dominant Feature**	**No of Results**	**Total time (ms) without clustering**	**Total time (ms) with clustering**
Color	Black	3905	3495.74	20.659329
	Grey	1876	1384.62	7.523582
	Orange	326	989.34	1.262171
	Blue	261	674.83	1.052927
	Red	79	24.66	0.29948
Texture	Non-Stochastic	3440	3102.38	17.608942
	Irregular	1106	0982.17	2.862654
	Regular	516	0972.36	2.714032
	Near-Regular	296	792.11	1.724521
	Stochastic	146	34.56	0.30786
Shape	Butterfly	29	2.32	0.038273
	Star	24	2.19	0.0352
	Guitar	15	0.87	0.025423
	Maple Leaves	10	0.65	0.020114
	Cup	9	0.12	0.012013

8.2.2 Correctness

We present the system correctness in terms of precision and recall values for a sample set of 100 images (for color and texture analysis) and another set of 91 images (for shape analysis). The reason we have taken a different sample set for both experiments is we need definite shape boundaries for evaluating shape based similar image search precision and recall values. Table 3 presents the precision/recall figures along with the number of true positive, false positive and false negative results for the experiment.

From table 3, the precision of the Color Extraction Engine is 0.65, for Texture Extraction Engine is 0.97, and for shape based extraction it is again 0.97. The recall value for Color Based Extraction is 0.54, for texture based extraction is 0.85 and for shape based extraction the recall is 0.95.

The precision and recall values evaluate the correctness of the image feature and shape extraction engines rather than evaluating the correctness of the Search Friendly Image Format (SFIF) based database architecture. Table 2 rightly shows the performance of the system which gives results in fraction of milliseconds even for a larger result set of 4K images.

8.2.3 Use of Clustering

To evaluate the performance of the database architecture, we stored the results of the feature and shape extraction engines in our database without creating clusters and without defining indexes. The database schema was the same. The improvement shown by the use of clusters and indexes is more than 200 percent for the best case. Table 3 illustrates the performance figures.

Table 3: System Evaluation: Precision and Recall Computation with 100 known images

Search Type	Dominant Feature	Total Images	Retrieved Images	Relevant Images	True Positive	False Positive	False Negative
Color	Black	15	15	5	5	10	10
	Grey	4	4	4	4	0	0
	Pink	9	7	7	7	0	2
	Orange	4	4	4	4	0	0
	Light Blue	13	12	12	12	0	1
	Dark Blue	6	2	2	2	0	4
	Red	10	5	4	4	1	5
	Green	15	2	2	2	0	13
	White	13	16	7	7	9	6
	Yellow	1	7	1	1	6	0
	Brown	10	8	6	6	2	4
		Total	**82**	**54**	**54**	**28**	**45**
Texture	Non-Stochastic	10	8	8	8	0	2
	Irregular	22	18	18	18	0	4
	Regular	13	10	9	9	1	4
	Near-Regular	7	5	5	5	0	2
	Stochastic	48	46	45	45	1	3
		Total	**87**	**85**	**85**	**2**	**15**
Shape	Butterfly	22	24	22	22	2	0
	Star	9	9	9	9	0	0
	Guitar	20	18	18	18	0	2
	Maple Leaves	4	4	4	4	0	0
	Cup	7	6	6	6	0	1
	Cap	19	20	19	19	0	1
	Boots	7	7	7	7	0	0
	Watch	3	3	3	3	0	0
		Total	**91**	**88**	**88**	**2**	**4**

CHAPTER IX

CONCLUSION

We have presented a scalable system for image retrieval based on the image content. The system is different from existing systems in the following ways.

(i) The system can take multi-modal queries such as text query and an input image as query with different search options.

(ii) We use the MPEG-7 color, shape and texture descriptors for image content description. We compute a search tag and a normalized score for each image. The meta data is stored as an inherent part of the image embedded in XMP (eXtensible Markup Platform). It is used for matching the input image to the images in the database.

(iii) A new image meta data representation mechanism called the Search Friendly Image Format (SFIF) is proposed which has a subset of image features that are extracted and embedded in the image using the ORDImage object, a Java based API from Oracle Multimedia.

(iv) The image content extraction engines extract visual descriptors - color, texture, shape - of an image and also the textual descriptors - size, orientation and style.

(v) An object dictionary is created which is an exhaustive list of all dominant object shapes within the search database. The feature dictionary includes an exhaustive set of color and texture samples to be supported by the system. The distinguishing element of this design is the scalable architecture of the database.

(vi) For text queries, the query expansion engine is used which creates a word cloud from the input query text.

(vii) An intuitive user interface is designed to enable the users to effectively browse the system with flexible querying mechanisms and refinement options.

All these technologies result in a significant improvement in the search precision and recall values of the system. They also improve the retrieval time of similar images for an input query. However, there is a good scope for future enhancements to the system. We can include a module to find exact (or nearly exact) matches for an image input as query. The module will consist of a combined search on the image features. We can make use of multi-threading to extract individual image features and then do a combination search on the database. Another scope for improvement could be the use of parallel processing techniques to construct the database. We can store the images to be exported in different source directories and build the image feature database by scheduling parallel processes. One more enhancement to the system could be an efficient use of image segmentation as demonstrated in Manjunath's [32] work. We can create clusters for segmented regions and handle queries which semantically mean: "give me similar images whose top right is blue and contains a bird".

The system is modular, and different modules - image feature extraction, database creation and similar image retrieval - can be used individually for a particular research problem. The system is shown to be scalable by expanding the set of images that can be used to construct the shape and feature dictionaries. The system is inter-operable; the SFIF tags can be extracted using XML parsing and the information can be used by other search engines for content based retrieval.

REFERENCES

1. Ramprasath Dorairaj, Kamesh R. Namuduri, "Compact combination of mpeg-7 color and texture descriptors for image retrieval", *38th Asilomar Conference on Signals, Systems and Computers, ACM 2004*

2. G. Almeida, F. Melicio, A. Pinheiro, "Multimodal Semantic Characterization of Images Using MPEG-7 Descriptors", *9th Symposium on Neural Network Applications in Electrical Engineering ACM, NEUREL-2008*

3. Jiawei Han, "Data mining for image/video processing: a promising research frontier", *Proceedings of the 2008 international conference on Content-based image and video retrieval ACM*

4. Keiji Yanai , Kobus Barnard, "Finding Visual Concepts by Web Image Mining", *Proceedings of the 15th international conference on World Wide Web, 2006 ACM*

5. Rokia Missaoui, Roman M. Palenichka , "Effective Image and Video Mining: an Overview of Model-Based Approaches", *Proceedings of the 6th international workshop on Multimedia data mining: mining integrated media and complex data ACM*

6. Ka-Man Wong, Kwok-Wai Cheung, Lai-Man Po, "MIRROR: An Interactive Content Based Image Retrieval System", *ISCAS (2) 2005*

7. T. Kato, "Database architecture for content-based image retrieval", *Image Storage and Retrieval Systems, Proc SPIE Vol.1662, 1992, pp.112-123*

8. Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petkovic, David Steele, Peter Yanker, "Query by Image and Video Content: The QBIC System," *IEEE Computer Society Press, Computer, vol. 28, no. 9, pp. 23-32, Sept. 1995*

9. A. Pentland , R. W. Picard, S. Sclaroff, "Photobook: Content-based manipulation of image databases", *International Journal of Computer Vision, Volume 18, Number 3 / June, 1996, pp. 233-254*

10. Y Rui, TS Huang, S Mehrotra, "Content-based image retrieval with relevance feedback in MARS", *Proc. IEEE Int. Conf. on Image Proc, 1997*

11. James Z. Wang, JiaLi, Gio Wiederhold, "SIMPLIcity: Semantics-Sensitive Integrated Matching for Picture LIbraries", *Proceedings of the 4th International Conference on Advances in Visual Information Systems, 2000, pp.360 - 371*

12. S. K. Chang and T. L. Kunii, "Pictorial data base systems", *Computer, vol. 25, no. 11, pp. 13-21, Nov. 1981*

13. N. S. Chang and K. S. Fu, "Query by pictorial example", *IEEE Trans. Software Eng., vol. SE-6, no. 6, pp. 519-524, Nov. 1980*

14. S.F. Chang, J. R. Smith, M. Beigi, and A. Benitez, "Visual information retrieval from large distributed online repositories", *Comm. ACM (Special Issue on Visual Information Retrieval) Dec. 1997, pp. 1220*

15. S.-K. Chang and A. Hsu, "Image information systems: Where do we go from here?", *IEEE Trans. on Knowledge and Data Engineering 4(5), 1992. pp. 431-442*

16. J. R. Bach, C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C. F. Shu, "The Virage image search engine: An open framework for image management", *in Proc. SPIE Storage and Retrieval for Still Image and Video Databases IV, Vol. 2670, No. 1. (1996), pp. 76-87*

17. "Retrievalware", demo page. *http://en.wikipedia.org/wiki/RetrievalWare*, March 2010.

18. J. Dowe, "Content-Based Retrieval in Multimedia Imaging," *Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases, 1993*

19. J.R. Smith and S.-F.Chang, "Querying by Color Regions Using the Visual SEEk Content- Based Visual Query System," *in Intelligent Multimedia Information Retrieval, Maybury, M. T., Ed., American Association for Artificial Intelligence (AAAI), Menlo Park, CA, 1997*

20. J.R. Smith and S.-F.Chang, "VisualSEEk: A Fully Automated Content-Based Image Query System," *Proc. ACM Multimedia '96, Boston, MA, November, 1996*

21. J. R. Smith and S.-F. Chang, "Visually searching the web for content", *IEEE Multimedia Magazine 4(3), 1997. [Columbia U. CU/CTR Technical Report 459-96-25] pp. 1220*

22. M. Swain and D. Ballard, "Color indexing", *International Journal of Computer Vision 7(1), 1991, pp. 11-32*

23. W. Niblack, R. Ba rber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C.Faloutsos, and G. Taubin, "The QBIC Project: Querying Images By

Content Using Color, Texture and Shape," *Proc. SPIE Storage and Retrieval for Image and Video Databases,pp. 173-187, 1993*

24. M. Stricker and M. Orengo, "Similarity of Color Images," *in Proc. SPIE Storage and Retrieval for Still Image and Video Databases III, pp. 381392, San Jose, CA, USA, February 1995*

25. H. Tamura, S. Mori, and T. Yamawaki, "Texture features corresponding to visual perception", *IEEE Trans. on Syst., Man. and Cybern. SMC-8(6), pp. 460-473 1978*

26. W. Y. Ma and B. S. Manjunath, "A comparison of wavelet transform features for texture image annotation", *Proc. Second International Conference on Image Processing (ICIP'95), Washington, D.C., vol. 2, pp. 256-259, Nov. 1995*

27. Y. Rui, A.C. She, and T.S. Huang, "Modified Fourier Descriptors for Shape Representation A Practical Approach", *Proc. First Int'l Workshop Image Databases and Multi Media Search, pp. 22-23, Aug. 1996*

28. D. Kapur, Y.N. Lakshman, T. Saxena, "Computing invariants using elimination methods," *ISCV, pp.97, International Symposium on Computer Vision, 1995*

29. An Open Source Java Content Based Image Retrieval Library, *http://www.semanticmetadata*
, March 2010

30. MPEG-7 Reference Software (C++), *http://mpeg.chiariglione.org/technologies/mpeg-7/mp07-rsw/index.htm* , March 2010

31. L1-Norm explanation, *http://mathworld.wolfram.com/L1-Norm.html* , March 2010

32. W.-Y. Ma and B.S. Manjunath, "NeTra: A Toolbox for Navigating Large Image Databases", *Multimedia Systems, vol. 7, no. 3, pp. 184-198, , May, 1999*

33. Oracle Multimedia Reference 11g Release 1 (11.1), *http://download.oracle.com/docs/cd/B28*
March 2010

34. Popat.K, Picard.R, "Novel cluster-based probability model for texture synthesis, classification, and compression", *Visual Communications and Image Processing, Proc. SPIE, Vol. 2094, pp.756-768, 1993*

35. B.S.Manjunath,Philippe Salembier,Thomas Sikora, "Introduction to MPEG-

7: Multimedia Content Description Language", *2002, Wiley Publications*

VITA

Prajakta Kalmegh was born in Nagpur on July 29, 1983. She holds a Bachelor of Engineering degree in Information Technology from University of Pune. She has worked as a System Engineer in IBM India Pvt. Ltd for two and a half years before joining Georgia Tech for MS in Fall 2008. Her areas of interest include databases systems, multi-tier application/database programming and computer networks. She has worked on research projects in the area of database systems in Georgia Tech for two years.