Islamic University – Gaza
Deanery of Higher Studies
Faculty of Engineering
Computer Engineering Department

# Efficient Content Based Image Retrieval

# استرجاع الصورة من خلال محتواها بكفاءة

**By**

**Ruba A. A. Salamah**

**Supervisor**

**Prof. Ibrahim S. I. Abuhaiba**

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering

1431H (2010)

# ACKNOWLEDGMENT

My thanks go to all people that were involved in my master's thesis. I would like to express my immense gratitude to my supervisor, Professor Ibrahim Abuhaiba, for his advice, support, guidance and inspiration to conduct my research.

I would also like to thank all my friends and my colleagues in the computer engineering department for their friendship and assistance during my research study. Special thanks to Dr. Wesam Ashour and Dr. Mohammad Al-Hanjouri for their assistance and expert advice.

My heartiest gratitude to my beloved family: my husband, mother, and children; without their support, understanding, and patience, it would have been impossible for me to finally complete my study.

Above all, I thank Allah for blessing me with all these resources, favors and enabling me to complete this thesis.

# Table of Contents

# List of Figures

# List of Tables

# List of ABBREVIATIONS

| | | |
|---|---|---|
| AMORE | : | A World Wide Web Image Retrieval Engine |
| BMU | : | Best Matching Unit |
| CBIR | : | Content Based Image Retrieval |
| CIE | : | International Commission on Illumination |
| DB | : | DataBase |
| ER | : | Extended Rectangle |
| ERBIR | : | Efficient Region Based Image Retrieval |
| GBIR | : | Global feature Based Image Retrieval |
| GCH | : | Global Color Histogram |
| GLCM | : | Gray-Level Co-occurrence Matrix |
| GRBIR | : | Global and Region feature Based Image Retrieval |
| HIT | : | Histogram Intersection Technique |
| HSV | : | Hue, Saturation, Value color space |
| HVS | : | Human Visual System |
| IR | : | Inner Rectangle |
| IRM | : | Integrated Region Matching |
| LCH | : | Local Color Histogram |
| PWT | : | Pyramid-structured Wavelet Transform |
| QBIC | : | Query By Image and video Content |
| RBIR | : | Region Based Image Retrieval |
| RGB | : | Red, Green, Blue color space |
| SOM | : | Self Organizing Map |

| | | |
|---|---|---|
| TBES | : | Texture and Boundary Encoding-based Segmentation |
| TWT | : | Tree structured Wavelet Transform (TWT) |
| VIR | : | Visual Information Retrieval |
| WALRUS | : | WAveLet-based Retrieval of User-specified Scenes |
| WBIIS | : | Wavelet Based Image Indexing and Searching |

# استرجاع الصورة من خلال محتواها بكفاءة

**مقدم من**

**ربا عبد الحميد سلامة**

## الملخص

لقد أصبح استرجاع الصور من خلال محتواها من قواعد بيانات كبيرة يتمتع باهتمام كبير هذه الأيام في العديد من المجالات.

في أطروحة الماجستير هذه نقدم نظاما لاسترجاع الصور من خلال محتواها بكفاءة على أساس المناطق الموجودة فيها وهذا النظام يستخدم اللون والنسيج كخصائص بصرية لوصف محتوى الصورة. مساهمتنا في هذا البحث تتكون من ثلاث توجهات. أولا، نحن نستخدم تقنية Gabor Filters لاستخراج ميزات النسيج من مناطق عشوائية الشكل تمَ فصلها من الصورة بعد عملية التجزئة. وثانيا، لتسريع عملية حساب التشابه و استرجاع الصور، نقوم بتقسيم الصور الموجودة في قاعدة البيانات الى مناطق،ثم نقوم بتنظيم هذه المناطق في مجموعات بناءً على الخصائص المستخرجة منها وذلك باستخدام تقنية الخرائط ذاتية التقسيم (SOM). هذه العملية تتم في مرحلة سابقة لعملية الاستعلام، وبالتالي للرد على استعلام معين فإن هذا النظام لا يحتاج للبحث في صور قاعدة البيانات بأكملها، بدلا من ذلك فإنه يبحث فقط في عدد من الصور المرشحة المطلوبة ليتم البحث فيها عن التشابه مع الصورة المطلوبة. وثالثا، لزيادة دقة نتائج الاسترجاع في نظامنا، نحن نستخدم الجمع بين الخصائص المستخرجة من مناطق الصورة مع الخصائص المستخرجة من الصورة بأكملها والتي هي النسيج باستخدام مرشحات جابور، والمدرج الاحصائي للون. في تحليل النظام من خلال المحاكاة نعطي مقارنة بين نتائج استرجاع الصورة على أساس الخصائص المستخرجة من الصورة كاملة، والخصائص المستخرجة من مناطق الصورة بعد تقسيمها، واستخدام هذين النوعين من الخصائص معاً .

النظام المقترح يعتبر تحسين لنظام استرجاع الصورة من خلال محتواها عن طريق زيادة دقة نتائج الاسترجاع بالاضافة الى تقليل الزمن الذي يحتاجه النظام لاسترجاع الصور. لقد تم تقييم النظام المقترح استنادا الى قاعدة صور مكونة من 1000 صورة ملونة من مجموعة كورال. النتائج العملية أثبتت تفوق النظام المقترح على عدد من النظم والتقنيات الموجودة من حيث الدقة والسرعة.

في تحليل النظام من خلال المحاكاة نعطي أيضا مقارنة بين نتائج استرجاع الصورة على أساس الخصائص المستخرجة من الصورة كاملة، والخصائص المستخرجة من مناطق الصورة بعد تقسيمها. وقد أثبتت النتائج أن كل نوع من هذه الخصائص يكون أكثر فعالية مع نوع معين من الصور بحسب دلالة محتوياتها، واستخدام هذين النوعين معا يعطي نتائج استرجاع أفضل تقريبا مع كل أنواع الصور باختلاف محتوياتها.

***الكلمات المفتاحية:*** استرجاع الصورة من خلال محتواها، خصائص مبنية على المناطق، مرشحات جابور، المدرج الاحصائي للون، خرائط ذاتية التنظيم.

# Efficient Content Based Image Retrieval

**By**

**Ruba A. A. Salamah**

# ABSTRACT

Content based image retrieval from large resources has become an area of wide interest nowadays in many applications.

In this thesis we present a region-based image retrieval system that uses color and texture as visual features to describe the content of an image region. Our contribution is of three directions. First, we use Gabor filters to extract texture features from arbitrary shaped regions separated from an image after segmentation to increase the system effectiveness. Second, to speed up retrieval and similarity computation, the database images are segmented and the extracted regions are clustered according to their feature vectors using Self Organizing Map (SOM) algorithm. This process is performed offline before query processing, therefore to answer a query our system does not need to search the entire database images; instead just a number of candidate images are required to be searched for image similarity. Third, to further increase the retrieval accuracy of our system, we combine the region based features extracted from image regions, with global features extracted from the whole image, which are texture using Gabor filters and color histograms.

Our proposed system has the advantage of increasing the retrieval accuracy and decreasing the retrieval time. The experimental evaluation of the system is based on a 1000 COREL color image database. From the experimental results, it is evident that our system performs significantly better and faster compared with other existing systems.

In our simulation analysis, we provide a comparison between retrieval results based on features extracted from the whole image, and features extracted from some image regions. The results demonstrate that each type of feature is effective for a particular type of images according to its semantic contents, and using a combination of them gives better retrieval results for almost all semantic classes.

***Keywords:*** Content Based Image Retrieval (CBIR), Region Based Features, Global Based Features, Texture, Gabor Filters, Self Organizing Map (SOM).

# Chapter 1

# INTRODUCTION

## 1.1 Information Retrieval

In the past decade, more and more information has been published in computer readable formats. In the meanwhile, much of the information in older books, journals and newspapers has been digitized and made computer readable. Big archives of films, music, images, satellite pictures, books, newspapers, and magazines have been made accessible for computer users. Internet makes it possible for the human to access this huge amount of information. The greatest challenge of the World Wide Web is that the more information available about a given topic, the more difficult it is to locate accurate and relevant information. Most users know what information they need, but are unsure where to find it. Search engines can facilitate the ability of users to locate such relevant information.

## 1.2 Image Retrieval Problem

In this computer age, virtually all spheres of human life including commerce, government, academics, hospitals, crime prevention, surveillance, engineering, architecture, journalism, fashion and graphic design, and historical research use images for efficient services. A large collection of images is referred to as image database. An image database is a system where image data are integrated and stored [1]. Image data include the raw images and information extracted from images by automated or computer assisted image analysis.

The police maintain image database of criminals, crime scenes, and stolen items. In the medical profession, X-rays and scanned image database are kept for diagnosis, monitoring, and research purposes. In architectural and engineering design, image database exists for design projects, finished projects, and machine parts. In publishing and advertising, journalists create image databases for various events and activities such as sports, buildings, personalities, national and international events, and product advertisements. In historical research, image databases are created for archives in areas that include arts, sociology, and medicine. In a small collection of images, simple browsing can identify an image. This is not the case for large and varied collection of images, where the user encounters the image retrieval problem. An image retrieval problem is the problem encountered when searching and retrieving images that are relevant to a user's request from a database. To solve this problem, text-based and content-based are the two techniques adopted for search and retrieval in an image database.

## 1.3 Text-Based and Content-Based Image Retrieval

In text-based retrieval, images are indexed using keywords, subject headings, or classification codes, which in turn are used as retrieval keys during search and retrieval [2]. Text-based retrieval is non-standardized because different users employ different keywords for annotation. Text descriptions are sometimes subjective and incomplete because they cannot depict complicated image features very well. Examples are texture images that cannot be described by text. Textual information about images can be easily searched using existing technology, but requires humans to personally describe every image in the database. This is impractical for very large databases, or for images that are generated automatically, e.g. from surveillance cameras. It is also possible to miss images that use different synonyms in their descriptions. Systems based on categorizing images in semantic classes like "cat" as a subclass of "animal" avoid this problem, but still face the same scaling issues [3].

The Content Based Image Retrieval (CBIR) technique uses image content to search and retrieve digital images. Content-based image retrieval systems were introduced to address the problems associated with text-based image retrieval. Content based image retrieval is a set of techniques for retrieving semantically-relevant images from an image database based on automatically-derived image features [4]. The main goal of CBIR is efficiency during image indexing and retrieval, thereby reducing the need for human intervention in the indexing process. The computer must be able to retrieve images from a database without any human assumption on specific domain (such as texture vs. non-texture, or indoor vs. outdoor).

One of the main tasks for CBIR systems is similarity comparison; extracting feature signatures of every image based on its pixel values and defining rules for comparing images. These features become the image representation for measuring similarity with other images in the database. An image is compared to other images by calculating the difference between their corresponding features.

## 1.4 Fields of Application

Image retrieval based on content is extremely useful in a plethora of applications such as publishing and advertising, historical research, fashion and graphic design, architectural and engineering design, crime prevention, medical diagnosis, geographical information and remote sensing systems, etc. [5]. A typical image retrieval application example is a design engineer who needs to search his organization database for design projects similar to that required by his clients, or the police seeking to confirm the face of a suspected criminal among faces in the database of renowned criminals. In the commerce department, before trademark is finally approved for use, there is need to find out if such or similar ones ever existed. In hospitals, some ailments require the medical practitioner to search and review similar X-rays or scanned images of a patient before proffering a solution.

The most important application, however, is the Web, as big fraction of it is devoted to images, and searching for a specific image is indeed a daunting task. Numerous commercial and experimental CBIR systems are now available, and many web search engines are now equipped with CBIR facilities, as for example Alta Vista, Yahoo and Google [6].

## 1.5 Principle of CBIR

Content-based retrieval uses the contents of images to represent and access the images. A typical content-based retrieval system is divided into off-line feature extraction and on-line image retrieval. A conceptual framework for content-based image retrieval is illustrated in Figure 1.1 [4]. In off-line stage, the system automatically extracts visual attributes (color, shape, texture, and spatial information) of each image in the database based on its pixel values and stores them in a different database within the system called a feature database. The feature data (also known as image signature) for each of the visual attributes of each image is very much smaller in size compared to the image data, thus the feature database contains an abstraction (compact form) of the images in the image database. One advantage of a signature over the original pixel values is the significant compression of image representation. However, a more important reason for using the signature is to gain an improved correlation between image representation and visual semantics [4].

In on-line image retrieval, the user can submit a query example to the retrieval system in search of desired images. The system represents this example with a feature vector. The distances (i.e., similarities) between the feature vectors of the query example and those of the media in the feature database are then computed and ranked. Retrieval is conducted by applying an indexing scheme to provide an efficient way of searching the image database. Finally, the system ranks the search results and then returns the results that are most similar to the query examples. If the user is not satisfied with the search results, he can provide relevance feedback to the retrieval system, which contains a mechanism to learn the user's information needs.

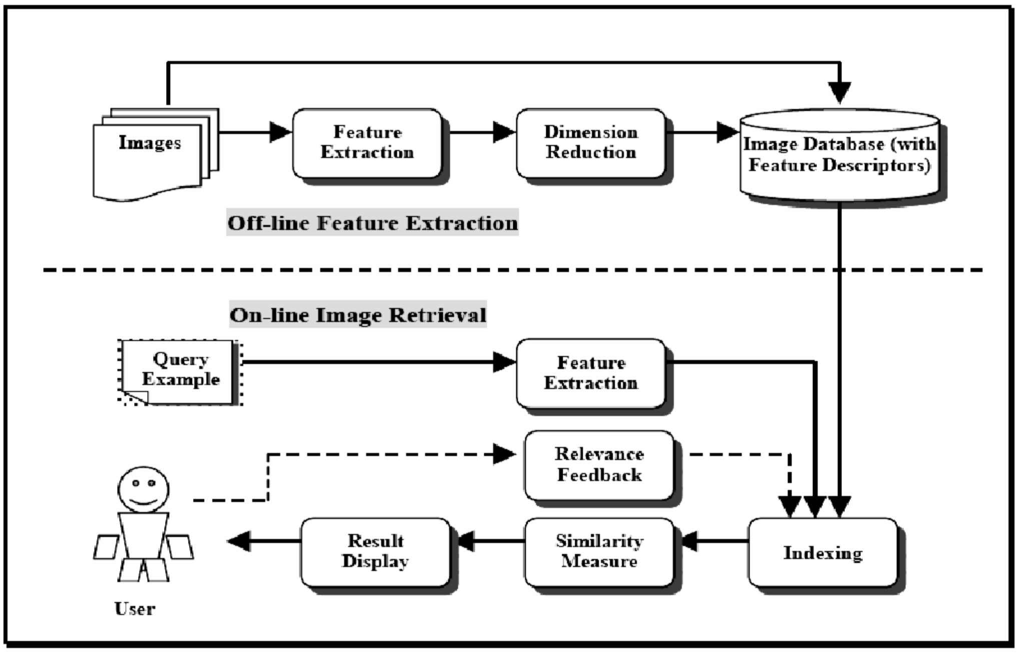



**Figure 1.1:** A Conceptual Framework for Content-Based Image Retrieval.

## 1.6 Region Based Image Retrieval (RBIR)

Early CBIR methods used global feature extraction to obtain the image descriptors. For example, QBIC [7] developed at the IBM Almaden Research Center extracts several features from each image, namely color, texture, and shape features. These descriptors are obtained globally by extracting information by means of color histograms for color features; global texture information on coarseness, contrast, and direction; and shape features about the curvature, moments invariants, circularity, and eccentricity. Similarly, the Photobook system [8], Visualseek [9], and VIR [10], use global features to represent image semantics.

These global approaches are not adequate to support queries looking for images where specific objects in an image with particular colors and/or texture are present, and shift/scale invariant queries, where the position and/or the dimension of the query objects may not be relevant. For example, suppose in one image there are two flowers with different colors: red and yellow. The global features describe the image as the average of the global average color which is orange. This description is certainly not the representation of the semantic meaning of the image. Therefore, the weakness of global features is observable.

Region-based retrieval systems attempt to overcome previous method limitations of global based retrieval systems by representing images as collections of regions that may correspond to objects such as flowers, trees, skies and mountains [11].

A key prerequisite for a good region based image retrieval system is a robust segmentation algorithm [12]. A segmentation algorithm takes an input image and clusters pixels of this image that seem to be similar with respect to some feature (e.g. color, texture, or shape). The result of this clustering phase is to decompose an image into regions, which correspond to physical objects (trees, people, cars, flowers) if the decomposition is ideal. The feature descriptors are then extracted from each object instead of global image. Color, texture, and shape features are extracted on each pixel that belongs to the object, and each object is described by the average value of these pixel features.

## 1.7 Motivation of This Work

The design and development of effective and efficient CBIR systems are still a research problem, because the nature of digital images involves two well-known problems: the semantic gap and the computational load to manage large file collections. The semantic gap is the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation [13]. It has linguistic and contextual consequences, and mainly depends on the domain knowledge to represent images. On the other hand, the computation load, when large image collections are managed, may make impractical use of CBIR systems [14].

The aim of this thesis is to propose a new CBIR system; an important task of the system is **1)** to reduce the "semantic gap" between low-level image features and the richness of human semantics and **2)** to reduce the overall retrieval time. The system first segments images into regions that correspond to the objects in it. A combination of texture, and color features are extracted from each region in the segmented image. The contribution of this work is of three directions:

1. Salient low-level texture features are extracted from arbitrary-shaped regions using Gabor filter, which has been a widely acclaimed natural and excellent tool in texture feature classification, segmentation, and extraction [15]. In many systems, texture features are obtained during segmentation from pixels or small blocks [8, 11]. Such features may not well represent the property of an entire region. In some other systems, segmentation does not produce texture features. Hence, it is necessary to study texture feature extraction from the whole region after segmentation.
2. Many of the existing systems attempt to compare the query image with every target image in the database to find the top matching images, resulting in an essentially linear search, which is prohibitive when the database is large. We believe that it is not necessary to conduct a whole database comparison. In fact, it is possible to exploit a priori information regarding the “organization” of the images in the database in the feature space before a query is posed, such that when a query is received, only a part of the database needs to be searched, while a large portion of the database may be eliminated in the search. This certainly saves significant query

processing time without compromising the retrieval precision. To speed up the retrieval process, the database images are segmented into distinct regions. A clustering algorithm, definitely the self organizing map (SOM), is used to cluster the image regions into groups. Regions with similar features are grouped together in the same class. This clustering process is performed offline, and each region's indexing data along with its associated class ID is recorded in the index files. To answer a query, the query image is segmented into its regions. The distances between each query region and all class centroids in the database is computed to determine which class of these query regions belong. The similar regions in the database are returned and all the images that have any of these regions are assigned as candidates. The query image is compared to the candidate image set instead of being compared to the whole database image.

3. To further increase the performance of the system, we develop a global searching algorithm (referred to as global features based CBIR) that uses texture and color features from the whole image to compute the distance between two images. This algorithm is combined with the region based searching algorithm using weighted sum of the two distances, and by this we use properties of image regions associated with the general properties of the image for similarity computation between a query and database images.
4. We make a comparison between image retrieval using region based features and global based features.

Results illustrate that the system developed in this thesis significantly improves the overall retrieval quality compared to the previous existing systems.

## 1.8 Organization of This Thesis

The rest of the thesis is organized as follows. Chapter 2 summarizes some of the related works in the topic of CBIR and the primary research issues. In Chapter 3, we introduce an overview of the CBIR system, its principle, and the techniques used for feature extraction, similarity measure, and indexing structures. Gabor filters, the essential technique used for feature extraction in the proposed systems, are discussed in Chapter 4. The development of the global features based retrieval and the region based retrieval systems is introduced in Chapters 5, and 6 respectively. We also present the combination of the two systems as our overall proposed system in Chapter 6. Simulation results and evaluation of the two systems, as well as the overall system, are detailed in Chapter 7. Finally, Chapter 8 concludes our work and suggests future work.

# Chapter 2

# Literature Review

Content based image retrieval for general-purpose image databases is a highly challenging problem because of the large size of the database, the difficulty of understanding images, both by people and computers, the difficulty of formulating a query, and the issue of evaluating results properly. A number of general-purpose image search engines have been developed. In the commercial domain, QBIC [7] is one of the earliest systems. Recently, additional systems have been developed such as T.J. Watson [18], VIR [10], AMORE [19], and Bell Laboratory WALRUS [20]. In the academic domain, MIT Photobook [8, 21] is one of the earliest systems. Berkeley Blobworld [22], Columbia Visualseek and Webseek [9], Natra [23], and Stanford WBIIS [24] are some of the recent well known systems.

The common ground for CBIR systems is to extract a signature for every image based on its pixel values and to define a rule for comparing images. The signature serves as an image representation in the “view” of a CBIR system. The components of the signature are called features. One advantage of a signature over the original pixel values is the significant compression of image representation. However, a more important reason for using the signature is to gain an improved correlation between image representation and semantics. Actually, the main task of designing a signature is to bridge the gap between image semantics and the pixel representation, that is, to create a better correlation with image semantics [11].

Existing general-purpose CBIR systems roughly fall into three categories depending on the approach to extract signatures: histogram, color layout, and region-based search. There are also systems that combine retrieval results from individual algorithms by a weighted sum matching metric [4], or other merging schemes [25].

After extracting signatures, the next step is to determine a comparison rule, including a querying scheme and the definition of a similarity measure between images. For most image retrieval systems, a query is specified by an image to be matched. We refer to this as global search since similarity is based on the overall properties of images. By contrast, there are also "partial search" querying systems that retrieve results based on a particular region in an image [26].

## 2.1 Global Feature Based CBIR Systems

Some of the existing CBIR systems extract features from the whole image not from certain regions in it; these features are referred to as Global features. Histogram search algorithms [7] characterize an image by its color distribution or histogram. Many distances have been used to define the similarity of two color histogram representations. Euclidean distance and its variations are the most commonly used. The drawback of a global histogram representation is that information about object location, shape and texture is discarded. Color histogram search is sensitive to intensity variations, color distortions, and cropping. The color layout approach attempts to overcome the drawback of histogram search. In simple color layout indexing [7], images are partitioned into blocks and the average color of each block is stored. Thus, the color layout is essentially a low resolution representation of the original image. A relatively recent system, WBIIS [24], uses significant Daubechies' wavelet coefficients instead of averaging. By adjusting block sizes or the levels of wavelet transforms, the coarseness of a color layout representation can be tuned. Hence, we can view a color layout representation as an opposite extreme of a histogram. At proper resolutions, the color layout representation naturally retains shape, location, and texture information. However, as with pixel representation, although information such as shape is preserved in the color layout representation, the retrieval system cannot perceive it directly. Color layout search is sensitive to shifting, cropping, scaling, and rotation because images are described by a set of local properties [4].

Image retrieval using color features often gives disappointing results, because in many cases, images with similar colors do not have similar content. This is due to the fact that global color features often fails to capture color distributions or textures within the image.

D. Zhang [27] proposed a method combining both color and texture features to improve retrieval performance. By computing both the color and texture features from the images, the database images are indexed using both types of features. During the retrieval process, given a query image, images in the database are firstly ranked using color features. Then, in a second step, a number of top ranked images are selected and re-ranked according to their texture features. Two alternatives are provided to the user, one is the retrieval based on color features, and the other is retrieval based on combined features. When the retrieval based on color fails, the user will use the other alternative which is the combined retrieval. Since the texture features are extracted globally from the image; they are not an accurate description of the image in some cases, which degrades the system performance.

## 2.2 Region Based CBIR Systems

Region-based retrieval systems attempt to overcome the deficiencies of global feature based search by representing images at the object-level. A region-based retrieval system applies image segmentation to decompose an image into regions, which correspond to objects if the decomposition is ideal [17]. The object-level representation is intended to be close to the perception of the human visual system (HVS). Since the retrieval system has identified what objects are in the image, it is easier for the system to recognize similar objects at different locations and with different orientations and sizes. Region-based retrieval systems include the Natra system [23], and the Blobworld system [22]. The Natra and the Blobworld systems compare images based on individual regions. The motivation is to shift part of the comparison task to the users. To query an image, a user is provided with the segmented regions of the image and is required to select the regions to be matched and also attributes, e.g., color and texture, of the regions to be used for evaluating similarity. Such querying systems provide more control to the user. However, the user's semantic understanding of an image is at a higher level than the region representation. For objects without discerning attributes, such as special texture, it is not obvious for the user how to select a query from the large variety of choices. Thus, such a querying scheme may add burdens on users without significant reward.

Recently, Natsev et al. considered the similarity model WALRUS [20], which is a robust model for scaling and translation of objects within an image. Each image is first decomposed into regions. The similarity measure between two images is then defined as the fraction of the area of the two images covered by matching regions. However, WALRUS focuses on the development of a fast and effective segmentation method instead of an image-to-image similarity measure. Consequently, region matching should be necessary before image matching. The authors proposed a greedy heuristic for computing the similar region pair set with the maximum area. The basic idea is to iteratively choose the best pair of matching regions that maximizes the area covered by the regions. The time complexity of the above greedy algorithm is $O(n^2)$, where n is the number of matching pairs obtained by the R*-tree search. In [28], the mean shift algorithm is used for segmentation of images and interested regions are indexed using cluster-based R*-tree to increase the efficiency of the retrieval process. However, this system uses only color as image signature, which is sensitive to shifting, cropping, scaling, and rotation. Also, query is by image region matching, while a user's semantic understanding of an image is at a higher level than region representation.

Region based image retrieval [29] uses low-level features including color, texture, and edge density. For color, the histogram of image regions are computed, for texture co-occurrence matrix based entropy, energy, etc, are calculated, and for edge density it is Edge Histogram Descriptor (EHD) that is found. To decrease the retrieval time of images, an idea is developed based on greedy strategy to reduce the computational complexity. In this strategy, the query image is compared to each of the target images in the database based on region matching in term of Euclidian distance between them. The system then arranges the segments of each image in decreasing order based on the size of each segment. When a query is presented to the system, it starts comparing from the first region, if the distance between the query region and the target region is less than the threshold value, the system continues to check the other regions; otherwise it exits and does not check the other segments marking the target image as an irrelevant one.

To measure the similarity between images, Li and Wang et al [11], proposed the Integrated Region Matching (IRM) algorithm, which allows matching a region of one image to several regions of another image. That is, the region mapping between any two images is a many-to-many relationship. As a result, the similarity between two images is defined as the weighted sum of distances in the feature space, between all regions from different images. Compared with retrieval systems based on individual regions, such as Blobworld, the IRM approach decreases the impact of inaccurate segmentation by smoothing over the imprecision in distances. IRM incorporates the properties of all the segmented regions so that information about an image can be fully used. To increase the robustness against segmentation errors, IRM allows a region to be matched to several regions in another image. Each matching is assigned a significance credit, which corresponds to the importance of the matching. There are several ways to assign the importance of a region. One can assume that every region is equally important. IRM views that important objects in an image tend to occupy larger areas, called an area percentage scheme. This scheme is less sensitive to inaccurate segmentation than the uniform scheme. If one object is partitioned into several regions, the uniform scheme raises its significance improperly, whereas the area percentage scheme retains its significance to the region.

Fuzzy Club [16] addresses the issue of effective and efficient content based image retrieval by presenting an indexing and retrieval system that integrates color, texture, and shape information for the indexing and retrieval, and applies these region features obtained through unsupervised segmentation, as opposed to applying them to the whole image domain. Fuzzy Club emphasizes improving on a color feature “inaccuracy” problem in the region based literature – that is color histogram bins are not independent. For instance, if the color spectrum is divided into 10 bins, these bins are not independent--some are closer or farther away from each other in the original color space. Fuzzy logic is applied to the traditional color histogram to solve this problem to some degree. Fuzzy Club first segments an image into regions of 4x4 blocks and extracts color and texture features on each block. The k-means algorithm is used to cluster similar pixels together to form a region. The Lab color space is used to extract color features and Haar wavelet transform is used to extract three texture features. A secondary clustering is performed to reduce query processing time. Regions with similar features are grouped together in the same class. This secondary clustering is performed offline, and each region’s indexing data along with its associated class ID are recorded in the index files. The distances between each query region and all class centroids in the database are computed to determine to which class these query regions belong. The similar regions in the database are returned and all the images that have any of the member regions are assigned as candidate. The query image is compared to the candidate image set.

## 2.3 Research Issues

Remarkable observations in the review of related works are as follows:

1. Despite the fact that Gabor filters are a widely acclaimed natural and excellent tool in texture feature classification, segmentation, and extraction, only few CBIR systems utilize Gabor filters for texture feature extraction.
2. In the existed RBIR systems, the texture features are obtained during segmentation from pixels or small blocks. Such features do not properly represent the property of an entire region; thus it is necessary to study texture feature extraction from the whole region after segmentation.
3. Even though RBIR systems increased the retrieval accuracy, they require high complex computations to calculate similarity, since these systems need to consider each region in the database images, resulting in high retrieval response time. Thus, we need a solution to reduce the number of database regions included in the similarity computation.
4. The existing CBIR systems use either global features, or region based features to represent the content of an image. Each type of these features can be significant in representing images with certain semantics. For example, global features are useful for retrieving textured images that have no specific regions in accordance to the user, such as natural scenes used as backgrounds. Thus, utilizing an integration of both types of features can improve the performance of the retrieval system.

# Chapter 3

# Content Based Image Retrieval

The search for similar images in large-scale image databases has been an active research area in the last couple of years. A very promising approach is content based image retrieval (CBIR). In such systems, images are typically represented by approximations of their contents. Typical approximations consist of statistics, and Fourier or wavelet transformations of the raw image data. This so-called feature extraction aims at extracting information that is semantically meaningful but needs a small amount of storage [30]. A detailed description of some feature extraction techniques is introduced in section 3.1.

The information gained by feature extraction is used to measure the similarity between two images. Images are represented by points in the high dimensional feature space. Each extent of the feature corresponds to one dimension in the feature space. A metric is defined to calculate the actual similarity between two of these points. An overview of common metrics is given in section 3.2.

In the basic model shown in Figure 3.1 [31], the search for images similar to a query image ‘q’ results in finding the ‘k’ nearest neighbors of ‘q’. The model can be extended to support more complex queries that can consist of more than one query image and more than one feature type.

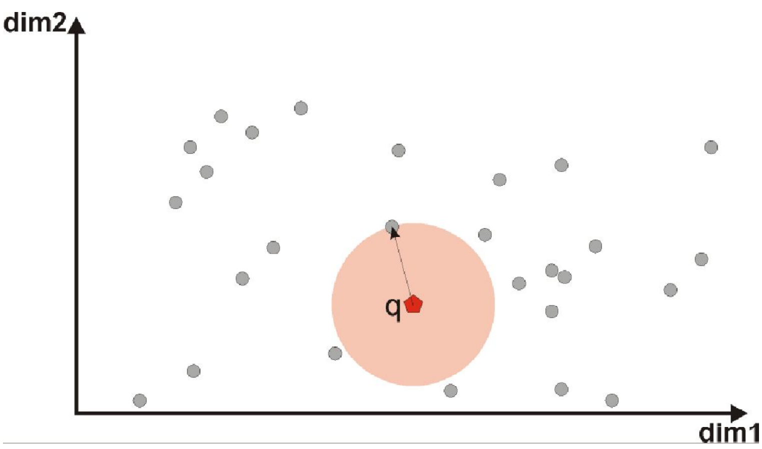



**Figure 3.1:** The Nearest Neighbor in Two Dimensions.

For fast retrieval, an indexing structure based on the query model is developed. In section 3.3, we present some of the indexing structures that are commonly used in CBIR systems.

## 3.1 Feature Extraction

Feature extraction is a means of extracting compact but semantically valuable information from images. This information is used as a signature for the image. Similar images should have similar signatures. If we look at the image shown in Figure 3.2, the white color and the texture of the building are characteristic properties. In a similar way, the sky can be described by its blue color. Furthermore, we can take the size of the objects in the image into account.

**Figure 3.2:** Example of Image Properties.

Representation of images needs to consider which features are most useful for representing the contents of images and which approaches can effectively code the attributes of the images. Feature extraction of the image in the database is typically conducted off-line so computation complexity is not a significant issue.  This section introduces three features: texture, shape, and color, which are used most often to extract the features of an image.

### 3.1.1 Color

One of the most important features visually recognized by humans in images is color. Humans tend to distinguish images based mostly on color features. Because of this, color features are the most widely used in CBIR systems and the most studied in literature.

Color is a powerful descriptor that simplifies object identification, and is one of the most frequently used visual features for content-based image retrieval. To extract the color features from the content of an image, a proper color space and an effective color descriptor have to be determined.

The purpose of a color space is to facilitate the specification of colors. Each color in the color space is a single point represented in a coordinate system. Several color spaces, such as RGB, HSV, CIE L*a*b, and CIE L*u*v, have been developed for different purposes [32]. Although there is no agreement on which color space is the best for CBIR, an appropriate color system is required to ensure perceptual uniformity. Therefore, the RGB color space, a widely used system for representing color images, is not suitable for CBIR because it is a perceptually non-uniform and device-dependent system [33]. The most frequently used technique is to convert color representations from the RGB color space to the HSV, CIE L*u*v, or CIE L*a*b color spaces with perceptual uniformity. The HSV color space is an intuitive system, which describes a specific color by its hue, saturation, and brightness values. This color system is very useful in interactive color selection and manipulation. The CIE L*u*v and CIE L*a*b color spaces are both perceptually uniform systems, which provide easy use of similarity metrics for comparing color [34].

After selecting a color space, an effective color descriptor should be developed in order to represent the color of the global or regional areas. Several color descriptors have been developed from various representation schemes, such as color histograms [35], color moments [36], color edge [37], color texture [38], and color correlograms [39].

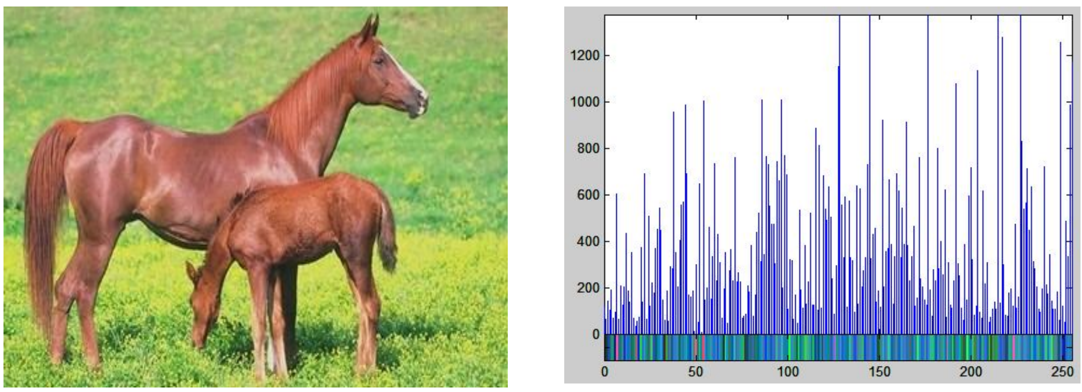
- **Color Histogram**

The most commonly used method to represent color feature of an image is the color histogram. A color histogram is a type of bar graph, where the height of each bar represents an amount of particular color of the color space being used in the image [32]. The bars in a color histogram are named as bins and they represent the x-axis. The number of bins depends on the number of colors there are in an image. The number of pixels in each bin denotes y-axis, which shows how many pixels in an image are of a particular color. The color histogram can not only easily characterize the global and regional distribution of colors in an image, but also be invariant to rotation about the view axis.

In color histograms, quantization is a process where number of bins is reduced by taking colors that are similar to each other and placing them in the same bin. Quantizing reduces the space required to store the histogram information and time to compare the histograms. Obviously, quantization reduces the information regarding the content of images; this is the tradeoff between space, processing time, and accuracy in results [40].

Color histograms are classified into two types, global color histogram (GCH) and local color histogram (LCH). A GCH takes color histogram of whole image and thus represents information regarding the whole image, without concerning color distribution of regions in the image. In the contrary, an LCH divides an image into fixed blocks or regions, and takes the color histogram of each of those blocks. LCH contains more information about an image, but when comparing images, it is computationally expensive. GCH is known as a traditional method for retrieving color based images. Since it does not include color distribution of the regions, when two GCHs are compared, one might not always get a proper result when viewed in terms of similarity of images [41].

An example of a color histogram in the HSV color space can be seen with the image in Figure 3.3.




a) Sample Image. b) Corresponding Color Histogram.

**Figure 3.3:** Sample Image and Its Corresponding Color Histogram.

We use the color histogram technique to extract the global color feature in our proposed global features based retrieval system, as will be detailed in Chapter 5.

### 3.1.2 Texture

In the field of computer vision and image processing, there is no clear-cut definition of texture. This is because available texture definitions are based on texture analysis methods and the features extracted from the image. However, texture can be thought of as repeated patterns of pixels over a spatial domain, of which the addition of noise to the patterns and their repetition frequencies results in textures that can appear to be random and unstructured. Texture properties are the visual patterns in an image that have properties of homogeneity that do not result from the presence of only a single color or intensity. The different texture properties as perceived by the human eye are, for example, regularity, directionality, smoothness, and coarseness, see Figure 3.4(a).

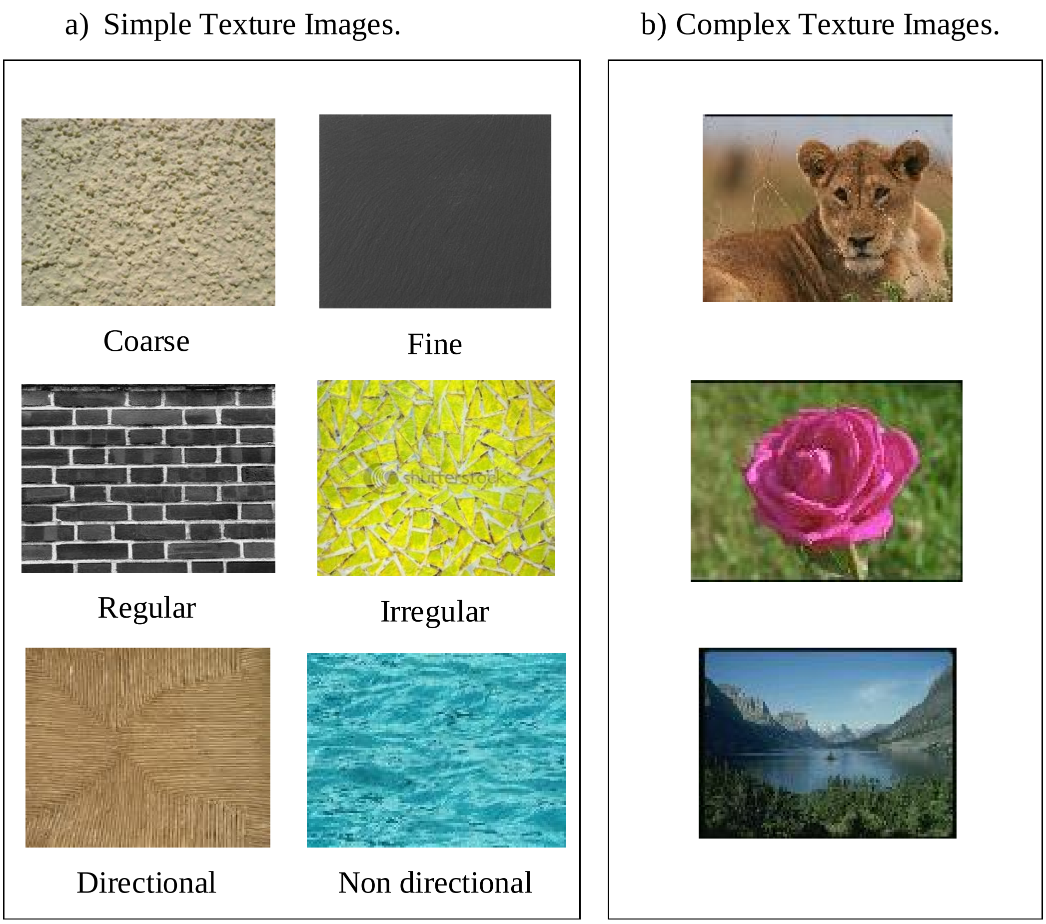



**Figure 3.4:** Examples of Simple and Complex Texture Images.

In real world scenes, texture perception can be far more complicated. The various brightness intensities give rise to a blend of the different human perception of texture as shown in Figure 3.4(b).

Image textures have useful applications in image processing and computer vision. They include: recognition of image regions using texture properties, known as texture classification, recognition of texture boundaries using texture properties, known as texture segmentation, texture synthesis, and generation of texture images from known texture models.

Since there is no accepted mathematical definition for texture, many different methods for computing texture features have been proposed over the years. Unfortunately, there is still no single method that works best with all types of textures. According to Manjunath and Ma [15], the commonly used methods for texture feature description are statistical, model-based, and transform-based methods. The texture feature description categories are explained below.

- **Statistical Methods**

Statistical methods analyze the spatial distribution of grey values by computing local features at each point in the image, and deriving a set of statistics from the distribution of the local features. They include co-occurrence matrix representation, statistical moments, gray level differences, autocorrelation function, and grey level run lengths.

The most commonly used statistical method is the Gray-level Co-occurrence Matrix (GLCM) [42]. It is a two-dimensional matrix of joint probabilities $P_{d,r}(i, j)$ between pairs of pixels, separated by a distance, d, in a given direction, r. It is popular in texture description and is based on the repeated occurrence of some gray level configuration in the texture; this configuration varies rapidly with distance in fine textures and slowly in coarse textures. Haralick [42] defined 14 statistical features from gray-level co-occurrence matrix for texture classification, such as energy, entropy, contrast, maximum probability, autocorrelation, and inverse difference moment. .

Gray-level co-occurrence matrix method of representing texture features has found useful applications in recognizing fabric defects, and in rock texture classification and retrieval [30].

- **Model Based Approaches**

Model-based texture methods try to capture the process that generated the texture. By using the model-based features, some part of the image model is assumed and an estimation algorithm is used to set the parameters of the model to yield the best fit [43]. To describe a random field, assume the image is modeled as a function f (r, $\omega$), where r is the position vector representing the pixel location in the 2-D space and $\omega$ is a random parameter. For a given value of r, f (r, $\omega$) is a random variable (because $\omega$ is a random variable). Once a specific texture $\omega$ is selected, f (r, $\omega$) is an image, which is a function over the two-dimensional grid indexed by r. Function f (r, $\omega$) is called as a random field. There are currently three major model based methods: Markov random fields by Dubes and Jain [44], fractals by Pentland [45], and the multi-resolution autoregressive features introduced by Mao and Jain [46].

- **Transform Domain Features**

The word transform refers to a mathematical representation of an image. There are several texture classifications using transform domain features in the past, such as discrete Fourier transform, discrete wavelet transforms, and Gabor wavelets.

Transform methods analyze the frequency content of the image to determine texture features. Fourier analysis consists of breaking up a signal into sine waves of various frequencies. On the other hand, wavelet analysis breaks up a signal into shifted and scaled versions of the original wavelet (mother wavelet), which refers to decomposition of a signal into a family of basis functions obtained through translation and dilation of a special function. Moments of wavelet coefficients in various frequency bands have been shown to be effective for representing texture [30].

Gabor filter (or Gabor wavelet) has been shown to be very efficient. Manjunath and Ma [15] have shown that image retrieval using Gabor features outperforms that using other transform features.

Therefore, in this research we use Gabor wavelet transform as our technique to extract the texture features. Gabor wavelet and its implementation for texture feature extraction are detailed in Chapter 4.

### 3.1.3 Shape

One of the common used features in CBIR systems is the shape. Shape of an object is the characteristic surface configuration as represented by the outline or contour. Shape recognition is one of the modes through which human perception of the environment is executed. It is important in CBIR because it corresponds to region of interests in images.

Shape feature representations are categorized according to the techniques used. They are boundary-based and region-based [30].

In region based techniques, all the pixels within a shape are taken into account to obtain the shape representation. Common region based methods use moment descriptors to describe shape [47]. Region moment representations interpret a normalized gray level image function as a probability density of a 2-D random variable. The first seven invariant moments, derived from the second and third order normalized central moments, are given by Hu [48]. Because moments combine information across an entire object rather than providing information just at a single boundary point, they capture some of the global properties missing from many pure contour-based representations.

Comparing with region based shape representation, contour based shape representation is more popular. Contour based shape representation only exploits shape boundary information. Simple contour-based shape descriptors include area, perimeter, compactness, eccentricity, elongation, and orientation. Complex boundary-based descriptors include Fourier descriptors, grid descriptors, and chain codes [40].

In our proposed system, we do not consider shape features during similarity distance computation. Including shape feature in the proposed system is one of our future works.

## 3.2 Similarity Measure

The similarity between two images (represented by their feature values) is defined by a similarity measure. Selection of similarity metrics has a direct impact on the performance of content-based image retrieval. The kind of feature vectors selected determines the kind of measurement that will be used to compare their similarity [4]. If the features extracted from the images are presented as multi-dimensional points, the distances between corresponding multi-dimensional points can be calculated. Euclidean distance is the most common metric used to measure the distance between two points in multi-dimensional space [49].

For other kinds of features such as color histogram, Euclidean distance may not be an ideal similarity metric or may not be compatible with the human perceived similarity. Histogram intersection was proposed by Swain and Bllard [50] to find known objects within images using color histograms. A number of other metrics, such as Mahalanobis distance, Minkowski distance, Earth Mover’s distance, and Proportional Transportation distance, have been proposed for specific purposes.

## 3.3 Indexing Structures

When manipulating massive databases, a good indexing is a necessity. Processing every single item in a database when performing queries is extremely inefficient and slow. When working with text-based documents, creating good indexes is not very difficult. Simply maintaining a list of all words in the database, and information on which documents contain which words, is quite good. When searching for a phrase, the system first checks the index for which documents contain all the necessary search words. Next, in-depth processing only needs to be done with these documents.

When searching for images, however, this approach is much more difficult. Raw image data is non-indexable as such, so the feature vectors must be used as the basis of the index. Popular multi-dimensional indexing methods include the R-tree and the R*-tree algorithms [4].

The Self Organizing Map (SOM) is also one of the indexing structures [51]. The SOM is trained to match the shape of the data in the feature space. After the training, the closest node in the SOM is calculated for every image in the database. This information about the closest nodes is stored. When a query is done, the first thing to be done is to calculate the closest SOM node, also known as the best matching unit (BMU), to the query image’s feature vector. When this is done, we know which images in the database are the closest to the query image: the ones that map to the same node as the query image. This cuts down processing time quite a lot. SOM, the indexing technique used in this thesis, will be discussed in Chapter 6.

## 3.4 Region-Based Image Retrieval

In traditional content based image retrieval, we take a signature for the whole image, and this is known as global features. In most cases, an image does not only contain one object. If we take signatures for each of these objects, we can describe the image more accurately. A simple approach is to manually divide the image into predefined regions, e.g., one region in the image center and four background regions as in Figure 3.5. The drawback of this approach is that the partitioning of the image and the assignment of the regions is static. An image that is not suited for the partitioning cannot be found easily. Newer approaches, therefore, partition the image dynamically. They use a segmentation algorithm to partition the image into homogenous regions [12, 52, 53].

**Figure 3.5:** Image with Static Regions.

### 3.4.1 Image Segmentation

In computer vision, segmentation refers to the process of partitioning a digital image into multiple segments (sets of pixels, also known as superpixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze [54]. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain visual characteristics.

The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image. Groups of pixels in each region are similar with respect to some characteristic or computed property, such as color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic(s) [54].

Different methods for image segmentation have been applied to region-based tasks for different goals, e.g., image retrieval, image annotation, and object recognition. However, the segmentation results greatly affect the performances of region-based tasks.

### 3.4.2 Region Based Features

For each of the regions in an image one or more features are extracted that are taken as a signature for that region. The feature descriptors we have discussed in section 3.1 can be extracted from each object instead of global image. Color and texture features are extracted on each pixel that belongs to the region through the segmentation process, and each region is described by the average value of these pixel features.

The method for exacting color features per pixel is straightforward since the three color components defining the pixel's color intensity are usually used as its color features. For groups of pixels representing an image region, the color features are represented by the average and standard deviation of the components' color values.

For typical pixel-level texture feature extraction, the texture values for each pixel are computed with a sliding window positioned such that the pixel is the center of the window and then applies some transform, such as a wavelet transform, to get the texture feature of that pixel. Other methods divide the whole image into many small non-overlapping pixel blocks, and then apply some transform to each block. These methods extract texture features for a block of pixels. The texture content of an image region is represented by the average texture features of the pixels or blocks belonging to it.

Our proposed approach suggests that these average values are not efficient to represent each region's content; the solution to this problem is to extract the feature descriptor from each image region after being extracted from the segmented image. The region based features that have been implemented into our proposed system are presented in Chapter 6.

### 3.4.3 Region Based Similarity Measure

For region-based extraction, different images may have different numbers of segmented regions. Hence, the total numbers of features extracted from different images are different in spite of the same number of features extracted from each region. Therefore, Euclidean distance cannot be applied directly for comparing region-based images. In region based image retrieval, similarity is calculated as a weighted sum of the similarity between the assigned regions. In order to find a match, the distances between regions must be computed. This is done using one of the metrics mentioned in section 3.3 and results in an m×n distance matrix, m being the number of regions of the query image and n being the number of regions of the database image.

Comparing images with region based features have different methods in the literature. One method was developed by Smith and Li [55], where each image is represented by a composite region template matrix and the distance of two images is measured by the closeness of the two matrices. However, their method is not robust to image shifting, scaling, and rotation [11].

Integrated region matching (IRM) [11] is another method to compare images with region-based features. It computes weighted region-wise distances for the distance of two images. For example, if there are two images $X = \{x_1, . . . , x_n\}$ and $Y = \{y_1, . . . , y_m\}$, where $x_i$ and $y_i$ are region feature vectors for image X and Y, respectively. IRM first computes the region-wise Euclidean distances between regions of image X and Y and obtains an $n \times m$ distance matrix D, where the $ij_{th}$ element is $D_{ij} = \|x_i - y_j\|$. Then IRM computes an $n \times m$ weight matrix W that considers the region importance between two images. Finally, IRM calculates the distance Dis between the two images by just the weighted sum of D

$$Dis(X, Y) = \sum_{ij} w_{ij} D_{ij} \tag{3.1}$$

Where $w_{ij}$ is the $i_{th}$ row and $j_{th}$ column element of the matrix W and satisfies $w_{ij} \in [0, 1]$ and $\sum_{ij} w_{ij}=1$. From Equation 3.1, it can be seen that the distance measure uses soft assignment since all pairs of the distances between regions of two images are considered and weighted.

However, the computation of the weight matrix W in IRM is quite complex and need considerable computation to process it. If only the nearest region of image Y from a region of image X is considered, which is called hard assignment [16], the computation of the matrix W can be avoided.

In our proposed system, we suggest using hard assignment in calculating similarity distance between images because of its simplicity and low computation cost. A comparative study of the effectiveness of both soft and hard assignment in image retrieval will be one of our future works. Details of image similarity distance computation are introduced in Chapter 6.

# Chapter 4

# Gabor Filters for Texture Feature Extraction

Gabor filters transform is a good multiresolution approach that represents the texture of an image in an effective way using multiple orientations and scales. This approach has a spatial property that is similar to mammalian perceptual vision, thereby providing researchers a good opportunity to use it in image processing. Gabor filters are found to perform better than wavelet transform and other multiresolution approaches in representing textures and retrieving images due to its multiple orientation approach [15].

We use the Gabor filter approach to extract global texture features from the whole image, and to extract texture features from image regions. In the following sections, we present Gabor filters in detail, and their use in representing image texture features.

## 4.1 Introduction

Before 1946, the Fourier system was the state-of-the art in signal analysis. The basis of the Fourier system is representation of arbitrary signal with trigonometric functions called Fourier series. The purpose of Fourier transform is to convert a time-domain signal into the frequency-domain, and to measure the frequency components of the signal.

The frequency components at specific locations of an image are used to represent the texture features of that image. Texture features computed from high frequency components are the main distinguishing factors between images that are used in CBIR.

Therefore, frequency information at specific locations is required to distinguish images in CBIR. However, the disadvantage of Fourier transform is that it captures only the global spectral features but does not provide any information about the exact location of these features [56].

Very often, Fourier transform in 2-D space fails to provide texture pattern discrimination information properly. Two completely different images may have similar patterns in their Fourier. Two original images may look different; however, their Fourier spectra have similar patterns. Based on this spectral pattern, these images may be considered as similar in a CBIR process but they can easily be differentiated by human perception [56].

As far back as 1946, Dennis Gabor, the 1971 Nobel Prize winner in Holography was, like every other scientist, interested in the problem of obtaining simultaneous localization in both time/space and frequency domains. He examined the two extreme cases of localization, the sine wave, which is not localized in time space domain but extremely localized in frequency domain, and the delta function, which is perfectly localized in time/space domain but with no localization in the frequency domain.

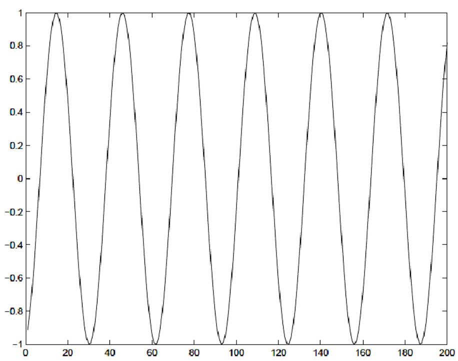
He proposed to represent signals in time and frequency domain by the use of elementary functions constructed from a single block, or "mother" signal, by translation and modulation. By his proposal, any signal of finite duration T and bandwidth F can be divided into a finite number of elementary information cells, called logons. Each logon is of duration $\Delta t$ and bandwidth $\Delta f$, and is localized at a different time and frequency in each of the domains.

If we let $\Delta t = T/M$ and $\Delta f = F/N$, then there are MN elementary information cells, and any signal is considered to represent MN logons of information, namely, the MN coefficients associated with the cells. This is the most information that can be represented by the signal, and these MN complex coefficients are sufficient to regenerate the signal.

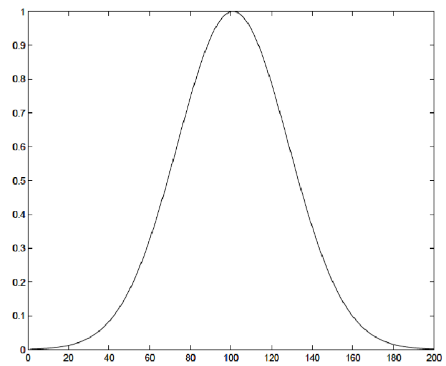
Gabor filter (or Gabor wavelet) is widely adopted to extract texture features from the images for image retrieval [15, 57, 57], and has been shown to be very efficient. Manjunath and Ma [15] have shown that image retrieval using Gabor features outperforms that using Pyramid-structured Wavelet Transform (PWT) features, Tree structured Wavelet Transform (TWT) features and multiresolution simultaneous autoregressive model features. Basically, Gabor filters are a group of wavelets, with each wavelet capturing energy at a specific frequency and a specific direction. Expanding a signal using this basis provides a localized frequency description, therefore capturing local features/energy of the signal. Texture features can then be extracted from this group of energy distributions. The scale (frequency) and orientation tunable property of Gabor filter makes it especially useful for texture analysis. Experimental evidence on human and mammalian vision supports the notion of spatial-frequency (multi-scale) analysis that maximizes the simultaneous localization of energy in both spatial and frequency domains [59].

## 4.2 The Complex Gabor Function

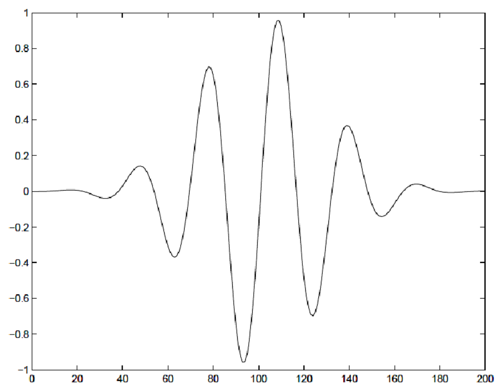
A Gabor function is obtained by modulating a complex sinusoid by a Gaussian envelope. For the case of one dimensional (1-D) signals, a 1-D sinusoid is modulated with a Gaussian. This filter will therefore respond to some frequency, but only in a localized part of the signal. This is illustrated in Figure 4.1 [60]. For two dimensional (2-D) signals such as images, consider the sinusoid shown in Figure 4.2(a). By combining this with a Gaussian shown in Figure 4.2(b), we obtain a Gabor filter as in Figure 4.2(c) [60].



(a) 1-D Sinusoid.

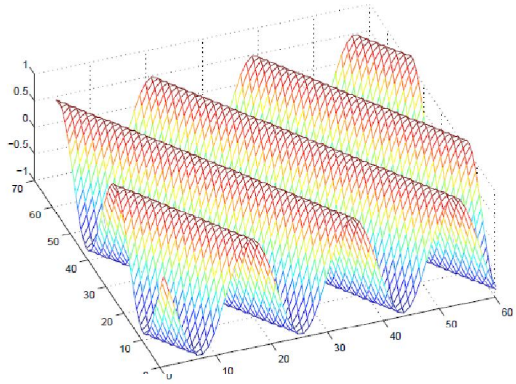


(b) A Gaussian Kernel.

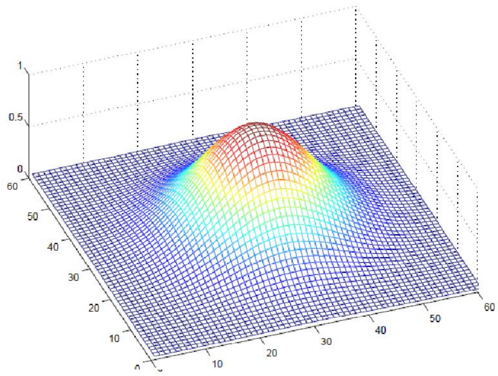


(c) The Corresponding Gabor Filter.

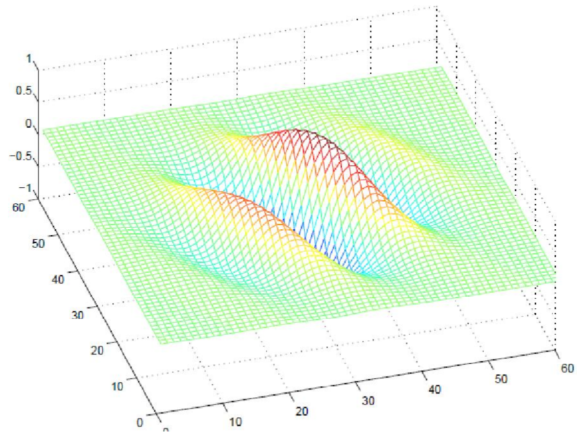
**Figure 4.1:** Gabor Filter Composition for 1-D Signals.



(a) 2-D sinusoid



(b) 2-D Gaussian kernel



(c) The Corresponding Gabor Filter.

**Figure 4.2:** 2-D Gabor Filter Composition.

The 2-D Gabor function can be specified by the frequency of the sinusoid $W$ and the standard deviation $\sigma_x$ and $\sigma_y$, of the Gaussian envelope as:

$$g(x, y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left(-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right) + 2\pi jWx\right) \tag{4.1}$$

Gabor functions form a complete but non-orthogonal basis set. Expanding a signal using this basis provides a localized frequency description.

## 4.3 Generation of Gabor Wavelets

Wavelets are families of basis functions generated by dilations (scaling) and translations of a basic wavelet called the mother wavelet. The basis functions are themselves basic functional building block of any wavelet family.

A class of self-similar functions referred to as Gabor wavelets, is now considered. Let $g(x,y)$ be the mother Gabor wavelet, then this self-similar filter dictionary can be obtained by appropriate dilations and rotations of $g(x,y)$ through the generating function :

$$g_{mn}(x, y) = a^{-m} \cdot g(\tilde{x}, \tilde{y}) \tag{4.2}$$

Where m and n are integers specifying the scale and orientation of the wavelets, respectively, with $m$ = 0, 1, 2, ..., $M$ −1, $n$ = 0, 1, 2, ..., $N$ −1, $M$ and $N$ are the total number of scales and orientations, respectively. And

$$\tilde{x} = a^{-m} \, (xcos\theta + ysin\theta) \tag{4.3}$$

$$\tilde{y} = a^{-m} \, (-xsin\theta + ycos\theta) \tag{4.4}$$

Where $a$>1 and $\theta = \frac{n\pi}{N}$.

The Fourier transform $G(u,v)$ of the mother Gabor shown in Figure 4.3 [15], is defined as:

$$G(u, v) = exp\left\{-\frac{1}{2}\left[\frac{(u-W)^2}{\sigma_u^2} + \left[\frac{v^2}{\sigma_v^2}\right]\right]\right\} \tag{4.5}$$

Where $\sigma_u = {}^{1}/_{2\pi\sigma_x}$ and $\sigma_v = {}^{1}/_{2\pi\sigma_y}$ and $W$ is the modulation frequency.

The non-orthogonality of the Gabor wavelets implies that there is redundant information in the filtered images, and the following strategy is used to reduce this redundancy [59]. Let $U_l$ and $U_h$ denote the lower and upper center frequencies of interest, then the Gabor filter design strategy is to ensure that the half-peak magnitude support of the filter responses in the frequency spectrum touch each other as shown in Figure 4.3.

To obtain this result, the following formulas are used for computing the filter parameters in Equation 4.1:

$$a = \left( {}^{U_h}/_{U_l} \right)^{\frac{1}{M-1}} \tag{4.6}$$

$$W_{m,n} = a^m U_l \tag{4.7}$$

$$\sigma_{x,m,n} = \frac{(a+1)\sqrt{2ln2}}{2\pi a^m (a-1) U_l} \tag{4.8}$$

$$\sigma_{y,m,n} = \frac{1}{2\pi \, tan\left(\frac{\pi}{2N}\right) \sqrt{\frac{U_h^2}{2ln2} - \left(\frac{1}{2\pi\sigma_{x,m,n}}\right)^2}} \tag{4.9}$$

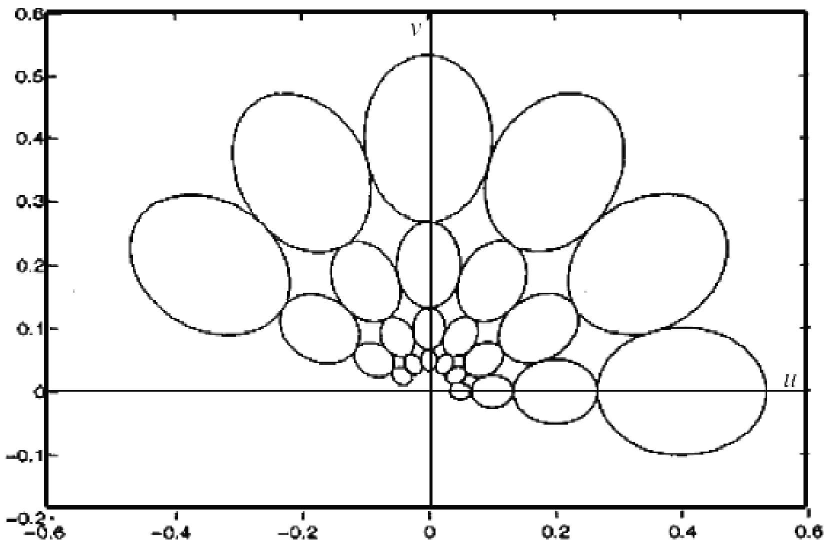



**Figure 4.3:** Frequency Response of Gabor Filters with 4 Scales and 6 Orientations.

In our implementation, we used the following constants as commonly used in the literature: $U_l$ = 0.05, $U_h$ = 0.4.

## 4.4 Texture Representation

The Gabor wavelet image representation is a convolution of that image within the same family of Gabor kernels given in Equation 4.2. Let $I\,(x, y)$ be the gray level distribution of an image, the convolution of the image $I$ together with a Gabor kernel $g_{mn}$ is defined as follows:

$$G_{mn}(x, y) = \sum_s \sum_t I\,(x - s, y - t) g^*_{mn}(s, t) \tag{4.10}$$

Where, s and t are the filter mask size variables, $g^*_{mn}$ is the complex conjugate of the mother Gabor function $g_{mn}$, and $G_{mn}$ is the convolution result corresponding to the Gabor kernel at orientation m and scale $n$.

After applying Gabor filters on the image with different orientation at different scale, we obtain an array of magnitudes:

$$E(m, n) = \sum_x \sum_y |G_{mn}(x, y)| \tag{4.11}$$

These magnitudes represent the energy content at different scale and orientation of the image.

The main purpose of texture-based retrieval is to find images or regions with similar texture. It is assumed that we are interested in images or regions that have homogenous texture, therefore the following mean $\mu_{mn}$ and standard deviation $\sigma_{mn}$ of the magnitude of the transformed coefficients are used to represent the homogenous texture feature of the region:

$$\mu_{mn} = \frac{E(m, n)}{P \times Q} \tag{4.12}$$

$$\sigma_{mn} = \frac{\sqrt{\sum_x \sum_y (|G_{mn}(x, y)| - \mu_{mn})^2}}{P \times Q} \tag{4.13}$$

A feature vector $F$ (texture representation) is created using $\mu_{mn}$ and $\sigma_{mn}$ as the feature components [15]. With $M$ scales and $N$ orientations used in common implementation the feature vector is given by:

$$F_{Texture} = (\mu_{00}, \sigma_{00}, \mu_{01}, \sigma_{01}, \ldots\ldots\ldots\ldots, \mu_{MN}, \sigma_{MN}) \tag{4.14}$$

# Chapter 5

# Global Feature Based CBIR System Design

In this chapter, we introduce the first part of our proposed CBIR system, the global features based image retrieval (GBIR). This system defines the similarity between contents of two images based on global features (i.e., features extracted from the whole image).

Texture is one of the crucial primitives in human vision and texture features have been used to identify contents of images. Moreover, texture can be used to describe contents of images, such as clouds, bricks, hair, etc. Both identifying and describing contents of an image are strengthened when texture is integrated with color, hence the details of the important features of image objects for human vision can be provided.

In this system, Gabor filters, a tool for texture features extraction that has been proved to be very effective in describing visual contents of an image via multiresolution analysis as mentioned in Chapter 4, is used. In addition, color histogram technique is applied for color feature representation combined with histogram intersection technique for color similarity measure. A similarity distance between two images is defined based on color and texture features to decide which images in the image database are similar to the query and should be retrieved to the user. The details of the proposed system are described in the following sections.

## 5.1 Texture Feature Extraction

Texture feature is computed using Gabor wavelets. Gabor function is chosen as a tool for texture feature extraction because of its widely acclaimed efficiency in texture feature extraction. Manjunath and Ma [15] recommended Gabor texture features for retrieval after showing that Gabor features performs better than that using pyramid-structured wavelet transform features, tree-structured wavelet transform features and multiresolution simultaneous autoregressive model. Detailed description of Gabor filters and texture extraction was previously mentioned in Chapter 4.

A total of twenty-four wavelets are generated from the "mother" Gabor function given in Equation 4.2 using four scales of frequency and six orientations.

Redundancy, which is the consequence of the non-orthogonality of Gabor wavelets, is addressed by choosing the parameters of the filter bank to be set of frequencies and orientations that cover the entire spatial frequency space so as to capture texture information as much as possible in accordance with filter design in [15]. The lower and upper frequencies of the filters are set to 0.04 octaves and 0.5 octaves, respectively, the orientations are at intervals of 30 degrees, and the half-peak magnitudes of the filter responses in the frequency spectrum are constrained to touch each other as shown in Figure 5.1 [15]. Note that because of the symmetric property of the Gabor function, wavelets with center frequencies and orientation covering only half of the frequency spectrum $\left(0, \frac{\pi}{6}, \frac{\pi}{3}, \frac{\pi}{2}, \frac{2\pi}{3}, \frac{5\pi}{6}\right)$ are generated.

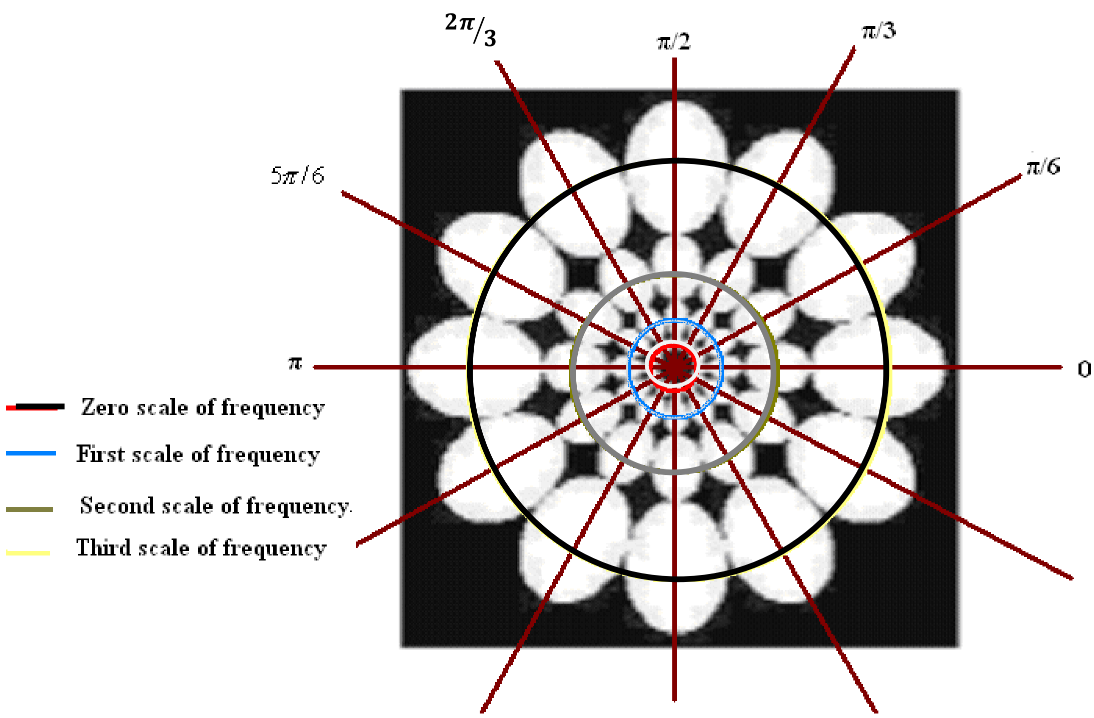
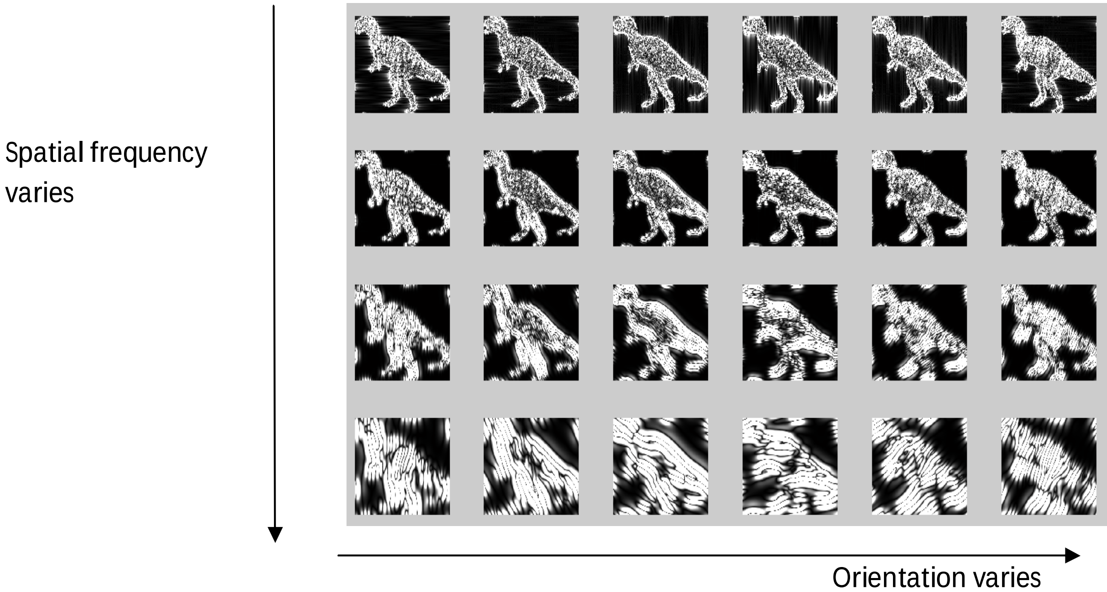



**Figure 5.1:** 2-D Frequency Spectrum View of Gabor Filter Bank.

a) Image in Gray Level.




b) Filtered Image Responses.

**Figure 5.2:** Example of Image Response to Bank of Gabor Filters.

To extract texture feature from an image, Algorithm 5.1 is applied. We first convert the image from the RGB color space into gray level and implement the group of designed Gabor filters. Twenty-four filtered images, $G_{mn}(x, y)$ , are produced by convolution of the gray level image and the Gabor filters as given in Equation 4.10, an example of the filter responses is shown in Figure 5.2. Using Equations 4.11, 4.12, and 4.13 given in the previous chapter, the mean $\mu_{mn}$and variance $\sigma_{mn}$of the energy distribution $E(m, n)$ of the filters responses are computed to finally get the texture feature vector $T$ with 48 attributes:

$$T = [\mu_{00}\ \sigma_{00}\ \ \mu_{01}\ \sigma_{01}\ \ \mu_{02}\ \ \sigma_{02} \dots \dots \dots . . \mu_{35}\ \sigma_{35}] \tag{5.1}$$

Algorithm 5.1: Texture Feature Extraction.

Purpose: to extract texture features from an image.

Input: RGB colored image.

Output: multi-dimension texture feature vector.

Procedure:

{

*Step1:* Convert the RGB image into gray level.

*Step2:* Construct bank of 24 Gabor filters using the mother Gabor function with 4 scales and 6 orientations.

*Step3:* Apply Gabor filters on the gray level of the image by convolution.

*Step4:* Get the energy distribution of each of the 24 filters responses.

*Step5:* Compute the mean μ and the standard deviation σ of each energy distribution.

*Step6:* Return the texture vector T consisting of 48 attributes calculated from step 5.

}

## 5.2 Texture Similarity Measure

To test the similarity between a query image $Q$ and a database image $I$ based on their texture feature we proposed to use the Euclidian distance for its simplicity.

The attributes of the texture features vector may have different ranges (one of very small value and one of very high value), therefore a normalization method should be applied to make all the texture features have the same effect in measuring image similarity. The Min-Max algorithm [31] is employed as a normalization technique; it performs a linear transformation on the original data. Suppose that $min_A$ and $max_A$ are the minimum and maximum values of the attribute, the Min-Max normalization maps a value, $v$, of $A$ to $\tilde{v}$ in the range [0, 1] by computing:

$$\tilde{v} = \frac{v - min_A}{max_A - min_A} \qquad (5.2)$$

Let $T_Q$ and $T_I$ denote the texture vector of a query image and an image in the database respectively, we define the distance between the two texture vectors denoted as $d_T(Q, I)$ by the Euclidian distance as:

$$d_T(Q, I) = \sqrt{\sum_{i=1}^{48} \left(T_{Q_i} - T_{I_i}\right)^2} \tag{5.3}$$

## 5.3 Color Feature Extraction

In this system we used global color histograms in extracting the color features of images. The main issue regarding the use of color histograms for image retrieval involves the choice of color space, color space quantization into a number of color bins, where each bin represents a number of neighboring colors, and a similarity metric [61].

### 5.3.1 HSV Color Space

In the literature, there is no optimum color space known for image retrieval, however certain color spaces such as HSV, Lab, and Luv have been found to be well suited for the content based query by color. We adopt to use the HSV (Hue, Saturation, and Value) color space for its simple transform from the RGB (Red, Green, Blue) color space, in which all the existing image formats are represented. The HSV color space is a popular choice for manipulating color, it is developed to provide an intuitive representation of color and to approximate the way in which humans perceive and manipulate color. RGB to HSV is a nonlinear, but reversible transformation. The hue (H) represents the dominant spectral component (color in its pure form), as in red, blue, or yellow. Adding white to the pure color changes the color: the less white, the more saturated the color is. This corresponds to the saturation (S). The value (V) corresponds to the brightness of color. The hue (color) is invariant to the illumination and camera direction, and thus suitable for object recognition. Figure 5.3 [62], shows the cylindrical representation of the HSV color space. The angle around the central vertical axis corresponds to “hue” denoted by the angle from 0 to 360 degrees, the distance from the axis corresponds to “saturation” denoted by the radius, and the distance along the axis corresponds to “lightness”, “value” or “brightness” denoted by the height.

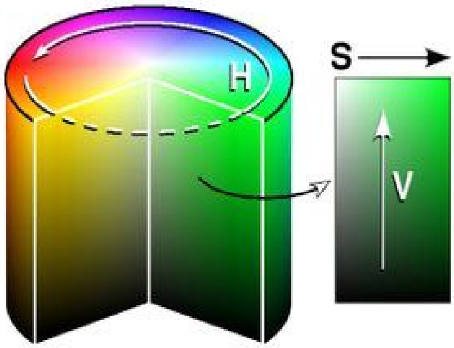



**Figure 5.3:** The HSV Color Space.

The HSV values of a pixel can be transformed from its RGB representation according to the following formulas:

$$H = \arctan\frac{\sqrt{3}(G-B)}{(R-G)+(R-B)} \tag{5.4}$$

$$S = 1 - \frac{min\{R, G, B\}}{V} \tag{5.5}$$

$$V = \left(\frac{R+G+B}{3}\right) \tag{5.6}$$

## 5.3.2 Quantization and Histogram Generation

The HSV color space is quantized into 108 bins by using uniform quantization (12 for H, 3 for S, and 3 for V), the choice of these parameters was motivated by [63]. Since Hue (H) has more importance in human visual system than saturation (S) and value (V), it is reasonable to assign bins in the histogram to the Hue more than the other components.

It is straightforward to generate the histograms of color images using the selected quantized color space. All we have to do is to count the number of pixels that correspond to a specific color in uniformly quantized color space regardless of which color space is used. One histogram bin corresponds to one color in the quantized color space.

## 5.3.3 Histogram Distance Measure

The similarity metric we used in deriving the distance between two color histograms is the Histogram Intersection Technique (HIT). The color histogram intersection was proposed for color image retrieval in [50], in this technique the similarity between two histograms is a floating point number between 0 and 1. Two histograms are equivalent when the similarity value is 1 and the similarity between two histograms decreases when the similarity value approaches to 0. Both of the histograms must be of the same size to have a valid similarity value.

Let $H_Q$ and $H_I$ denote the histograms of the query image and an image in the database, respectively, and $S(H_Q, H_I)$ denotes the similarity value between $H_Q$ and $H_I$. Then, $S(H_Q, H_I)$ can be expressed by:

$$S(H_Q, H_I) = \frac{\sum_{x\in X, y\in Y, z\in Z} min\left(H_Q(x,y,z), H_I(x,y,z)\right)}{\sum_{x\in X, y\in Y, z\in Z} H_Q(x,y,z)} \tag{5.7}$$

Where $X$, $Y$, and $Z$, are the arguments of the discretized color channels.

The similarity measure given in Equation 5.7 is not a distance in the strict sense, since it does not satisfy the condition of associatively. Smith and Chang [64] extended (5.7) by modifying the denominator of the original definition slightly:

$$S(H_Q, H_I) = \frac{\sum_{x\in X, y\in Y, z\in Z} min\left(H_Q(x,y,z), H_I(x,y,z)\right)}{min\left[\sum_{x\in X, y\in Y, z\in Z} H_Q(x,y,z), \sum_{x\in X, y\in Y, z\in Z} H_I(x,y,z)\right]} \tag{5.8}$$

Finally, the distance between the query image and the database image according to the extracted color feature denoted as $d_C(Q, I)$ is defined as follows:

$$d_C(Q, I) = 1 - S(H_Q, H_I) \tag{5.9}$$

## 5.4 Image Matching

Once we have extracted texture and color feature vectors from the query image, as well as the database images, we can use these features to measure the similarity between images in order to retrieve the most similar DB images to the query. The similarity between a query image $Q$ and a DB image $I$ is defined by a distance between them denoted as $d(Q, I)$, which is assessed according to the extracted color and texture features. Two images are equivalent when the similarity distance value between them approaches zero, and the similarity between them decreases as the distance value between them increases.

Using the texture distance $d_T$ given in (5.3) and the color histogram distance $d_C$ given in (5.9) we define the distance $d(Q, I)$ as:

$$d(Q, I) = w_t d_T + w_c d_c \tag{5.10}$$

Where $w_t$ and $w_c$ are weights for the texture and color features. We used $w_t$= 0.35 for texture and $w_c$= 0.65 for color as have been used effectively in the literature [16].

## 5.5 Image Retrieval Methodology

The image retrieval methodology proposed in this system goes throughout the steps shown in the block diagram Figure 5.4. The user enters a query image for which the system extracts both color and texture features as explained in the previous sections, the feature vectors of database images are previously extracted and stored. Using the similarity metrics defined for color and texture, the similarity distances between the query image and every image in the database are calculated according to Equation 5.10 and then are sorted in ascending order. The first *N* similar target images (with smallest distance value to the query) are retrieved and shown to the user, where *N* is the number of the retrieved images required by the user.

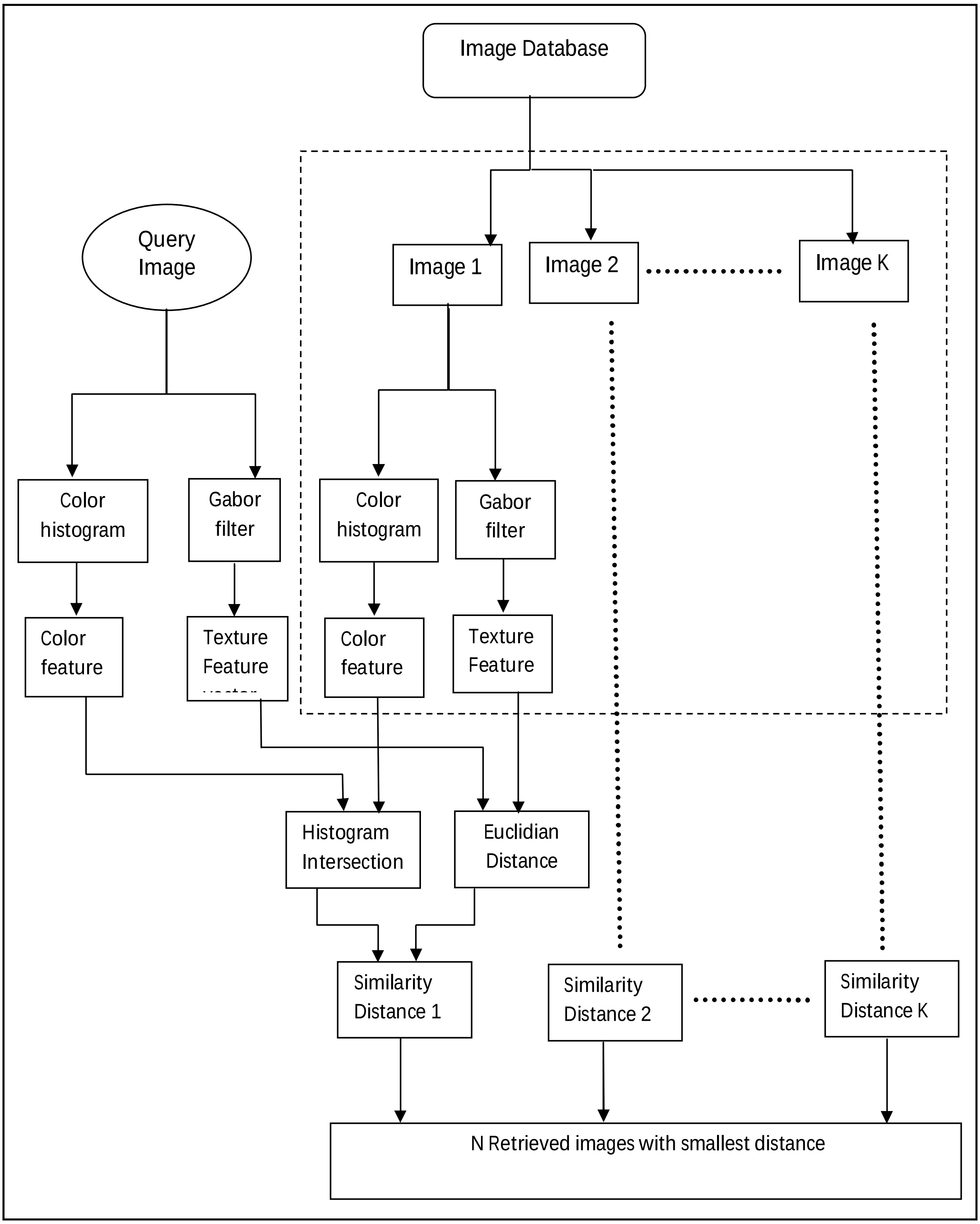



**Figure 5.4:** Block Diagram of Proposed GBIR System.

# Chapter 6

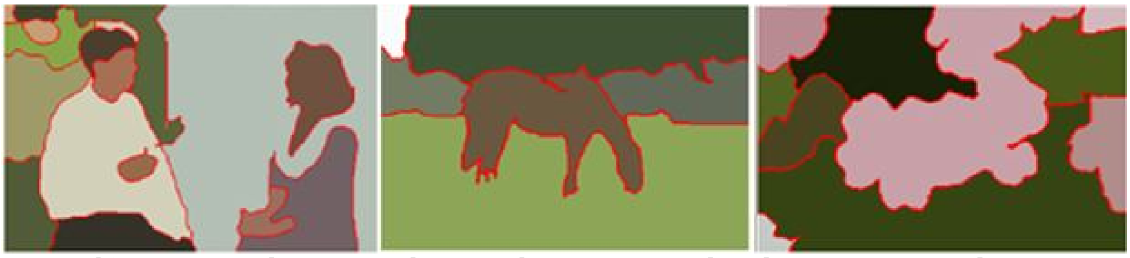
# Region Based CBIR System Design

The second part of our proposed system, which is Efficient Region Based Image Retrieval (ERBIR), is introduced in this chapter. The ERBIR system attempts to overcome the deficiencies of the GBIR systems by representing images at the object-level. The system applies image segmentation to decompose an image into regions, which correspond to objects if the decomposition is ideal. Since the retrieval system has identified what objects are in the image, it is easier for the system to recognize similar objects at different locations and with different orientations and sizes, thus results in more accurately retrieved results. However, this system has the drawback of high computation complexity since it compares each region in a query image with each region in the database images in order to determine the overall similarities between the whole images, making the system inefficient. Our proposed solution to this problem is to index the image regions into subgroups via unsupervised clustering algorithm that maps the database images regions of similar visual features into separate clusters. The query image regions will also be mapped into these clusters and thus we need not to search all images in the database, but only those images that have regions in the same cluster with any of the query image regions, this will reduce the searching time to high extent.

## 6.1 TBES Image Segmentation Algorithm

Image segmentation, which is a technique to decompose an image into distinct regions that correspond to objects or subjects on the image, is the first step in a ERBIR system. Humans can easily divide an image into objects, however there are currently no typical algorithms that can perform this job correctly in all cases, and as good as humans could. There are many proposed algorithms for automatic image segmentation; in our system we used the Texture and Boundary Encoding-based Segmentation (TBES) proposed by Allen, et al [52]. The TBES is an unsupervised image segmentation algorithm based on observations that a homogeneously textured region of a natural image can be well modeled by a Gaussian distribution and the region boundary can be effectively coded by an adaptive chain code. The optimal segmentation of an image is the one that gives the shortest coding length for encoding all textures and boundaries in the image, and is obtained via an agglomerative clustering process applied to a hierarchy of decreasing window sizes. The optimal segmentation also provides an accurate estimate of the overall coding length and hence the true entropy of the image [52]. Examples of segmented images obtained by the TBES are shown in Figure 6.1.

a) The Original Images.



b) Segmented Images Where Each Region Is Colored By Its Mean Color.

**Figure 6.1:** Qualitative Results of TBES Segmentation Algorithm.

## 6.2 Texture Feature Extraction

In our proposed ERBIR system, we use the same features we have used in the GBIR system proposed in Chapter 5, which are texture and color, as visual features to represent each region extracted from the segmented image.

In the existing region based CBIR systems, visual features are extracted on each pixel that belongs to the region, and each region is described by the average value of these pixel features. However, we have found out that these average feature values are not efficient in describing the region’s content. Also, these features are extracted from each pixel or texton for the purpose of segmentation and differ with different segmentation algorithms. We propose to extract the color and texture features from each image region as a whole after being extracted from the segmented image, this will help in representing the region efficiently and will make us free to use any image segmentation method without being obliged to use the same features used in that segmentation method.

There are two problems to be considered when extracting texture features for RBIR systems:

1. In many systems, texture features are obtained during segmentation from pixels or small blocks. Such features may not well represent the property of an entire region. In some other systems, segmentation does not produce texture features. In order to make our system robust to any segmentation algorithm, it is necessary to study texture feature extraction from the whole region after being extracted from the segmented image.
2. Transforms such as Gabor filtering require the input image to be rectangular, which is not always true for regions resulting from image segmentation. An instinctive way is to obtain an inner rectangle (IR) from a region on which filtering can be performed. This works when the size of the filtering mask is much smaller than the size of the IR. But many regions in RBIR systems are small, and the coefficients obtained can not well describe the region. To solve this problem, we present an extended rectangle (ER) texture feature extraction algorithm. By initial padding, our algorithm extends an arbitrary-shaped region into a larger rectangle onto which Gabor filtering is applied. Then a set of coefficients best describing the region is obtained, from which texture features can be extracted.

### 6.2.1 Initial Padding

Extracting texture feature from image region requires that the shape of the region be rectangular, which is not always true for regions resulting from image segmentation.

Given an arbitrary-shaped region $f(x, y)$, $(x, y) \in A$ containing $M$ pixels as shown in Figure 6.2(a), with A being the region interior and the boundary. To apply Gabor filtering, we first extend it into a rectangular block $E$ of size $N$ ($>M$) enclosing the region interior by padding some values outside the boundary to zero as shown in Figure 6.2(b).

Let $x_1$, $x_2$, $y_1$, and $y_2$ be the minimum and maximum $x$ and $y$ values in region $A$ respectively, then we define the size of region $E$ to be $N = (x_2 - x_1 + 1, y_2 - y_1 + 1)$. Each pixel in E is assigned the value:

$$E(x, y) = \begin{cases} f(x, y) & if\ x, y \in A \\ 0 & otherwise \end{cases} \tag{6.1}$$

In the literature, there are many other padding techniques such as mirror padding and object-based padding; we have chosen zero padding for its simplicity and low cost computation.

a) Original Region. b) Extended Region.

**Figure 6.2:** Example of Zero Padding.

## 6.2.2 Gabor Filter Implementation

As in the GBIR system, we implement a bank of Gabor filters with 4 scales and 6 orientations to extract texture features from the ER of an image region. Then, we select the M largest coefficients from the N coefficients in each of the 24 filtered output regions, since the high frequency components represent the object region and its boundary. By assuming spatial homogeneity of texture in each image region, the texture features are computed as the mean of the magnitude of the selected coefficients according to the formula:

$$\mu_{mn} = \frac{1}{M} \sum_{x} \sum_{y} G_{mn}(x, y) \tag{6.2}$$

Where $G_{mn}(x, y)$ is the group of selected coefficients from the region resulting from applying the filter with scale $m$ and orientation $n$ on the extended region ER using Equation 4.10.

Texture feature vector is constructed using the computed means $\mu_{mn}$according to the formula in equation 6.3.

$$Texture\_Feature = [\mu_{00}\ \mu_{01}\ \mu_{02} \ldots \ldots \ldots .. \mu_{35}] \qquad (6.3)$$

Figure 6.3 shows the diagram of the proposed texture extraction algorithm.

Image region texture features may have different ranges (one with very small value and one region with very large value), therefore a normalization method should be applied on each of them. We use the Min-Max normalization; the same formula we use in the GBIR system proposed in Chapter 5 and given in Equation 5.2.

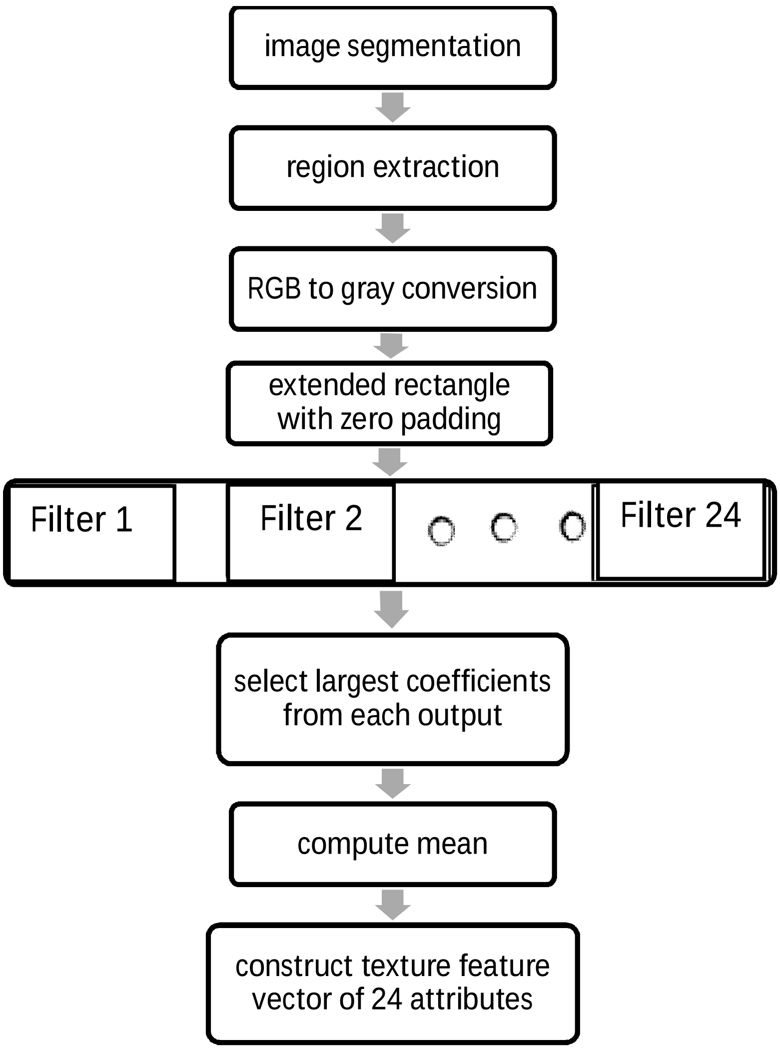



**Figure 6.3:** Block Diagram of Proposed Texture Extraction Algorithm.

## 6.3 Color Feature Extraction

We use the HSV color space for color feature extraction, since it is natural, perceptually uniform, and easy to be converted to RGB space and vice versa. As the image regions extracted from the image after segmentation are approximately color homogeneous, it is possible to use the average HSV value in each channel of all pixels in the region as its perceptual color. We also use the standard deviation for each color channel resulting in six color features. The Min-Max normalization formula given in Equation 5.2 is used to have the values of each color feature in the range [0, 1].

## 6.4 Region Percentage Area

The last feature we use is the region percentage area of an image. We propose that the area occupied by a region in an image gives information on the importance of this region and this importance should be great for regions with larger areas proportionally to the image area.

Let the number of pixels in a given region is $N$, and the size of the original image is $P \times Q$ then the region percentage is given by:

$$A = \frac{N}{P \times Q} \tag{6.4}$$

## 6.5 Region Matching

An image region is described by a feature vector of 31 normalized attributes named as $f_1$ to $f_{31}$. The first 24 features are for texture, $f_{25}$ to $f_{30}$ are for color, and $f_{31}$ for region percentage. To measure the similarity between two images we have to compare each region in one image to all the regions in the other, and this comparison is based on the extracted region features. We use the Euclidian distance between the feature vectors to match two regions such that as the distance increases the matching between the two region decreases and vice versa. The distance between two image regions $R_i$ and $R_j$ denoted by $d_{ij}$ is defined as:

$$d_{ij} = \sqrt{w_T \sum_{k=1}^{24} \left(f_{ki} - f_{kj}\right)^2 + w_C \sum_{k=25}^{31} \left(f_{ki} - f_{kj}\right)^2} \tag{6.5}$$

Where $f_{ki}$ and $f_{kj}$ are the $k_{th}$ feature of the regions $R_i$ and $R_j$, respectively, and $w_T$ and $w_C$ are weights for texture and color features. Experimentally in our simulation we examined some values for $w_T$ and $w_C$ and we chosen to set $w_T = 1$, and $w_C = 2$ , since we have 24 texture features, whereas the color and area features are only seven, and thus we have to increase the effect of the color features on the distance measure between image regions, the effect of changing the values of $w_T$ and $w_C$ on the retrieval performance will be one of our future work.

The distance $d_{ij}$ between any two image regions will be used to measure the overall similarity between a query image and a database image, as will be shown in the next section.

## 6.6 Image Similarity Search

Given the definition of the distance between two regions, we are ready to compute the global similarity between two images.

Suppose that we have query image $I_Q$ with $M$ regions and database image $I_D$ with $N$ regions, we compute the global similarity between the two images $I_Q$ and $I_D$ using the following procedure:

***Step 1:*** Using Equation 6.5, compute the distance between one region in $I_Q$ and all regions in $I_D$. For each region $R_i$ in the query image $I_Q$, the distance between this region and the database image $I_D$ is defined as:

$$R_{i,I_D} = Min\left(d_{ij}\right) \quad \forall\, j \in I_D \tag{6.6}$$

Where $d_{ij}$ is the distance between $R_i$ and any region $R_j$ in the database image. This definition takes the minimum distance between the query region $R_i$ and all the regions in the database image $I_D$, which maximizes the similarity between the region and the database image.

***Step 2:*** We compute the similarity between the query image $I_Q$ and the database image $I_D$ as follows:

$$D1\left(I_Q, I_D\right) = \sum_{i=1}^{M} \alpha_i R_{i,I_D} \tag{6.7}$$

Where $\alpha_i$ is the weight for region $R_i$ in image $I_Q$, we use the percentage of the region in an image $f_{31}$ as its weight (i.e. $\alpha_i = f_{i31}$), since we think a region with a larger area plays a more significant role in contributing to the overall similarity value between two images than a region with a smaller area.

***Step 3:*** The similarity distance between the query image and the database image given in Equation 6.7 is not symmetric, to make it symmetric we compute the distance between the database image and the query image by repeating steps 1 and 2 for the regions in the database image, we define the distance between region $R_j$ in the database image and the query image as:

$$R_{j,I_Q} = Min\left(d_{ji}\right) \quad \forall\, i \in I_Q \tag{6.8}$$

This definition takes the minimum distance between the database image region $R_j$ and all the regions in the query image $I_Q$, which maximizes the similarity between the region and the query image.

***Step 4:*** The distance between $I_D$ and $I_Q$ can be defined as:

$$D2\left(I_D, I_Q\right) = \sum_{j=1}^{N} \alpha_j R_{j,I_Q} \tag{6.9}$$

Where $\alpha_j$ is the weight for region $R_j$ in image $I_D$, and also we use it as the $f_{j31}$ just as for the query image regions.

In Figure 6.4, a line from a query region to a DB region corresponds to the minimum distance from the region in image $I_Q$ (for example with 7 regions) to the region in database image $I_D$ (with 9 regions). Whereas, a line from a DB region to a query region corresponds to the minimum distance from the region in image $I_D$ to the region in $I_Q$. These distances are then added and divided by two to get the symmetric distance between image $I_Q$ and $I_Q$ as in step 5.

***Step 5:*** The overall distance between the two images $I_Q$ and $I_D$ is defined as:

$$Dist\left(I_D, I_Q\right) = \frac{D1\left(I_Q, I_D\right) + D2\left(I_D, I_Q\right)}{2} \tag{6.10}$$

This definition of the distance between two images captures the overall similarity measure based on regional and global matching. As compared with many existing similarity measures in the literature, this definition strives to incorporate as much semantic information as possible, and at the same time also achieves a computational efficiency. Given this definition, for each query image $I_Q$, it is straightforward to compute $Dist(I_D, I_Q)$ for every image $I_D$ in the database in the retrieval process.

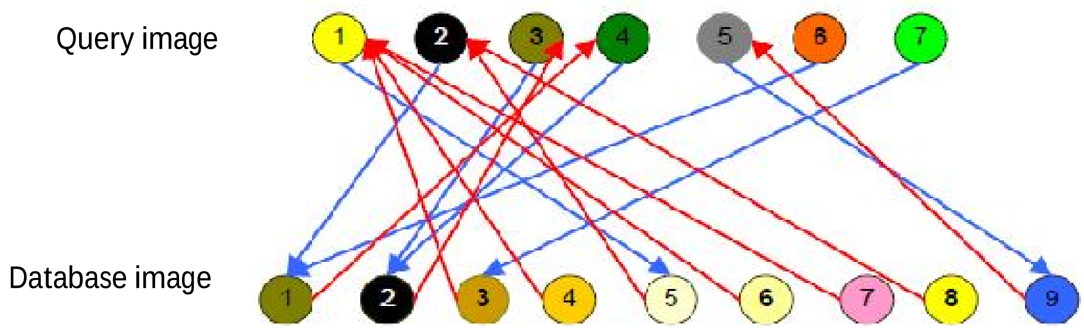



**Figure 6.4:** Minimum Distance of Regions from Image $I_Q$ to Image $I_D$ and Vice Versa.

## 6.7 Database Regions Clustering

The time of image retrieval in almost all RBIR systems depends in a large degree on the number of images in the database and the number of regions in both the query image and database image. Many existing systems attempt to compare the query image with every target image in the database to find the top matching images, resulting in an essentially linear search, which is highly computationally inefficient when the database is large. We believe that it is not necessary to conduct a whole database comparison. In fact, it is possible to make use of a priori information regarding the organization of the images in the database in the feature space before a query is processed, such that when a query is received, only a part of the database needs to be searched, while a large portion of the database may be eliminated in the search. This certainly saves significant query processing time without compromising the retrieval precision. To achieve this goal, in our proposed ERBIR we add a pre-retrieval phase to the feature space after database features are constructed and stored by applying a clustering algorithm to the distance $d_{ij}$ in the region feature vector space to cluster all the regions in the database into classes. The theory behind this is that regions with similar (color, and texture) features should be grouped together in the same class. This clustering is performed offline, and each region's features data, along with its associated class number, is recorded in the database files.

For clustering purpose we use the Kohonen Self Organizing Map (SOM) [65] algorithm.

One advantage using SOM with respect to other clustering algorithms is the spatial organization of the feature map that is achieved after the learning process. Basically, more similar clusters are closer than more different ones. Consequently, the distance among prototypes in the output layer of the SOM can be considered as a measure of similarity between patterns in the clusters.

The main advantage of using the SOM network is that SOM automatically (self-organizing) clusters the input space and not sensitive to initialization. The SOM network also can be applied to a large scale of data, and most importantly, the learning of SOM can be incremental such that only new images can be used for new training of SOM if new images are not included in the system.

- ## The Self Organizing Map

The Self Organizing Map is an artificial neural network that performs clustering by means of unsupervised competitive learning. In the SOM the neurons are usually arranged in a two dimensional lattice (feature map). Each neuron receives inputs from the input layer and from the other neurons in the map. The input samples are described with real vectors $x(t) \in R^n$, where t is the index of the sample. Each neuron contains a model vector $m_i \in R^n$ that can be regarded as a prototype of the patterns in the cluster. During the learning, the network performs clustering and the model vectors are changed so as to reflect the similarity of neighboring clusters. The goal of the mapping is to represent the points in the source space by corresponding points in a lower dimensional target space. In particular, the training is aimed at preserving as much as possible the distance and proximity relationships among input samples.

The initial values of the model vectors, $m_i(0)$, may be selected at random or can be initialized in some orderly fashion, for instance arranging the vectors along a two-dimensional subspace spanned by the two principal eigenvectors of the input data. The two main SOM learning algorithms are the on-line and the batch ones. The on-line algorithm computes the mapping by processing each training pattern $x(\mathrm{t})$ with the following steps and repeating the overall loop several times.

***Step1.*** The vector $x(\mathrm{t})$ is compared with all the model vectors $m_i(\mathrm{t})$ and the Best Matching Unit (BMU) on the map is identified. The BMU is the node having the lowest distance with respect to the input pattern $x(\mathrm{t})$. The final topological organization of the map is heavily influenced by the distance function considered in this step. In most cases the Euclidean distance is considered, and the BMU $m_b(x)$ is identified by:

$$\|x(\mathrm{t}) - m_b(x)\| = \min_i\{\|x(\mathrm{t}) - m_i(\mathrm{t})\|\} \qquad (6.11)$$

***Step2.***The model vector of the BMU as well as some of its neighboring nodes are changed so as to move towards the current input pattern $x(\mathrm{t})$ according to the following equation :

$$m_i(t + 1) = m_i(t) + h_{b(x),i}(t)(x(t) - m_i(t)) \qquad (6.12)$$

Where $h_{b(x),i}$ is the neighborhood function, implemented with a smoothing kernel that is time-variable and is defined over the lattice points. The neighborhood function is a decreasing function of the distance between the $i_{th}$ and $b(x)_{th}$ models on the map grid. The extension of the kernel is also decreasing monotonically during the iterations. A widely used neighborhood function is based on the Gaussian function:

$$h_{b(x),i}(t) = \alpha(t)\exp\left(-\frac{\|r_i - r_{b(x)}\|^2}{2\sigma^2(t)}\right) \qquad (6.13)$$

Where $0 < \alpha(t) < 1$ is the learning-rate factor that decreases with the iterations, $\mathrm{r_i} \in \Re^2$ and $r_{b(x)} \in \Re^2$ are the locations of the neurons in the lattice, and $\sigma(t)$ defines the width of the neighborhood function and is also decreasing monotonically.

The above steps are repeated until all the patterns in the training set have been processed. To achieve a better convergence towards the desired mapping it is usually required to repeat the previous loop until some convergence criteria are met [65].

A typical SOM is a two-dimensional grid with 10×10 units, each of these units is considered as a cluster center. Figure 6.5 shows an example of SOM weight vectors (nodes) distributed over the input data in 2-D space.

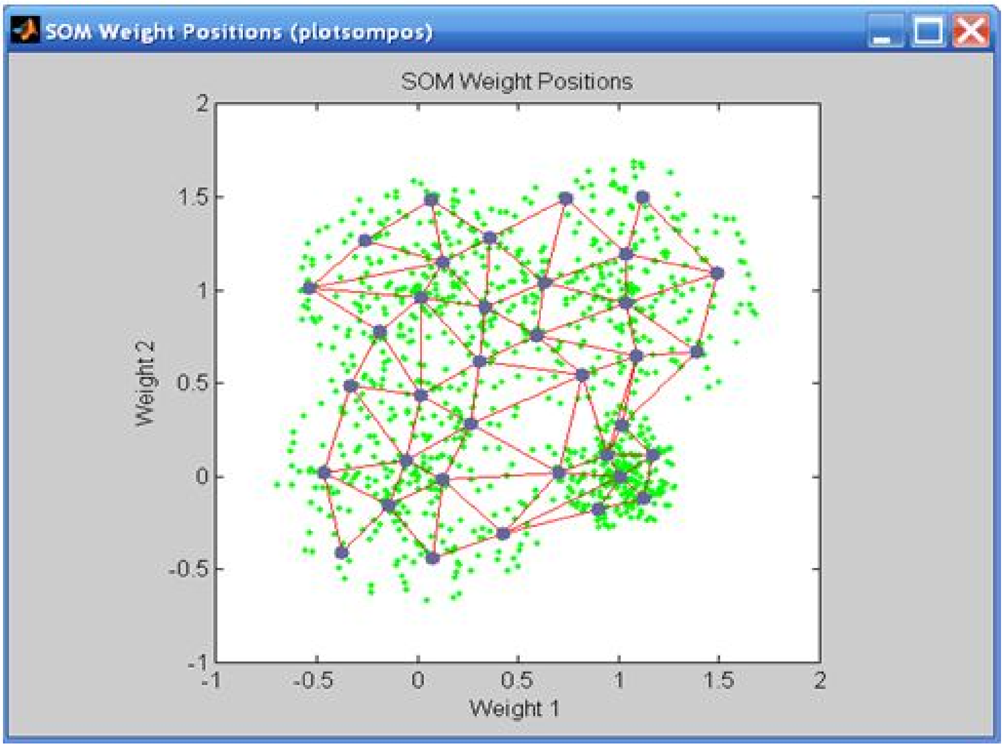



**Figure 6.5:** The SOM Weight Vectors in Two Dimensional Space.

## 6.8 Image Retrieval Methodology

### 6.8.1 Data Insertion

The image retrieval system we propose in this chapter first segments each image in the database into distinct regions considered as objects in that image using the TBES algorithm. Features are extracted from each image region using the techniques described in section 6.2; these features are stored in the database files. We implement clustering with self-organizing map algorithm in the database feature space to group those regions of similar visual features into separate clusters to reduce the searching time in the query process. The SOM is chosen to have two dimensional 10×10 nodes in gridtop topological organization, each of these nodes is considered as a cluster center. Each image region in the database is given a cluster number stored with it at the end of SOM training using the region's features.

### 6.8.2 Query Image Processing

Given a query image, our system processes the query as follows:

***Step 1:*** Perform the query image segmentation to obtain all the regions, say we have $N$ regions ($Q_i$ : $i$ = 1to $N$) in the query image.

***Step 2:*** Calculate the closest SOM node, also known as the best matching unit (BMU), to the query image region's feature vector to determine which class $Q_i$ belongs to. Assume that region $Q_i$ belongs to class $C_j$.

***Step 3:*** Retrieve all the regions in the database that belong to the class $Cj$. These regions constitute a region set $X$. The images containing any regions in the set $X$ is subsequently retrieved. These images comprise an image set $T$ and are the candidate images.

***Step 4:*** Compare the query image with the images in the image set $T$. The distance $Dist(Q, I)$ given in Equation 6.10 is used to measure the similarity between the query image and a candidate image, and the top-least-distance images are returned to the user.

## 6.9 Integrated GBIR and ERBIR System

The capability of existing CBIR systems is limited in large part by fixing a set of features used for retrieval. Apparently, different image features are suitable for the retrieval of images in different semantic types. For example, a global features based method may be good for certain pictures, while a region-based indexing approach is much better for other pictures. Similarly, global texture matching is suitable only for textured pictures, and a combination of both global and local features can also be good for another type of image semantic.

Since the region based approach consumes more time in image retrieval due to the segmentation process besides working on each image region for feature extraction and similarity measure, the global features based method may give us good results in shorter time according to picture semantics. We propose our overall system to have three alternatives to answer an image query:

1. First we return to the user the results using global features based approach, if the results are not satisfying according to him we move to the second choice.

2. We apply region based approach and return the retrieved images; if the results are not satisfying we can use the third choice.

3. Our third choice to improve the results is an integrated Global and Region-Based Image Retrieval GRBIR model, in which we propose to use a combination of the two methods, ERBIR and GBIR, such that the similarity distance between a query image and a database image $d(Q,I)$ is defined as:

$$d(Q,I) = d_1 + d_2 \qquad (6.14)$$

Where $d_1$ is the global features based distance given in Equation 5.10, $d_2$ is the region based distance given in Equation 6.10.

A comparative study of the three alternative systems will be demonstrated in Chapter 7.

# Chapter 7

# Results and System Evaluation

In this chapter we present an evaluation of the proposed CBIR systems: GBIR, ERBIR, and GRBIR, that we introduced in the previous chapters. We also compare their performance with four of already existing CBIR systems.

## 7.1 WANG Database

The database we used in our evaluation is WANG database [67]. The WANG database is a subset of the Corel database of 1000 images, which have been manually selected to be a database of 10 classes of 100 images each.


**Figure 7.1:** Example Images from each of the 10 Classes of WANG Database.

The images are subdivided into 10 classes, such that it is almost sure that a user wants to find the other images from a class if the query is from one of these 10 classes. This is a major advantage of this database because due to the given classification it is possible to evaluate retrieval results. The images are of size 384 × 256 or 256 × 384 pixels, Figure 7.1 shows 10 sample images in each image class.

This database was used extensively to test many CBIR systems [68, 17, 29, 66] because the size of the database and the availability of class information allows for performance evaluation as can be seen in the following sections. This database was created by the group of professor Wang from the Pennsylvania State University and is available for download [67]. This database was also used for classification experiments.

## 7.2 Implementation Environment

The image retrieval system is implemented using MATLAB image processing tools and statistical tools. During the implementation, we use a platform of Intel Core 2 Due Processing power of 2.4 GHz CPU with 4GB RAM. 1000 image database went through image segmentation algorithm to obtain more than 5800 objects. These images are manually divided into 10 classes such as Africans, buses, buildings, and flowers.

## 7.3 Performance Evaluation Metrics of CBIR

The level of retrieval accuracy achieved by a system is important to establish its performance. If the outcome is satisfactory and promising, it can be used as a standard in future research works. In CBIR, precision-recall is the most widely used measurement method to evaluate the retrieval accuracy. We have found some recent literature [16, 29, 60] that uses this pair to measure the retrieval performance.

**Precision**, P, is defined as the ratio of the number of retrieved relevant images to the total number of retrieved images [69].

$$P = \frac{Number\ of\ relevant\ images\ retrieved}{Total\ number\ of\ images\ retreived} \tag{7.1}$$

Let the number of all retrieved images be n, and let r be the number of relevant images according to the query then the precision value is: P = r / n. Precision P measures the accuracy of the retrieval.

**Recall**, R, is defined as the ratio of the number of retrieved relevant images to the total number of relevant images in the whole database [69].

$$R = \frac{Number\ of\ relevant\ images\ retrieved}{Number\ of\ relevant\ images\ in\ the\ database} \qquad (7.2)$$

Let r be the number of relevant images among all retrieved images according to the query, and M be the number of all relevant images to the query in the whole database then the Recall value is: R = r / M. Recall R measures the robustness of the retrieval.

In information retrieval, a perfect precision score of 1.0 means that every result retrieved by a search was relevant (but says nothing about whether all relevant documents were retrieved), whereas a perfect recall score of 1.0 means that all relevant documents were retrieved by the search (but says nothing about how many irrelevant documents were also retrieved).

In a classification task, a precision score of 1.0 for a class *C* means that every item labeled as belonging to class *C* does indeed belong to class *C* (but says nothing about the number of items from class *C* that were not labeled correctly), whereas a recall of 1.0 means that every item from class *C* was labeled as belonging to class *C* (but says nothing about how many other items were incorrectly also labeled as belonging to class *C*).

Often, there is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other. For example, an information retrieval system (such as a search engine) can often increase its recall by retrieving more documents, at the cost of increasing number of irrelevant documents retrieved (decreasing precision).

Usually, precision and recall scores are not discussed in isolation. Instead, either values for one measure are compared for a fixed level at the other measure (e.g. precision at a recall level of 0.75) or both are combined into a single measure, such as precision/recall graph. Precision-recall pair is a good standard of performance evaluation. It provides meaningful result when the database type is known and has been effectively used in some earlier research. For other data sets, especially those that have been created by collecting user generated images, the result may vary due to different human concepts of image classification.

Because the ground truth is known for the whole database, every image in the database can be used as the query. For each query, the precision of the retrieval at each level of the recall is obtained.

## 7.4 ERBIR System Evaluation

In this section, we test the main part of our proposed system; the ERBIR system. We evaluate the system regarding two metrics: the effectiveness in terms of precision and recall, and the efficiency in terms of the time the system takes to answer a query.

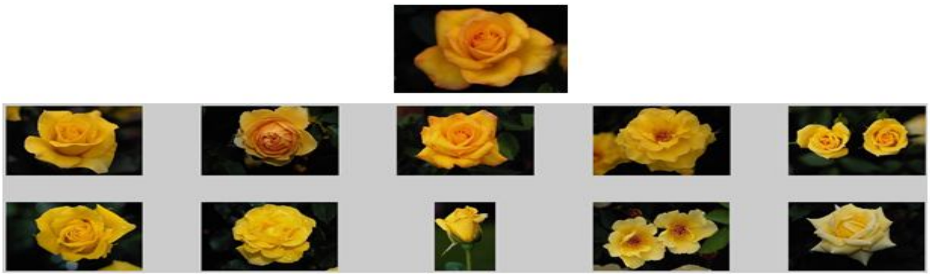
### 7.4.1 Effectiveness

To test the effectiveness of our algorithm, we randomly select 4 images from different classes, namely Flowers, Dinosaurs, Buses, and Elephants. Each query returns the top 10 images from the database. The four query retrievals are shown in Figure 7.2.

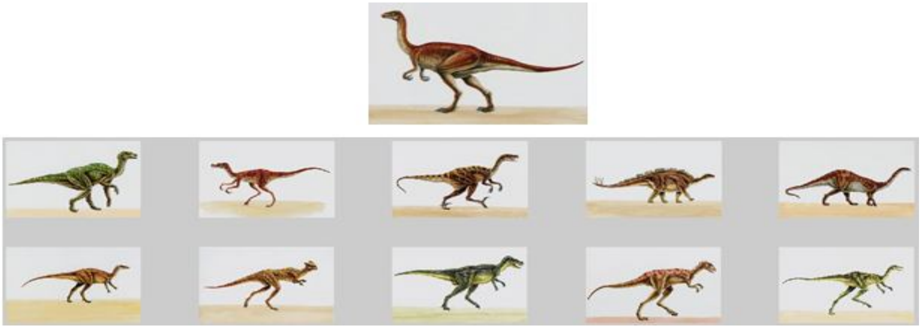
As can be seen from Figure 7.2 our ERBIR system has very good retrieving results over the randomly selected images as queries. It can be also shown that it has the same good retrieval results for most of the other images in the database if they are chosen as queries. The precision values of the retrieval results for top 5, 10, 20, and 50 retrieved images in response to each of the four queries are given in Table 7.1. As can be noticed from this table, the precision values are high for small number of retrieved images, and these values decrease as the number of retrieved image increases, indicating that the system gives a good ranking of the retrieved images, for example in Flowers query the first top 5 and top 10 of the retrieved images are relevant, whereas in the top 20 retrieved images only 2 of them are found to be irrelevant.

**Table 7.1:** Precision of ERBIR for top 5, 10, 20, and 50 retrieved images for different queries.

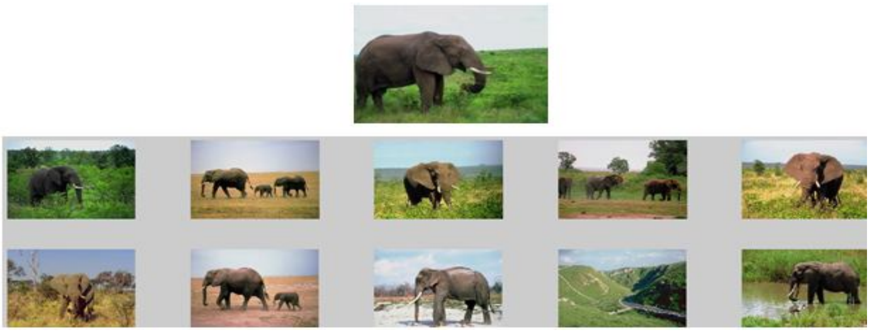
| Query | Top 5 | Top 10 | Top 20 | Top 50 |
|---|---|---|---|---|
| Flowers | 1 | 1 | 0.9 | 0.6 |
| Dinosaurs | 1 | 1 | 1 | 1 |
| Elephants | 1 | 0.9 | 0.65 | 0.58 |
| Buses | 1 | 1 | 0.9 | 0.74 |



a) Flowers Query, 10 Matches from Top 10 Retrieved Images.



b) Dinosaurs Query, 10 Matches from Top 10 Retrieved Images.



c) Elephants Query, 9 Matches from Top 10 Retrieved Images.

d) Buses Query, 10 Matches from Top 10 Retrieved Images.

**Figure 7.2:** Four Query Response Examples of ERBIR.

To further evaluate our proposed ERBIR system, 20 images are randomly selected as queries from each of the 10 semantic classes in the database, for each query the precision of the retrieval at each level of the recall is obtained by gradually increasing the number of retrieved images. The 200 retrieval results are averaged to give the final precision/recall chart of Figure 7.3.

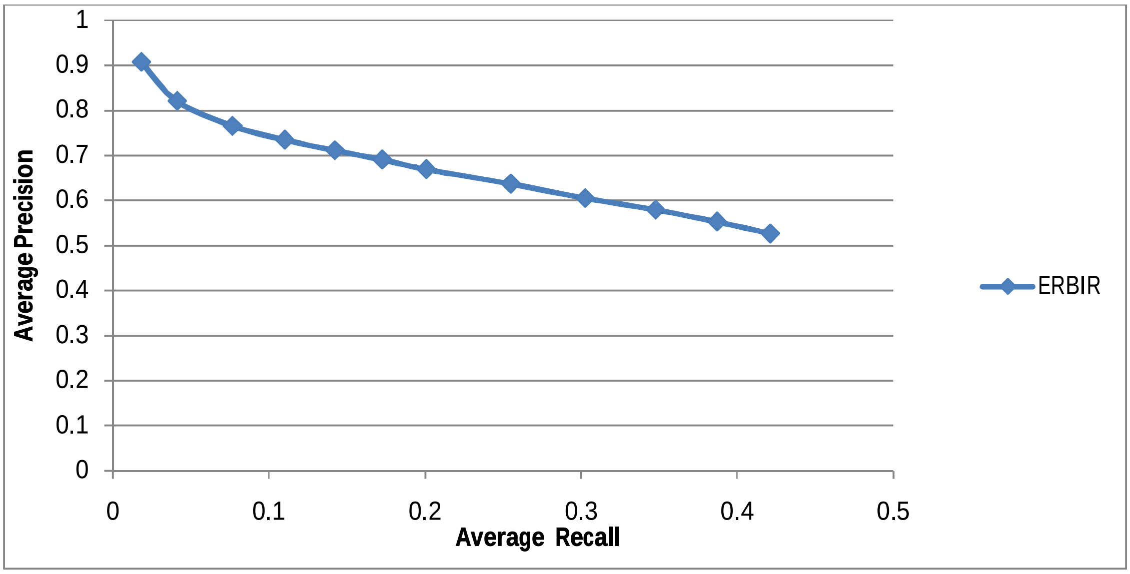



**Figure 7.3:** The Average Precision/Recall Chart of ERBIR over 200 Randomly Selected Queries.

From Figure 7.3 it can be noted that the system has good average precision values over different recall levels. It has a maximum average precision of 0.9 at recall level of .01, this value decreases to 0.52 precision value at 0.43 of recall level .For example, for an average recall value of 10%, we have an average precision value of 70% (i.e., if the user intends to get 10% of the relevant images in the database, he can get them with 70% of the retrieved images are relevant and 30% of them are irrelevant). As expected from a good retrieval system, the precision values are shown to decrease little as the recall levels increase.

## 7.4.2 Efficiency

We have improved the efficiency of our proposed ERBIR using clustering of the database image regions using the SOM algorithm as mentioned in Chapter 6. To show that the clustering process will decrease the time required responding to a query without sacrificing the accuracy of the results, we randomly selected 20 images from 10 different semantic classes in the database as queries. We applied these queries twice, the first for the ERBIR system with using clustering before image retrieval process, and the second for the ERBIR system without using clustering as pre-process (i.e., concerning all image regions in the database for image matching step). The average precisions for each group based on the returned top 20 images are shown in charts of Figure 7.4.

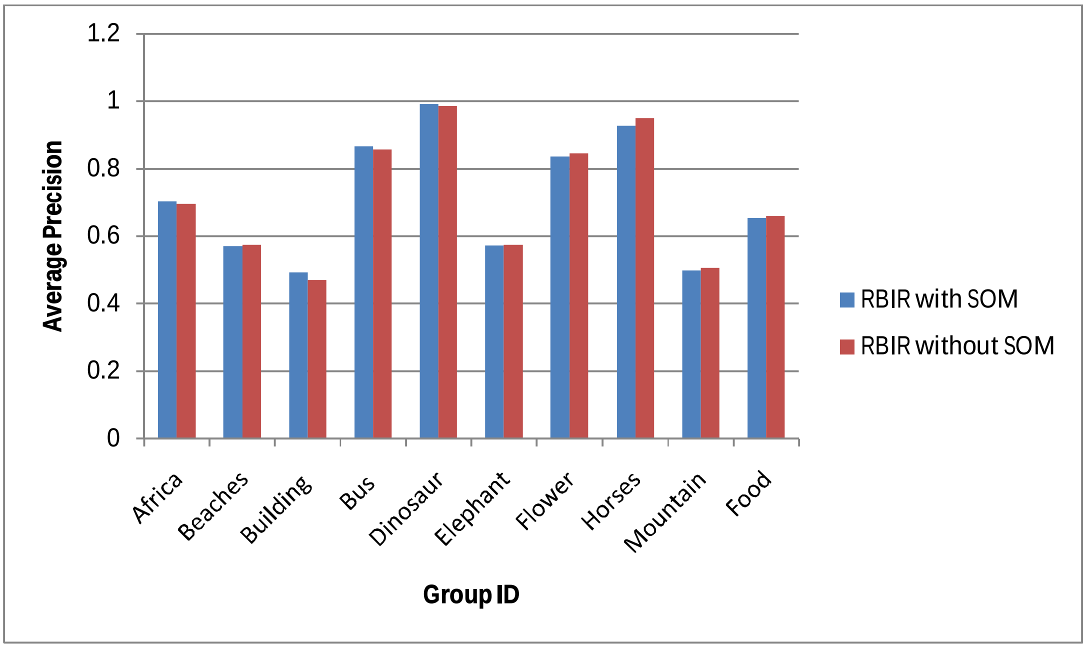



**Figure 7.4:** Comparison of Precision of ERBIR System Applied With Clustering and Without Clustering Pre-Process.

The average time the ERBIR system takes for feature extraction is about 2 seconds per image. A comparison of the average time required for returning top 20 images, per query in seconds, recorded for each semantic group in the database over 20 randomly selected queries by the ERBIR system applied with clustering, and without using clustering, pre-process is shown in Figure 7.5.

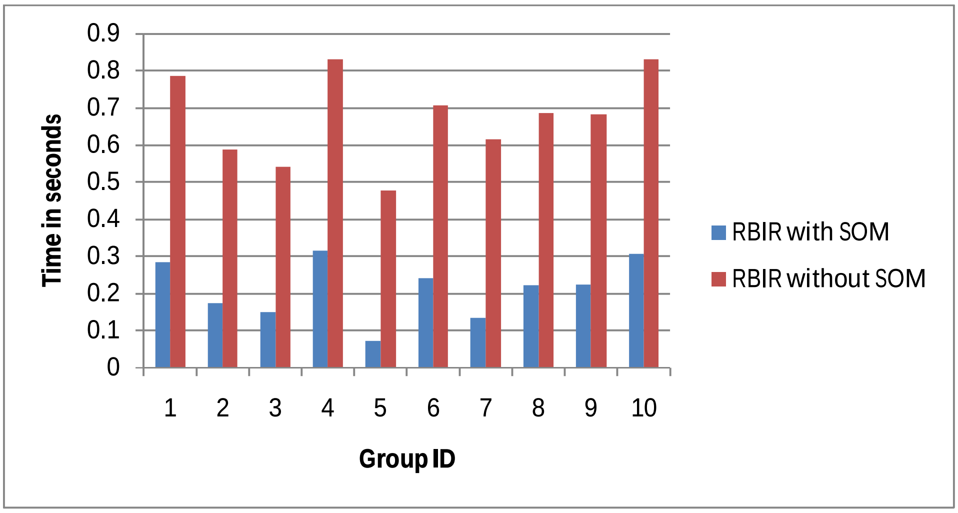



**Figure 7.5:** Comparison of Average Retrieval Time Required by ERBIR Applied with Clustering and without Clustering Pre-Process.

As depicted in Figure 7.4, the usage of the clustering pre-process of the database image regions via SOM algorithm does not degrade the average precision values of the system for the different semantic classes; even these precision values can be seen to increase slightly in some semantic classes such as groups of Africa, Buildings, Buses, and Dinosaurs. This can be reasoned to the exclusion of some target images from being searched for image matching. These target images may have small overall sum of distances (see Equations 6.7 and 6.9) between regions in these target images and query image regions, whereas none of these regions can be said to be similar to each other, i.e., the distances between them are relatively high.

Using clustering pre-process of the database image regions via SOM algorithm decreases the average query response time, and thus increases the efficiency of the system, as can be shown in Figure 7.5. Note that the system has the same feature extraction time, but using the clustering pre-process decreases the similarity search time for image matching, since it avoids searching all the database images, but some candidate images only. The query response time is reduced significantly using the clustering process, as noticed in Figure 7.5.

## 7.5 Comparison of ERBIR with Other Systems

In this subsection we evaluate the retrieval accuracy of our system and compare it with some of the existing region based algorithms. In order to calculate the performance, we used the same approach as that of Lakshmi et al [66], since we used their comparison results. For each category in the 1000 database images, we randomly selected 20 images as queries. Since we have 10 categories in the database, we have 200 query images. For each query, we examined the precision of the retrieval based on the relevance of the semantic meaning between the query and the retrieved images. Each of the 10 categories in the database portrays a distinct semantic topic, therefore this assumption is reasonable to calculate the precision. The average precisions for each group based on the returned top 20 images were recorded.

The result of this study is compared against the performance of IRM [11], Fuzzy Club [16], Geometric Histogram [70], and Signature Based [66]; the comparison is recorded in Table 7.2.

**Table 7.2:** Comparison of Precision of ERBIR with Previously Existed Systems.

| Semantic Group | Fuzzy Club | IRM | Geometric Histogram | Signature Based CBIR | Proposed ERBIR |
|---|---|---|---|---|---|
| **Africa** | 0.65 | 0.47 | 0.125 | 0.42 | 0.7025 |
| **Beaches** | 0.45 | 0.32 | 0.13 | 0.46 | 0.57 |
| **Building** | 0.55 | 0.31 | 0.19 | 0.25 | 0.4925 |
| **Buses** | 0.70 | 0.61 | 0.11 | 0.83 | 0.8675 |
| **Dinosaurs** | 0.95 | 0.94 | 0.16 | 0.92 | 0.9925 |
| **Elephants** | 0.30 | 0.26 | 0.19 | 0.95 | 0.5725 |
| **Flowers** | 0.30 | 0.62 | 0.15 | 0.96 | 0.835 |
| **Horses** | 0.85 | 0.61 | 0.11 | 0.89 | 0.9275 |
| **Mountains** | 0.35 | 0.23 | 0.22 | 0.32 | 0.4975 |
| **Foods** | 0.49 | 0.49 | 0.15 | 0.28 | 0.655 |

The comparison results in Table 7.2 show that our proposed system (ERBIR) performs significantly better than the Fuzzy Club, Geometric Histogram, and IRM in all semantic classes. Our algorithm outperforms the Signature Based algorithm in all image groups, except groups 6 and 7, which are Horses, and Flowers.

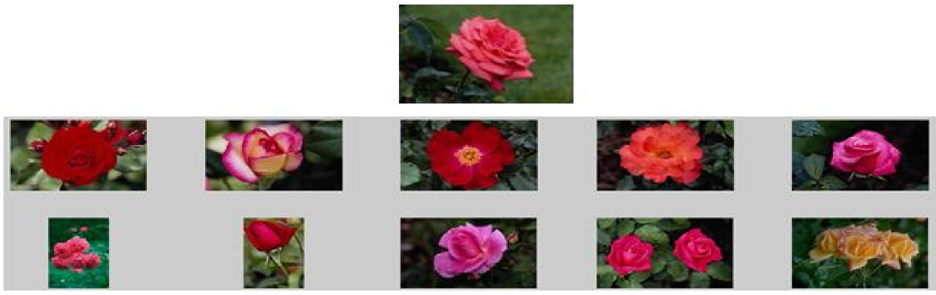
## 7.6 GBIR System Evaluation

To improve the performance of our CBIR system, we proposed to combine the GBIR system with the ERBIR system. In this section, we evaluate the performance of our proposed GBIR system.

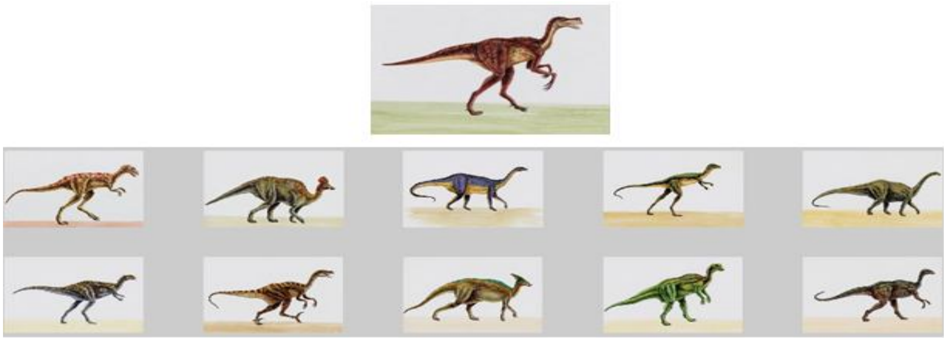
Four images are randomly selected as queries from different classes, namely Flowers, Dinosaurs, Buses, and Elephants. Each query returns the top 10 images from database. The four retrieval results in Figure 7.6 show that the GBIR system has good retrieving results for all of the four randomly selected queries.

It can be depicted from Figures 7.2 and 7.6 that the proposed GBIR system has almost the same good retrieval results as that of our proposed ERBIR system.

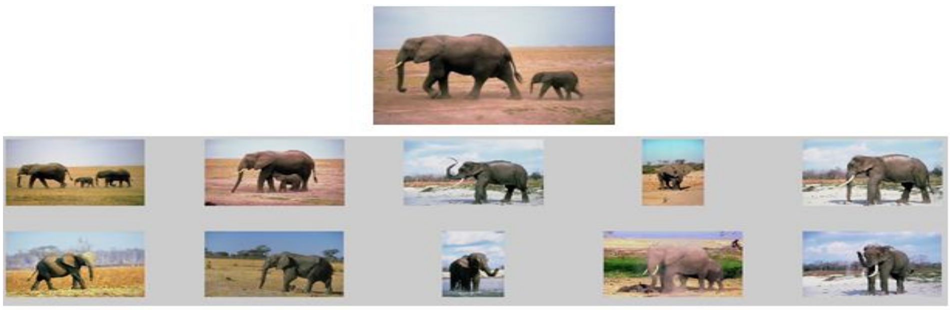
We also use the precision/recall curve to evaluate the GBIR system using the same steps we used in testing the ERBIR. The average precision/recall curve for the GBIR system is shown in Figure 7.7, from this figure it can be noticed that the system has good average precision values over different recall levels. As expected from a good retrieval system, the precision values are shown to decrease little as the recall levels increase.



a) Flowers Query, 10 Matches from the Top 10.



b) Dinosaurs Query, 10 Matches from the Top 10.



b) Elephants Query, 10 Matches from the Top 10.

d) Buses Query, 10 Matches from the Top 10.

**Figure 7.6:** Four Query Response Examples of GBIR System.

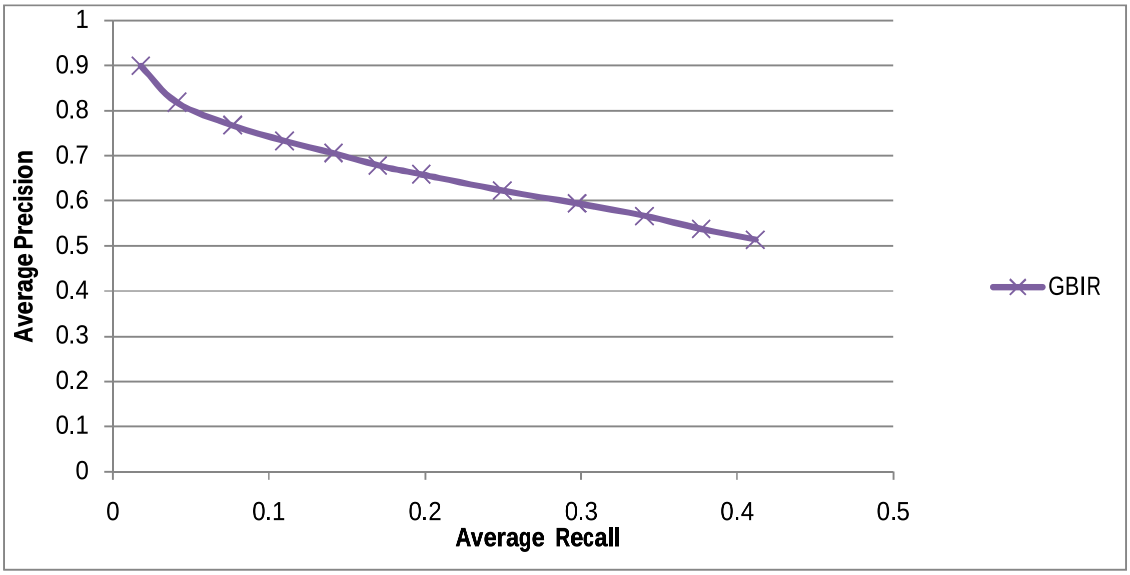



**Figure 7.7:** Average Precision/Recall Chart of GBIR over 200 Randomly Selected Queries.

## 7.7 Integrated Model GRBIR System Evaluation

As can be seen from the previous sections both of the systems we propose, GBIR and ERBIR, have good retrieval results and high average precision versus recall values. In this section, we make a comparison between them and test the result of using a combination of them in the Global Region Based Image Retrieval System (GRBIR).

To compare the effectiveness of the three systems; ERBIR, GBIR, and GRBIR, we recorded their average precision/recall curves over 200 random selected images from different semantic classes in the database as queries. The three precision/recall curves are shown in Figure 7.8.

It can be noticed from Figure 7.8 that the average precision/recall values are higher in the integrated model GRBIR than the region based RBIR and global features based GBIR approaches. Thus, using a combination of both region based and global features improves the performance of the retrieval system. The two approaches ERBIR and GBIR have approximately the same precision/recall values when the number of the retrieved images is small, but as the number of the retrieved images increases, we find out that the ERBIR system slightly overcomes the performance of the GBIR system.

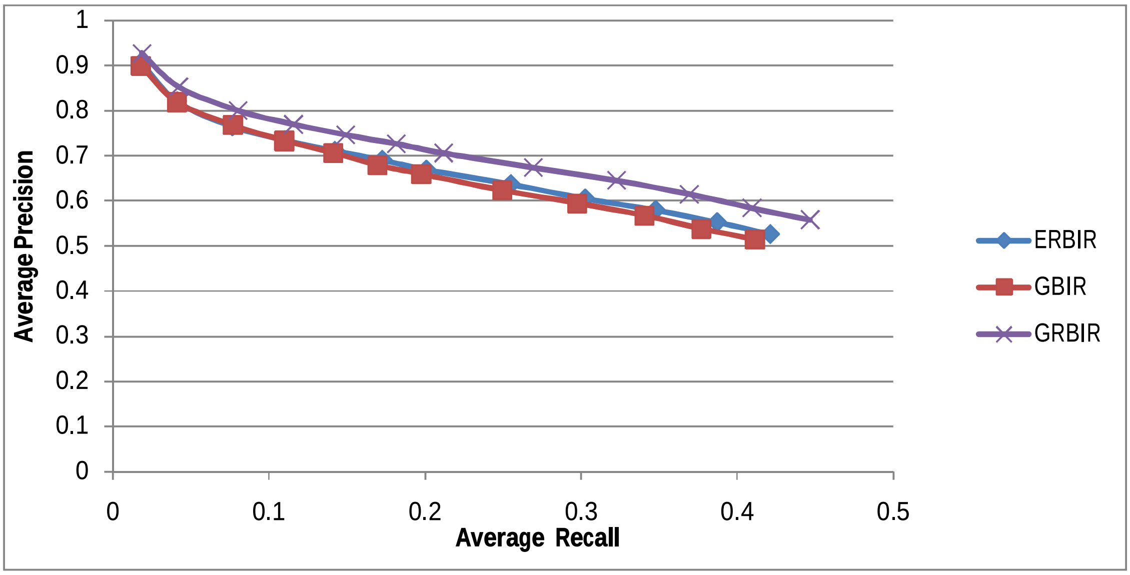



**Figure 7.8:** Average Precision/Recall Chart of ERBIR, GBIR, and GRBIR Approaches over 200 Randomly Selected Queries.

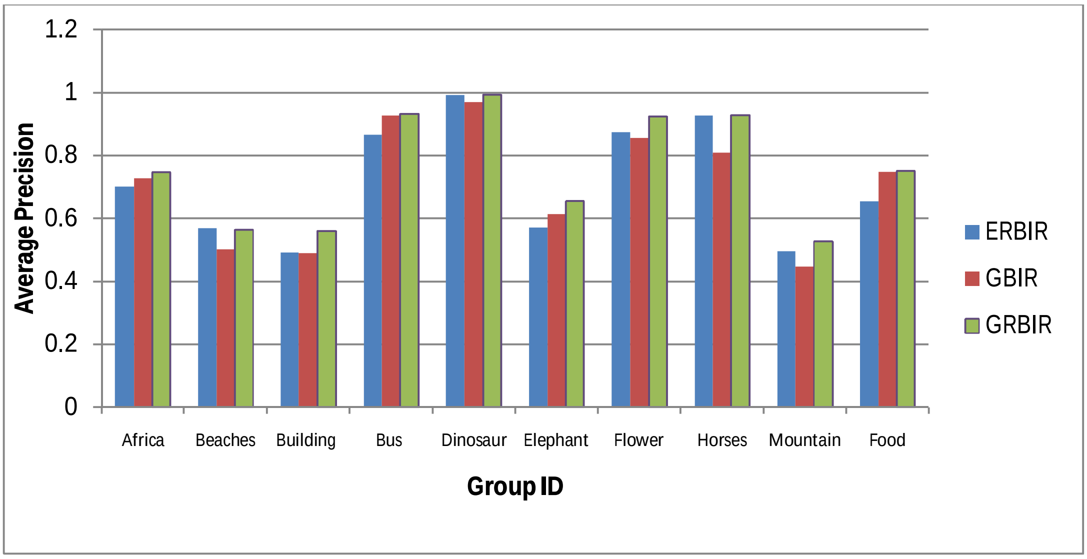



**Figure 7.9:** Precision of Integrated GRBIR compared to that of ERBIR and GBIR for Different Semantic Classes in the Database.

Even though the two systems ERBIR and GBIR have approximately equal performance results on average over all semantic classes in the test database, from our experiments we found that each of them provide better results than the other for certain images, and worse for other images according to the semantics of these images. To demonstrate this, we randomly selected 20 images as queries from each semantic class in the database, and we recorded the precision of the three systems for top 20 retrieved images responding to a selected query. The average precisions of the three systems over the 20 queries in each class are shown in Figure 7.9.

As can be noticed from Figure 7.9, the GBIR outperforms the ERBIR system in classes: Africa, Buses, Elephants, and Foods. Whereas the ERBIR has higher precisions in classes: Beaches, Dinosaurs, Horses, and Mountains. The precision of the integrated GBIR system has higher retrieval precision than the other two systems over all semantic classes, since it makes use of both types of features.

To further explain the difference between the two systems in responding to different images, we show in Figures 7.10 and 7.11 the top 10 retrieval results of the two systems responding to different two query images from the Elephants class in the database.

<u>**Query#1:**</u>

a) The GBIR Top 10 Retrieval Result to Query#1.

b) The ERBIR Top 10 Retrieval Result To Query#1.

**Figure 7.10:** The Top 10 Retrieval Result to Query#1 of GBIR, and ERBIR.

**<u>Query#2:</u>**

a) The GBIR Top 10 Retrieval Result to Query#2.

b) The ERBIR Top 10 Retrieval Result to Query#2.

**Figure 7.11:** The top 10 retrieval result to query#2 of GBIR, ERBIR.

From Figures 7.10 and 7.11, we can notice that the ERBIR gives good results for the pictures that have distinct objects with color contrast from the background and this is because these images give good results in the segmentation process, which is the first stage in the ERBIR system. In Figure 7.11, the query image has a distinct object of large area which is an Elephants, and a background of less area; this makes working on image regions better than working on global features of the image. The objects in query #2 are not clear and have smaller area than the background; this makes global features more effective than features extracted from image regions as shown in the figure.

Table 7.3 gives the retrieval precision of the GBIR and ERBIR systems, for top 20 retrieved images responding to the two queries in Figures 7.10 and 7.11.

**Table 7.3:** Precision of ERBIR, and GBIR for Top 20 Retrieved Results Responding to Two Different Queries.

| | ERBIR | GBIR | GRBIR |
|---|---|---|---|
| Query #1 | 0.85 | 0.5 | 0.9 |
| Query #2 | 0.3 | 0.56 | 0.7 |

The results in Table 7.3 ensure the difference between the three systems for top 20 retrieved images responding to the two selected queries. ERBIR is more effective for query #1, whereas GBIR is more effective for query #2, and as can be noticed the GBIR system gives better results in both cases.

# Chapter 8

# Conclusion and Future Work

This chapter presents the conclusions from this thesis. In section 8.1, we provide a summary of the thesis. Future works are proposed in section 8.2.

## 8.1 Conclusion

Content based image retrieval is a challenging method of capturing relevant images from a large storage space. Although this area has been explored for decades, no technique has achieved the accuracy of human visual perception in distinguishing images. Whatever the size and content of the image database is, a human being can easily recognize images of same category.

From the very beginning of CBIR research, similarity computation between images used either region based or global based features. Global features extracted from an image are useful in presenting textured images that have no certain specific region of interest with respect to the user. Region based features are more effective to describe images that have distinct regions. Retrieval systems based on region features are computationally expensive because of the need of segmentation process in the beginning of a querying process and the need to consider every image region in similarity computation.

In this research, we presented a content based image retrieval that introduces three alternatives to answer an image query, which are to use either region based, global based features, or a combination of them.

We use Gabor filter, which is a powerful texture extraction technique either in describing the content of image regions or the global content of an image. Color histogram as a global color feature and histogram intersection as color similarity metric combined with Gabor texture have been proved to give approximately as good retrieval results as that of region based retrieval systems.

We have increased the effectiveness of the RBIR system by estimating texture features from an image region after segmentation instead of using the average value of group of pixels or blocks through the segmentation process.

Furthermore, we have improved the efficiency of the RBIR system by not considering the whole database images for similarity computation but a number of candidate images are only considered. A candidate image is any database image that has at least one of its regions in the same cluster with any of the query image regions. The clustering process of the database image regions is performed offline using SOM algorithm. The simulation results have proved the benefit of this clustering process in decreasing the retrieval time without sacrificing the retrieval accuracy.

The performance of our algorithm has been shown to perform better compared to a number of recent systems such as Geometric Histogram, Fuzzy Club, IRM, and signature based CBIR.

Both of our proposed systems, ERBIR and GBIR, have good retrieval results and high precession/recall values. According to our simulation results, the GBIR system can be used as the first option in our retrieval system, since it gives accepted results and avoids the complex computations of the segmentation process and region comparison that are present in the ERBIR system, which can be used next to further improve the retrieval results in case of not satisfying the user.

## 8.2 Future Work

The following developments can be made in the future:

1. Region based retrieval systems are effective to some extent, but their performance is greatly affected by the segmentation process. Development of an improved image segmentation algorithm is one of our future works.
2. To further improve the performance of the retrieval system, the study of taking shape features into account during similarity distance computation can be considered.
3. To obtain better performance, the system can automatically pre-classify the database into different semantic images (such as outdoor vs. indoor, landscape vs. cityscape, texture vs. non texture images) and develop algorithms that are specific to a particular semantic image class.
4. Demonstration of using different color and texture weights in Equation 6.5 and their effect on the retrieval results.

# REFERENCES


[1] H. Tamura, and N. Yokoya, "Image Database Systems: A Survey," Pattern Recognition, vol. 17, no 1, pp.29–49, Sep. 1984.

[2] S. Gerard, C. Buckely, "Term-Weighting Approaches in Automatic Text Retrieval," Information Processing and Management, vol. 24, no.5, pp. 513-523, Jan. 1988.

[3] Y. Chen, J. Wang, "Image Categorization by Learning and Reasoning with Regions," Journal of Machine Learning Research, vol. 5, pp. 913–939, May 2004.

[4] F. Long, H. Zhang, H. Dagan, and D. Feng, "Fundamentals of content based image retrieval," in D. Feng, W. Siu, H. Zhang (Eds.): "Multimedia Information Retrieval and Management. Technological Fundamentals and Applications," Multimedia Signal Processing Book, Chapter 1, Springer-Verlag, Berlin Heidelberg New York, 2003, pp.1-26.

[5] V. Gudivada and V. Raghavan, "Content-based image retrieval systems," IEEE Computer, vol. 28, no 9, pp18-22, Sep. 1995.

[6] M. Kherfi, D. Ziou, and A. Bernardi, " Image Retrieval From the World Wide Web: Issues, Techniques, and Systems," ACM Computing Surveys, vol. 36, no. 1, pp. 35–67, March 2004.

[7] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, and P. Yanker, "Query by image and video content: The QBIC system," IEEE Computer, vol. 28, no 9, pp.23-32, Sep. 1995.

[8] A. Pentland, R. Picard, and S. Sclaroff , "Photobook: Content based manipulation of image databases," International Journal of Computer Vision, vol.18, no 3, pp.233–254, June 1997.

[9] J. Smith and S. Chang, "Visualseek: A Fully Automated Content-Based Image Query System," Proceedings of the 4th ACM international conference on Multimedia table of contents, Boston, Massachusetts, United States, Nov. 1996, pp. 87-98.

[10] A. Gupta, and R. Jain, "Visual information retrieval," Comm. Assoc. Comp. Mach., vol. 40, no. 5, pp. 70–79, May. 1997.

[11] J. Li, J. Wang, and G. Wiederhold, "Integrated Region Matching for Image Retrieval," In Proceedings of the 2000 ACM Multimedia Conference, Los Angeles, October 2000, pp. 147-156.

[12] Y. Deng, B. Manjunath, “Unsupervised Segmentation of Color -Texture Regions in Images and Video,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23, no. 8, pp. 800-810, Aug. 2001.

[13] A. Smeulders, “Content-Based Image Retrieval at the End of the Early Years,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 12, pp. 1349-1380, May. 2000.

[14] J. Caicedo, F. Gonzalez, E. Romero, E. triana, “Design of a Medical Image Database with Content-Based Retrieval Capabilities,” In Proceedings of the 2nd Pacific Rim conference on Advances in image and video technology, Santiago, Chile, December 17-19, 2007.

[15] B. Manjunath and W. Ma, “Texture features for Browsing and retrieval of image data,” IEEE transactions on pattern analysis and machine intelligence, vol. 18. No. 8, pp. 837-842, August 1996

[16] R. Zhang, and Z. Zhang, “A Clustering Based Approach to Efficient Image Retrieval,” Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI’02), Washington, DC, Nov. 2002, pp. 339-346.

[17] J. Wang, J. Li, G. Wiederhold, “Simplicity: semantics-sensitive integrated matching for picture libraries,” IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 23, no. 9, pp. 947–963, Sep. 2001.

[18] J. Smith and C. Li, “Image Classification and Querying Using Composite Region Templates,” Int. J. Computer Vision and Image Understanding, vol. 75, no. 2, pp. 165-174, July 1999.

[19] S. Mukherjea, K. Hirata, and Y. Hara, “AMORE: A World Wide Web Image Retrieval Engine,” Proc. World Wide Web, vol. 2, no. 3, pp. 115-132, June. 1999.

[20] A. Natsev, R. Rastogi, and K. Shim, “WALRUS: A Similarity Retrieval Algorithm for Image Databases,” IEEE Trans. On Knowledge and Data Engineering, vol.16, pp. 301-318, Mar. 2004.

[21] R. Picard and T. Kabir, “Finding Similar Patterns in Large Image Databases,” Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing, vol. 5, pp. 161-164, New York 1993.

[22] C. Carson, M. Thomas, S. Belongie, J.M. Hellerstein, and J. Malik, “Blobworld: A System for Region-Based Image Indexing and Retrieval,” Proc. Visual Information Systems, pp. 509-516, June 1999.

[23] W. Ma and B. Manjunath, “Natra: A Toolbox for Navigating Large Image Databases,” Proc. IEEE Int'l Conf. Image Processing, Santa Barbara, 1997, pp. 568-571.

[24] J. Wang, G. Wiederhold, O. Firschein, and X. Sha, “Content-Based Image Indexing and Searching Using Daubechies' Wavelets,” Int. J. Digital Libraries, vol. 1, no. 4, pp. 311-328, 1998.

[25] M. Ghahroudi, M. Sarshar, and R. Sabzevari, “A Novel Content-Based Image Retrieval Technique Using Tree Matching,” Proceedings of The World Congress on Engineering, pp1797-180, May 2008.

[26] C. Chiang, Y. Hung, H. Yang, and G. Lee, “Region-based image retrieval using color-size features of watershed regions,” Journal of Visual Communication and Image Representation , vol. 20, no. 3, pp. 167-177, April 2009.

[27] D. Zhang, "Improving Image Retrieval Performance by Using Both Color and Texture Features," In Proc. of IEEE 3rd International Conference on Image and Graphics (ICIG04), Hong Kong, China, Dec.18-20, 2004, pp.172-175.

[28] M. Sudhamani, and C. Venugopal, “Image Retrieval from Databases: an Approach using Region Color and Indexing Technique,” International Journal of Computer Science and Network Security, vol.8 no.1, pp. 54-64, January 2008.

[29] S. Nandagopalan, B. Adiga, and N. Deepak, “A Universal Model for Content-Based Image Retrieval,” Proc. of world academy of science, engineering and technology, vol. 36, pp. 659-662, DEC. 2008.

[30] R. Datta, J. Li, and J. Wang, “Content-based image retrieval - approaches and trends of the new age,” ACM Computing Surveys, vol. 40, no. 2, Article 5, pp. 1-60 April 2008.

[31] J. Han and M. Kambr, “Data Mining Concepts and Techniques,” $2^{nd}$ Ed., Morgan Kaufmann Publisher, 2006, ch.2, pp 71-73.

[32] R. Gonzales, R. E. Woods, “Digital Image Processing,” 2nd Ed., New Jersey Prentice Hall, 2002.

[33] T. Gevers and A. Smeulders, “Pictoseek: Combining color and shape invariant features for image retrieval,” IEEE Trans. Image Processing, vol. 9, no. 1, pp.102–119, Nov. 2000.

[34] Y. Haeghen,, J. Naeyaert, I. Lemahieu, and W. Philips, “An imaging system with calibrated color image acquisition for use in dermatology," IEEE Transaction on Medical Imaging, vol. 19, no. 7, pp. 722-730, June 2000.

[35] A. Ouyang, and Y. Tan, “A novel multi-scale spatial-color descriptor for content-based image retrieval,” Proceedings of the 7th International Conference on Control, Automation, Robotics and Vision, Mexico, August 2002, vol. 3, pp. 1204-1209.

[36] H. Yu, M. Li, H. Zhang, and J. Feng, “Color texture moments for content-based image retrieval,” Proceedings of the International Conference on Image Processing, Rochester, New York, USA, September 22-25, 2002, vol. 3, pp. 929-932.

[37] T. Gevers, and H. Stokman, “Classifying color edges in video into shadow-geometry, highlight, or material transitions,” IEEE Transactions on Multimedia, vol. 5, no. 2, pp. 237-243, Sep. 2003.

[38] H. Guan, and S. Wada, “Flexible color texture retrieval method using multi-resolution mosaic for image classification,” Proceedings of the 6th International Conference on Signal Processing, vol. 1, pp. 612-615, Feb. 2002.

[39] H. Moghaddam, T. Khajoie, and A. Rouhi, “A new algorithm for image indexing and retrieval using wavelet correlogram,” Proceedings of the International Conference on Image Processing, vol. 3, pp. 497-500, May 2003.

[40] M. Lew, N. Sebe, C. Djeraba and R. Jain, “Content-Based Multimedia Information Retrieval: State of the Art and Challenges,” ACM Transactions on Multimedia Computing, Communications and Applications, vol. 2, no. 1, pp. 1-19, February 2006.

[41] J. Fuertes, M. Lucena, N. Perez, and J. Martinez, “A Scheme of Color Image Retrieval from Databases,” Pattern Recognition Letters, vol. 22, pp.323–337, June 2001.

[42] R. Haralick, K. Shanmugam, and I. Dinstein. “Texture Features for Image Classification,” IEEE Trans. on Systems, Man and Cybernetics, SMC, vol.3, no 6, pp. 610–621, Nov. 1973.

[43] M. Chantler, and J. Wu, “Rotation Invariant Classification of 3D Surface Textures using Photometric Stereo and Surface Magnitude Spectra,” Proceedings of British Machine Vision Conference, Vol.2, pp 486-495, Jan. 2002.

[44] R. Dubes, and A. Jain, “Random field models in image analysis,” Journal Applied Statistic, vol. 16, no. 2, pp.131-164, Nov. 1989.

[45] A. Pentland, “Fractal-Based Description of Natural Scenes,” IEEE Transaction on Pattern Analysis Machine Intelligence, vol. 6, no. 6, pp. 661-674, 1984.

[46] J. Mao, and A. Jain, "Texture Classification and Segmentation using Multi-Resolution Simultaneous Autoregressive Models," Pattern Recognition, vol. 25, no. 2, pp. 173-188, 1992.

[47] C. Teh and T. Roland, "On image analysis by the methods of moments," IEEE Trans. On Pattern Analysis and Machine Intelligence, vol. 10, no. 4, pp. 496-513, 1988.

[48] M. Hu. "Visual pattern Recognition by Moment Invariants," IEEE Transactions on Information Theory, IT, vol. 8, pp. 179-187, Feb. 1962.

[49] G. Qian, S. Sural, Y. Gu, and S. Pramanik, "Similarity between Euclidean and cosine angle distance for nearest neighbor queries," Proceedings of ACM Symposium on Applied Computing, vol. 12, no. 22, pp. 1232-1237, 2004.

[50] M. Swain, D. Ballard, "Color indexing," International Journal of Computer Vision vol. 7, no 1, pp 11-32, Nov. 1991.

[51] J. Laaksonen, M. Koskela, S. Laakso, and E. Oja, "Picsom - content-based image retrieval with self-organizing maps," Pattern Recognition Letters, vol. 21, pp. 1199–1207, Feb. 2000.

[52] A. Yang, J. Wright, Y. Ma, and S. Sastry, "Unsupervised segmentation of natural images via lossy data compression," Computer Vision and Image Understanding (CVIU), vol. 110, no. 2, pp. 212-225, May 2008.

[53] M. Sudhamani, and C. Venugopal, "Segmentation of Images through clustering to Extract Color Features: An application for image Retrieval," International Journal of Computer Science, vol. 2, No.1, pp.54-61, August 2007.

[54] L.G. Shapiro and G.C. Stockman, "Computer Vision," 3$^{rd}$ Ed., New Jersey, Prentice-Hall, 2001, ch.5, pp 279-325.

[55] J.R. Smith, C.S. Li, "Image classification and querying using composite region templates," International Journal of Computer Video Database, vol. 75, pp. 165–174, 1999.

[56] M. Porat and Y. Zeevi., "The generalized Gabor scheme of image representation in biological and machine vision," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 10, no.4, pp. 452-468, July 1988.

[57] J. Smith. "Integrated Spatial and Feature Image System: Retrieval, Analysis and Compression," Ph.D dissertation, Dept. of Elect. Eng., Columbia University, New York, 1997.

[58] Y. Deng. “A Region Representation for Image and Video Retrieval,” Ph.D dissertation, University of California, Santa Barbara, 1999.

[59] V. Kyrki, J. Kamarainen, and H. Kalviainen, “Simple Gabor feature space for invariant object recognition,” Pattern Recognition Letters, vol. 25, no. 3, pp. 311–318, Feb. 2004.

[60] V. Prasad, and J. Domke, “Gabor Filter Visualization, Technical Report,” Dept. Comp. Sci., University of Maryland, Maryland , Tech. Rep, March 2005.

[61] M. Tico, T. Haverinen, and P. Kuosmanen, “A method of color histogram creation for image retrieval,” Proceedings of the Nordic Signal Processing Symposium (NORSIG'2000), Kolmarden, Sweden, June 13-15, 2000, pp. 157-160.

[62] The Online Encyclopedia and Dictionary, “HSV color space,” [Online], Available: http://www.fact-archive.com/encyclopedia/HSV_color_space, last visited June 12th 2010.

[63] T. Ojala, M. Rautiainen, E. Matinmikko, and M. Aittola. “Semantic image retrieval with HSV correlograms,” In 12th Scandinavian Conference on Image Analysis, Bergen, Norway 2001, pp. 621 - 627.

[64] J.R. Smith, S.F. Chang, “Tools and techniques for color image retrieval,” In IST/SPIE—Storage and Retrieval for Image and Video Databases IV, San Jose, CA, February 1996, vol. 2670, pp. 426–437.

[65] T. Kohonen, S. Kaski, K. Lagus, J. Salojvi, J. Honkela, V. Paatero, and A. Saarela, “Self organization of a massive document collection,” IEEE Trans. on Neural Networks, vol. 11, no 3, pp.1025–1048, May 2000.

[66] D. Lakshmi, A. Damodaram, M. Sreenivasa and J. Lal, “Content based image retrieval using signature based similarity search,” Indian Journal of Science and Technology, Vol.1, No 5, pp.80-92, Oct. 2008.

[67] J.Z. Wang, “Wang Database,” [Online], Available at: http://wang.ist.psu.edu/ , last visited August 12th 2010.

[68] F. Jing, M. Li, H.-J. Zhang, B. Zhang, “An efficient and effective region-based image retrieval framework,” IEEE Transaction on Image Processing, vol.13, no.5, pp. 699-709, May 2004.

[69] Y. Liu, D. Zhang, G. Lu, and W. Ma, “A survey of content based image retrieval with high-level semantics,” Journal of Pattern Recognition, vol. 40, pp. 262–282, Nov. 2007.

[70] A. Rao, R. Srihari, Z. Zhang, “Geometric Histogram: A Distribution of Geometric Configuration of Color Subsets,” Internet Imaging, Proceedings of the SPIE Conference on Electronic Imaging 2000, Vol. 3964-09, San Jose, CA, pp.91-101, Jan 2000.