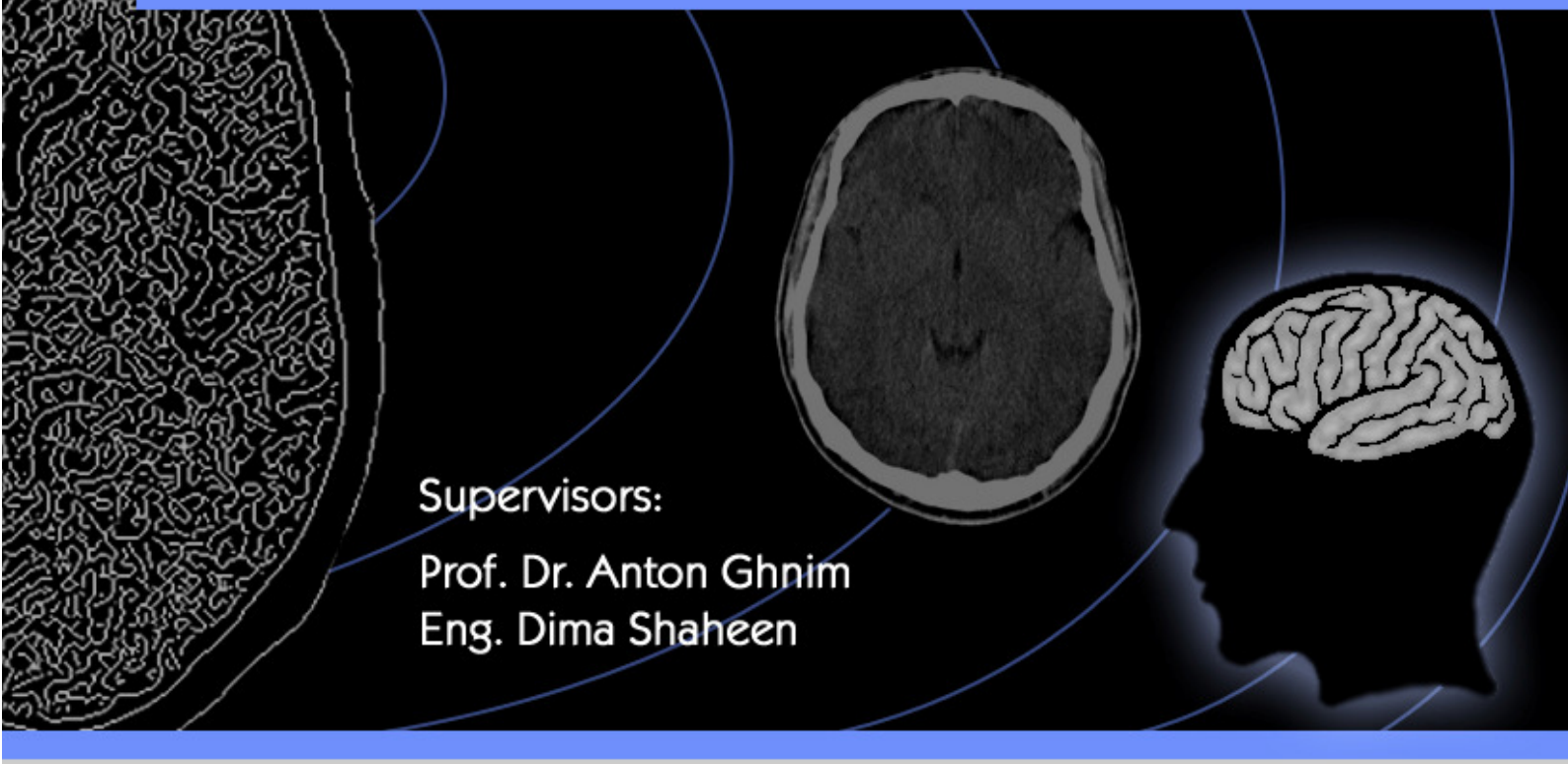
UNIVERSITY OF KALAMOON
Faculty of Engineering
Department of Information Technology

CONTENT-BASED IMAGE RETRIEVAL

AHMAD HAJJAR
OSAMA AL-KHATIB



Supervisors:
Prof. Dr. Anton Ghnim
Eng. Dima Shaheen

CBIR

2009

University of Kalamoon

Faculty of Engineering

Department of Information Technology

جامعة القلمـــون الخاصـــــة

كلية الهندســـــــة

قســـــــم تقانة المعلـــومات

Final Project

Content-Based Image Retrieval (CBIR)

Prepared by:

Ahmad Hajjar ***Osama Al-Khatib***

An Internship Report Presented in Partial Fulfillment of the Requirements for the Degree Bachelor of Science in Information Technology

Supervisors:

Prof. Dr. Anton Ghnim

Eng. Dima Shaheen

Has Been Approved

Fabruary 2009

ACKNOLWLEDGMENT

To my parents, who always take care of me and light up my way, I ascribe my successes through my life to you. I would like to thank my brother and sisters who supported me in good and bad times. I would like also to thank my family and my friends for their understanding why I have neglected them in the last few months.

This project would never have seen the light without the continuous guidance and support of my supervisors, to *Prof. Dr. Anton Ghnim* and *Eng. Dima Shaheen*, who always helped us developing the correct and correcting the wrong of our information and always eased the difficulties by their sweet words in the moments of fear.

I would like to forward very special thanks to my uncle, *Dr. Rabee Hajjar* who always guided me through my life, inspired me and provided me with all the medical information needed to complete this project. Also I would like to thank my partner *Osama Al-Khatib* who constantly motivated me by his creative ideas.

Finally, I would like to thank everyone participated, directly or indirectly, in the success of this project, especially teaching staff in the University Of Kalamoon and the staff of Al-Zahra Hospital.

Ahmad S. Hajjar

To my parents, whom taught me how to navigate life, gave me inspiration, trust and love; I have relied on throughout my last five years. My father, the person who puts the fundament to my learning character, showing me the joy of intellectual pursuit ever since I was a child. My mother, the one who sincerely raised me the faith with her caring and gently love. And to my family, brothers and sisters, who always believed in me, thanks for being supportive and caring siblings… I am honored for being your brother.

In the first place, we are deeply indebted to our supervisor *Prof. Dr. Anton Ghnim* who guided us through the project process, never accepting less than our best efforts. Special thanks to *Eng. Dima Shaheen* who was there for us from the beginning. Without her knowledge, advice and assistance this project would not have been successful.

Especially, I would like to extend my gratitude to my partner *Ahmad Hajjar* for standing by me side-to-side until the last stage of this project. It is a pleasure to convey my compliment to the members of engineering faculty and the teaching staff who played a great role in my academic life and their encouragement in difficult times.

Finally to my colleagues and friends who deserve special mention for their vital support and for giving me the motivation with all they had … and for all who made it possible for us to complete this work.

Osama Al-KhatiB

February 15th 2009

ABSTRACT

Advances in scanning, networking, compression and video technology -and the proliferation of multimedia computers- have led to the generation of large collections of images and videos. These collections or databases have created a need for new methods to locate specific images or video clips. A new content-based approach was then proposed and efficient **content-based image retrieval (CBIR)** systems become increasingly important in business and in the everyday lives of people around the world.

This document reviews the current state of the art in content-based image retrieval (CBIR), a technique for retrieving images automatically based on their visual features (such as color and shape), and relationship features (texture or spatial relations among the objects in a picture) because they are the main features human beings as well as computers use to recognize images. In addition to this, we will clarify some of the issues raised by this new technology, by reviewing its current capabilities and limitations, and its potential usefulness to users in higher education and elsewhere. The document is based on a review of contemporary research and professional literature in the field of vision science and particularly computational models written by users and managers of large collections of image data, multimedia authors, researchers, software developers, and representatives of standards bodies.

The aim of this project is to talk in details about CBIR in addition to the needed steps to implement an effective image search by using specific algorithms, these topics and more are handled in the first part of this document. Furthermore, the second part will present a biomedical application (Brain CT Scans diagnosing) as a practical sample of using this method.

TABLE OF CONTENTS

ABSTRACT **2**
TABLE OF CONTENTS **3**

PART ONE THE CBIR SYSTEM
CHAPTER 1 INTRODUCTION **5**

1.1 Overview 6
1.1.1 Image Retrieval History 6
1.1.2 What is CBIR? 7
1.1.3 Content-Based vs. Text-Based 8
1.2 Statement of the Need for CBIR Systems 10
1.2.1 CBIR Application Domain 10
1.2.2 Standards relevant to CBIR 12
1.3 CBIR System Functionality and Techniques 12
1.3.1 Query Classification According to Features Complexity 13
1.3.2 Query Classification According to Retrieval Method 14

CHAPTER 2 ANALYTICAL STUDY OF CBIR SYSTEMS **16**

17
2.1 CBIR System Architecture 18
2.1.1 Features Extraction Component 21
2.1.2 Features Indexing Component 23
2.1.3 Retrieval (Query) Engine 24
2.2 CBIR System Operations 25
2.2.1 Image Insertion (Archival) 26
2.2.2 Image Retrieval (Query) 28
2.3 Challenges and Limitations of CBIR Systems

29
CHAPTER 3 CBIR QUERY TECHNIQUES
30
3.1 Query by Example 32
3.2 Query by Semantic 33
3.2.1 Reducing Semantic Gap 34
3.2.2 Semantic CBIR System Design

PART TWO SELECTED APPLICATION AS A SAMPLE OF CBIR SYSTEM
CHAPTER 4 BIOMEDICAL CBIR SYSTEM DESIGN **37**

4.1 Biomedical Domain 38
4.1.1 Brain CT Scans Characteristics 38
4.1.2 Brain Lesions From Computer Vision Point of View 38
4.2 Biomedical CBIR System Layers 40
4.2.1 Data Layer 40
4.2.2 Processing Layer 41
4.2.2.1 Image Archival 41
4.2.2.2 Image Retrieval 44
4.2.3 Interfacing Layer 45

TABLE OF CONTENTS (con.)

4.3 Technical Issues 46
4.3.1 Computer Related Issues 46
4.3.2 Medical Issues 46

CHAPTER 5 TESTS AND RESULTS 48
5.1 Performance Measurement 49
5.2 Future Work 50

SUMMARY 52
GLOSSARY 53
REFERENCES 55

APPENDICES
Appendix A. (Arabic Summary) 56
Appendix B. (UML Models) 58
Appendix C. (Delivery List) 74
Appendix D. (CODE) 75
Appendix E. (Screenshots) 79

Introduction

The increasing amount of digitally produced images requires new methods to archive and access this data. Conventional databases allow for textual searches on meta data only. Content Based Image Retrieval is a technique which uses visual contents, normally called as features (like color, shape, textures, and edges), to search images from large scale image databases according to users' requests in the form of a query image.

This chapter describes the CBIR concept with a brief history about images retrieval, including the present need to CBIR technology in business and different domains. Basic functionalities of Content-Based image Retrieval are represented in the last section in addition to two suggested methods to classify image queries according to features complexity or the system itself.

1.1 Overview

Interest in the potential of digital images has increased enormously over the last few years, fuelled at least in part by the rapid growth of imaging on the World-Wide Web since 1990s. Users in many professional fields are exploiting the opportunities offered by the ability to access and manipulate remotely-stored images in all kinds of new and exciting ways. However, they are also discovering that the process of locating a desired image in a large and varied collection can be a source of considerable frustration. The problems of image retrieval are becoming widely recognized, and the search for solutions an increasingly active area for research and development.

Once computerized imaging became affordable (thanks largely to the development of a mass market for computer games), it soon penetrated into areas traditionally depending heavily on images for communication, such as engineering, architecture and medicine. Photograph libraries, art galleries and museums, too, began to see the advantages of making their collections available in electronic form. The creation of the computer networks, enabling users to access data in a variety of media from anywhere on the planet, has provided a further massive stimulus to the exploitation of digital images. The number of images available on the Web was recently estimated to be between 85 and 90 billion (2008) [3]– a figure which some observers consider to be a significant underestimate.

1.1.1 Image Retrieval History

The research in Image Retrieval began in the 1970s. Initially, a text-based approach was adopted. In this approach, human first manually annotates each image using metadata such as captions or keywords, and then images are retrieved based on the keywords in the text annotation, which is a simple and easy approach to manipulate. However, there are two major problems with this method. First, it requires a huge amount of human labor in the manual annotation when the image collection is large which is time consuming. Moreover, the keywords are inherently subjective and not unique so it is hard to precisely annotate the rich content of an image by humans due to perception subjectivity, which means that different persons (recipients) or the same person in different situations may judge visual content differently. Due to these disadvantages, automatic indexing and retrieval based on image content becomes more desirable for developing large volume image retrieval applications. [Section 1.1.3 will talk in details about the differences between text-based and content-based retrieval approaches]

The text-based approach remained popular until early 1990s when many large-scale image collections emerged the drawbacks of text-based approach became more and more notorious[2].

A new image retrieval approach based on analyzing the content of the images and their features starts to appear, being known as Content-Based image retrieval (CBIR). Since then, a lot of researches have been done to improve the effectiveness of using CBIR. Nowadays, three of the largest commercial CBIR systems are available – IBM's **QBIC**, **Virage's VIR Image Engine**, and **Excalibur's Image Retrieval Ware**. CBIR systems are beginning to find a foothold in the marketplace; prime application areas include crime prevention (fingerprint and face recognition), intellectual property (trademark registration), journalism and advertising (video asset management) and Web searching. Both the Altavista and Yahoo! Search engines now have CBIR facilities, courtesy of Virage and Excalibur respectively [3].

1.1.2 What is CBIR?

Content-based image retrieval (CBIR), also known as query by image content (QBIC) and content-based visual information retrieval (CBVIR) is the application of computer vision to the image retrieval problem, that is, the problem of searching for digital images in large databases. "Content-based" means that the search will analyze the actual contents of the image. The term 'content' in this context might refer to colors, shapes, textures, or any other data that can be derived from the image itself. Without the ability to examine image content, searches must rely on metadata such as captions or keywords, which may be laborious or expensive to produce. Querying or retrieval is performed by comparing feature vectors of a search image with the feature vectors of all images in the database using a similarity algorithm to rank these images according to their degree of resemblance to the query image.

While the requirements of image users can vary considerably, as we show in section 1.3 below, it can be useful to characterize image queries into three levels of abstraction: *primitive* features such as color or shape, *logical* features such as the identity of objects shown, and *abstract* attributes such as the significance of the scenes depicted. While CBIR systems currently operate effectively only at the lowest of these levels, most users demand higher levels of retrieval.

CBIR draws many of its methods from the field of image processing and computer vision, and is regarded by some as a subset of that field. It differs from these fields principally through its emphasis on the retrieval of images with desired characteristics from a collection of significant size. Image processing covers a much wider field, including image enhancement, compression, transmission, and interpretation. While there are grey areas (such as object recognition by feature analysis), the distinction between mainstream image analysis and CBIR is usually fairly clear-cut. An example may make this clear. Many police forces now use automatic face recognition systems. Such systems may be used in one of two ways. Firstly, the image in front of the camera may be compared with a single individual's database record to verify his or her identity. In this case, only two images are matched, a process few observers would call CBIR. Secondly, the entire database may be searched to find the most closely matching images. This is a genuine example of CBIR.

Research and development issues in CBIR cover a range of topics, many shared with mainstream image processing and information retrieval. Some of the most important are:

- Understanding image users' needs and information-seeking behavior.
- Identification of suitable ways of describing image content.
- Extracting such features from raw images.
- Providing compact storage for large image databases.
- Matching query and stored images in a way that reflects human similarity judgments.
- Efficiently accessing stored images by content.
- Providing usable human interfaces to CBIR systems.

1.1.3 Content-Based vs. Text-Based

Indexing and retrieval of images using techniques and methods from traditional information retrieval has *semantic understanding* as one of its main strengths. It relies on textual description of images, mostly using keywords or free text. This has high expressive power; it can be used to describe most aspects of image content. In addition, the process of searching and retrieval can be automated by a wide range of existing text retrieval software. However, it has been shown that there are three major difficulties to this approach; subjectivity, volume, and explicability.

Textual description and annotation is mainly a manual process. The combination of rich image content and differences in human perception makes it possible for two individuals to have diverging interpretations of the same image. As a result, the description might be subjective and incomplete. In addition, the use of different words by the indexer and the searcher, such as synonyms or general / specific terms to describe similar images makes retrieval based on annotation even more difficult, so text-based retrieval approach is valid only for one language. These points at an important distinction between indexing and retrieval of images using these techniques; while it is relatively easy to describe and index images, the search results depend on the enquirer's knowledge of the indexing terms used.

The second problem, the potentially large amount of images in a collection, can make textual description and annotation a tedious and very time consuming process, with indexing times quoted up to as much as 40 minutes per image.

Finally, while text based description has a high expressive power, there are some limitations when dealing with objects that are visible in nature. Some structural image characteristics are difficult to describe with words. For example, although we have a set of terms describing the different colors, none of these terms are exact. Every color has a broad range of different shades

and intensities. Although most people are able to differentiate between two different shades, it is difficult to express the differences verbally without using fuzzy terms like “more” or “less” red. We call this the problem of explicability.

The limitations inherent in metadata-based (text-based) systems, as well as the large range of possible uses for efficient image retrieval leads to grow the interest in content-based image retrieval (CBIR). Textual information about images can be easily searched using existing technology, but requires humans to personally describe every image in the database. Current indexing practice for images relies largely on text descriptors or classification codes, supported in some cases by text retrieval packages designed or adapted specially to handle images. Remarkably little evidence on the effectiveness of such systems has been published. User satisfaction with such systems appears to vary considerably. 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 [11].
CBIR operates on a totally different principle from keyword indexing. Primitive features characterizing image content, such as color, texture and shape, are computed for both stored and query images, and used to identify the stored images most closely matching the query. Semantic features such as the type of object present in the image are harder to extract. Although the 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 [7].

The effectiveness of all current CBIR systems is inherently limited by the fact that they can operate only at the primitive feature level. None of them can search effectively for, say, a photo of a horse – though some semantic queries can be handled by specifying them in terms of primitives. A beach scene, for example, can be retrieved by specifying large areas of blue at the top of the image, and yellow at the bottom. There is evidence that combining primitive image features with text keywords or hyperlinks can overcome some of these problems, though little is known about how such features can best be combined for retrieval.

The process of digitization does not in itself make image collections easier to manage, and image retrieval can not relay only on analyzing the content of the images. Some form of cataloguing and indexing is still necessary, the only difference being that much of the required information can *potentially* be derived automatically from the images themselves [3]. One way to improve the search results is by using an algorithm to mix between search by keywords and content-based image retrieval methods to search, This algorithm first retrieves the results of a keyword query from an existing image search engine, clusters the results based on extracted image features, and returns the cluster that is inferred to be the most relevant to the search query. Furthermore, it may be used in ranking the remaining results in order of relevance.

1.2 Statement of the Need for CBIR Systems

One of the main problems were highlighted was the difficulty of locating a desired image in a large and varied collection. While it is perfectly feasible to identify a desired image from a small collection simply by browsing, more effective techniques are needed with collections containing thousands of items. Journalists requesting photographs of a particular type of event, designers looking for materials with a particular color or texture, and engineers looking for drawings of a particular type of part, all need some form of access by image content.

The need for such Content-Based image retrieval systems can be described generally by two reasons; the continuous increasing in images stores scale with technology growing such as the internet and computer networks, and the interested in searching for fragments that are similar to a query, rather than a total data item that is similar to a query. The search interest is for "contains", not "is". In other words, Image data query can be classified into two different approaches: "a-whole-picture search," and "in-picture search." Each approach generates a different type of query result. "A-whole-picture" or "thumbnail-based" search approach searches for data that is globally similar to the query input; on the other hand, an "in-picture" search approach searches for a large piece of data contains a fragment that is similar to the query [12]. More information about analyzing images and segmentation techniques mentioned in section 2.1.1.

1.2.1 CBIR Application Domain

Users needing to retrieve images from a collection come from a variety of domains, including crime prevention, medicine, architecture, fashion and publishing. Remarkably little has yet been published on the way such users search for and use images, though attempts are being made to categorize users' behavior in the hope that this will enable their needs to be better met in the future.

A wide range of possible applications for CBIR technology has been identified. Potentially there are varieties of domains in which CBIR systems can work meaningfully include:

- Crime prevention
- Military application
- Intellectual property
- Architectural and engineering design
- Fashion and interior design
- Publishing and advertising
- Medical diagnosis
- Geographical information systems (GIS) and remote sensing
- Cultural heritage

- Education and training and general image search applications
- Historical researches
- Web searching
- Home entertainment

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

From this point of view, CBIR systems can be classified according to domains extensiveness to two classes:

Narrow; e.g. medical imagery retrieval, finger print retrieval and satellite imagery retrieval.
Broad; e.g. retrieving from photo collections through internet services and websites.

The next table (Table. 1.1.) distinguishes between these classes:

	Narrow	Broad
Variance of content	low	high
Source of knowledge	specific	generic
Semantics	homogeneous	heterogeneous
Ground truth	likely	unlikely
Content description	objective	subjective
Scene and sensor	possibly controlled	unknown
Aimed application	specific	generic
Type of application	professional	public
Tools	model-driven, specific invariants	perceptual, cultural, general invariants
Interactivity	limited	pervasive, iterative
Evaluation	quantitative	qualitative
System architecture	tailored database-driven	modular interaction-driven
Size	medium	large to very large
A source of inspiration	object recognition	information retrieval

Table. 1.1. CBIR classification according to domain extensiveness [13].

1.2.2 Standards relevant to CBIR

Potentially, a number of different types of standard could affect, and be affected by, developments in CBIR technology. These include [3]:

- Network protocols such as TCP/IP, governing the transmission of data between hosts holding stored data and clients running applications making use of such data.
- Image storage formats such as TIFF or JPEG, specifying how images should be encoded for long-term storage or transmission.
- Image data compression standards such as JPEG and MPEG-2, specifying standard methods for compressing image (and video) data for efficient transmission.
- Database command languages such as SQL, providing a standard syntax for specifying queries to a relational database.
- Metadata standards such as RDF, providing a framework for describing the content of multimedia objects, and languages such as XML in which to write content descriptions.

Some of these standards are unlikely to pose any implications for the development of CBIR. For example, low-level network transmission protocols such as TCP/IP handle all types of data in the same way, regarding them simply as packets of binary data whose meaning, if any, is left to the sending and receiving applications to sort out. CBIR applications are no different from any others in this respect. Similarly, storage formats for image data are not really a CBIR issue either. All commercial and many experimental CBIR systems can accept images in a wide variety of formats, converting them to their own native format for feature extraction if required. Image matching and retrieval is always performed on a database of extracted features, with the original images used purely for display purposes. Hence the format in which these images are stored has no effect on the operations of query formulation, matching or retrieval.

1.3 CBIR System Functionality and Techniques

Access to a desired image from a repository might thus involve a search for images depicting specific types of object or scene, evoking a particular mood, or simply containing a specific texture or pattern. Potentially, images have many types of attribute which could be used for retrieval, including:

- The presence of a particular combination of color, texture or shape features (e.g. green stars).
- The presence or arrangement of specific types of object (e.g. chairs around a table).
- The depiction of a particular type of event (e.g. a football match).
- The presence of named individuals, locations, or events (e.g. the Queen greeting a crowd).

- Subjective emotions one might associate with the image (e.g. happiness).
- Metadata such as who created the image, where and when.

Each listed query type (with the exception of the last one which refers to the Text-Based query) represents a higher level of abstraction than its predecessor, and each is more difficult to answer without reference to some body of external knowledge.

1.3.1 Query Classification According to Features Complexity

Features complexity differences lead naturally on to a classification of query types into three levels of increasing complexity [8]:

Level 1 comprises retrieval by *primitive* features such as color, texture, shape or the spatial location of image elements. Examples of such queries might include "find pictures with long thin dark objects in the top left-hand corner", "find images containing yellow stars arranged in a ring" – or most commonly "find me more pictures that look like this". This level of retrieval uses features (such as a given shade of yellow) which are both objective, and directly derivable from the images themselves, without the need to refer to any external knowledge base. Its use is largely limited to specialist applications such as trademark registration, identification of drawings in a design archive, or color matching of fashion accessories.

Level 2 comprises retrieval by *derived* (sometimes known as *logical*) features, involving some degree of logical inference about the identity of the objects depicted in the image. It can usefully be divided further into:

a) Retrieval of objects of a given type (e.g. "find pictures of a double-decker bus").

b) Retrieval of individual objects or persons ("find a picture of the Eiffel tower").

To answer queries at this level, reference to some outside store of knowledge is normally required – particularly for the more specific queries at level 2(b). In the first example above, some prior understanding is necessary to identify an object as a bus rather than a lorry; in the second example, one needs the knowledge that a given individual structure has been given the name "the Eiffel tower". Search criteria at this level, particularly at level 2(b), are usually still reasonably objective. This level of query is more generally encountered than level 1 – for example, most queries received by newspaper picture libraries appear to fall into this overall category.

Level 3 comprises retrieval by *abstract* attributes, involving a significant amount of high-level reasoning about the meaning and purpose of the objects or scenes depicted. Again, this level of retrieval can usefully be subdivided into:

a) Retrieval of named events or types of activity (e.g. "find pictures of Scottish folk dancing").

b) Retrieval of pictures with emotional or religious significance ("find a picture depicting suffering").

Success in answering queries at this level can require some sophistication on the part of the searcher. Complex reasoning, and often subjective judgment, can be required to make the link between image content and the abstract concepts it is required to illustrate. Queries at this level, though perhaps less common than level 2, are often encountered in both newspaper and art libraries.

This classification of query types can be useful in illustrating the strengths and limitations of different image retrieval techniques. The most significant gap at present lies between levels 1 and 2. We can refer to level 1 as image retrieval *by example*; ant to levels 2 and 3 together as *semantic* image retrieval, and hence the gap between levels 1 and 2 as the *semantic* gap.
Note that this classification ignores a further type of image query – retrieval by associated metadata (Text-Based) such as, who created the image, where and when. This is not because such retrieval is unimportant. It is because (at least at present) such metadata is exclusively textual, and its management is primarily a text retrieval issue.

1.3.2 Query Classification According to Retrieval Method

As we describe in the previous section, different implementations of CBIR make use of different methods of user queries. Basically, there are two main methods for images query basis on the CBIR itself:

A. Query by example

Query by example is a query technique that involves providing the CBIR system with an example image that it will then base its search upon. The underlying search algorithms may vary depending on the application, but result images should all share common elements with the provided example. This query technique removes the difficulties that can arise when trying to describe images with words.

B. Semantic retrieval

The ideal CBIR system from a user perspective would involve what is referred to as semantic retrieval, where the user inputs a text query for category search using preprocessed sparse binary data or just makes a request like "find pictures of flowers" or even "find pictures of Maradona". This type of open-ended task is very difficult for computers to perform - pictures of Lilies and Gardenia look very different, and Maradona may not always be facing the camera or in the same pose. Current CBIR systems therefore generally make use of lower-level features like texture, color, and shape, although some systems take advantage of very common higher-level features like faces. Not every CBIR system is generic. Some systems are designed for a specific domain, e.g. shape matching can be used for finding parts inside a CAD-CAM database[7].

Other query methods include browsing for example images, navigating customized/hierarchical categories, querying by image region (rather than the entire image), querying by multiple example images, querying by visual sketch, querying by direct specification of image features, and multimodal queries (e.g. combining touch, voice, etc.). Chapter 3 will view a detailed description of performing such queries techniques.

Analytical Study of CBIR Systems

In order to understand the functionalities of CBIR systems and how does the retrieval operation work, it is important to analyze the system architecture and describe the design of its components.

In this chapter, we illustrate a general CBIR system framework. This architecture can works for retrieving purposes from any general-purpose picture library, or it can be specialized to any special application. The two main operations in any CBIR system; image insertion and image retrieval, are defined by the representation of image information and the calculated data through each component within the system.

The last section of this chapter takes a general look at the disadvantages of CBIR systems which are playing a role in limiting its applications and stand in front of using it in some domains.

2.1 CBIR System Architecture

Typically, any content-based image retrieval system consists of three fundamental components in addition to a database engine to store the image's information with the extracted features. These components are:

A. Feature extraction component, extracts the visual feature information from the images or from database image.
B. Feature indexing component, organizes the visual feature information to speed up the query processing.
C. Retrieval engine processes the user query and provides a user interface (GUI).

Note that these components are not actually separated in physical way, the boundaries between them are logical to isolate them from each other according to their functionalities.
The general architecture of a CBIR system is shown in Figure 2.1. This framework can works fluently for general image search application, with the ability to reconfigure to adapt any specialist field. This section descries its components in detail.

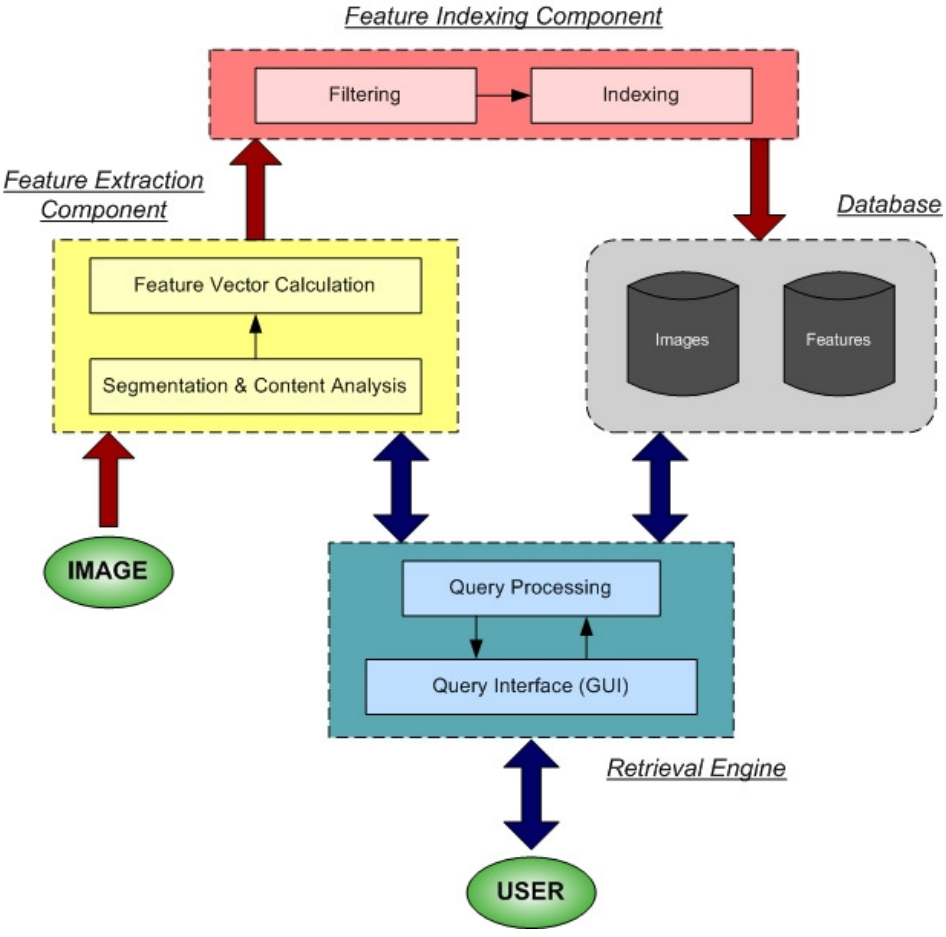


Figure. 2.1. CBIR System Main Architecture block diagram.

2.1.1 Features Extraction Component

Feature extraction is the first operation done on the images before storing in the database and searching in them. When we say feature extraction we mean extracting the visual characteristics of the image such as color, texture and shape. This operation may be applied on the whole image, or the image might be segmented into special regions before extracting the features from it.

The *Features Extraction Component* basically contains two sub blocks; *Segmentation & Content* Analysis block where any necessary partitioning can be done to focus the feature extraction process within a special region of the image instead of considering the whole image array which is the function of *Feature Vectors Calculation* block.

Segmentation & Content Analysis

Content analysis or visual content descriptor can be either global or local. A global analysis uses the visual features of the whole image, whereas a local descriptor uses the visual features of *regions* or *objects* to describe the image content. To obtain the local visual descriptors, an image is often divided into parts first. The simplest way of dividing an image is to use a *partition*, which cuts the image into tiles of equal size and shape. A simple partition does not generate perceptually meaningful regions but is a way of representing the global features of the image at a finer resolution. A better method is to divide the image into homogenous regions according to some criterion using *region segmentation* algorithms that have been extensively investigated in computer vision. A more complex way of dividing an image, is to undertake a complete *object segmentation* to obtain semantically meaningful objects (like ball, car, horse) i.e. using *Region of Interest (ROI)* algorithms [2].

Feature Vectors Calculation

Feature Vectors Calculation block uses the visual contents of an image (or a selected segmentation) such as color, texture and shape to represent the image. In typical content-based image retrieval systems, the visual contents of the images in the database are extracted and described by multi-dimensional feature vectors. The way to extract these vectors is differs from one visual content or feature to another. Typically, most common features are introduced in this section.

- Color Extraction:

The color features are the most widely used visual features in colored image retrieval because they are more robust to change due to scaling, orientation, perspective and occlusion of images. Humans perceive a color as a combination of three stimuli, R (red), G (Green), and B (Blue), which form a color space. Separating chromatic information and luminance information can generate more color spaces. To extract color information, a color space must be chosen first.

Many schemes, such as color histogram, color moments, color coherence vector, and color autocorrelogram, can be used to describe the color information in an image. Color histogram is the most widely used method since it is more robust to changes due to scaling, orientation, perspective, and occlusion of images. Color histogram represents the joint distribution of three color channels in an image in the case of the colored images, and the distribution of the gray band (luminance scale) for grayscale images (Figure. 2.2. shows an example of color image histogram). Therefore, it characterizes the global color information in an image. Color features extracted from an image are histogram related values and they represents a statistical calculations e.g. mean, median, entropy, energy, standard deviation and variance.

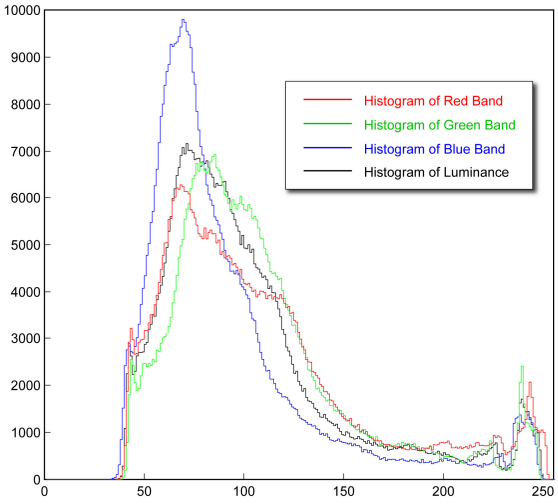


Figure. 2.2. Histogram of a color image [14].

- Texture Extraction:

There is no precise definition for texture. However, one can define texture as the visual patterns that have properties of homogeneity that do not result from the presence of only a single color or intensity. In other worlds, the ability to match on texture similarity can often be useful in distinguishing between areas of images with similar color (such as sky and sea, or leaves and grass). Texture determination is ideally suited for medical image retrievals.

Basically, texture representation methods can be classified into two categories: *structural* and *statistical*. **Structural** methods, including *morphological operator* and *adjacency graph*, describe texture by identifying structural primitives and their placement rules. They tend to be most effective when applied to textures that are very regular. **Statistical** methods, including *Fourier power spectra*, *co-occurrence matrices*, *shift-invariant principal component analysis* (*SPCA*) [5], *Tamura feature*, *Wold decomposition*, *Markov random field*, *fractal model*, and *multi-resolution filtering* techniques such as *Gabor and wavelet transform*, characterize texture by the statistical distribution of the image intensity.

- Shape Extraction:

Shape provides information about image contents i.e. the objects themselves in the image, information like: Area, circularity, eccentricity, major axis orientation, and moment invariants.

To extracting the shapes of the image; the system records the color, location, height, width, and area information of the objects for similarity measurement and identifies the color and spatial relations. This operation can be done by dividing the image first into a number of equivalent boxes and the representative color of a box is calculated based on the average color of all the pixels. Then the objects extracted by combining as many boxes in a region as possible if the color similarity between the seed box and the neighboring box is within a threshold [5]. The result will contain many regions (Figure. 2.3(c)). Each region will contain some boxes. The scattered small regions removed because, in general, these regions are useless in image retrieval and will worsen performance (Figure. 2.3(d)).

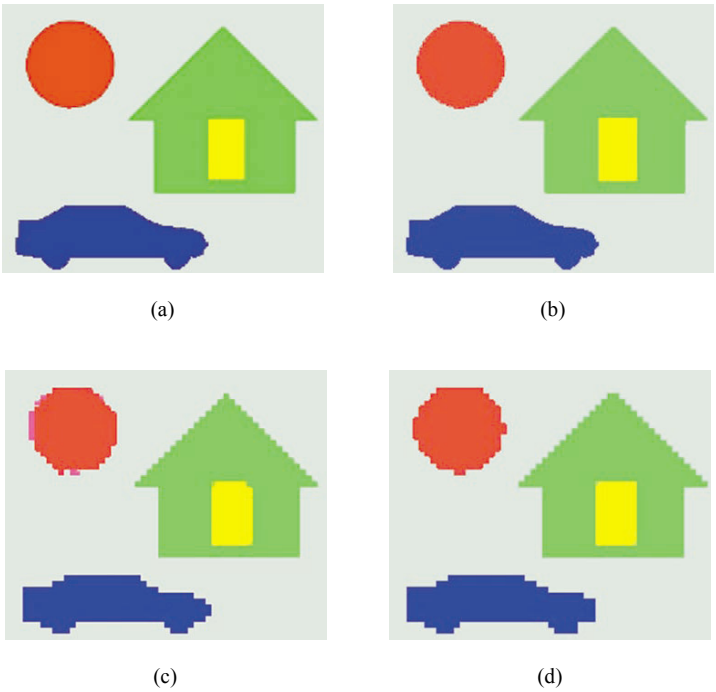


Figure. 2.3. Example of a processed image [5].
(a) Original image (b) color clustering image, (c) shape extraction image, (d) refined image.

After extracting the shapes, they normalized and convert from regions to contours using **edge detection**. Edge detection and edge tracing are very important tasks in the feature calculation of any retrieval system to locate edge pixels. There are two well-known signal edge detectors: the *Canny* operator and the *Shen-Castan (ISEF)* method (Canny, 1986; Shen and Castan, 1992). The Canny algorithm convolves the image with the derivative of a Gaussian function, and then performs non-maximum suppression and hysteresis thresholding; the Shen-Castan algorithm convolves the image using the Infinite Symmetric Exponential Filter (ISEF), computes the binary Laplacian image, suppresses false zero crossings, performs adaptive gradient thresholding, and also applies the hysteresis threshold. Canny's edge detector seems to be better under most certain circumstances for CBIR systems. Non-maximum suppression is meant to ensure that the edge line is thinned and only one pixel wide [5]. Chapter 4 includes more information about Canny algorithm for edge detection and its technical implementation.

Figure. 2.4. An example image and the result after the edge detection process [5].

The next process after edge detection is **edge tracing**, which is the process of following the edges, while usually collecting the edge pixels into a list. This is done in a consistent direction, either clockwise or counterclockwise, around the objects. The result is a non-raster representation of the objects, which can be used to compute shape measurements or otherwise identity or classify the object. However, this method is influenced by the rotation and scale of the object.
Shape representation is a set of turning angles as shown in Figure. 2.5. This method is invariant for translation and scaling of the object. In addition, it is invariant for rotation after normalization. The turning angle of the

object can express the edge's subtle differences, including those in curvature and distance. Based on the turning angle variation, feature can be classified based on their curvature properties.

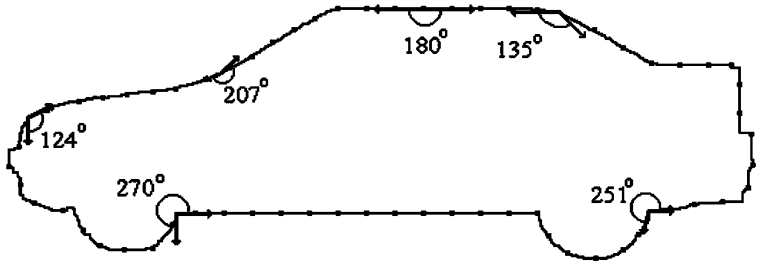


Figure. 2.5. The turning angle representation of the object [5].

An important criterion of shape feature representation is that shape features should be invariant to translation, rotation, and scaling since human beings tend to ignore such variations for recognition and retrieval purposes. In general, shape representations can be divided into two categories, *boundary-based* and *region-based*. For this reason it is important to find the point of the boundary which is farthest from the centroid and will be the starting point of turning angle representation (see Figure. 2.6).

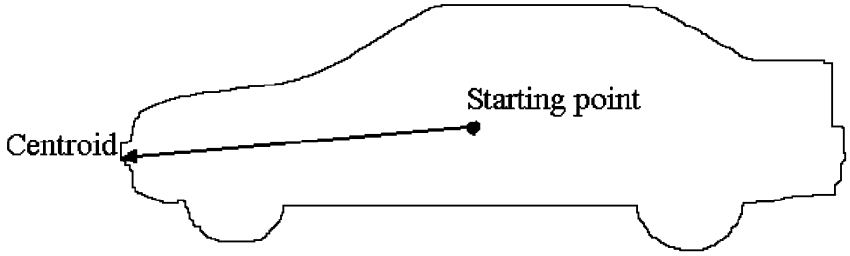


Figure. 2.6. The starting point of turning angle representation [5].

The most successful representation for these two categories is the Fourier Descriptor and Moment Invariant. The main idea of the Fourier Descriptor is to use the Fourier transformed boundary as the shape feature. The main idea of the Moment Invariant is to use region-based moments, which are shape feature invariant to transformation.

2.1.2 Features Indexing Component

Index scheme in an image database is very important for improving retrieval performance. For this reason, some filters might be done on the extracted features before storing their images records into the database, which is the *Filtering* sub-block functionality. After filters done, it is necessary for large images collection databases to be arranged in organized way to speed up the fetching time. Because storing the images randomly or even ascending according to their IDs it not enough for huge images databases, it needs a kind of indexing and clustering operations first. Features and images organizing and indexing are done in the *Indexing* block.

Filtering

General purpose retrieval systems usually contain different strategies for extracting general features. Some of these strategies may produce unnecessary features for special applications, and some feature normalization can be useful to define accurately the query space or it might reduces the needed time to execute the retrieval process. As an example, shape recognition features are not always required to distinguish natural scenes since the color seems to be more important in this case. Otherwise, multi-level color histogram is not useful for grayscale images such as X-ray scans in the biomedical applications, so such features need to be quantized before indexing them and inserting them into the database; this operation named as *Dimensionality Reduction*.
Another normalization scheme may be required to optimize the retrieval results, because different feature categories may be at different scales. This leads to uncertain similarity results when measuring the distance between feature vectors, such as the ratio of the color to the image spatial information (e.g. shape, texture and edge). In this case, *total distance algorithm* can be used to speed up the query processing [2].

Dimensionality Reduction **[6]**

Another important issue in content-based image retrieval is effective indexing and fast searching of images based on visual features. Because the feature vectors of images tend to have high dimensionality and therefore are not well suited to traditional indexing structures, *dimension reduction* is usually used before setting up an efficient indexing scheme.

One of the techniques commonly used for dimension reduction is *principal component analysis* (PCA). It is an optimal technique that linearly maps input data to a coordinate space such that the axes are aligned to reflect the maximum variations in the data. The QBIC system uses PCA to reduce a 20-dimensional shape feature vector to two or three dimensions. In addition to PCA, many researchers have used *Karhunen-Loeve* (KL) *transform* to reduce the dimensions of the feature space. Although the KL transform has some useful properties such as the ability to locate the most important sub-space, the feature properties that are important for identifying the pattern similarity may be destroyed during blind dimensionality reduction. Apart from PCA and KL transformation, *neural network* has also been demonstrated to be a useful tool for dimension reduction of features.

After dimension reduction, the multi-dimensional data are indexed. A number of approaches have been proposed for this purpose, including *R-tree* (particularly, *R*-tree*), *linear quad-trees*, *K-d-B tree* and *grid files*. Most of these multi-dimensional indexing methods have reasonable performance for a small number of dimensions (up to 20), but explore exponentially with the increasing of the dimensionality and eventually reduce to sequential searching. Furthermore, these indexing schemes assume that the underlying feature comparison is based on the Euclidean distance, which is not necessarily true for many image retrieval applications. One attempt to solve the indexing problems is to use hierarchical indexing scheme based on the *Self-Organization Map (SOM)*. In addition to benefiting indexing, SOM provides users a useful tool to browse the representative images of each type [6].

Indexing

For a small image database, sequentially searching the image during the retrieval process will be fast and provide acceptable response time. However, this is not feasible for large image database. Therefore, indexing mechanism is proposed to eliminate irrelevant images before the more complex and expensive similarity measurement is conducted.

Finding index structures which allow efficient searching of an image database is still an unsolved problem. None of the index structures proposed for text retrieval has proved applicable to the problem. The most promising approach so far has been multidimensional indexing; using structures basis of trees such as the R-tree (see Figure. 2.7.)[10], binary trees, 2D-S Tree, Graph-based, containment tree, fuzzy-based, relationship tree, etc.[1], but the overheads of using such complex index structures are considerable. A more recent approach, which seems to offer better prospects of success, is the use of similarity clustering of images, allowing hierarchical access for retrieval and providing a way of browsing the database as a bonus.

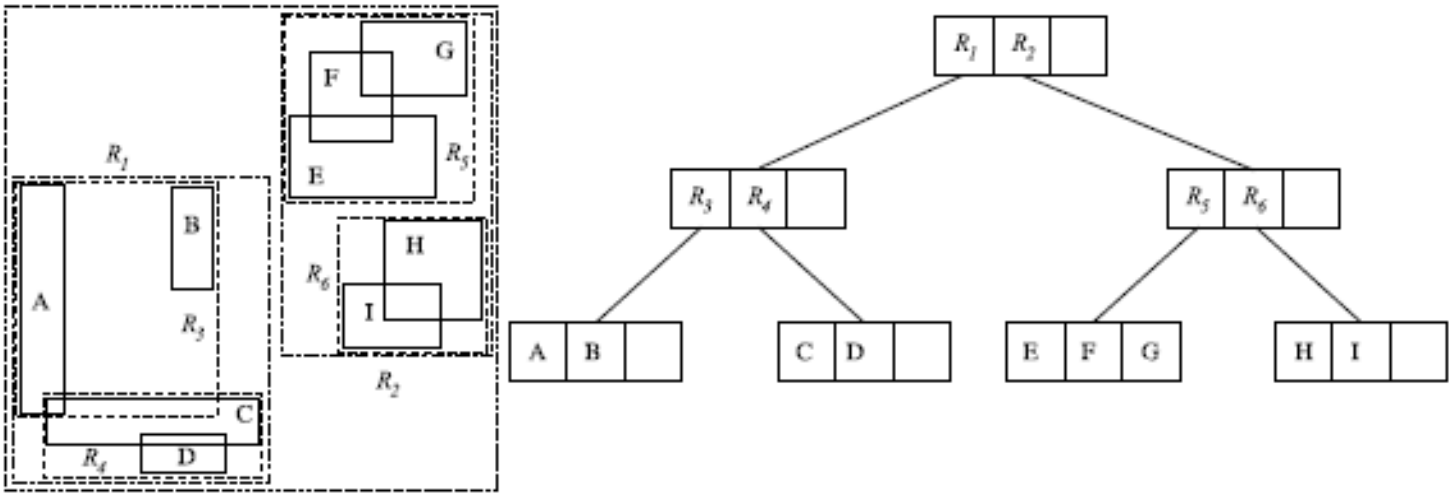


Figure. 2.7. An R-tree example in two dimensions [15].

2.1.3 Retrieval (Query) Engine

To achieve effective retrieval, an image system must be able to accurately characterize and quantify perceptual similarity. Thus, the retrieval system has to present a good way to identify the query, with a sufficient approach for retrieving various similar images using a powerful similarity measuring algorithm. *Retrieval Engine* component can be considered as the most important component in the entire retrieval system; because it is responsible for giving users the full ability to setup the query, then to execute this query applying the suitable processing and similarity measurements to produce the best results.

Query Interface (GUI):

The strength of any system can be driven from the user interface (UI); for this reason, it is very significant to have a friendly interface in case to help users to access any operation within the system as much simple as possible. For CBIR systems, image search is the most important function so it has to be well-defined and easy to use, starts from query creation (for both methods; semantic, and by example) and ends with results displaying. User interface in CBIR systems provides a number of operations such as:

- A user interface for Semantic Search and Search by Example since the ability for users to express their search needs accurately and easily is crucial in any retrieval system.
- Image insertion processes; insert, modify and delete images from system's database.
- An image processing tools to enhance the quality of the entered image samples or (e.g. changing image Brightness or Contrast).
- View query results ordered by their similarity.
- Provide some extra controls on the system itself to customize the processes within the system with flexible settings.
- Feedback ability to modify the retrieval process parameters in order to generate perceptually and semantically more meaningful retrieval results.

Query Processing:

Query processing block which known as the (Similarity Measurement Block) as well, is the core of the retrieval system because it is responsible of all similarity-related operations; i.e. apply similarity algorithms and distance measurement, results ranking, query refinement .etc. This part of the system produces the following operations:

- Process the user query by measuring the similarity (distance) between features vectors of the query image and the stored images in the database.
- Optimize the retrieval process by additional information about the query such as features weights.
- Rank the query results with the ability to filter any undesirable results.
- Refine the retrieval process when optimizing the results is required.

2.2 CBIR System Operations

Any CBIR system basically consists of two main operations; image insertion, and image retrieval. All needed extracted features are stored in the features database as a single vector to represent each image in the insertion state. Two ways can be applied for image retrieval; Query By Example (QBE), and by semantic contents. When an example query image is submitted by the user, the same work is done for the example image to get its feature vector. For similarity comparison between the query image and the database image, a similarity algorithm method is used to measure the distance between these images. For the semantic retrieval, the similarity of its corresponding features vectors are directly compared with features database. Using an

appropriate threshold, images that are semantically closer are retrieved from the database and displayed as a thumbnail.

2.2.1 Image Insertion (Archival)

As we mentioned before, image retrieval process needs a large collection of images stored in databases. These images have to be represented in a well-defined form, in case to be comparable with a given query or any image example. Image analysis or region segmentation could be required for some application to identify special regions within the entered image before extraction its features and saving them into the database. In general, Image insertion can be done as shown in figure. 2.8.

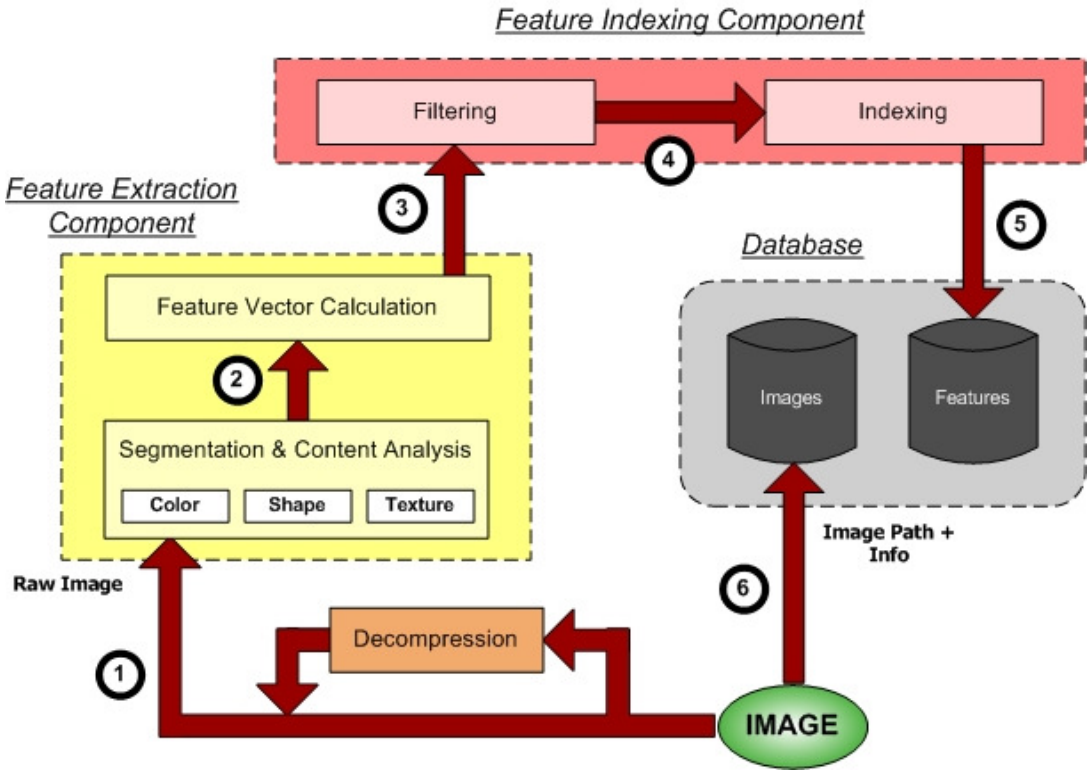


Figure. 2.8. Data flow over Image Insertion (Archival) process.

Step 1:

The feature extraction operation is based on row format (Bitmap), since its pixel stream can represent spatial information. For an image in another format, it must be converted into the bitmap format only in the feature extraction phase.

Step 2:

Feature extractions is domain related operation because of two reasons; first, some areas in the image may need to be focused or concentrated on and other least significant areas can be omitted to reduce the processing cost for some applications. Additionally, not all features spaces are necessary to calculated for all applications. For example, most of the biomedical applications work with grayscale images such as x-ray scans. In this case, just one band is required to create the histogram. Furthermore, shape similarity may not be useful for some application, so it is better not to consider it for features extraction. If any segmentation process is needed, it is done first and then feature vectors are computed.

Step 3:
Actually, there is not a clear boundary between this step and the previous one. Generally, unneeded feature vectors are ignored before storing the rest. This operation can be considered as a quantization process, so it also known as "Dimensionality Reduction"; which explained previously in this chapter.

Step 4-5:
The collection of image objects in image databases can be quite large. Efficient indices are required to accelerate the searching process. Indexing process was described in details in sections 2.1.2 of this document.

Step 6:
After inserting the features, the system saves image information in the original format to the database, or it just stores the needed information about these images such as the dimensions or the color depth. Storing some extra information about the image such as a textual description or meta-data can be useful to justify the query results.

2.2.2 Image Retrieval (Query)

The main process of any CBIR system is image retrieval process. In this process, data flow over a number of operations step-by-step moving from query processing component, going through similarity measurement ending with displaying ranked results. Figure. 2.9. depicts the data flow over this process with numbers referring to each step.

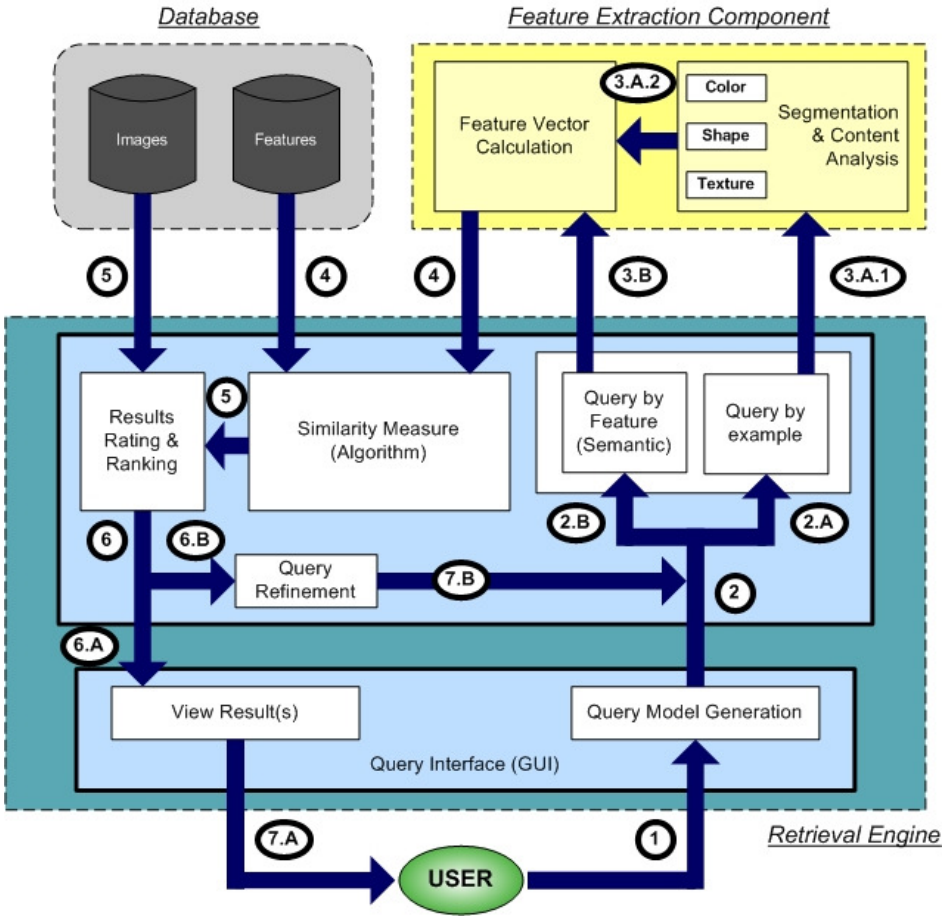


Figure. 2.9. Data flow over Image Retrieval (Query) process.

Step 1-2:
The query is generated for the example (2.A) or the semantic (2.B) entered by the user; additional information may be used to optimize the retrieval process. This can be done by selecting weights for the features in order to indicate the importance of the selected features, or it can be done by identifying threshold values to describe the minimum or the maximum distance between an image and the search image. This method is called the computer centric approach (CCA) [3]. Another strategy includes filtering the query by features, greatly reduces the search range and therefore speeds up the query processing.

Step 3:
A. (Query by example) Features are calculated after that using the same strategy used for image insertion process.
B. (Semantic) Features corresponding to the given semantic are selected to be matched with all image features from the database.

Step 4:
Calculated features vectors will be compared to the features stored in the database using a similarity measure algorithm to evaluate the nearest distances to the example image or the selected semantic.

Features filtering:
The query engine may employ the idea of filtering the features to reduce the search ranges at different stages within the retrieval process so as to speed up the query processing. Multi levels of filtering can be performed.
As an example, two filters may be used, one is the color filter, and the other is the spatial filter. For the *color filter*, the system can filter the image based on human color sensation. According to chromatology, the hue value influences the sense of warmth or coldness, the saturation value influences the sense of gray or vividness, and the intensity value influences the sense of brightness or darkness [5]. The User indicates small thresholds to represent the color-sensations to filter out images in the database that are dissimilar to the query image in color and therefore need not be compared in the second stage. The color filter effectively eliminates around 80% of the images in the database from the search range for the later stage. The images passing the color filter are sorted based on the color distances between them and the query image. In the second stage, the *spatial filter* computes the distance in spatial information between the query image and those images that pass the color filter. A threshold is also used in the spatial filter to eliminate those images that are similar to the query image in color and but are dissimilar to the query image in spatial information. This avoids the unnecessary computation time at the third stage. About 50% of images passing the color filter are removed by the spatial filter. The images in the search range are not sorted at the second stage because the final ranking is not based solely on the distance in spatial information. At the last stage, the total normalized distances between the images passing the two filters and the query image are calculated. The images in the query result are then sorted based on the total normalized distances [2].

Step 5:
Results will be ranked according to the minimum distances between the generated query and the stored images. Full images space can be considered in the ranking or a selected range; according to the ranking method.

Step 6:
Information of the best ranked images is fetched from the database to be viewed.

Step 7:
A refinement process or (relevance feedback) can be added after ranking and displaying the results where the user progressively refines the search results by marking images in the results as "relevant", "not relevant", or "neutral" to the search query, then repeating the search with the modified information in order to generate perceptually and semantically more meaningful retrieval results.

2.3 Challenges and Limitations of CBIR Systems

This domain is a wide new domain and its techniques, tools and algorithms extends to fields such as statistics, pattern recognition, signal processing, and computer vision, which made a problems in finding reliable sources. Furthermore, most of CBIR systems developed to serve special domains or science applications; which in fact is an important problem for non-specialists (e.g. biomedical applications such as the selected example in this project needs a good knowledge in brain CT scans diagnosis). In other words, some of these applications may be critical which needs enough study and researches in specific domains before implementing it.

From another point of view, digitizing some processing from the real life still causes some problems; because of the limitations of the computer itself, or it just might be hard to implementing using the available computer techniques. Currently, the computer centric approach in CBIR suffers from several disadvantages:

- **Semantic gap:**
  - Bad results due to the semantic gap and the subjectivity of human perception: This point stands for the difference between the high-level CBIR concepts usually presented to users and the low-level features actually employed. The latter addresses the fact that different persons (recipients) or the same person in different situations may judge visual content differently.

  - User seeks semantic similarity, but the database can only provide similarity by data processing.

  - Single object can represent uniform texture with different color feature, for example, sky can represented as light blue (color), upper (spatial).

- **Bad querying performance: Using (computational often very complex) distance functions for the comparison of feature vectors leads to bad, sometimes unacceptable response times.**

- **Huge amount of objects to search among.**

- **Incomplete query specification.**

- **Incomplete image description.**

CBIR Query Techniques

Different implementations of CBIR make use of different types of user queries. In this chapter, we distinguish between query-by-example (QBE) as using an image or images as the query, and query by semantic. The first section shows in detail how to retrieve images by giving an example and how to compare the similarity between visual features, in addition to representing a number of distance measuring methods. In the second section, we talk about the concepts of query by semantic and how to use a textual annotation to retrieve images contents.

3.1 Query by Example

Using this technique, CBIR System retrieves stored images from a collection by comparing features automatically extracted from the images themselves. It allows users to formulate queries by submitting an example of the type of image being sought, though some systems offer alternatives such as selection from a palette or sketch input.
Options for providing example images to the system include:

- A preexisting image may be supplied by the user or chosen from a random set.
- The user draws a rough approximation sketch and drawing of the image they are looking for, for example with blobs of color or general shapes.
- Selected color and texture patterns to appear in the resulted images.

The system then identifies those stored images whose feature values match those of the query most closely, and displays thumbnails of these images. Some of the more commonly used types of feature used for image retrieval are described below.

Color Retrieval:

One effective way in retrieving color features is by proposing a color label histogram to extract global color information. Then the color space can be quantized into a number of bins by categorizing the pixel colors into the same number of categories. The resulting color histogram is effective and efficient to obtain objects with similar colors.

Each image added to the collection in the database is analyzed to compute its histogram which shows the proportion of pixels of each color within the image; where the majority of the proposed features are variations on the color histogram initially proposed for color recognition, e.g. color coherence vectors, color correlograms, color moments, color mean and entropy, etc.
At search time, the user can either specify the desired proportion of each color (75% green and 25% red, for example), or submit an example image from which a color histogram is calculated. The matching process then retrieves those images whose color histograms or their features match those of the query most closely. Variants of this technique are now used in a high proportion of current CBIR systems. Methods of improving original technique include the use of cumulative color histograms, combining histogram intersection with some element of spatial matching, and the use of region-based color querying.

Texture Retrieval:

The ability to retrieve images on the basis of texture similarity may not seem very useful because most texture databases consist of homogeneous images. Therefore, texture retrieval usually assumes Gaussian distributed features for which simple similarity metrics, such as the

Euclidean or Mahalanobis distances are optimal. A variety of techniques has been used for measuring texture similarity; but the best-established rely on comparing values of what are known as second-order statistics calculated from query and stored images. Essentially, these calculate the relative brightness of selected pairs of pixels from each image. From these it is possible to calculate measures of image texture such as the degree of contrast, coarseness, directionality and regularity, or periodicity, directionality and randomness. Alternative methods of texture analysis for retrieval include the use of Gabor filters and fractals. Texture queries can be formulated in a similar manner to color queries, by selecting examples of desired textures from a palette, or by supplying an example query image. The system then retrieves images with texture measures most similar in value to the query.

Shape Retrieval:

The ability to retrieve by shape is perhaps the most obvious requirement at the primitive level. Unlike texture, shape is a fairly well-defined concept – and there is considerable evidence that natural objects are primarily recognized by their shape. A number of features characteristic of object shape (but independent of size or orientation such as edges) are computed for every object identified within each stored image. Queries are then answered by computing the same set of features for the query image, and retrieving those stored images whose features most closely match those of the query.

Two main types of shape feature are commonly used – global features such as aspect ratio, circularity and moment invariants and local features such as sets of consecutive boundary segments. Alternative methods proposed for shape matching have included elastic deformation of templates, comparison of directional histograms of edges extracted from the image and shocks e.g. *Edge Histogram Descriptor (EDH)*, skeletal representations of object shape that can be compared using graph matching techniques. Queries to shape retrieval systems are formulated either by identifying an example image to act as the query, or as a user-drawn sketch.

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

Retrieval by other types of primitive feature:

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

Several other types of image feature have been proposed as a basis for CBIR. Most of these rely on complex transformations of pixel intensities which have no obvious counterpart in any human description of an image. Most such techniques aim to extract features which reflect some aspect of image similarity which a human subject can perceive, even if he or she finds it difficult to describe. The most well-researched technique of this kind uses the *wavelet transform* to model an image at several different resolutions. Promising retrieval results have been reported by matching wavelet features computed from query and stored images. Another method giving interesting results is *retrieval by appearance*. Two versions of this method have been developed, one for whole-image matching and one for matching selected parts of an image. The part-image technique involves filtering the image with Gaussian derivatives at multiple scales, and then computing differential invariants; the whole-image technique uses distributions of local curvature and phase.

The advantage of all these techniques is that they can describe an image at varying levels of detail (useful in natural scenes where the objects of interest may appear in a variety of guises), and avoid the need to segment the image into regions of interest before shape descriptors can be computed.

3.2 Query by Semantic

In semantic retrieval, the system analyzed object labeling, drawings each with a set of possible interpretations and their probabilities. These were then used to derive likely interpretations of the scene within which they appeared.

More recent research has tended to concentrate on one of two problems. The first is scene recognition. It can often be important to identify the overall type scene depicted by an image, both because this in an important filter which can be used when searching, and because this can help in identifying specific objects present. One system of this type uses color, texture, region and spatial information to derive the most likely interpretation of the scene, generating text

descriptors which can be input to any text retrieval system. Other researchers have identified simpler techniques for scene analysis, using low-frequency image components to train a neural network, or color neighborhood information extracted from low-resolution images to construct user-defined templates.

The second focus of research activity is object recognition, an area of interest to the computer vision community for many years. Techniques are now being developed for recognizing and classifying objects with database retrieval in mind. Such techniques are based on the idea of developing a model of each class of object to be recognized, identifying image regions which might contain examples of the object, and building up evidence to confirm or rule out the object's presence. Evidence will typically include both features of the candidate region itself (color, shape or texture) and contextual information such as its position and the type of background in the image.

In contrast to these fully-automatic methods is a family of techniques which allow systems to learn associations between semantic concepts and primitive features from user feedback. This invites the user to annotate selected regions of an image, and then proceeds to apply similar semantic labels to areas with similar characteristics. The system is capable of improving its performance with further user feedback. Another approach is the concept of the *semantic visual template*. Here, the user is asked to identify a possible range of color, texture, shape or motion parameters to express his or her query, which is then refined using relevance feedback techniques. When the user is satisfied, the query is given a semantic label (such as "sunset") and stored in a query database for later use. Over time, this query database becomes a kind of *visual thesaurus*, linking each semantic concept to the range of primitive image features most likely to retrieve relevant items.

3.2.1 Reducing Semantic Gap

Different techniques are used to reduce the semantic gaps:

Object Ontology:

Descriptors form a vocabulary which provides a definition for high level concepts; each region is described by average color in color space, its position based on access, its size and shape. These descriptors are mapped to high level semantics based on our knowledge.

Machine Learning:

Training the machine to perform human action

Supervised Learning:

A set of images with labels and low level feature description are fed into the system. When a similar new picture is fed to the system, after it extracts the low level features it recognizes the picture by comparing it to its own picture database.

Unsupervised:

It clusters based on the image similarities and semantics and it retrieves images clusters instead of set of ordered images. And if a new image is fed, it assigns to a cluster which matches with the image.
Example: ALIPR retrieval system; trained their application using a 50,000 database images that was already been tagged. It is an automatic image annotation engine. According to the MIT tech review the ALIPR system provided at least one accurate word 98% of the time for an image query and 51% of the time the first word was most accurate.

Relevant Feedback:

The user inputs a query and a set of images are retrieved. This image is used a feedback from user and the query is adjusted and new set of images is searched. Machine learning techniques are used to learn the user feedback; such as *Neural Networks (NNs)*, *Hidden Markov Models (HMMs)*, *Bayesian Networks (BNs)*, *Support Vector Machines (SVMs)* and *Genetic Algorithms (GAs)* [13].

Semantic template:

The semantic template method involves a visual template that describes a high level concept like "sunset" or a "meeting". This approach needs the user to build a template by specifying the spatial and temporal constraints and the weights assigned to each feature for each object. However this method expects the user to know the image features and might not be easy to use for ordinary users.

WWW image retrieval:

Google and AltaVista use this method. The URL of the image, HTML documents, information surrounding the image and hyperlinks are used to derive information about the image.

3.2.2 Semantic CBIR System Design

The overall architecture of general CBIR system for semantic image retrieval is illustrated in Figure. 3.1. Initially, a *segmentation* algorithm is applied in order to divide the given image into regions (i.e. Regions of Interest (ROI)), which are likely to represent meaningful semantic objects. Then for every resulting segment, *low-level descriptions* and *fuzzy spatial relations* are estimated, the latter according to the relations supported by an explicitly defined *knowledge infrastructure* in the form of *domain ontology*.

Estimated low-level descriptions for each region are employed for generating initial *hypotheses* regarding the region's semantic label (i.e. generating an initial hypothesis set for every image region). This is realized by evaluating the compound low-level *descriptor* vector by one or more machine learning approach with generalization ability and efficiency in solving high-dimensionality pattern recognition problems; e.g. *Support Vector Machines (SVMs)*. Each descriptor is trained to identify instances of a single concept defined in the ontology.
The generated *hypothesis sets* for each region with the associated degrees of confidence for each hypothesis along with the spatial relations computed for every image segment, are subsequently employed for selecting a globally optimal set of semantic labels or annotations for the image regions by introducing them to a *genetic algorithm (GA)* which used extensively in a wide variety of global optimization problems [8].

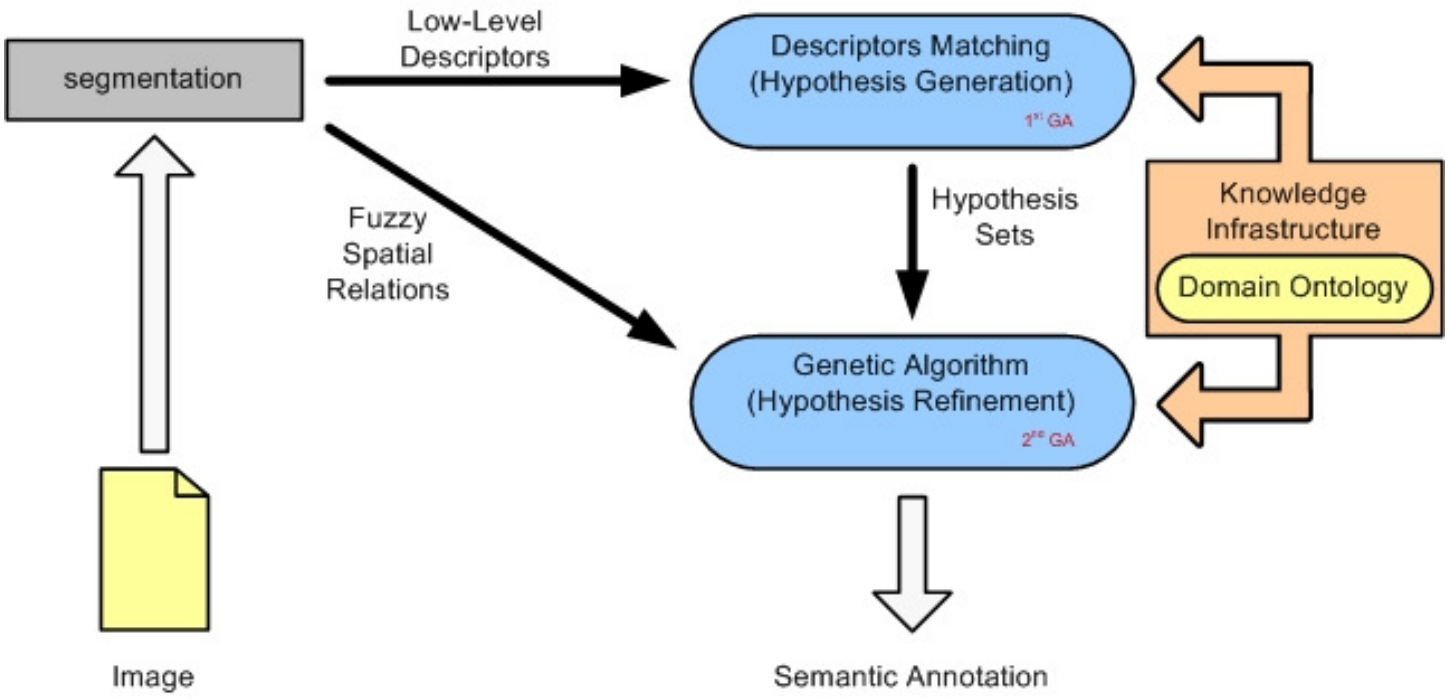
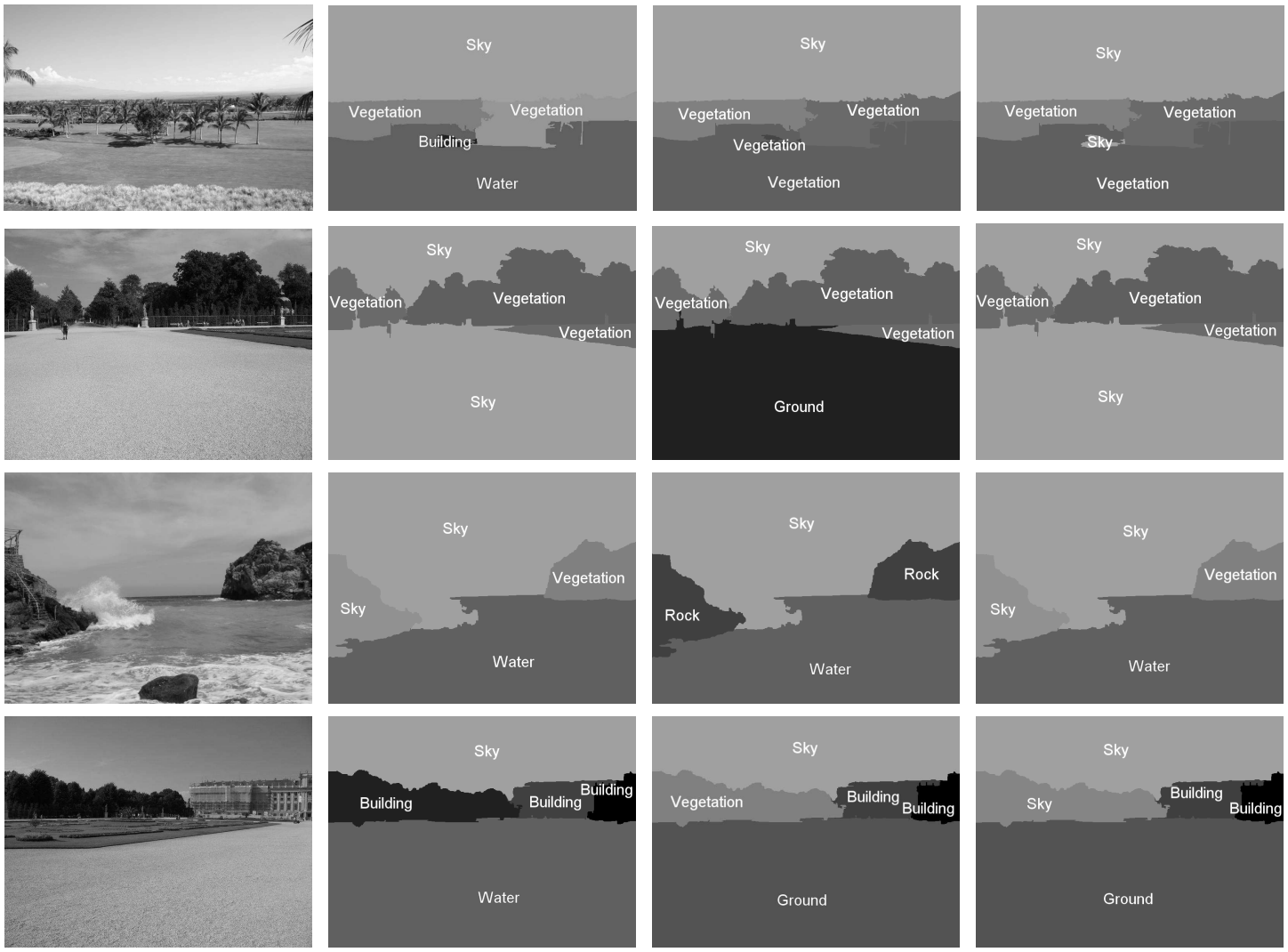


Figure. 3.1. General Semantic CBIR System Architecture [8].

In Figure. 3.2. indicative results are given showing the input image, the annotation resulting from the initial hypotheses set, considering for each image segment the hypothesis with the highest degree of confidence, and the final interpretation after the application of the genetic algorithm.



Input image | Initial hypotheses | First genetic algorithm interpretation | Second genetic algorithm interpretation

Figure. 3.2. Indicative results for the outdoor images domain [8].

Biomedical CBIR System Design

In the last two decades the medical domain has been developed which made this domain demands very big amount of data processing from archival to analysis of this data. One of the most demanding fields in medical domain is biomedical images which have a lot of types such as *microscope* images, *X-Ray* images, *MRI* (*Magnetic Resonance Imaging*) and *CT* (*Computerized Tomology*) images.

4.1 Biomedical Domain

In spite of the increasing demands of analysis in the biomedical domain, it was insignificant the development in the *computer vision field* taking place at the same period of time, which makes this domain a good domain to study.

To narrow the range of our study we concentrate on brain CT scans because this kind of images has a limited number of contents and can be well-categorized in terms of diseases.

4.1.1 Brain CT Scans Characteristics

Every kind of images in specific domain has a number of characteristics related to this domain which differentiate the domain from other domains. For brain CT scans there are number of characteristics, in what follows list of these characteristics:

1. Brain CT images are usually taken in series of sequential images for many slices in patient's brain, each series contains an average of 20 images.
2. Slices differ in their thickness for example in posterior fossa (base of skull) the slices are thinner than the slices in the upper levels or slices. The standard for slice thickness is 5mm in the posterior fossa and 10mm in the upper parts.
3. The angle in which the series of images is taken differs from one series to another, which affects the shape of the slice; the standard is to take slices parallel to posterior fossa.
4. Brain CT images are usually 512 by 512 pixels grayscale images.
5. These images come from the CT Device in DICOM format which is the standard for medical images. This format describes images in terms of tissue densities in the imaged location, where each pixel is 12-Bit depth its value ranges from -1023 to 3072, the less density the material is the less value the pixel is described and vice versa. Tissues densities are measured in Hounsfield Units (HU).

4.1.2 Brain Lesion from Computer Vision Point of View

A lesion in CT terminology is one or more *abnormal observation* in the image. From computer vision point of view a lesion is the variations in the computed image features from the features of normal brain image, so the lesions of the brain can be classified on the basis of images features variations in color, shape, or location, happened as a result of those lesions.

As mentioned before CT images are digitized in terms of (organic tissues densities) using Hounsfield Units, these units range from -1023 to +3072 according to the material being imaged e.g. for air the HU is less than -900 and for bone the HU falls between +400 and +1000.

The differences in densities from low to high make differences in image colors from black to white respectively (See figure 4.1). Abnormal variations in colors refer to existence of a lesion differs from a color range to another.

Lesions also may make variations in the shape of some parts of the brain without variations in the colors of the injured part. These variations differ from stretching to narrowing to shifting (Global brain shape change) and may be random change in the shape.

Lesions usually differ in their locations in the brain from the outer border on the bones of the skull to the cisterns in the center of the brain.

The following is a table of some common brain lesions along with their colors and shapes and location in the image. See Figure 4.2. which depicts some other lesions.

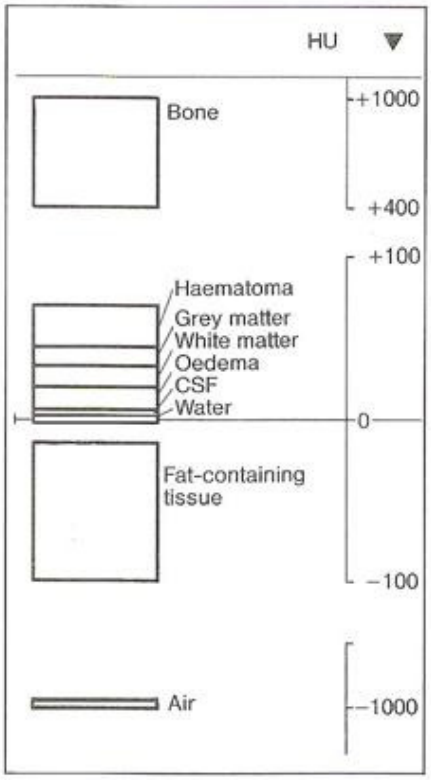


Figure. 4.1. Density values for the structures occurring in a CT scan [9].

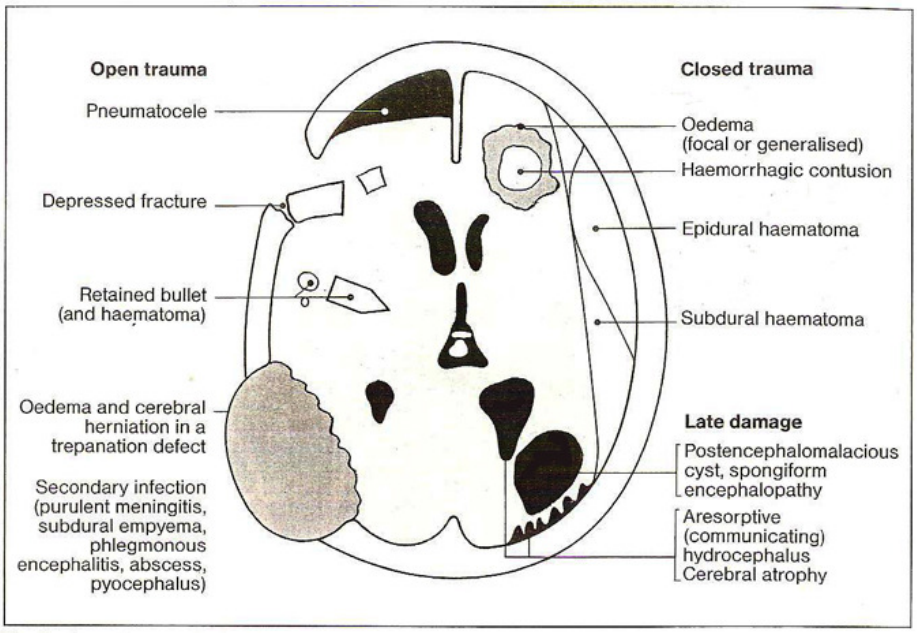


Figure. 4.2. Synopsis of head trauma [9].

Lesion	Color	Shape	Location to Image
Infractions	Dark Grey	-	Any where in the white or grey matter
Epidural Haematomas	White	convex	Between the brain and the skull bones
Subdural Haematomas	White	concave	Between the brain and the skull bones
Tumors	Black	Global change	Any where in the brain

Table 4.1 Visual features for common brain lesions

4.2 Biomedical CBIR System Layers

In order to meet the CBIR system needs, a full flexible design is introduced; this design is divided into three parts isolating the data from processing as listed below from the inner to the outer part. In the following sub-sections each layer will be explained singly.

1. Data Layer: Contains the database and system data files.
2. Processing Layer: Contains the processing methods of the system.
3. Interface Layer: Interfaces between the system and the user.

4.2.1 Data Layer

The data layer in this system consists of two parts; the first is the database the second is system data files. In this system the database forms the basic part which the whole system depends on. The system data files will contain the system settings and search history. The following diagram is the ERD (Entity-Relationship Diagram) for the database.

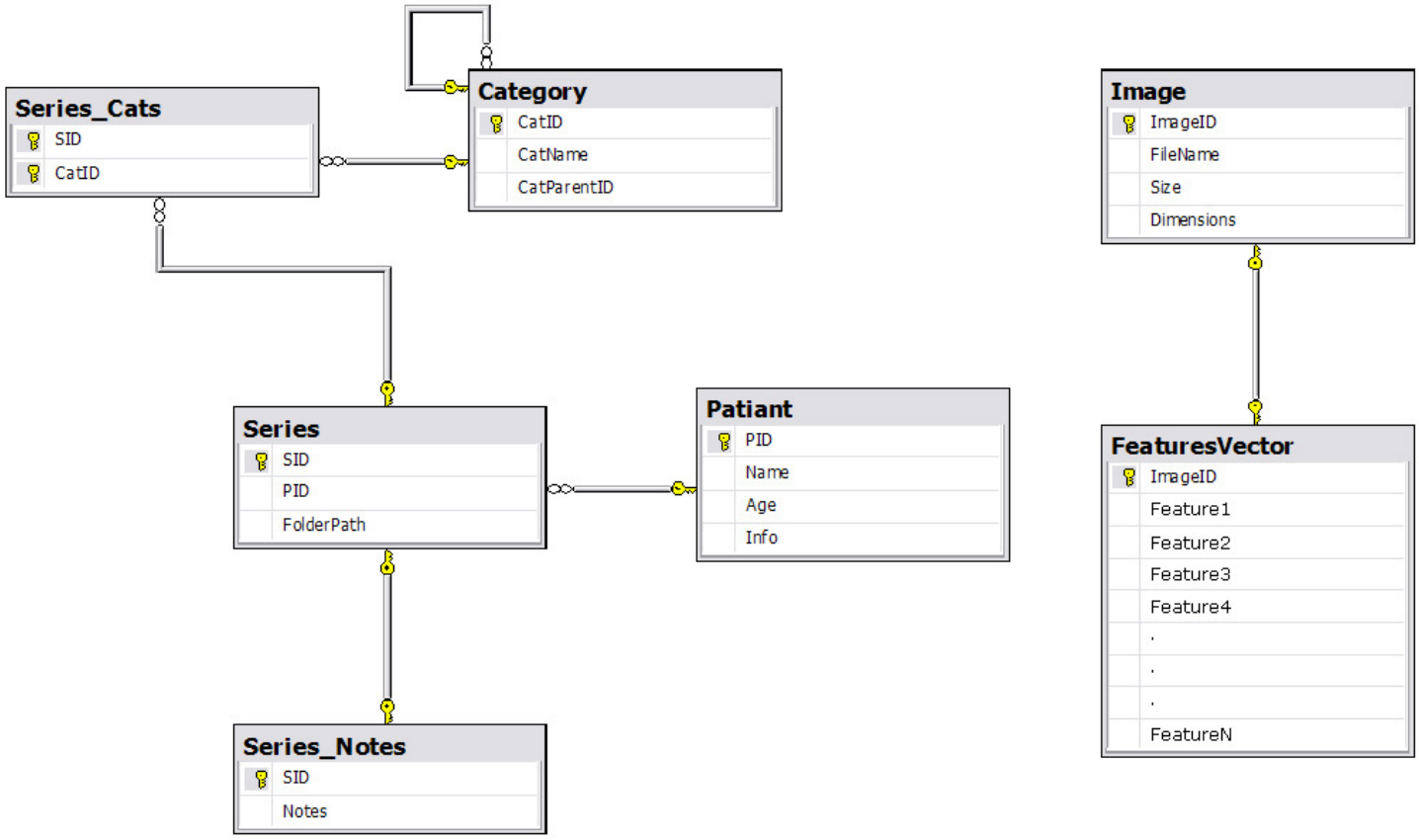


Figure. 4.3. Entity Relationship Diagram

4.2.2 Processing layer

The processing in this system is also divided into two main parts as usual CBIR systems:

1. Image Archival.
2. Image Retrieval.

4.2.2.1 Image Archival:

This process stores images in the database as series after extracting their features. In this sub-section the logic of image archival will be discussed.

The most important task in image archival process is features extraction while other tasks are simple database manipulation tasks. Feature extraction is done using MATLAB© libraries (See section 4.3.1), this is done by interfacing between the system and MATLAB© .Net© generated DLL (Dynamic Linking Library) using interface class *featureExtractor* this class takes in its constructor the image path then *AllFeatures ()* method calculates the brain location in the image and crops it then calculates all image features for the cropped image and returns a *list* of features we call it feature vector. The following is a list of features extracted from each image:

Feature	Category
Histogram Mean	Color
Histogram Standard Deviation	Color
Image Entropy	Color
Image Energy	Color
Moment One	Shape
Moment Two	Shape
Moment Three	Shape
Moment Four	Shape
Moment Five	Shape
Moment Six	Shape
Moment Seven	Shape
Contrastθ	Texture
Correlationθ	Texture
Energyθ	Texture
Homogeneityθ	Texture
Entropyθ	Texture
Maximumθ	Texture
36 Edges Directions	Edge

Table 4.2 - $\theta \in \{0, 45, 90, 135\}$

Color features are calculated using the statistics of first degree on the image it self by going through each pixel value. Here are the color features equations:

Mean:

$$m= (\Sigma_i^H \Sigma_j^W (I (i, j)))/N$$

Standard Deviation:

$$s = \sqrt{(\sum_i^W \sum_j^H (I(i,j)-m)^2)}$$

Entropy:

$$H=- \Sigma_i^H \Sigma_j^W [P (I (i, j)) \times log_2 (P (I (i, j)))]$$

$$If\ P (I (i, j)) =0\ then\ P (I (i, j)) \times log_2 (P (I (i, j))) =0$$

Energy:

$$E=\Sigma_i^H \Sigma_j^W P (I (i, j))^2$$

Where H is image height, W is image width, I (i, j) is image value on i^{th} row and j^{th} column and N is number of pixels N=H×W.

Shape features are calculated from the *invariant moments* which are computed by computing the *raw moment* and *central moment*

Raw moment of order (i+j) is defined as:

$$M_{ij} = \sum_x \sum_y x^i y^j I(x, y)$$

Central moment of order (p+q) for a digital image $f(x, y)$

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x, y)$$

Where:

$$\bar{x} = \frac{M_{10}}{M_{00}} \qquad \bar{y} = \frac{M_{01}}{M_{00}}$$

Invariant moments are calculated using the central moments μ_{pq}

$$M_1= \mu_{20}+\mu_{02}$$

$$M_2= (\mu_{20}-\mu_{02})^2+\mu_{11}^2$$

$$M_3= (\mu_{30}-3\times\mu_{12})^2+ (3\times\mu_{21}-\mu_{03})^2$$

$$M_4= (\mu_{30}+\mu_{12})^2+ (\mu_{21}+\mu_{03})^2$$

$$M_5= (\mu_{30}+\mu_{12}) + (\mu_{30}-3\times\mu_{12}) \times(((\mu_{30}+\mu_{12})^2)-3\times((\mu_{21}+\mu_{03})^2)) + (3\times\mu_{21}-\mu_{03})\times(\mu_{21}+3\times\mu_{03})\times(3\times((\mu_{03}+\mu_{21})^2)-((\mu_{21}-\mu_{03})^2))$$

$$M_6= (\mu_{20}+\mu_{02}) \times(((\mu_{30}+\mu_{12})^2)-((\mu_{21}+\mu_{03})^2)) + (4\times\mu_{11}\times(\mu_{30}+\mu_{12}) \times(\mu_{21}+\mu_{03}))$$

$$M_7= (3\times\mu_{21}-\mu_{03}) \times(\mu_{30}+\mu_{12}) \times(((\mu_{30}+\mu_{12})^2)-3\times((\mu_{21}+\mu_{03})^2)) - (\mu_{30}-3\times\mu_{12})\times(\mu_{21}+\mu_{03})\times(3\times((\mu_{03}+\mu_{21})^2)-((\mu_{21}-\mu_{03})^2))$$

Texture features for an image are computed by calculating a group of statistic on the GLCM (Gray Level Co-occurrence Matrix) of the image. These statistics and their equations are listed below:

Contrast: Measures the local variations in the gray-level co-occurrence matrix.

$$\sum_{i,j} |i-j|^2 p(i,j)$$

Correlation: Measures the joint probability occurrence of the specified pixel pairs.

$$\sum_{i,j} \frac{(i-\mu i)(j-\mu j)p(i,j)}{\sigma_i \sigma_j}$$

Energy: Provides the sum of squared elements in the GLCM.

$$\sum_{i,j} p(i,j)^2$$

Homogeneity: Measures the closeness of the distribution of elements in the GLCM to the GLCM diagonal.

$$\sum_{i,j} \frac{p(i,j)}{1+|i-j|}$$

Maximum probability in the GLCM:

$$\max (p(i, j))$$

Entropy for the GLCM:

$$H = -\Sigma_i^H \Sigma_j^W [P(I(i,j)) \times \log_2 (P(I(i,j)))]$$

If $P(I(i,j)) = 0$ *then* $P(I(i,j)) \times \log_2 (P(I(i,j))) = 0$

Edge features are a histogram of edge directions this histogram is derived from an edged image (see figure 4.4) for each white point in this image the angle θ between X-axis and the vector from the centroid to that point considering the center of the coordinates is the center of the image. This angle is calculated using the following equation:

$$\theta = \arctan (py/px)$$

Where $\theta \in [0, 360]$, px is the projection of the point on X-axis and py is the projection of the point on Y-axis.

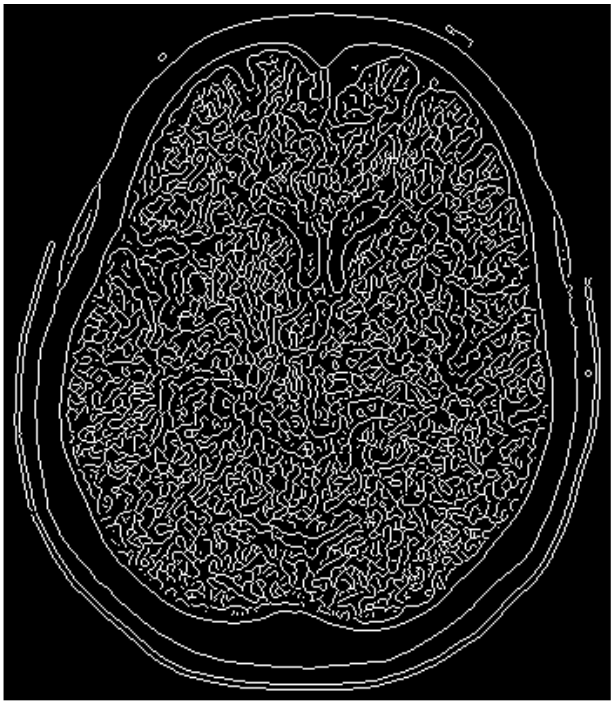


Figure 4.4. An edged brain image using Canny Edge Detection Algorithm

Then the histogram is computed by counting how many times each angle is repeated, to get a histogram of 360 elements then this histogram is quantized to 36 elements, this is what we call EDH (Edge Direction Histogram).

In image archival task the user has the choice to edit the image after committing edition changes the system will extract features from the edited image then it will insert it.

In image archival task the following database tables are affected: *Images*, *FeaturesVector*, *Series*, *Series_Notes* and *Series_Cats*.

4.2.2.2 Image Retrieval:

This process represents the main objective of this system, this process is responsible for either finding images similar to some image given by the user (Search by Example) or finding images belongs to some category by writing the name of this category (Semantic Search). This divides the tasks of this process into two tasks:

1. *Search by Example:* in this task the system searches for images similar to the image provided by the user. Firstly the system extracts the features from the provided example in the same way of feature extraction explained in the *archival process*, then the system

computes distances from the example feature vector to each feature vector stored in the database this distance in the proposed system will be computed using Minkowski Distance Function clarified in the following equation:

$$D=\sqrt[n]{\Sigma_i (fe_i-f_i)^n}$$

Where n is the Minkowski order (n=1, 2, 3 . . .) and (i) is the compound (i) of feature vectors being compared. In the proposed system we will use Minkowski of 5^{th} order n=5, this system extracts 71 features so $i \in [1, 71]$.

2. *Semantic Search:* in this task system searches for images that satisfy some conditions, these conditions are specified by the user by typing a text called *semantic query*, this query is analyzed by the system to get its *ontologies*, each ontology is related to a number of features each of them is assigned a threshold value, this threshold determines weather an image satisfy the condition of this feature or not.

   The semantic search needs some kind of learning, the objective of learning process is to determine which image features are feasible to ontology and which are not, also to give those feasible features classifying threshold values.

   Similarity in semantic search is measured by specifying which images fall into the allowed range and which are not, for number of features related to the ontology specified in the search query.

In image retrieval process the user is free to choose one or more of image features to retrieve images according to them. For Search by Example the user may also edit the example image before retrieving similar images so the system searches for images looks like the edited image.

4.2.3 Interfacing Layer

This layer is composed of three parts these parts interface the system with its environment, here is a list of these parts:

1. GUI: Interfaces between users and the system.
2. Image Manipulation Interface: Interfaces between the system and Image Manipulation Component.
3. Image Features Extractor Interface: Interfaces between the system and Feature Extraction Component.

4.3 Technical Issues

In this chapter we will point to some technical issues related to the project –the subject of this document. The technical issues will be discussed are divided into two parts Computer Related Issues and Medical Issues.

4.3.1 Computer Related Issues

The following list is a summary of these issues that must be considered in order to develop the system:

1. *Programming languages:* An object-oriented programming language was selected to meet the modern trends, the selected language is Microsoft© C-Sharp© as a main language, beside the Mathworks© MATLAB© R2007a as image processing language. The codes written in MATLAB are all then packaged into DLLs (.Net Assemblies), these codes are listed in Appendix D.
2. *Database:* The system uses the Microsoft© SQL Server© 2005 Database to store images information and features. The user can add and select database connection strings. By making some changes to *DatabaseController class* system can also connect to other types of databases.
3. *Remote Access:* In this version the remote access is not supported, but can be supported by just adding some methods to *databaseController class* and by adding some security components.
4. *The CT Samples:* Our set of samples contains about 10,000 images in 470 series processed to remove all information referring to the patient.

4.3.2 Medical Issues

Here we list a summary of the medical issues that must be taken in consideration to develop the system.

1. Image quality is not always good because it is subjected to external circumstances; most important circumstances are:
   i. Patient cooperation, by not moving his/her head while scanning.
   ii. CT scanning device quality.
   iii. Technical errors by the operator of the device such as slices thickness errors, image tilt and patient positioning which could lead to variations in shape and size of image contents or could give useless images.
   iv. The existence of lesions that are hardly diagnosed which needs a DD (Differential Diagnosis) by using another scanning technique such as MRI.

2. Image quality also is related to internal issues in the CT scan device such as the inability to depend on CT scans to judge the existence of a lesion in the posterior fossa, because of the artifacts resulting from bone affection in that area.
3. Lesions can not be studied independently, without looking at the clinical precedents.
4. Brain lesions may have the same appearances such as hemorrhagic contusion and hemorrhagic tumor.

Tests and Results

In this chapter, we have two parts, in the first, we will explain the method of testing in the progress of application development and we will detail in the performance measurement, tests and results. In the second part we will introduce some recommendations to improve the performance of the application and some suggestions to make better use of the application in the real life.

5.1 Performance Measurement

In the project life time, to confirm each component availability and reliability we applied *white box strategy* as testing strategy for each component by itself then we applied the *black box strategy* in testing the whole system as a unit to assure the system capabilities. By testing the system and its components we found the following:

1. By testing, it was found that, image processing operations are better to be done in MATLAB than to be done in C-Sharp; this is because in the most functions, the time taken by C-Sharp is greater than time taken by MATLAB, to complete the same task in a rate reaches sometimes to 30:1, the MATLAB can be replaced by some other tool (*See section 5.2*)

2. When MATLAB libraries are called from the application some kind of pre-processing is done before executing the first command, this pre-processing, often takes a lot of time, estimated to be about 15 seconds, and to overcome this problem, the first command is called from the splash screen, this command is the initialization for the classes of the MATLAB libraries.

3. Operation of feature extraction is time consuming operation, where it takes about 3.5-4.5 seconds to extract 71 features from each 512X512 image.

4. All database operations (Insertion – Updating – Deletion) are executed in real-time, except *series insertion* because, the application has to extract features from each image in the series and as mentioned before, feature extraction is time consuming.

5. Image editing operations (Brightness – Contrast) are being done in a good time measured by 0.5 seconds.

6. Search operations:
   Search is acting well when the retrieval is acquired on the basis of any group of features (*Color – Shape – Texture or Edge*). The retrieval is done in a good time measured by about 6 seconds. By the time, this document is being written, search accuracy was not the best, and the accuracy was measured to be about 60%.

All application tests have been done on the following system specifications:

1. **CPU** Intel® Pentium® 4 CPU 3.00GHz, Socket 775 LGA
2. **L2 Cache :** 2 MB
3. **Memory**: DDR2, 512 Mbytes, Frequency 267.6 MHz.
   Platform: Microsoft Windows XP Professional Service Pack 2 (Build 2600).

5.2 Future Work

- **Retrieval Enhancements:**

  To optimize the application some features or algorithms may be added to the retrieval system to improve the performance or to increase the retrieval quality. Some suggestions are:

  - Adding new features in future helps for better retrieval efficiency.

  - A weighting algorithm; for an automatic derivation of weights for all features in a query model. This purpose can be done by clustered the global feature vectors (all feature vectors merged) of images in the database. The weight of a feature for a specific search image is defined as the contribution of this feature to build the cluster of the search image.

  - An ordering algorithm; for the performance-optimized ordering of features. The ordering algorithm sorts query models before their execution according to their predicted number of returned images and the performance of the distance functions. This is a hard task but allows an enormous increase of performance.

  - A query generation algorithm; for an automatic generation of query models out of a search image or out of a group of search images and expert knowledge. The task of such algorithm is to select features and suitable thresholds for one or more query models.

  - Automated learning algorithm for image segmentation and region recognition.

- **DICOM format support**

  Supporting DICOM format will provide the application with powerful set of additional information about the patient –owner of image, technical information about the image series and each image by itself and most important is the information about the contents of the image by having the information about densities in Hounsfield units rather than an image quantized in a color range of 256 levels.

- **Develop the system to be Web Service**

  By developing Biomedical CBIR web service, can help in building some kind of (National Bank for CT Scans) where radiology specialists anywhere can utilize the CBIR.

- **Connect directly to a station that works with the CT scanning device.**

  In future, this application can be installed on a computer connected directly to a CT station which could facilitate archival operations for clinics and hospitals.

- **Threading**

  Threading for processes like feature extraction and image retrieval, is the next step in the future to improve the performance of the application.

- **Improve image processing tool**

  This can be done by adding more processing abilities or filters, or by using more compatible techniques with C#.net instead of MATLAB libraries; e.g. *AFrorge.NET* from Google (http://code.google.com/p/aforge/) is a powerful open source product with great processing capabilities.

- **Projection Tool**

  A projection tool can be added to the application for medical or teaching purposes. This tool can be used to point to special points within the brain image or highlighting injured areas.

- **From medical point of view, to improve search performance:**

  - Images must be inserted in accordance to the most important 9 slices in the brain anatomy.
  - Series must be inserted along with the clinical story for the patient, to know the context in which the brain image was acquired.
  - Some method must be found to make the inserted images technically, standard images.

- **From medical point of view the application can be developed:**
  - To give the observations in the CT scan image.
  - To give differential diagnosis for differential cases, by building a comprehensive diagnosis database.

SUMMARY

By ending this research, we achieved a good reference in CBIR systems to researchers how to work in this domain later. At the beginning of this document we have had a brief history of image retrieval, and then explained what is a CBIR and why is it needed, after that we classified CBIR systems by *Features Complexity* and by *Retrieval Method*. Then we made an analytical study for CBIR systems, explaining their architecture, their main operations and the challenges and limitations of them. Following that a spot light on CBIR querying techniques. That was a brief summary of the first part –The theoretical part.

The second part was the practical part listing at the beginning CT scans characteristics and then explained the effects of lesions on CT images observations, after that the methods of implementation was discussed starting with database structure going through the processes of image archival and image retrieval ending with the interfacing techniques. Then we discussed technical issues related to system implementation. System tests and development ability in the future was the subject to discuss at the end of the second part.

An example was programmed to clarify the discussed materials in this document, trying to project the theoretical concepts on a real life problem

GLOSSARY

English Terminology	المصطلح العربي
Proliferation	تكاثر
Multimedia Computers	حواسيب الوسائط المتعددة
Content-Based Approach	منهجية حسب المحتوى
Content-Based Image Retrieval	استرجاع الصور حسب المحتوى
Visual Features	الخصائص المرئية
Relationship Features	الخصائص العلائقية
Vision Science	علم المرئيات
Computational Models	نماذج حسابية
Biomedical Application	تطبيق طبي
Brain CT Scans Diagnosing	تشخيص صور الطبقي المحوري للدماغ
Text-Based	معتمد على النصوص
Standards	معايير
Domain	مجال
Query	سؤال
Classification	تصنيف
Features Complexity	عقدية الخصائص
Retrieval Method	طريقة الاسترجاع
Features Extraction Component	مكون استخراج الخصائص
Features Indexing Component	مكون فهرسة الخصائص
Retrieval Engine	محرك البحث
Insertion (Archival)	الإدخال أو الأرشفة
Retrieval (Query)	الاسترجاع أو السؤال
Limitation	المحدودية
Query By Example	الطلب باستخدام مثال
Query By Semantic	الطلب باستخدام الإملاء
Brain Lesions	آفة الدماغ
Computer Vision	رؤية الحاسوب
Data Layer	طبقة البيانات
Processing Layer	طبقة المعالجة
Interfacing Layer	طبقة التخاطب

English Terminology	المصطلح العربي
Performance Measurement	قياس الأداء
Meta Data	المعلومات العالية
Query Image	صورة المثال
Imaging	التصوير
Remotely-Stored	المخزن عن بعد
Computerized Imaging	التصوير المحوسب
Media	وسط التخزين
Perception Subjectivity	لاموضوعية الرؤية
Crime Prevention	منع الجرائم
Fingerprint Recognition	التعرف على البصمة
Face Recognition	التعرف على الوجه
Laborious	يتطلب عمالة
Feature Vector	شعاع الخصائص
Similarity Algorithm	خوارزمية المطابقة
Rank Images	ترتيب الصور
Degree of resemblance	درجة التطابق
Primitive Features	الخصائص البدائية
Logical Features	الخصائص المنطقية
Abstract Attributes	الخصائص المجردة
Image Processing	معالجة الصور
Object Recognition	التعرف على الأشياء
Feature Analysis	تحليل الخصائص
Image Enhancement	تحسين الصورة
Information Seeking	البحث عن المعلومات
Raw Images	الصور الخام
Human Similarity Judgments	الحكم الإنساني على التطابق
Textual-Description	الوصف اللفظي
Subjectivity	اللاموضوعية
Metadata-Based	الاعتماد على المعلومات العالية
Semantic Features	الخصائص اللفظية
Semantic Classes	الصفوف اللفظية

REFERENCES

[1] S. Nandagopalan, **A Universal Model for Content-Based Image Retrieval**, 2008.

[2] Xiuqi Li, **An Effective Content-based Visual Image Retrieval System**, Florida Atlantic University, 2002.

[3] John Eakins, and Margaret Graham, **Content-based Image Retrieval**, University of Northumbria at Newcastle, 1999.

[4] Horst Eidenberger, **Query Model-Based Content-Based Image Retrieval**, 2000.

[5] TIMOTHY K. SHIH, **An Intelligent Content-based Image Retrieval System Based on Color, Shape and Spatial Relations**, Tamkang University, 2000.

[6] Dr. Fuhui Long, Dr. Hongjiang Zhang and Prof. David Dagan Feng, **Fundamentals of Content-Based Image Retrieval**, 2003.

[7] Murari Mohan Sardar, Krishnendu Basuli and Saptarshi Naskar, **CONTENT-BASED IMAGE RETRIEVAL SYSTEM**, University of Calcutta, 2008.

[8] G. Th. Papadopoulos, V. Mezaris, S. Dasiopoulou and I. Kompatsiaris, **Semantic Image Analysis Using a Learning Approach and Spatial Context**, Aristotle University of Thessaloniki, Greece, 2006.

[9] S. Lange, T. Grumme, W. Kluge, K. Ringel and W. Meese, **Cerebral and Spinal Computerized Tomography**, Schering AG, 1989.

[10] Ming-Yang Kao, **Encyclopedia of Algorithms,** © 2008 SpringerScience.

[11] G Henning Muller, Nicolas Michoux, David Bandon and Antoine Geissbuhler, **A Review of Content-Based Image Retrieval Systems in Medical Applications**, University Hospital of Geneva, 2007.

[12] Chia-Hung Wei, Chang-Tsun Li and Roland Wilson, **A Content-Based Approach to Medical Image Database Retrieval**, University of Warwick, 2006.

[13] Dima shaheen, **Image Retrieval: survey of the Current Techniques, Promising Directions, and Open Issues**, Al-Baath University, 2008.

[14] Konstantions N. Plataniotis, **Color Image Processing; Methods and Applications**, Taylor & Francis Group, 2007.

[15] Ming-Yang Kao, **Encyclopedia of Algorithms**, Northwestern University, 2008.

[16] http://en.wikipedia.org/wiki/CBIR

* **MSDN® Library for Microsoft Visual Studio 2005,** ©2005 Microsoft Corporation.

* **MATLAB® R2007a** Help library, The MathWorks, Inc.

Appendix A. (ملخص البحث باللغة العربية)

إن التطور الملحوظ في مجال تقنيات الحاسب الآلي ووسائل حفظ المعلومات المرئية والمسموعة خلال العقدين المنصرمين والذي أدّى بدوره إلى زيادة انتشار سبل مشاركة المعلومات عن طريق الشبكات الداخلية والخارجية؛ كل ذلك كان سبباً لظهور الحاجة إلى طرق جديدة للبحث الدقيق بشكل فعّال ضمن قواعد البيانات الكبيرة. و لم يكن مجال البحث عن الصور بشتى أنواعها وأحجامها مختلفاً عن غيره، بل وإنه أصبح مجالاً هاماً جداً للبحث والتطوير خصوصاً بعد ظهور شبكة الانترنت في بداية التسعينات والتي ساهمت في توسّع استخدامات الصور وبالتالي زيادة الحاجة إلى حفظها بشكل منظم يسمح للباحثين بالحصول على أفضل النتائج.

كانت عمليات البحث عن الصور في بادئ الأمر تعتمد بشكل أساسي على المعلومات الأساسية التي تصف هذه الصور سواءً بشكل مباشر كحفظ اسم الصورة أو شرح لما تمثّله ضمن قواعد بيانات الصور، أو بشكل غير مباشر عن طريق ربط الصورة بما يدل عليها كربطها بنص أو خبر ما، أو حتى بصورة أخرى كما هو الحال في المواقع الالكترونية. لكن الحاجة إلى البحث بشكل أكثر تفصيلاً يصف المحتوى الفعلي للصور كان العامل الأساسي لظهور تقنية "البحث الصوري عن طريق المحتوى" أو ما يسمى Content-Based Image Retrieval.

عند مقارنة تقنية البحث الصوري عن طريق المحتوى بالطرق القديمة للبحث والتي تعتمد على المعلومات التي تصف الصور والتي تسمى "البحث الصوري عن طريق الوصف"، نجد فرقاً واضحاً في مجال التطبيق وبالتالي في جودة النتائج التي يمكن الحصول عليها. حيث أن البحث الوصفي على سبيل المثال يعتمد على الكلمات المفتاحية وبالتالي قد يلعب العامل البشري خطأ كبيراً في عملية البحث عند إساءة وصف محتوى الصورة. وبالمثل نجد الحاجة الماسة لوجود طريقة للبحث عن الخائص المرئية التي تصف محتوى الصور غير مجدية لمثل هذا النوع من البحث.

إذن، كيف تتم عملية البحث عن طريق المحتوى وكيف يمكن الاستفادة من قدرات الحاسب على معالجة الصور في عمليات البحث؟ إن عملية البحث عن طريق المحتوى تعتمد بشكل أساسي على تحليل الصفات المرئية لمحتويات الصور كاللون (color) والشكل (shape) والخامات (texture) أو غيرها من الصفات التي تصف البنية الداخلية للصور، وباستخدام خورزميات رياضية لمقارنة نسبة التقارب بالنسبة للأشعة التي تصف هذه الصفات يتم تحديد درجة التشابه بين الصور. وبشكل عام، يمكن تطبيق هذا النوع من البحث بطريقتين أساسيتن:

- *الطلب باستخدام مثال (Query by example):*

ويتم هذا النوع من البحث عن طريق إدخال صورة كمثال، أو رسم شكل تقريبي للشكل المراد البحث عنه أو عن طريق تحديد مجال البحث المراد بالنسبة للصفة المرئية؛ كالطلب عن طريق درجات لونية محددة.

- *الطلب باستخدام الإملاء (Query by semantic):*

وهو الطلب باستخدام وصف للمحتوى المراد البحث عنه حيث أن نظام البحث يقوم بربط هذا الوصف بالصفات المرئية المقابلة له ثم تحديد نسبة التشابة بالنسبة لجمل الصور الموجودة ضمن النظام.

ويمكن استخدام كلتا الطريقتين للبحث الصوري في العديد من المجالات المختلفة كمطابقة البصمات أو الوجوه وغيرها من الاستخدامات الأمنية، أو في المجالات الطبية والعلمية والتعليمية. ولقد جاء الهدف من هذا المشروع انطلاقاً من أهمية البحث عن طريق المحتوى والمكانة العلمية التي اكتسبها في مجال البحوث العلمية في الوقت الحاضر.

حاولنا في بداية هذا المشروع عرض نبذة تاريخة عن البحث الصوري بشكل عام، ومن ثم تم عرض الصفات العامة التي تميز البحث الصوري عن طريق المحتوى ومجالات استخدامه وطرق تنفيذ هذه العملية وذكر بعض العقبات التي تواجه الباحثين في هذا المجال. وبعد ذلك تم تحليل النظام العام للبحث الصوري والبنية الداخلية التي يتكون منها، وبعض أساليب استخراج القيم الحسابية للصفات المرئية للصور وطرق فهرستها ضمن قواعد البيانات مع ذكر لأهم الخوارزميات المستخدمة لمقارنة الصور. وكان لابد من خلال المشروع أن يتم شرح تصميم النظام من الناحية التقنية وطريقة تنفيذ البحث وبنية قواعد المعطيات اللازمة لتنفيذ البحث.

وجاء القسم الثاني من المشروع ليتناول تطبيق طبي كمثال لاستخدام نظام البحث الصوري عن طريق المحتوى. قمنا من خلال هذا التطبيق بتنفيذ برنامج لمطابقة الصور الدماغية والبحث عنها ضمن قاعدة بيانات مكونة من مجموعة كبيرة من الصور بتقنية "الطبقي المحوري" ومحاولة تقديم مساعدة لمعرفة بعض الأمراض التي قد يتعرض لها الدماغ لدعم القرارات الطبية أو التشخيص العلمي لها. لذلك كان لابد من ذكر الصفات التي تميز هذه الصور الطبية وكيفية استخدامها ضمن النظام البحثي. وفي ختام المشروع كان من المهم ذكر بعض الدراسات والاختبارات التي تمت على النظام وكيفية استغلال بعض النقاط أو الميزات لزيادة أداءه، إضافة لبعض المقترحات للتطوير المستقبي على المشروع.

- تم بعون الله -

Appendix B. (UML Models)

Through this appendix we will use the international standards in modeling the system; these standards are the **Unified Modeling Language - UML standards** for modeling diagrams.

*Why choosing **UML** standards?*
An important benefit of the **UML** is that it is consistent, and it is part of common modeling elements across different diagrams, in addition to considering it as a good start because it defines the notation and semantics for common object-oriented models.

B.1 Use-Case Modeling

Use case modeling is a form of requirements engineering. It shows the relationships among actors and use-cases within a system. They are often used to:

- **Provide an overview of all or part of the usage requirements for a system or organization in the form of an essential model or a business model.**
- **Communicate the scope of a development project.**
- **Model the analysis of usage requirements in the form of a system use-case model.**

The use-case model provides a prime source for objects and classes. It is the primary input to class modeling.

B.1.1. Use-Case Diagram

Use-case diagram differs from what we might call the "traditional" way of requirements models, which comprising functional and non-functional requirements. Use-case modeling is a different and complementary way of eliciting and documenting requirements. There are four components of this model:

- ***System boundary*:** a box drawn around the use cases to denote the edge or boundary of the system, subsystem, being modeled.
- ***Actors*:** roles played by people or things that use system.
- ***Use cases*:** things that the actors can do with the system.
- ***Relationship*:** meaningful relationships between actors and use cases.

Here is the use-case diagram modeling what this system should do and how these operations mentioned before are done. Firstly, the system is divided into four main subsystems which are:

1. **Search Subsystem.**
2. **Image Processing Component.**
3. **Database Controller.**
4. **System Controller.**

Figure B.1 shows the general use-case diagram of the system showing these subsystems.

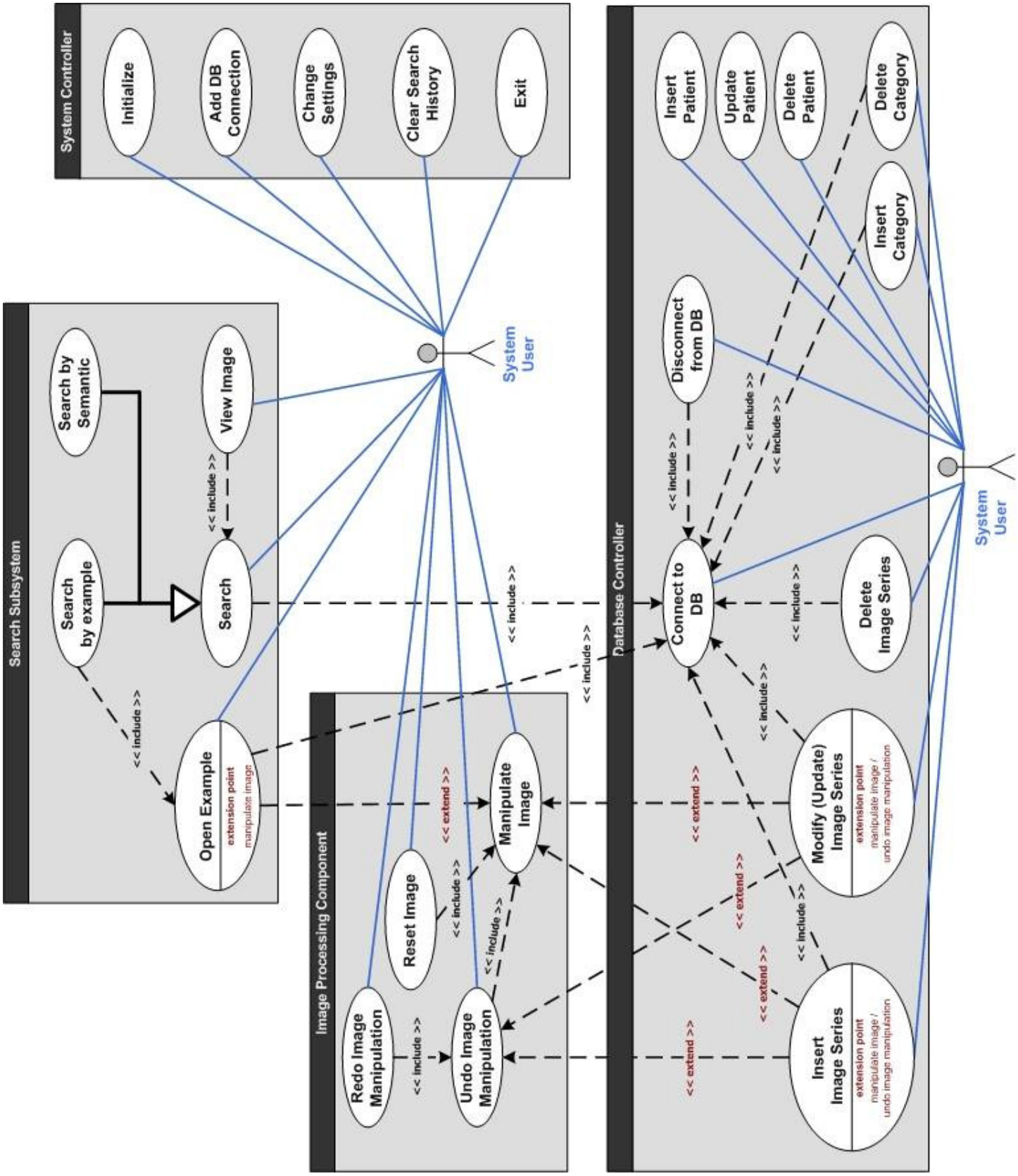


Figure. B.1. General Use-Case Diagram
(Larger copy of this diagram was attached with this document)

B.1.2. Use-Case Specification (narratives)

The use-case narratives give a general understanding of the events and magnitude of the system. This specification starts with documenting the use-cases at a high level, then returns to each use-case and expands it to a fully documented business requirement narrative.

Now, each use-case displayed in the diagram in the previously section is detailed by the narratives (following tables) where a full description of what that use-case should do, the scenario (the main flow), the alternative flow and who is/are the primary and secondary actors involved in that use-case.

1. Search Subsystem:

Use Case :	Search
ID :	1.1
Brief Description :	Finds an image that matches a certain criteria.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user selects search criteria. 2. The user enters the criterion and clicks Search. 3. The system performs measuring on the entered criterion. 4. The system gets the matching images from the DB and performs ranking on retrieved images. 5. The system views results.
Alternatives :	1. (4) There is no matching images 2. (5) The system asks the users to refine their query

Use Case :	Search by Example
ID :	1.2
Parent ID :	1
Brief Description :	Finds an image that matches an example image.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Open Example). 2. (o1) The user selects "**search by example**" method. 3. (o2) The user enters an example image and clicks Search. 4. (o3) The system performs a feature extraction on the entered example. 5. (4) The system gets the matching images from the DB and performs ranking on retrieved images. 6. (5) The system views results.
Alternatives :	1. (4) There is no matching images 2. (5) The system asks the users to refine their query

Use Case :	Search by Semantic
ID :	1.3
Parent ID :	1
Brief Description :	Finds an image that matches a semantic query.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. (o1) The user selects "**search semantically**" method. 2. (o2) The user enters the query and clicks Search. 3. (o3) The system performs an "**Expression Evaluation**" on the entered query. 4. (4) The system gets the matching images from the DB and performs ranking on retrieved images. 5. (5) The system views results. 6. The system inserts the query in the **search history file**.
Alternatives :	3. (4) There is no matching images 4. (5) The system asks the users to refine their query

Use Case :	Open Example
ID :	1.4
Brief Description :	Opens an example for the (**example search**).
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user clicks open example. 2. The system views open file dialog. 3. The user selects an image file. 4. The system views the image on the preview board. **Extension point: Manipulate Image**
Alternatives :	None.

Use Case :	View Image
ID :	1.5
Brief Description :	Views an image from results.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Search). 2. The user double-clicks an image on the results list. 3. The system views the selected image on separated form.
Alternatives :	None

2. Image Processing Component:

Use Case :	Manipulate Image
ID :	2.1
Brief Description :	Makes processing operation on an image.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user clicks manipulate image. 2. The system views the image in **Edition Mode** in a separate form and saves a temporary copy of the original image. 3. While the user is editing the image: a. The user selects a tool and uses it on the image. b. The system shows its effect in real-time. 4. The user asks to close **Edition Mode**. 5. The system asks if the user wants to save changes made to the image. 6. The user clicks YES. 7. The system saves the image.
Alternatives :	1. (6) The user clicks no. 2. (7) The system closes with out saving and recovers the temporary copy of the image.

Use Case :	Undo Image Manipulation
ID :	2.2
Brief Description :	Undo processing operation done on an image.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Manipulate Image). 2. The user clicks undo image manipulation. 3. The system gets the temp copy of the image from the temp directory. 4. The system views the image. 5. The system updates the **REDO** list.
Alternatives :	None

Use Case :	Redo Image Manipulation
ID :	2.3
Brief Description :	Redoes last undone processing operation on an image.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Undo Image Manipulation). 2. The user clicks redo undone image manipulation. 3. The system gets the temp copy of the image from the temp directory. 4. The system views the image. 5. The system updates the **UNDO** list.
Alternatives :	None

Use Case :	Reset Image
ID :	2.4
Brief Description :	Undoes all processing operations done on an image.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Manipulate Image). 2. The user clicks reset done processing on image. 3. The system gets the first temp copy of the image from the temp directory. 4. The system views the image. 5. The system clears both the **UNDO** list and **REDO** list.
Alternatives :	None

3. Database Controller:

Use Case :	Connect to DB
ID :	3.1
Brief Description :	Connects the system to the selected database.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user selects a database from a list and clicks **Connect**. 2. The system gets its connection string and connects to it. 3. The system shows an indicator on the status bar indicating that the system is connected.
Alternatives :	1. (1) If the list of databases is empty the system asks the user to add a database. 2. (2) If the connection failed for any reason the system pops up a message clarifying the reason. 3. (3) If (Alt-1) or (Alt-2) is true the system shows an indicator on the status bar indicating that the system is disconnected.

Use Case :	Insert Patient
ID :	3.2
Brief Description :	Inserts new patient into the database
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Connect to DB). 2. The user clicks **Insert Patient**. 3. The system shows a separate form in which user can insert patient to database. 4. The user enters the required patient information and clicks **Insert**. 5. The system inserts the patient into the database.
Alternatives :	1. (4) The user inserts wrong or incomplete information. 2. (5) The system shows a message box indicting the error.

Use Case :	Update Patient
ID :	3.3
Brief Description :	Updates patient information in the database.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Connect to DB). 2. The user clicks **Update Patient**. 3. The system shows a separate form in which user can select a patient to update. 4. The user then selects a patient from a list of patients' names or by writing the first few litters of the patient name to select from a list with limited number of options. 5. The system shows the patient information. 6. The user edits the information they want to update. 7. Then user clicks **Update**. 8. The system asks the user to confirm. 9. The user confirms. 10. The system updates the patient.
Alternatives :	1. (6) If the entered information were wrong or incomplete then the system shows a message box indicates the error.

Use Case :	Delete Patient
ID :	3.4
Brief Description :	Deletes a patient from the database.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Connect to DB). 2. The user clicks **Delete Patient**. 3. The system shows a separate form in which user can select a patient to delete. 4. The user then selects a patient from a list of patients' names or by writing the first few litters of the patient name to select from a list with limited number of options. 5. The system shows the patient information. 6. The user clicks **Delete**. 7. The system asks the user to confirm. 8. The user confirms. 9. The system deletes the patient.
Alternatives :	1. (8) The user does not confirm. 2. (9) The system cancels the operation.

Use Case :	Insert Images Series
ID :	3.5
Brief Description :	Inserts images series to the database
Primary Actors :	User
Secondary Actors :	None

Main Flow :	1. Include (Connect to DB). 2. The user clicks **Insert Image Series**. 3. The system shows a separate form in which user can insert image series to database. 4. The user selects a patient to associate with the series. 5. The user checks the categories to be associated to the series. 6. The user opens an image file(s) to insert. 7. The system views the images in the preview picture list. ***For selected image:*** Extension Point: Manipulate Image. Extension Point: Undo Image Manipulation. 8. The user clicks **Insert**. 9. The system **extracts features** for each image in the picture list and inserts it. 10. The system then performs indexing and filtering on the inserted image(s). 11. The system stores the indexed feature vector into the feature table. 12. Finally the system stores the manipulated images on the disk and stores images' **URLs** and other images information into the database also it inserts the series information into database.
Alternatives :	1. (4) If no patient is selected. 2. (6) If no images is loaded. 3. A message box is shown describes the error

Use Case :	Modify (Update) Images Series
ID :	3.6
Brief Description :	Modifies images series in the database
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Connect to DB). 2. The user clicks **Modify Image Series**. 3. The system shows a separate form in which user can modify image series to database. 4. The user selects a patient from the list of patients' names. 5. The system then fills the list of series associated with selected patient. 6. The system then fills the categories and checks the categories associated with this series 7. The system then fills all images associated with this series in the picture list. 8. The users modify the information they want (Add-Edit-Delete an image *or* update series notes and categories). ***For selected image:*** Extension Point: Manipulate Image. Extension Point: Undo Image Manipulation. 9. The user clicks **Modify**. 10. The system **extracts features** for each changed image in the picture list and updates it in the database. 11. The system then performs indexing and filtering on the updated image(s). 12. The system stores the indexed feature vector into the database. 13. Finally the system stores the manipulated images on the disk and stores images' **URLs** and other images information into the database also it updates the series information into database.
Alternatives :	1. (4) If no patient is selected. 2. (5) If no series is selected. 3. A message box is shown describes the error

Use Case :	Delete Images Series
ID :	3.7
Brief Description :	Deletes images series from the database
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Connect to DB). 2. The user clicks **Delete Image Series**. 3. The system shows a separate form in which user can delete image series to database. 4. The user selects a patient from the list of patients' names. 5. The system then fills the list of series associated with selected patient. 6. The system then fills the categories associated with this series 7. The system then fills all images associated with this series in the picture list. 8. The user clicks **Delete**. 9. The system asks the user to confirm. 10. The user confirms. 11. The system asks the users if they want to delete the image file in addition to image record. 12. The user clicks YES. 13. The system deletes the file from the disk and the record from the database.
Alternatives :	1. (10) The user does not confirm the deletion. 2. (11) The system cancels the operation. 3. (12) The user clicks NO. 4. (13) The system deletes the record only and keeps the image file.

Use Case :	Insert Category
ID :	3.8
Brief Description :	Inserts an image category.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Connect to DB). 2. The user clicks **Insert Category**. 3. The system shows a separate form in which user can insert category to database. 4. The user enters the required category information and clicks **Insert**. 5. The system inserts the category into the database.
Alternatives :	3. (4) The user inserts wrong or incomplete information. 4. (5) The system shows a message box indicting the error.

Use Case :	Delete Category
ID :	3.9
Brief Description :	Deletes an image category from the database.
Primary Actors :	User
Secondary Actors :	None

Main Flow :	1. Include (Connect to DB). 2. The user clicks **Delete Category**. 3. The system shows a separate form in which user can select an image category to delete. 4. The user then selects a category from a list of categories' names or by writing the first few litters of the category name to select from a list with limited number of options. 5. The system shows the category information. 6. The user clicks **Delete**. 7. The system asks the user to confirm. 8. The user confirms. 9. The system deletes the category.
Alternatives :	3. (8) The user does not confirm. 4. (9) The system cancels the operation.

Use Case :	Disconnect From DB
ID :	3.10
Brief Description :	Disconnects the connected database.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. Include (Connect to DB). 2. The user clicks **Disconnect from Database**. 3. The system closes the connected database connection. 4. The system shows an indicator on the status bar indicating that the system is disconnected
Alternatives :	1. (3) If the system failed to close the connection for any reason the system pops up a message clarifying the reason. 2. (4)The system shows an indicator on the status bar indicating that the system is **still connected**.

4. System Controller:

Use Case :	Initialize
ID :	4.1
Brief Description :	Initializes the system (This operation may include login task).
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user starts the system. 2. The system loads system settings from **settings file**. 3. The system loads Connection Strings List from **connection strings file**. 4. The system loads search history from **history file**. 5. The system connects to the **default database**. 6. Finally the system shows the main window.
Alternatives :	1. (2-3-4-5) the system failed to load any of the components. 2. (6) The system shows a message box indicating the error and where it happened and then the **system exits**.

Use Case :	Add DB Connection
ID :	4.2
Brief Description :	Adds new connection string to the system.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user clicks **Add Database Connection**. 2. The system shows a separate form in which user can add connection string information. 3. The user inserts required information. 4. The system checks the **connectivity** of the inserted connection string. 5. The system inserts the connection string into the **file** of connection strings.
Alternatives :	1. (4) If the connection string is not valid then the system shows a message box clarifying the reason. 2. (5) The insertion of the connection string in the file failed for any reason the system shows a message box clarifying the reason.

Use Case :	Change Settings
ID :	4.3
Brief Description :	Changes system settings
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user clicks **Settings**. 2. The system shows the current system settings. 3. The users change the settings they want to change and click **Apply**. 4. The system checks the validity of the changed settings. 5. The system saves the new settings in the **settings file**.
Alternatives :	1. (4) The changed setting are invalid 2. (5) The system shows a message indicating the error(s).

Use Case :	Clear Search History
ID :	4.4
Brief Description :	Clears search history of the semantic queries.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user clicks **Clear History**. 2. The system deletes the contents of the history file. 3. The system also clears the list in which history is listed in the run-time.
Alternatives :	1. (2) The deletion from the file failed for any reason the system pops up a message clarifying the reason.

Use Case :	Exit
ID :	4.5
Brief Description :	Quits the system.
Primary Actors :	User
Secondary Actors :	None
Main Flow :	1. The user clicks **Exit**. 2. The system asks the user to confirm. 3. The user confirms. 4. The system ends all the current operations and closes any opened connection to database and clears all temp files and exits the system.
Alternatives :	1. (3) If the user does not confirm the system will cancel the operation. 2. (4) If any operation of pre-exit operations failed to be done then the system shows a message indicating the error.

B.2. Class Model

This model focuses on ***what*** the system needs to do, but leaves the details of ***how to*** do this to the design workflow in diagrams such as (DFDs) and the sequence diagrams.

As a definition, a class is a specification or template that all objects, data and functions, of that class (instances) must follow. Each object of the class has specific values for the attributes defined by the class and will respond to messages by invoking the operations defined by the class, and it can interact with other classes by relationships between them. In other words, all objects of the same class must have the same set of operations, the same set of attributes, and the same set of relationships, but may have different attribute values.

We can summarize the functions of this model as:

- **Modulate the domain concepts.**
- **Analyze requirements in the form of a conceptual/analysis model.**
- **Depict the detailed design of object-oriented or object-based software.**

Class Diagram

After analyzing the requirements shown in section 4.1 of this document, we found that the following classes are required for the operation of the system.
The following are two diagrams (Figure. B.2, Figure. B.3) showing the classes of the system. The first one is the *analysis class diagram* which shows the needed classes to build the system without describing their structure. The other diagram is the *design class diagram* that illustrates the design details of the classes including the returned data types of each method with its parameters, in addition to the internal fields and other information about these classes. Note that these diagrams were generated automatically using *Microsoft Visual Studio© 2005 Professional Edition*.

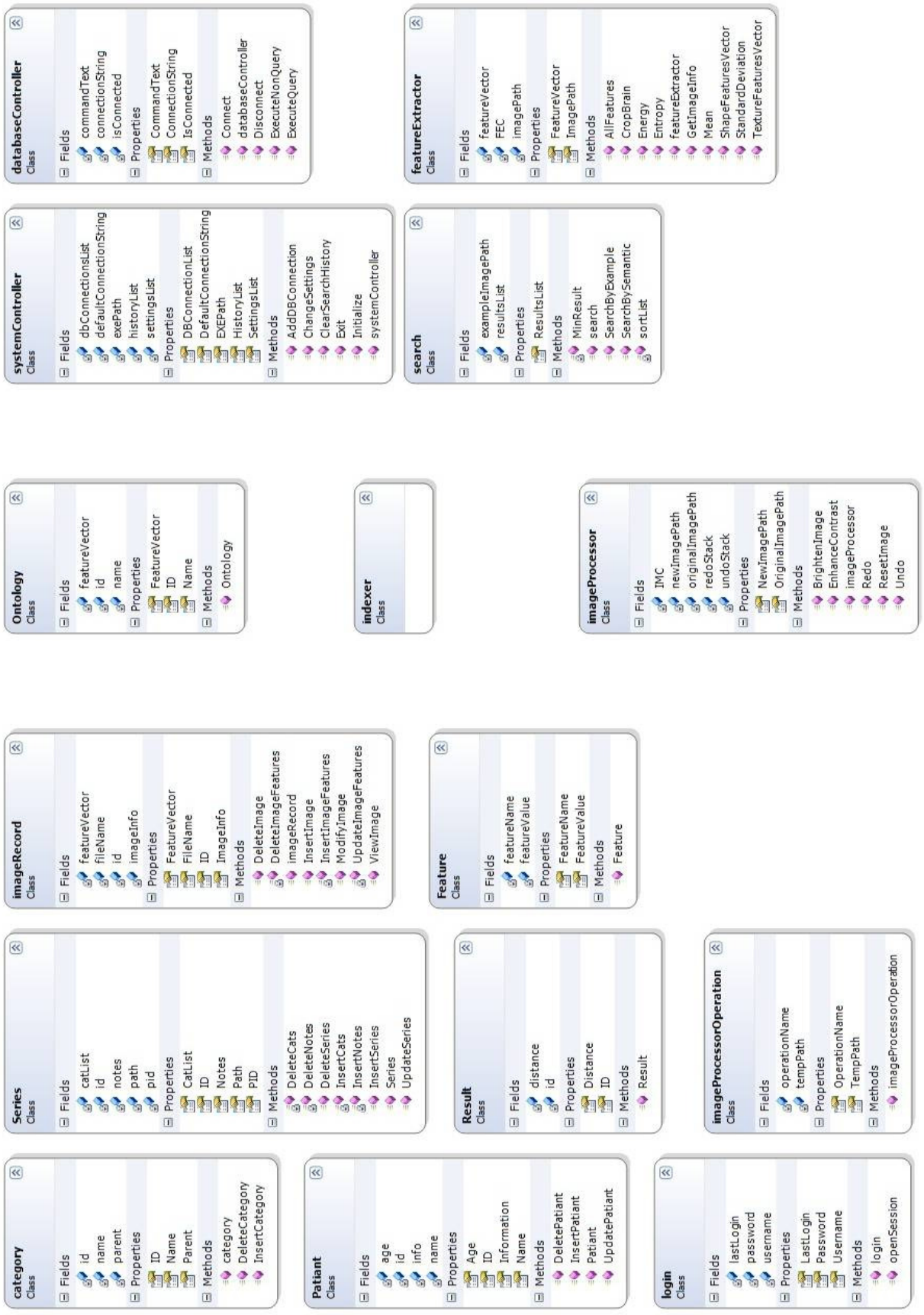


Figure. B.2. Analysis Class Diagram
(Larger copy of this diagram was attached with this document)

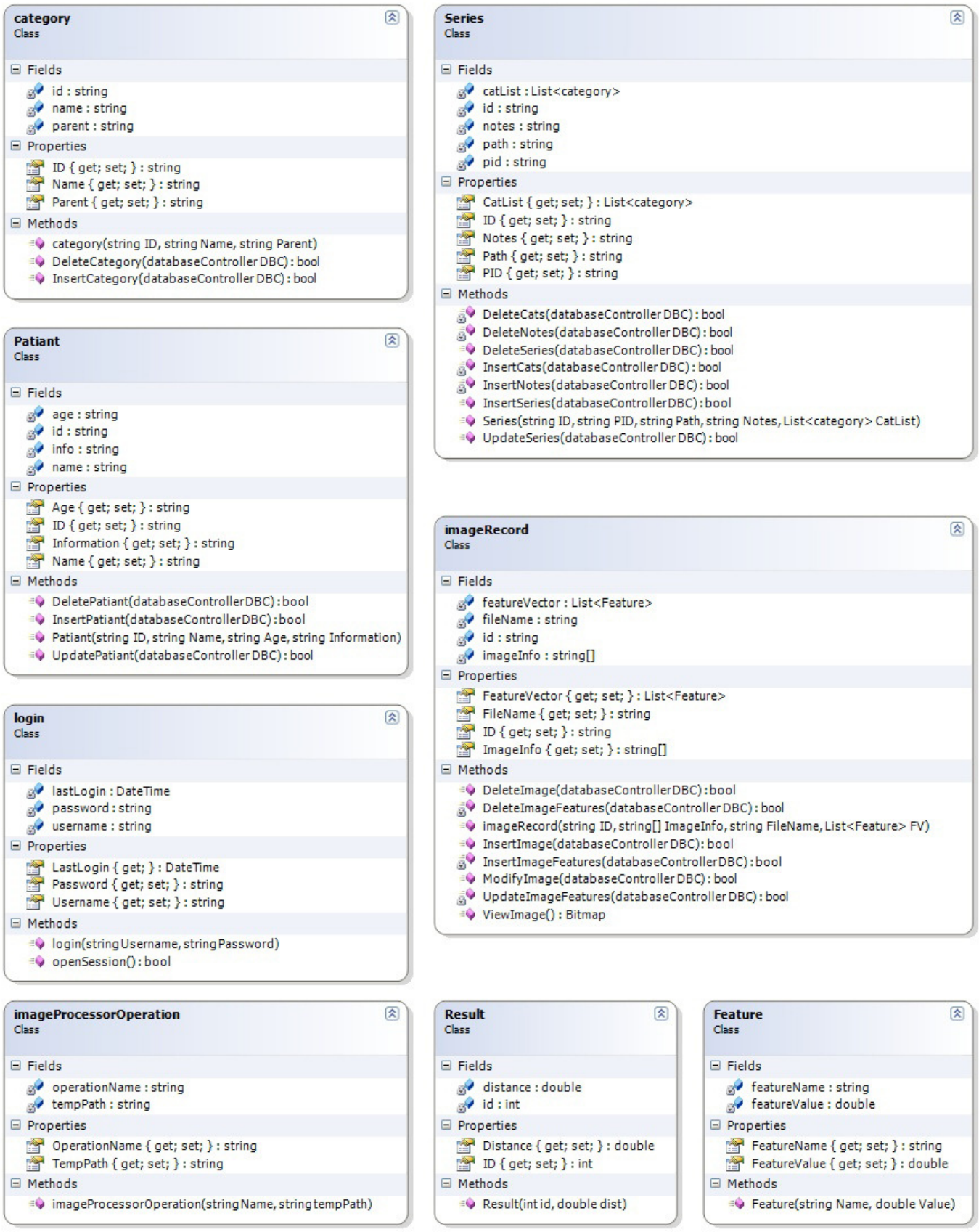


Figure. B.3. Design Class Diagram
(Larger copy of this diagram was attached with this document)

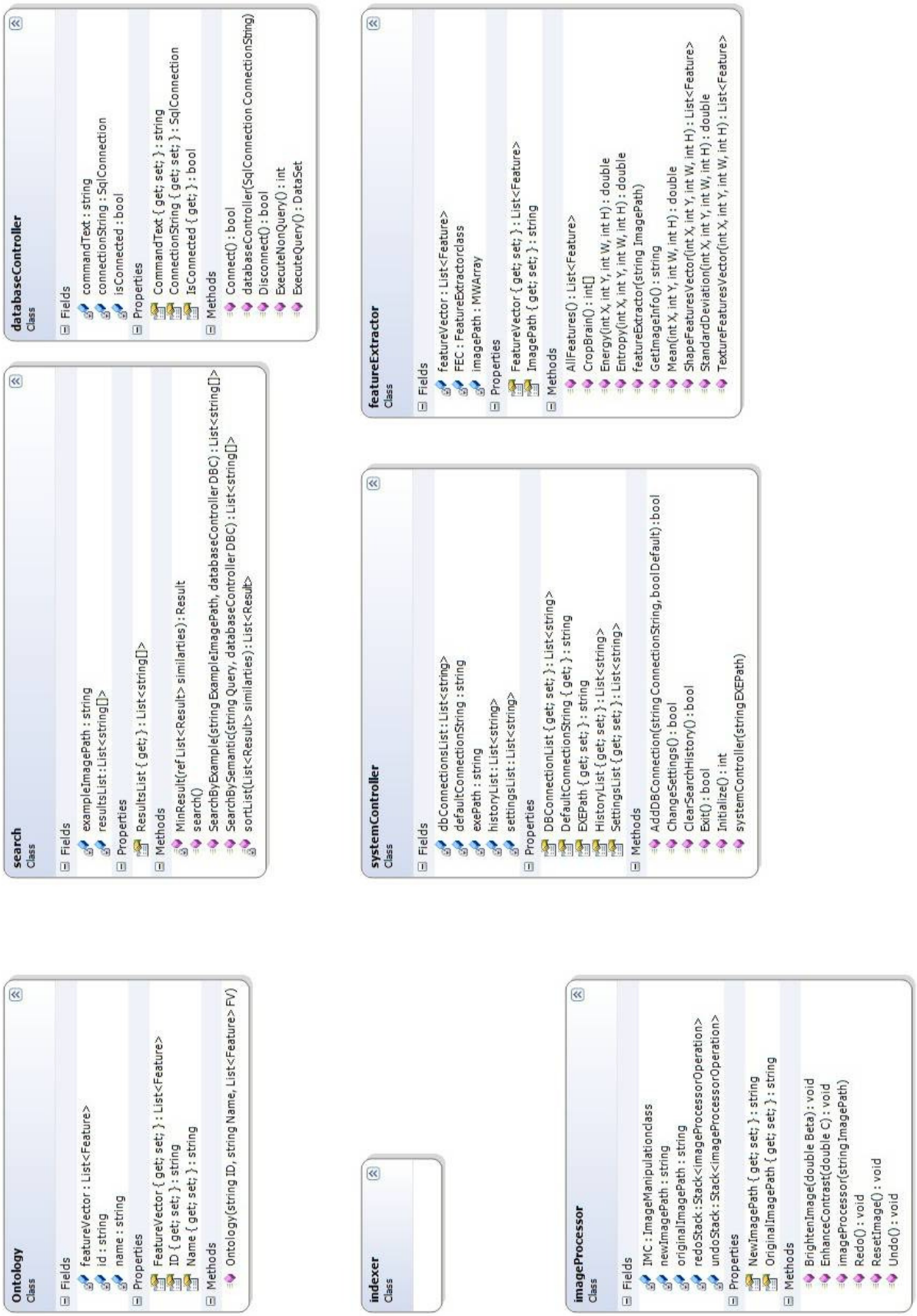


Figure. B.3. Design Class Diagram
(Larger copy of this diagram was attached with this document)

The following table describes in general each class:

SN	Class	Summary
1	Category	Represents series category in the database
2	databaseController	Controls the link between the system and the database
3	Feature	Represents an image feature
4	FeatureExtractor	Interfaces between the system and the external processing libraries for features extraction purpose
5	imageProcessingOperation	Represents an image processing operation
6	imageProcessor	Interfaces between the system and the external graphical libraries for image processing purposes
7	imageRecord	Represents image record in the database
8	Indexer	Holds indexing and filtering operations
9	Login	Represents login agent for the system
10	Ontology	Represents a semantic ontology
11	Patient	Represents patient record in the database
12	Result	Represents a search result
13	Search	Processing core of the system holds the main operations
14	Series	Represents series of images in the database
15	SystemController	Loads the system and manages its settings

Appendix C. (Delivery List)

SN	Item	Comment
1	Report	Project documentation
2	SourceCode.rar	The source of the project
3	XCBIR.exe	The executable of the project
4	FeatureExtractor.dll	Feature extraction module
5	ImageManipulation.dll	Image editing tools module
6	MWArray.dll	MathWorks® Array Data Type objects
7	PureComponents.ActionSet.dll	GUI libraries Action Set
8	PureComponents.EntrySet.dll	GUI libraries Entry Set
9	settings.lst	File contains system settings
10	history.lst	File contains search history
11	DBConnections.lst	Contains application database connection strings
12	MathWorks® Framework	The framework which will execute MATLAB codes
13	.Net Framework	.Net Libraries
14	Help.rar	User's Manual

Appendix D. (CODE)

Image Insertion Process

```
//Check if the series information are filled well . . .
if ((rbnCboPatiant.SelectedItem != "Select Patiant") &&
(picLstSerImage.Items.Count > 0) && (rbnCboPatiant.SelectedItem != null))
{
//Preparing the progress bar . . .
int xx = picLstSerImage.Items.Count;
xx = 100 / xx;
proBarOpr.Visible = true;
proBarOpr.BringToFront();
proBarOpr.Value = 0;
//Create the physical path where the image and the series is contained
int sid = int.Parse(rbnTxtSeriesID.Text);
string serFolder = SC.SettingsList[0].TrimEnd('\\') + "\\" + sid.ToString();
Directory.CreateDirectory(serFolder);
sid = sid * 100;
int imgID = 0;
//Extract image features for each image in the series :
foreach (PictureListItem P in picLstSerImage.Items)
{
Application.DoEvents();
fe = new featureExtractor(P.Tag.ToString());
fv = fe.AllFeatures();
//Get all image information :
string[] info = fe.GetImageInfo().Split('\n');
for (int i = 0; i < info.Length; i++)
{
    info[i] = info[i].Trim();
}
//Get image size and Dimensions
string size = "", h = "", w = "";
for (int i = 0; i < info.Length; i++)
{
    if (info[i].StartsWith("FileSize"))
    {
        size = info[i];
    }
    else if (info[i].StartsWith("Width"))
    {
        w = info[i];
    }
    else if (info[i].StartsWith("Height"))
    {
        h = info[i];
    }
}
info = new string[2];
info[0] = size.Split(':')[1].Trim();        //Image size . . .

info[1] = h.Split(':')[1].Trim() + "X"
        + w.Split(':')[1].Trim();           //Image Dimensions . . .
imgID++;                                    //Image ID in the series
int ID = sid + imgID;                       //Image ID in the database . . .
```

```
//Create image record . . .
ImageRecord = new imageRecord(ID.ToString(), info, P.Text, fv);
// . . . and insert that record in the opened database
if (!ImageRecord.InsertImage(DBC))//If insertion is not done
{
    throw new Exception("Failed to insert the image record with ID: " + ID.ToString() +
"!");//Throw an exception
}
//Copy image file from its location to where each image in the database is stored
File.Copy(P.Tag.ToString(), serFolder + "\\" + P.Text, true);
Application.DoEvents();
proBarOpr.Value += xx;
}
sid = sid / 100;
//Determine series categories . . .
List<category> catList = new List<category>();
for (int i = 0;
i < mListCats.Items.Count; i++)
{
if (mListCats.Items[i].CheckState == CheckState.Checked)
{
    catList.Add(new category(mListCats.Items[i].Description, null, null));
}
}
//Create series record . . .
SeriesRecord = new Series(sid.ToString()
                    , rbnCboPatiant.SelectedItem.Split('\t')[0]
                    , sid.ToString()
                    , this.rbnTxtNotes.Text, catList);
//Try to insert the series . . .
if (!SeriesRecord.InsertSeries(DBC))    //In insertion is not done
{
throw new Exception("Failed to insert this series !");  //Throw an exception
}
proBarOpr.Value = 100;
proBarOpr.SendToBack();
proBarOpr.Visible = false;
this.DialogResult = DialogResult.OK;
this.Close();//Close when done
}
else
{
throw new Exception("Please fill all the fields !");
}
```

Image Retrieval Process

```
List<string[]> res = new List<string[]>();
//Get the features of the example image
featureExtractor fe = new featureExtractor(ExampleImagePath);
List<Feature> fv = fe.AllFeatures();
DBC.CommandText = "Select * From MultiplyFactors";
DataSet mf_ds = DBC.ExecuteQuery();
double[] exF = new double[fv.Count];
```

```
for (int i = 0; i < fv.Count; i++)
{
exF[i] = fv[i].FeatureValue *
double.Parse(mf_ds.Tables[0].Rows[0].ItemArray[i].ToString());
if (exF[i].Equals(double.NaN))
{
    exF[i] = 0;
}
}
double area = exF[exF.Length - 1] * exF[exF.Length - 2];
double max_area = area + (area * 0.15);
double min_area = area - (area * 0.15);
double mean = exF[0];
double max_mean = mean + (8 *
double.Parse(mf_ds.Tables[0].Rows[0].ItemArray[0].ToString()));
double min_mean = mean - (8 *
double.Parse(mf_ds.Tables[0].Rows[0].ItemArray[0].ToString()));
//Calculate similarities between example image and images in the database
FeatureExtractorclass fec = new FeatureExtractorclass();
MWArray mwA = new MWNumericArray(exF.Length, 1, exF);
//Get the features of all images in the database that match the condition of area . .
.!
string filter1 = "SELECT * FROM NormalizedFeatures Where (CropW*CropH > " +
min_area.ToString() + ") AND ";
filter1 += "(CropW*CropH < " + max_area.ToString() + ") ";
DBC.CommandText = "Select * From (" + filter1 + ") as AreaFilter ";
DBC.CommandText += "Where (( Mean > " + min_mean.ToString() + " ) AND ( Mean < " +
max_mean.ToString() + " )) ;";
DataSet ds = DBC.ExecuteQuery();
//If no images match the conditions of area then bring all images with out constraints
if (ds.Tables[0].Rows.Count < 1)
{
DBC.CommandText = "SELECT * FROM NormalizedFeatures";
ds = DBC.ExecuteQuery();
if(ds.Tables[0].Rows.Count<1)
    throw new Exception("The database is empty or there are no matches!");
}
List<Result> similarties = new List<Result>();
object[] ftable = new object[ds.Tables[0].Rows.Count];  //Array of rows in feature
vector table.
double[] frow;                                          //Array of features in the
current row.
MWArray mwRes = new MWNumericArray(1);                  //The current similarity
for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
{
frow = new double[ds.Tables[0].Rows[i].ItemArray.Length - 1];
for (int j = 1; j < ds.Tables[0].Rows[i].ItemArray.Length; j++)
{
    frow[j - 1] = double.Parse(ds.Tables[0].Rows[i].ItemArray[j].ToString());
}
MWArray mwB = new MWNumericArray(frow.Length, 1, frow);
mwRes = fec.minkowski(mwA, mwB, 5);

int ID = int.Parse(ds.Tables[0].Rows[i].ItemArray[0].ToString());
double Distance = double.Parse(mwRes.ToString());
Result sim = new Result(ID, Distance);
similarties.Add(sim);
ftable[i] = frow;
}
```

```
similarties = sortList(similarties);
string[] resStr;

//Ranking :
double farestImg = (double)similarties[similarties.Count - 1].Distance;   //The worst
match.
double rank = 0;

for (int i = 0; i < similarties.Count; i++)
{
resStr = new string[3];
int x = similarties[i].ID;
//"SELECT I.FileName, S.FolderPath" "FROM Image AS I INNER JOIN Series AS S ON
FLOOR(I.ImageID / 100) = S.SID" "WHERE (I.ImageID = 107)"

//DBC.CommandText = "SELECT FileName,ImageID FROM IMAGE WHERE IMAGEID = " + x + " ;";
DBC.CommandText = "SELECT I.FileName, S.FolderPath ";
DBC.CommandText += "FROM Image AS I INNER JOIN Series AS S ON FLOOR(I.ImageID / 100) =
S.SID ";
DBC.CommandText += "WHERE (I.ImageID = " + x + ")";

ds = DBC.ExecuteQuery();
//rank = (1 - ((double)similarties[i].Distance / farestImg)) * 100;
rank = similarties[i].Distance;
//rank = Math.Round(rank, 5);
resStr[0] = ds.Tables[0].Rows[0].ItemArray[0].ToString();
resStr[1] = ds.Tables[0].Rows[0].ItemArray[1].ToString();
resStr[2] = " - " + rank.ToString(); // +"%";
res.Add(resStr);
resStr = null;
}
return res;
```

Appendix E. (Screenshots)

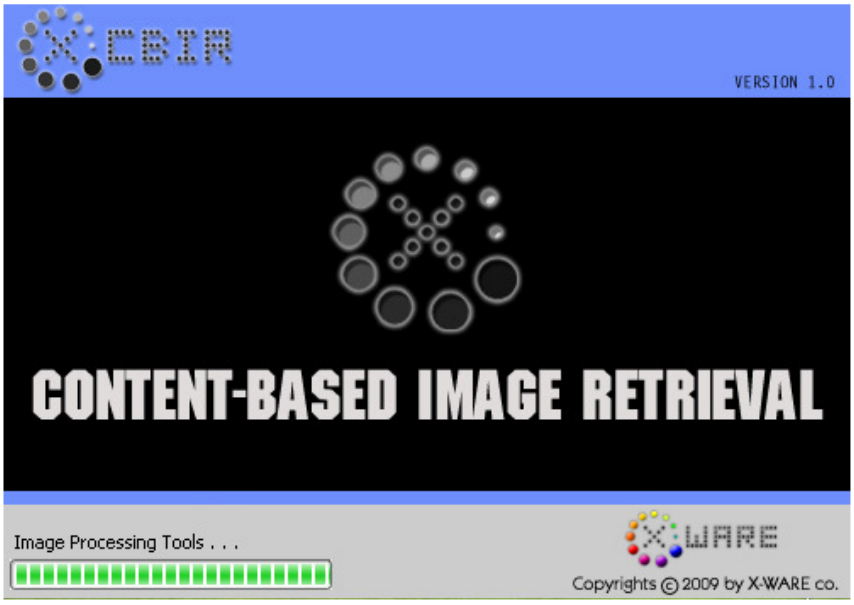


Figure. E.1. System Splash

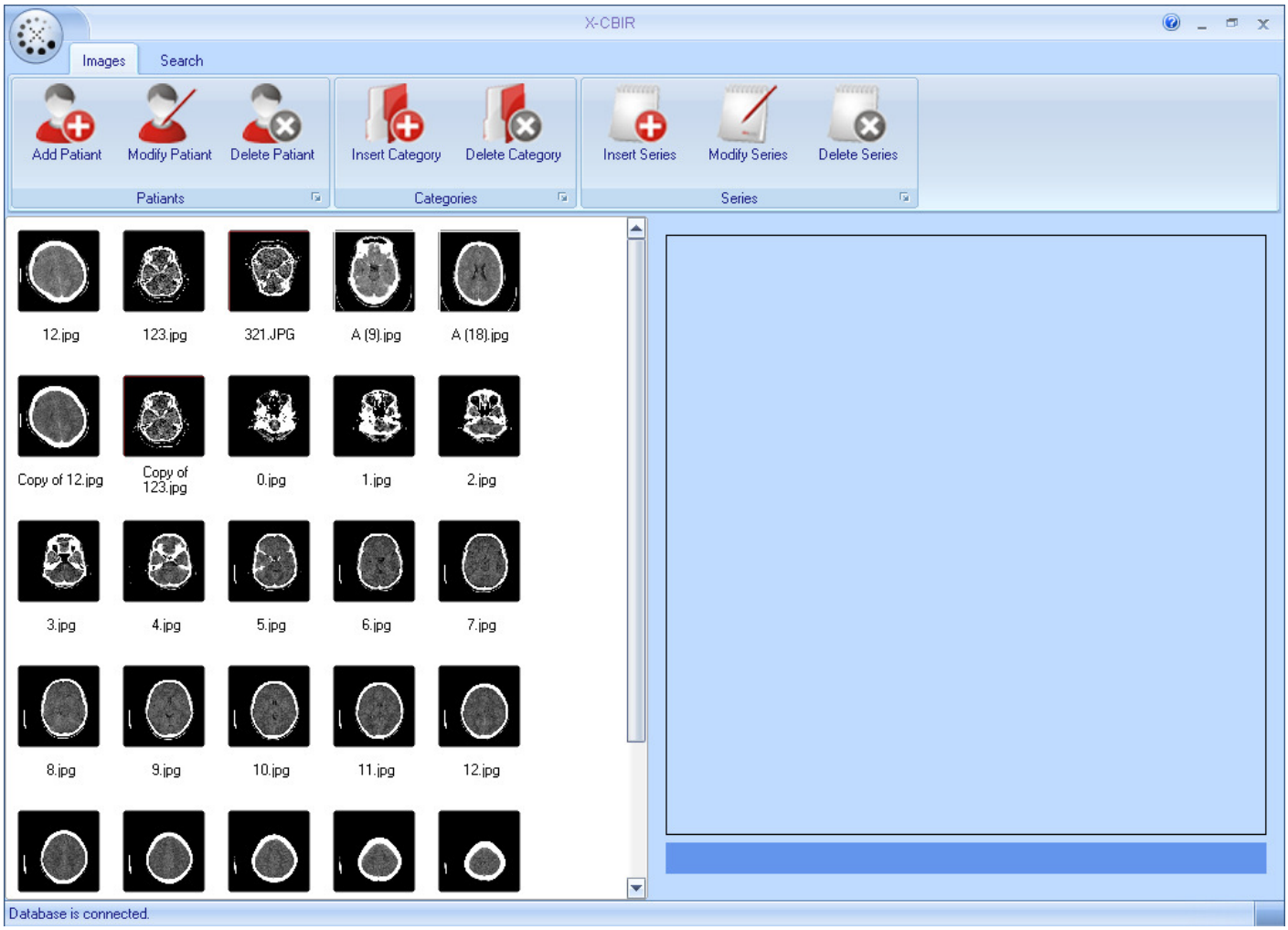


Figure. E.2. System Main Window - Search Tap

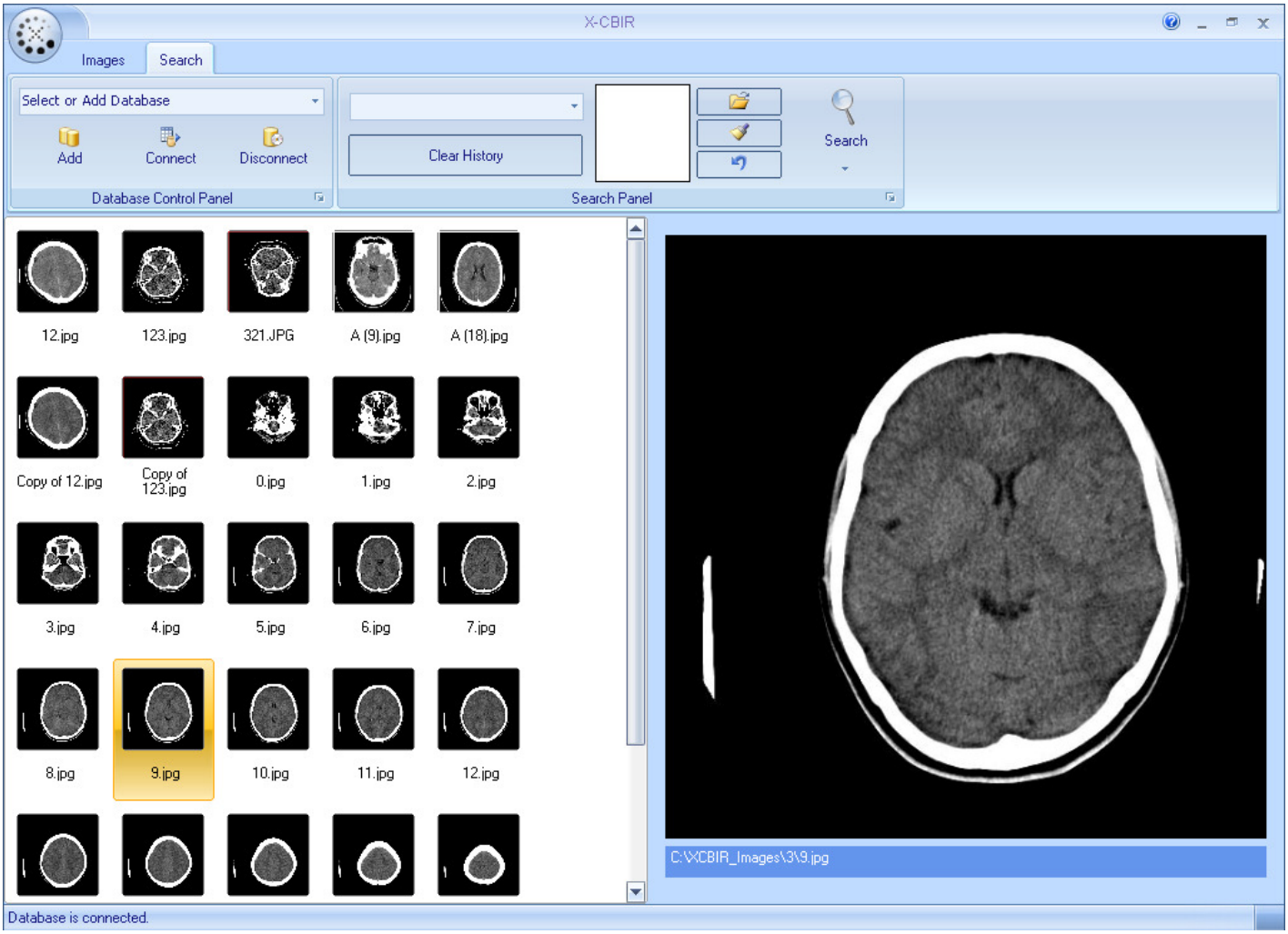


Figure. E.3. System Main Window - Data Manipulation

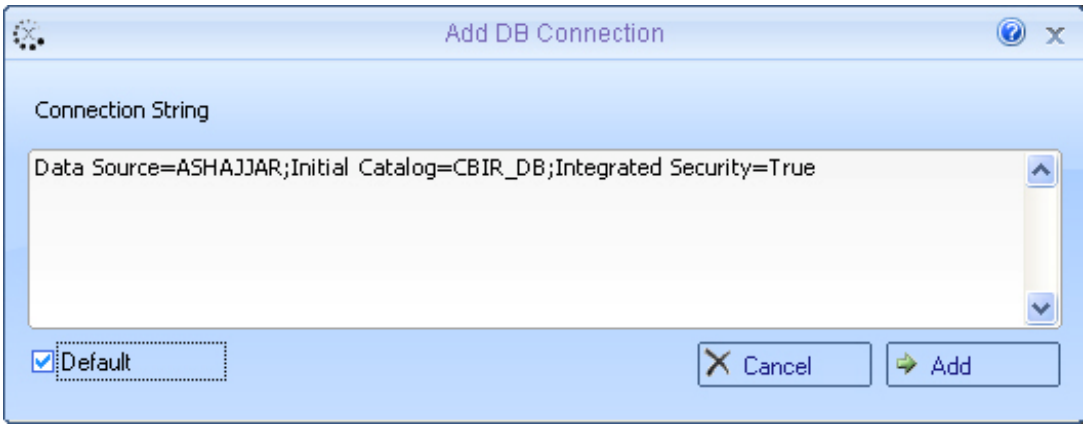


Figure. E.4. Database Connection String From

Figure. E.5. Insert Patient Form

Figure. E.6. Category Control Form

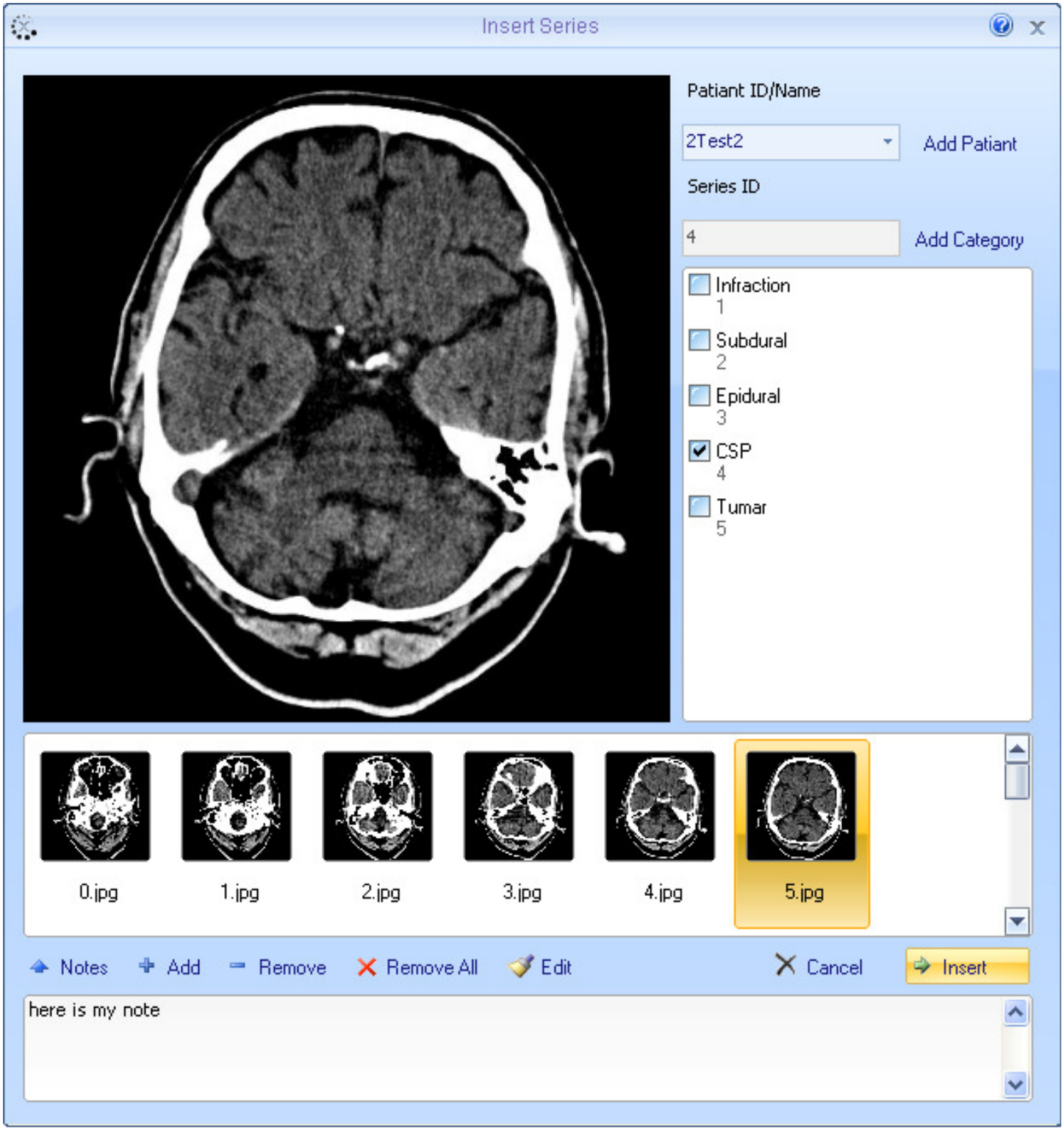


Figure. E.7. Insert Series Form