

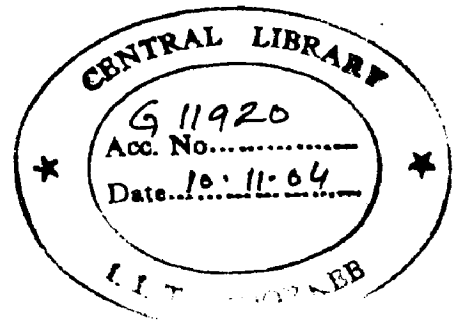
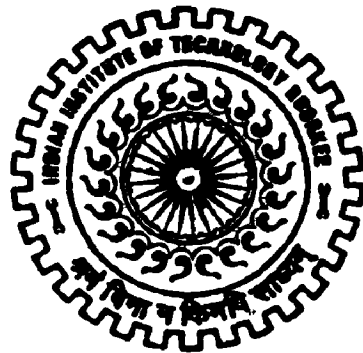
CLUSTERING TECHNIQUES FOR CONTENT BASED IMAGE RETRIEVAL USING MATHEMATICAL MORPHOLOGY

A DISSERTATION

*Submitted in partial fulfillment of the
requirements for the award of the degree
of*
MASTER OF TECHNOLOGY
in
INFORMATION TECHNOLOGY

By

SURESH VARIGANJI



**IIT Roorkee - CDAC, NOIDA,
c-56/1, "Anusandhan Bhawan"
Sector 62, Noida-201 307**

JUNE, 2004

CANDIDATE'S DECLARATION

I hereby declare that the work presented in this dissertation titled "**Clustering Techniques For Content Based Image Retrieval Using Mathematical Morphology**", in partial fulfillment of the requirements for the award of the degree of **Master of Technology in Information Technology**, submitted in **IFT-Roorkee - CDAC campus, NOIDA**, is an authentic record of my own work carried out during the period from June 2003 to June 2004 under the guidance of **Dr.Moinuddin**.Professor, Jamia Millia Islamia (Central University), New Delhi.

I have not submitted the matter embodied in this dissertation for the award of any other degree or diploma.

Date:

Place: NOIDA



(SURESH VARIGANJI)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

Date: 26.6.2004

Place: NOIDA


(Dr. Moinuddin)

Professor

Jamia Millia Islamia (Central University)

New Delhi.

ACKNOWLEDGEMENT

I express my sincere thanks and gratitude to my Guide **Dr. MOINUDDIN, Professor, Jamia Millia Islamia (Central University), New Delhi**, for his inspiring guidance and sustained interest throughout the progress of this dissertation.

I hereby take the privilege to express my deep sense of gratitude to **Prof. PREM VRAT**, Director, Indian Institute of Technology, Roorkee, and **Mr. R.K.VERMA**, Executive Director, CDAC, Noida for providing me with the valuable opportunity to carry out this work. I am very grateful to **Prof. A.K.AWASTI**, Programme Director, **Prof. R.P. AGARWAL**, course coordinator, M.Tech(IT), IIT, Roorkee and **Mr. V.N.SHUKLA**, course coordinator, M.Tech(IT), CDAC, NOIDA for providing the best of the facilities for the completion of this work and constant encouragement towards the goal.

I am thankful to **Mr. R.K.SINGH** and **Mr. MUNISH KUMAR**, project engineer, CDAC Noida, for providing necessary infrastructure to complete the dissertation in time.

I owe special thanks to **K. SATISH, J. NEELIMA, GLADVIN**, all my classmates and other friends who have helped me formulate my ideas and have been a constant support.

I thank my parents, my brother **KISHORE** and my sister **LAVANYA** for their moral support.

(**SURESH VARIGANJI**)

Enroll. No. 029025

ABSTRACT

“*Clustering Techniques For Content Based Image Retrieval Using Mathematical Morphology*” is to check the applicability of the Mathematical Morphology in Content Based Image Retrieval (CBIR) to extract the derived features of the images and to retrieve to relevant images from the image database using efficient Data Mining Clustering algorithms.

The present Proposed System uses CBIR technology to retrieve the relevant images from the image database on the basis of derived feature such as color, size, shape and texture. Before CBIR was developed, the images were retrieved from large databases by appending some index or address to the images. But this did not work efficiently when user queries consisted of *primitive, logical* and *abstract* features.

The purpose of this dissertation work is to check the applicability of the Mathematical Morphology in Content Based Image Retrieval to extract the *shape* feature from the images using Morphological operation *pattern spectrum* as the *pattern spectrum* acts as a shape descriptor for extracting features from the images. Data-Mining clustering techniques such as RObust hierarchical Clustering with linKs (ROCK) and the Clustering Using REpresentatives (CURE) were used for the image retrieval . Concept of *bins* was used for fast and efficient image retrieval from the image data collection. The performance evaluation of these two clustering algorithms were calculated in terms of domain size and time.

CONTENTS

CANDIDATE'S DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iii
CONTENTS	iv
LIST OF FIGURES	vii
1. INTRODUCTION	1
1.1. Motivation	1
1.2. Objective	4
1.2.1. Problem Statement	4
1.2.2. Proposed System	4
1.3. Organization of the Dissertation	5
2. LITERATURE SURVEY ON CONTENT BASED IMAGE RETRIEVAL	7
2.1. Introduction	7
2.2. Content Based Image Retrieval techniques	9
2.2.1. Colour retrieval	9
2.2.2. Texture retrieval	9
2.2.3. Shape retrieval	10
2.2.4. Retrieval by primitive feature	10
2.2.5. Retrieval by semantic image features	11
2.3. Available Content Based Image Retrieval software	11
2.3.1. Commercial systems	11
2.3.2. Experimental systems	12
2.4. Applications of Content Based Image Retrieval	13
2.5. Summary	14

3.	MATHEMATICAL MORPHOLOGY FOR FEATURE EXTRACTION	15
3.1.	Introduction	16
3.2.	Morphological Operations	17
3.3.	Fundamental Definitions	18
	3.3.1. Dilation and Erosion	19
	3.3.2. Open and Close	20
	3.3.3. Pattern Spectrum	20
3.4.	Binary Morphology – Gray Scale Morphology	22
3.5.	Examples of Pattern Spectrum	22
3.6.	Summary	32
4.	CLUSTERING TECHNIQUES FOR CONTENT BASED IMAGE RETRIEVAL	33
4.1.	Introduction	33
4.2.	Applications of clustering techniques	34
4.3.	Classification of clustering techniques	34
	4.3.1 Partitional clustering algorithms	34
	4.3.2. Hierarchical clustering algorithms	35
4.4.	Proposed Clustering Techniques	36
	4.4.1. RObust hierarchical Clustering with linKs (ROCK)	36
	4.4.2. Clustering Using Representatives (CURE)	39
4.5.	Summary	42
5.	DESIGN AND IMPLEMENTATION OF CONTENT BASED IMAGE RETRIEVAL SYSTEM	43
5.1.	Introduction	43
5.2.	Different Phases of Proposed Retrieval System	43
5.3.	Design and Working of Proposed Retrieval System	44
	5.3.1. Extraction of feature vector	45
	5.3.2. Clustering the images in the database	46
	5.3.3. Query matching	46

5.4.	Implementation Details	48
5.4.1.	Choice of Programming Language	48
5.4.2.	Different Classes and Data Structures	49
5.4.3.	System Requirements	52
5.5.	Summary	52
6.	EXPERIMENTAL RESULTS	53
7.	CONCLUSION AND FUTURE WORK	63
	REFERENCES	65

LIST OF FIGURES

Figure No	Caption	Page No
Figure 2.1	Content Based Image Retrieval System	7
Figure 3.1	Rectangle with 17*13 pixel size	22
Figure 3.2	Pattern Spectrum of Figure 3.1 opened with <i>Square</i> Structuring Element	23
Figure 3.3	Pattern Spectrum of Figure 3.1 opened with <i>horizontal linear</i> Structuring Element	23
Figure 3.4	Rectangle of 17*13 pixel size	24
Figure 3.5	Pattern Spectrum of Figure 3.4 opened with square Structuring Element	24
Figure 3.6	Rectangle of 17*13 pixel size with 5*3 hollow space	25
Figure 3.7	Pattern Spectrum of Figure 3.6 opened with <i>horizontal</i> Structuring Element	25
Figure 3.8	A Rocket image with 13*13 pixel size	26
Figure 3.9	Pattern Spectrum of Figure 3.8 opened with <i>45° linear</i> Structuring Element	26
Figure 3.10(a)	Query Image1	27
Figure 3.10(b)	Query Image2	27
Figure 3.11(a)	Pattern Spectrum of the Figure 3.10(a) opened with <i>diamond</i> Structuring Element	27
Figure 3.11(b)	Pattern Spectrum for the Figure 3.10(b) opened with <i>diamond</i> Structuring Element	28
Figure 3.12(a)	Query Image1	28
Figure 3.12(b)	Query Image 2 with similar shapes	28
Figure 3.13(a)	Pattern Spectrum of the Figure 3.12(a) opened with <i>diamond</i> Structuring Element	29
Figure 3.13(b)	Pattern Spectrum of the Figure 3.12(b) opened with <i>diamond</i> Structuring Element	29

Figure 3.14	Method to Find Bitstring for given Pattern Spectrum	30
Figure 5.1	Different Phases of the Proposed Retrieval System	43
Figure 5.2	Architectural design of the working of the Proposed Retrieval System	43
Figure 6.1	Original Query Image 1	53
Figure 6.2	Binary Image for the given Query Image1	54
Figure 6.3:	Dilated Image for the given Query Image 1	54
Figure 6.4	Binary Image after holes filled for the given Query Image 1	55
Figure 6.5	Erosion operation on the Query Image 1	55
Figure 6.6	Edge Detection technique to identify the object	56
Figure 6.7	Pattern spectrum for the given query image 1	56
Figure 6.8	Bitstring for the Pattern Spectrum of Query Image1	57
Figure 6.9	Similar Images Retrieved for the given Query Image 1	57
Figure 6.10	Original Query Image 2	58
Figure 6.11	Pattern Spectrum for the given Query Image 2	58
Figure 6.12	Bitstring for the pattern spectrum of the given Query Image 2	59
Figure 6.13	Similar Images Retrieved for the given Query Image 2	59
Figure 6.14(a)	Performance evaluation of ROCK and CURE algorithms with Domain size of 50 to 250 on X axis and Time interval on Y axis	60
Figure 6.14(b)	Performance evaluation of ROCK and CURE algorithms with Domain size of 50 to 500 on X axis and Time interval on Y axis	61

INTRODUCTION

1.1 Motivation

“*Content Based Image Retrieval is a technology for retrieving images on the basis of automatically derived features.*” The derived features include primitive features such as color, texture and shape. The features may also include logical features like identity of objects given, abstract features like significance of some scene depicted etc., depending on the application.

Before Content Based Image Retrieval (CBIR) [1] was developed, the images were retrieved from large databases by appending some index or address to the images. But this did not work efficiently when user queries consisted of abstract and logical features. Some form of cataloguing and indexing is still necessary – the only difference being that much of the required information can now potentially be derived automatically from the images themselves. In common images are required for a variety of reasons, including[1]:

- conveying information or emotions difficult to describe in words.
- illustration of text articles.
- display of detailed data (such as radiology images) for analysis.
- formal recording of design data (such as architectural plans) .

Access to a desired image from a repository thus involves a search for abstract features in images depicting specific types of object or scene, evoking a particular mood, or simply containing a specific texture or pattern.

Content Based Image Retrieval System [1] is a system which retrieves the images from an image collection based on a query specified by content. The *query image* is one in which a user is interested and wants to find similar images from the image collection.

Potentially images have many derived features which can be used for retrieval :

- the presence of a particular combination of colour, 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).

Each of the above listed query type (with the exception of the last) represents a higher level of abstraction than its predecessor, and each is more difficult to answer without reference to some body of external knowledge. This leads on to a classification of query types into 3 levels of increasing complexity [1]:

Level 1 [1] comprises retrieval by *primitive* features such as colour, 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 colour matching of fashion accessories.

Level 2 [1] 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:

1. retrieval of objects of a given type (e.g. “find pictures of a double-decker bus”);
2. retrieval of individual objects or persons (e.g. “find a picture of the Eiffel tower”).

To answer queries at this level, reference to some outside store of knowledge is 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.

Level 3 [1] 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:

1. retrieval of named events or types of activity (e.g. “find pictures of Scottish folk dancing”);
2. retrieval of pictures with emotional or religious significance (e.g. “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 judgement, can be required to make the link between image content and the abstract concepts it is required to illustrate. Levels 2 and 3 together are referred as *semantic* image retrieval. Queries related to this are often encountered in both newspapers and art libraries.

All the existing CBIR systems operate only at the primitive feature *Level 1* [1]. The current system developed (described later) also operates only at the primitive feature level like “find me more pictures that look like this”. The logical features are difficult to extract and is still a research topic.

The features used for retrieval using CBIR can be either primitive or semantic, but the extraction process must be predominantly automatic. CBIR differs from classical information retrieval in that image databases are essentially unstructured, since digitized images consist purely of arrays of pixel intensities, with no inherent meaning. One of the key issues with any kind of image processing is the need to extract useful information such as features from the raw data, such as recognizing the presence of particular shapes or textures before any kind of reasoning about the image’s contents is possible.

1.2 Objective

The objective of this dissertation work is to apply the techniques of Mathematical Morphology for Content Based Image Retrieval. In the processing and analysis of images, it is important to be able to extract features, describe shapes and recognize patterns. Such tasks refer to geometrical concepts such as size, shape, and orientation. Mathematical Morphology [2] uses concepts from set theory, geometry and topology to analyze geometrical structures in an image. Mathematical morphology simplifies an image by removing irrelevant details and errors with which the essential characteristics of the images remain intact. This characteristic of mathematical morphology is the motivation behind using it for feature extraction in content-based image retrieval.

1.2.1 Problem Statement

To check the ability of mathematical morphology in extracting the feature vectors from the images and to check the performance evaluation of the clustering techniques for efficient image retrieval.

1.2.2 Proposed System

The Proposed system is divided into feature extraction phase and clustering for image retrieval, which works as follows

- ***Mathematical Morphology for feature extraction***

First the feature vectors of all the images in the database are found. The system uses morphological operations such as opening distribution and pattern spectrum, which acts as shape descriptors in analyzing an image.

Second, given a query image, its feature vector is computed, compared to the feature vectors in the feature database, and images most similar to the query image are returned to the user.

- ***Data mining clustering algorithms for image retrieval***

The clustering algorithm considers the feature vectors of all the images of the collection and will divide the whole collection into a number of clusters. The idea behind

doing so is that the images belonging to the same cluster are similar or relevant to each other when compared to images belonging to different cluster.

Every care has to be taken to ensure that the features and the similarity measure used to compare two feature vectors are efficient enough to match similar images and to discriminate dissimilar ones. The main aim of this system is to retrieve almost all relevant images as well as to minimize the number of irrelevant images. To achieve this, the system considers all the close clusters (Inter-Cluster Search) and then doing the search within that cluster (Intra-Cluster search).

The proposed system operates at the *Level 1* as described in section 1.1 as *Level 1* comprises retrieval by primitive feature such as color, texture, shape or the spatial location of image elements.

1.3 Organization of the Dissertation

In Chapter 2, various commercial and experimental techniques of Content Based Image Retrieval (CBIR) systems and the methods applied to extract the features from the images with different search techniques for image retrieval. It concludes with the practical application of the CBIR techniques.

In Chapter 3, Mathematical Morphology definitions and the properties are briefly discussed followed by the concept of pattern spectrum used for the present retrieval system is discussed with examples.

In Chapter 4, Clustering techniques for Content Based Image Retrieval (CBIR) were discussed and the performance evaluation of the ROBust hierarchical Clustering with linKs (ROCK) and Clustering Using REpresentatives (CURE) are calculated.

In Chapter 5, Design and Implementation details of all the concepts of the present proposed retrieval system are explained. Details of the programming language used, different classes and data structures and the user interface are explained.

In Chapter 6, Experimental results of different test cases were given. It concludes with the experimental setup of the system.

Chapter 7, which is the final chapter gives the conclusion and explains about the enhancements that can be carried out in this system.

LITERATURE SURVEY ON CONTENT BASED IMAGE RETRIEVAL

2.1 Introduction

There are basically two steps involved in Content Based Image Retrieval System:

1. For each image in the database, a feature vector characterizing some image properties is computed and stored in a feature database.
2. Given a query image, its feature vector is computed, compared to the features in the feature database, and the images most similar to the query image are returned to the user.

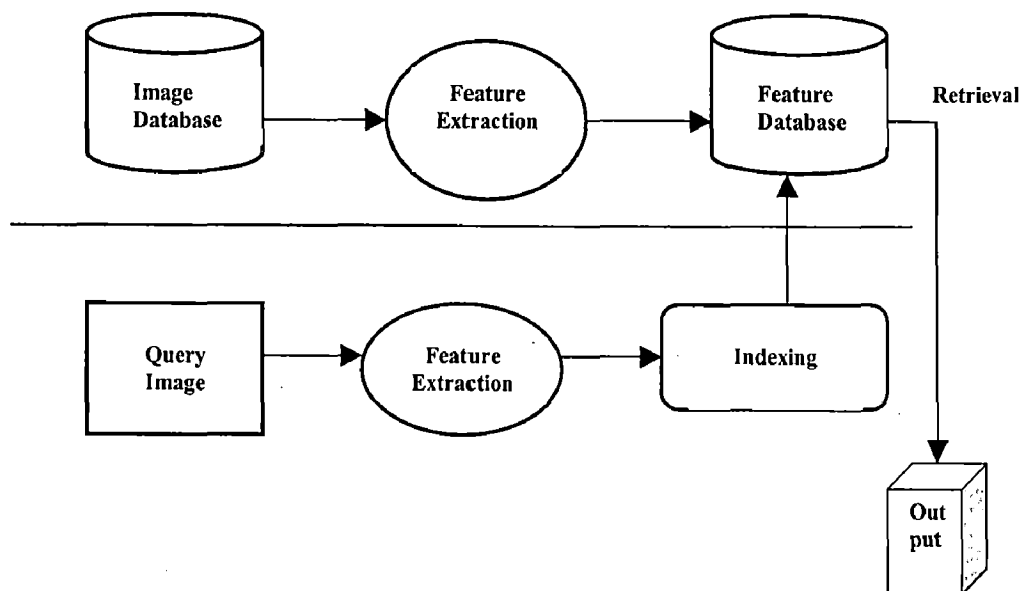


Figure 2.1: Content Based Image Retrieval System

The *feature extraction* is the reduction of the images into mathematical “feature vectors” that capture the various properties of interest. The feature extraction can be done based on shape, texture, color or combination of any based on a particular task. For example queries like “retrieve all the images containing yellow stars” require feature

extraction based on color and shape. For queries like “retrieve images containing long thin lines” feature extraction is done based on texture and shape.

Retrieval is the user interaction with the CBIR system to retrieve desired images from the database. The retrieval can be performed in two methods:

- The feature vector of the query image is compared with the feature vectors of all the images in the database
- The feature vectors of all the images in the database are clustered based on specific criteria (similarity, difference etc.) and the feature vector of the query image is compared with the feature vector of the cluster.

The various *multiresolution techniques* such as quad trees, pyramids, fractal Imaging, scale-spaces, etc., are used for feature extraction, all have their merits and limitations. For example, fractals have been exploited with great success in image compression but to a much lesser extent for segmentation problems. The emergence of *wavelet techniques* has considerably boosted the multiresolution approach. Unfortunately, application of wavelets to problems in image processing and computer vision is sometimes hindered by its linearity. Coarsening an image by means of linear operators may not be compatible with a natural coarsening of some image attribute of interest (shape of object, for example), and hence use of linear procedures may be inconsistent in such applications.

Mathematical morphology [2] is a nonlinear technique compared to linear wavelets. It considers images as geometrical objects rather than as elements of a linear space. Many of the existing morphological operations and techniques, such as granulometries, skeletons, pattern spectrum and alternating sequential filters, are essentially multiresolution techniques.

R*-trees [22], region-based searching techniques [34], existing database management systems like Oracle, QBE are used for efficient retrieval.

2.2 Content Based Image Retrieval techniques

In contrast to the text-based approach all current CBIR systems, whether commercial or experimental, operate at at the colour, texture or shape. Some of the CBIR systems are described below.

2.2.1 Colour retrieval [22]

Several methods for retrieving images on the basis of colour similarity have been described in the literature, but most are variations on the same basic idea. The colour histogram for each image is then stored in the database. At search time, the user can either specify the desired proportion of each colour (75% olive green and 25% red, for example), or submit an example image from which a colour histogram is calculated. Either way, the matching process then retrieves those images whose colour histograms match those of the query most closely.

2.2.2 Texture retrieval [25]

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

2.2.3 Shape retrieval [27]

The ability to retrieve by shape is perhaps the most obvious requirement at the primitive level. Unlike texture, shape is a fairly well-defined concept – and there is considerable evidence that natural objects are primarily recognized by their shape. A number of features characteristic of object shape (but independent of size or orientation) are computed for every object identified within each stored image. Queries are then answered by computing the same set of features for the query image, and retrieving those stored images whose features most closely match those of the query[27].

2.2.4 Retrieval by primitive feature [1]

One of the oldest-established means of accessing pictorial data is retrieval by its spatial location. Similar techniques have been applied to image collections, allowing users to search for images containing objects in defined spatial relationships with each other. Improved algorithms for spatial retrieval are still being proposed. Spatial indexing is seldom useful on its own, though it has proved effective in combination with other cues such as colour. Several other types of image feature have been proposed as a basis for CBIR. The most well-researched technique of this kind uses the *wavelet transform* [26] 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.

2.2.5 Retrieval by semantic image features

1) Level 2

The vast majority of current CBIR [1] techniques are designed for primitive-level retrieval. 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 is an important filter which can be used when searching, and because this can help in identifying specific objects present. The second focus of research activity is object recognition, an area of interest. Techniques are now being developed for recognizing and classifying objects with database retrieval in mind.

2) Level 3

Reports of automatic image retrieval at *Level 3* [1] are very rare. The only research that falls even remotely into this category has attempted to use the subjective connotations of colour (such as whether a colour is perceived to be warm or cold, or whether two colours go well with each other) to allow retrieval of images evoking a particular mood. It is not at present clear how successful this approach will prove.

2.3 Available CBIR software

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

2.3.1 Commercial systems

QBIC: IBM's QBIC[15] system is probably the best-known of all image content retrieval systems. It offers retrieval by any combination of colour, texture or shape – as well as by text keyword. The system extracts and stores colour, shape and texture features from each image added to the database, and uses R*-tree indexes to improve search efficiency.

Virage: Another well-known commercial system is the VIR Image Engine from Virage[16]. This is available as a series of independent modules, which systems

developers can build in to their own programs. This makes it easy to extend the system by building in new types of query interface, or additional customized modules to process specialized collections of images such as trademarks. Alternatively, the system is available as an add-on to existing database management systems such as Oracle or Informix.

Excalibur: A similar philosophy has been adopted by Excalibur[14] Technologies, a company with a long history of successful database applications, for their Visual RetrievalWare product. This product offers a variety of image indexing and matching techniques based on the company's own proprietary pattern recognition technology. It is marketed principally as an applications development tool rather than as a standalone retrieval package. Its best-known application is probably the Yahoo! Image Surfer, allowing content-based retrieval of images from the World-wide Web.

2.3.2 Experimental systems

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

Photobook: Like the commercial systems above, aims to characterize images for retrieval by computing shape, texture and other appropriate features. This Photobook[16] systems aims to calculate *information-preserving* features, from which all essential aspects of the original image can in theory be reconstructed. This allows features relevant to a particular type of search to be computed at search time, giving greater flexibility at the expense of speed.

Chabot: Another early system which has received wide publicity is Chabot [17], which provided a combination of text-based and colour-based access to a collection of digitized photographs held by California's Department of Water Resources. The system has now been renamed Cypress, and incorporated within the Berkeley Digital Library project at the University of California at Berkeley (UCB).

VisualSEEk: The VisualSEEk [20] system is the first of a whole family of experimental systems developed at Columbia University, New York. It offers searching by image region colour, shape and spatial location, as well as by keyword.

MARS: The MARS [19] project at the University of Illinois is aimed at developing image retrieval systems which put the user firmly in the driving seat. Relevance feedback is thus an integral part of the system, as this is felt to be the only way at present of capturing individual human similarity judgements. The system characterizes each object within an image by a variety of features, and uses a range of different similarity measures to compare query and stored objects. User feedback is then used to adjust feature weights, and if necessary to invoke different similarity measures.

Surfimage: An example of European CBIR technology is the Surfimage[21] system from INRIA. This has a similar philosophy to the MARS system, using multiple types of image feature which can be combined in different ways, and offering sophisticated relevance feedback facilities.

Netra : The Netra[18] system uses colour texture, shape and spatial location information to provide region-based searching based on local image properties. An interesting feature is its use of sophisticated image segmentation techniques.

2.4 Practical applications of CBIR[1]

A wide range of possible applications for CBIR technology has been identified. Potentially fruitful areas include:

- Crime prevention
- The military
- Intellectual property
- Architectural and engineering design
- Fashion and interior design
- Journalism and advertising
- Medical diagnosis
- Geographical information and remote sensing systems

- Cultural heritage
- Education and training
- Home entertainment
- Web searching.

2.5 Summary

This chapter discussed the available Content Based Image Retrieval techniques with the methods used for the feature extraction and image retrieval techniques. It also included the various commercial and experimental softwares available. This chapter concludes with the practical applications of the Content Based Image Retrieval techniques.

MATHEMATICAL MORPHOLOGY FOR FEATURE EXTRACTION

3.1 Introduction

Mathematical Morphology [2] is an important basis for image processing to study the form of objects. “*Morphology is the name of a specific methodology designed for the analysis of the geometrical structure in an image.*” Mathematical Morphology has three aspects as listed below

- An algebraic one dealing with image transformations derived from set-theoretical operations.
- A probabilistic one dealing with models of random sets applicable to the selection of small samples of materials.
- An integral geometrical one dealing with image functionals.

Present study exclusively concerns with the algebraic aspects of the theory of Mathematical Morphology.

Mathematical Morphology [2] involves rules for changing the state of a pixel based upon its state and the states of its surrounding neighbours. It employs specific sequences of neighbourhood transformations to measure useful geometric features on given images. It examines the geometrical structure of an image by probing it with small patterns of varying size and shape, just the way a blind man explores the world with his fingers or a stick. This procedure results in nonlinear image operators, which are well suited to exploring geometrical and topological structures. Successions of such operators are applied to an image in order to make certain features apparent, distinguishing meaningful information from irrelevant distortions, by reducing it to a form skeleton. Theoretical aspects were given below.

3.1.1 Theoretical Basis

Morphological operations are among the first kinds of image operators. But although the techniques are being used in the industrial world the basis and theory of mathematical Morphology [2] tend to be not covered in the textbooks or journals, which discuss image processing or computer vision.

Peter Maragos [5] introduced the concept of pattern spectrum, which appears to be a promising concept that quantifies various aspects of the shape-size content of a signal. Song and Delp [4] expose the limitations of using a single structuring element and propose the utilization of multiple structuring elements that can enhance the performance of morphologic based filtering. S.S.Wilson [3] introduces the theory of matrix morphology in which the current concept of mathematical morphology has been labeled as scalar morphology and a new concept of matrix morphology formalism has been introduced in which there exists a matrix of input images and another matrix of structuring elements. Dilations, Erosions, Openings, and closings between such matrices are defined. It is shown that whereas scalar morphology deals with sets that are used to describe shapes, matrix morphology deals with the relationships of shapes used to describe feature space.

3.1.2 Applications of Mathematical Morphology

Morphology calls on many areas of mathematics, including integral geometry, statistics, algebra, topology, analysis, and computer science. Machines that perform morphologic operations are not new [2]. They are the essence of cellular logic machines such as DIFF3, CLIP, DAP, MPP, PICAP, PHP, Cytocomputer, Genesis, Texture Analyzer, and many more of interest. A number of companies now manufacture industrial vision machines, which incorporate video rate morphologic operations. These companies include Machine Vision International, Maitre, Synthetic Vision Systems, Vicom, and Applied Intelligence Systems Inc. & Leitz among others.

Apart from the above sources, there exists a host of published material dealing with real-life applications for which mathematical morphology has been made use of. F.Meyer [7] uses morphological operations for the automatic screening of cytological

smears for the early detection of cancer. Use of the morphological operations has been made for autofocussing the microscope, segmentation of the nuclei, artifact recognition and leucocyte elimination. Michael M.Skolnick[8] applied morphological transformations to the analysis of two-dimensional electrophoretic gels of biomedical materials. C.Bhagvati and A.Grivas[10] make use of morphological opening size distributions for the classification of various categories of pavement distress. H.Boerner [9] compares the conventional DOG filter and a morphological filter based on opening with a spherical structuring element for the detection of bright blobs in low signal/ high noise images containing other objects like step edges and corners, especially with regard to selectivity, susceptibility to noise and stability of segmentation. Currently CWI is investigating a general axiomatic pyramid scheme encompassing most existing linear as well as nonlinear morphological pyramids [11]. Next section discusses the some of the morphological operations.

3.2 Morphological Operations

The mathematical morphology uses the concept of *Minkowski set operations* which are a background for the mathematical morphology, and the applications of mathematical morphology such as the size distribution, pattern spectrum are discussed in this section.

The language of Mathematical Morphology is that of set theory where, in our case, the sets represent binary and gray-level images. The set of all white pixels in a black and white image constitutes a complete description of the image. Morphological transformations apply to sets of any dimensions, those like Euclidean N-space, or those like discrete or digitized equivalent –the set of N-tuples of integers Z^n [16]. It uses the concept of *structuring element* to study the geometric structure of an object or an image.

3.2.1 Structuring Element

All the operations of mathematical morphology are usually defined between two sets (say A and B) where A is the set we want to examine and B is called a *Structuring Element*. This is a key concept of mathematical morphology because it offers us a great flexibility that we can design it easily with different shapes and sizes according to our purpose. Performing morphological operations on set A with different structuring elements $\{B_i\}$, we get a resulting set series $\{Y_i\}$. Each specific structuring element B_i filters out a specific kind of information from the original image. By analyzing the resulting set sequence $\{Y_i\}$, we can obtain sufficient heuristical information. Then we can combine this with the specific goal in our analysis and decide what structuring element to use next to filter out further information we are interested in. This interaction between set A and structuring element B facilitates image analysis by continuously finding the most suitable structuring element.

3.3 Fundamental Definitions

Images are modeled in mathematical morphology as sets of points in the n-dimensional Cartesian product space of the reals. The fundamental operations associated with an object are the standard set operations *union*, *intersection*, and *complement* $\{\cup, \cap, \complement\}$ plus *translation* [9]

- *Translation* - Given a vector \mathbf{x} and a set A , the *translation*, $A + \mathbf{x}$, is defined as:

$$A + \mathbf{x} = \{\alpha + \mathbf{x} \mid \alpha \in A\} \quad \text{----- (3.1)}$$

From the equation 3.1 it is known that A be a digital image with composed of pixels, \mathbf{x} be the size of the structuring element and α belongs to A . Note that, since we are dealing with a digital image composed of pixels at integer coordinate positions (\mathbb{Z}^2), this implies restrictions on the allowable translation vectors \mathbf{x} .

$$\text{Minkowski addition} \quad : \quad (A \oplus B) = \bigcup_{\beta \in B} (A + \beta) \quad \text{----- (3.2)}$$

$$\text{Minkowski subtraction: } (A \otimes B) = \bigcap_{\beta \in B} (A + \beta) \quad \text{-----}(3.3)$$

Equations 3.2 and 3.3 specifies the basic *Minkowski set operations*--addition and subtraction--can now be defined by two sets *A* and *B*.

3.3.1 Dilation and Erosion

The two Minkowski operations, addition and subtraction, we define the fundamental mathematical morphology operations *dilation* and *erosion*.

Dilation - Dilation is the morphological transformation, which combines two sets using vector addition of set elements. Let *A* and *B* be subsets of Z^2 . Then Dilation of *A* and *B* is defined as the union of all the translates of the set *A* by each of the elements of the set *B*(dilation shown in equation 3.4).

$$D(A,B) = (A \oplus B) = \bigcup_{\beta \in B} (A + \beta) \quad \text{-----}(3.4)$$

Erosion - Erosion is the morphological dual to dilation. It is the morphological transformation, which combines two sets using vector subtraction of set elements. Let *A* and *B* be subsets of Z^2 . Then Erosion of *A* and *B* is defined as the intersection of all the translates of the set *A* by each of the elements of the set *(-B)*.Equation 3.5 represents the erosion operation of the mathematical operation.

$$E(A,B)=A \otimes (-B)= \bigcap_{\beta \in B} (A - \beta) \quad \text{-----}(3.5)$$

Dilation, in general, causes objects to dilate or grow in size; *erosion* causes objects to shrink. The amount and the way that they grow or shrink depend upon the choice of the structuring element. *Dilation* and *erosion* satisfies the properties like commutative,non commutative,associative,duality.

When A is an object and A^c as the background, equation 3.5 says that the *dilation* of an object is equivalent to the *erosion* of the background. Likewise, the *erosion* of the object is equivalent to the *dilation* of the background.

3.3.2 Opening and Closing

The opening is just a sequential application of erosion followed by dilation with the same structuring element. The opening of an image A by a structuring element B is defined as

$$O(A,B) = A \circ B = D(E(A,B),B) \quad \text{-----}(3.6)$$

Equation 3.6 shows the iterative application of dilations and erosions results in elimination of specific image detail smaller than the structuring element without the geometric distortion of unsuppressed features.

The closing is just a sequential application of dilation followed by erosion with the same structuring element. The closing of an image A by a structuring element B is defined as

$$C(A,B) = A \bullet B = E(D(A,-B),-B) \quad \text{-----}(3.7)$$

Equation 3.7 represents the closing operation of on the image A with the structuring element of size B .

3.3.3 Pattern Spectrum

The Mathematical Morphological operation Pattern spectrum is a shape size descriptor of an object content of a digital image. It is a technique for extracting various shape and resolution information of a digital image. It quantifies various aspects of the shape-size content of an image [7]:

- Shape, as Maragos [5] defined it, is any image conveying information about intensity or range or any other finite extent signal whose graph is viewed as an image object conveying pictorial information.

- Scale, is defined as the smallest size of a shape pattern generated by a prototype pattern of unit size that can fit inside the image.

Before giving the defining expression for pattern spectrum, one point needs to be stressed: *Image A, when opened with structuring element B eliminates all objects of size less than B (i.e., objects inside which B cannot fit).*

Definition: Let A be finite extent discrete binary image and let the discrete binary pattern B be any finite subset of Z^2 . It can be shown that for all $n \geq 1$,

$$\dots \subseteq A \circ (n+1) B \subseteq A \circ n B \subseteq A \circ (n-1) B \dots \subseteq A \dots \quad \text{-----(3.8)}$$

From the equation 3.8 the *Pattern Spectrum* is defined by the following expression:

$$PS_A = (n, B) = \text{Area}(A \circ n B) - \text{Area}(A \circ (n+1) B) \text{ for } n \geq 0 \quad \text{-----(3.9)}$$

In the equation 3.9 *Area(A ∘ nB)* indicates the area of the image when opened with B of size n.

An opening operation by a structuring element of a certain size removes all features from an image that are smaller than the size and geometry of the structuring element. Thus the area obtained by opening A by a structuring element B of size r is a measure of the pattern content of A relative to the pattern rB.

By varying both sizes r and the shape of B we obtain a shape-size spectrum of A which is the full pattern spectrum of A relative to all the patterns that can fit inside A. By keeping B fixed we get a size histogram of A relative to B.

3.4 Binary Morphology----Gray-Scale Morphology

Binary Morphology deals with input images and structuring elements with only two intensity values – one representing the foreground and the other representing the background. Gray-scale morphology deals with input images whose intensity values range from 0 to 255.

In the present system we deal only with binary images. The gray-scale images are converted into binary using the concept of Thresholding. In order to create the two-valued binary image a simple threshold may be applied so that all the pixels in the image plane are classified into object and background pixels. A binary image function is constructed such that pixels above the threshold are considered as foreground (“1”) and below the threshold are background (“0”) [8].

3.5 Examples of Pattern Spectrum

Example 1: Consider a rectangle of size 17 x 13 as shown in the figure below. The actual picture is represented as white pixels.



Figure 3.1: Rectangle with 17*13 pixel size

The image shown in Figure 3.1 is opened with *square* structuring elements of increasing size starting from 1. The pattern spectrum of the image is shown in the form of a graph below. The size of the structuring elements is given along x-axis and the values of pattern spectrum are given along y-axis. The end of the graph is the point at which the area of the image becomes zero when opened with the structuring element of size equal to the value at the end point.

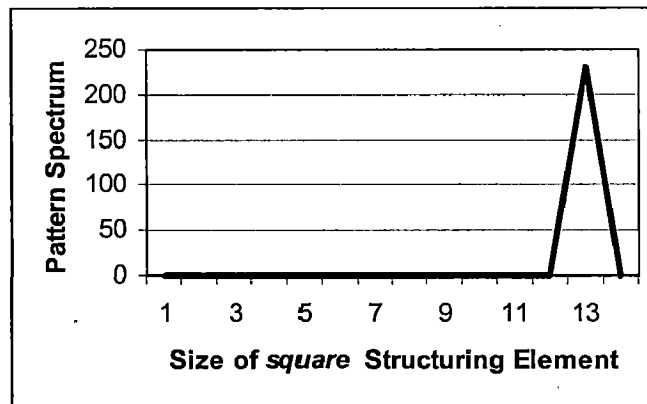


Figure 3.2: Pattern Spectrum of Figure 3.1 opened with *Square* Structuring Element

From the Figure 3.2, it is clear that the pattern spectrum shows an impulse at a scale that corresponds to the smaller of the two dimensions of the rectangle. That means a square of size greater than 13 does not fit into the object. This clearly can have applications like finding the width of an object in the image.

When the picture in Example-1 is opened with *horizontal linear* structuring elements of increasing size, the results are as shown:

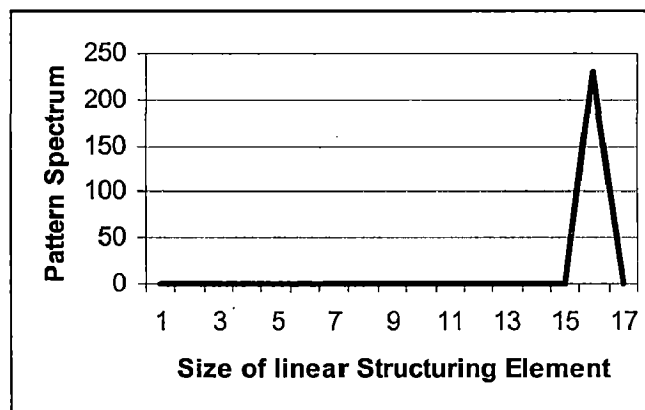


Figure 3.3: Pattern Spectrum of Figure 3.1 opened with *horizontal linear* Structuring Element

From the pattern spectrum in Figure 3.3, it is clear that there exists no object of spatial extent larger than 17 pixels in the horizontal direction.

Example 2: The rectangle in example-1 is modified that it is slightly bulged horizontally near its bottom as shown in the figure below.



Figure 3.4: Rectangle of 17*13 pixel size

The image in Figure 3.4 is bulged by 3 columns from the rows 7 to 12. The picture is opened with horizontal linear structuring elements of increasing size starting from size 1. The pattern spectrum in the form of a graph is as shown below:

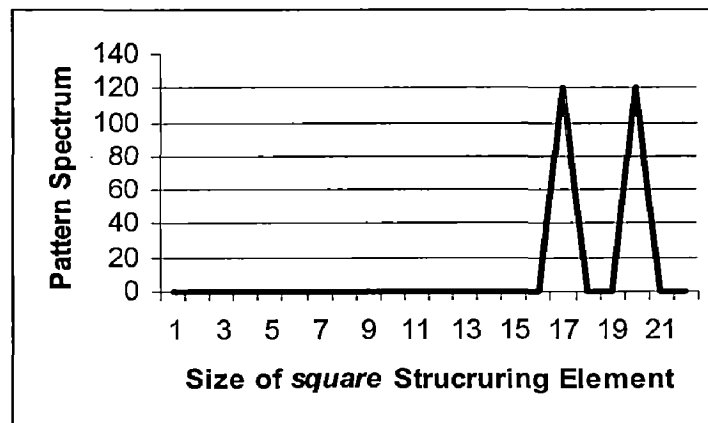


Figure 3.5: Pattern Spectrum of Figure 3.4 opened With *square* Structuring Element

From the pattern spectrum of Figure 3.5, it is clear that there exists no object of spatial extent larger than 20 pixels in the horizontal direction. Also it gives an impulse at size 17 and 20. That means the image can accommodate horizontal lines of 17 and 20 only in it. This is clear from the image above. It also indicates the sudden bulging of the image. This clearly has application like detecting whether any part of the object has expanded in its size, that is change in the size and shape of an object can be detected

Example 3: Consider a rectangle that has some hollow space in it as shown in the figure below.



Figure 3.6: Rectangle of 17*13 pixel size with 5*3 hollow space

The image in Figure 3.6 having a hollow space is of size 5 by 3 starting at row 5 and column 5. The image is opened with *horizontal linear* structuring elements of increasing size starting from size 1. The pattern spectrum is as shown below:

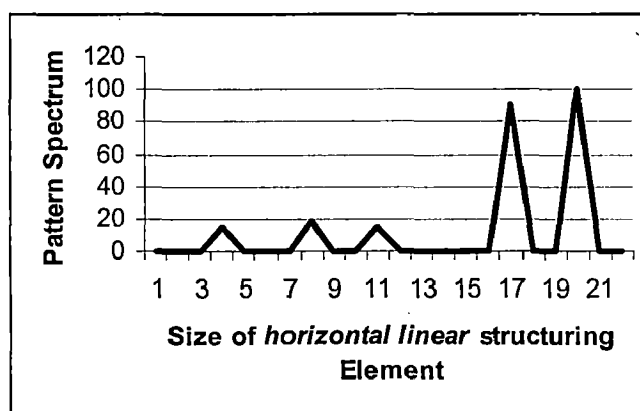


Figure 3.7: Pattern Spectrum of Figure 3.6 opened with *horizontal linear* Structuring Element

From the Figure 3.7 Pattern spectrum, it is clear that as the hollow space starts at 5th column, there is an impulse at size 4 of the horizontal linear structuring element. Also there is an impulse at 8, which is equal to $((17-4)-5)$ where 5 is the length of hollow space and 4 is the position of the hollow space. The impulse at 11 indicates that the hollow space also extends to the bulged portion of the image. The gap between the impulses at 17 and 20 indicates that the bulge is of size 3 in the horizontal direction. Now this can be applied in applications to identify the presence of any cavity in an object.

Example 4: Consider the image of a rocket. The pattern spectrum of the image using 45° diagonal structuring element is:



Figure 3.8: A Rocket image with 13×13 pixel size

The image in Figure 3.8 of size 13 by 13 pixels size is opened with a 45° angular Structuring Element.

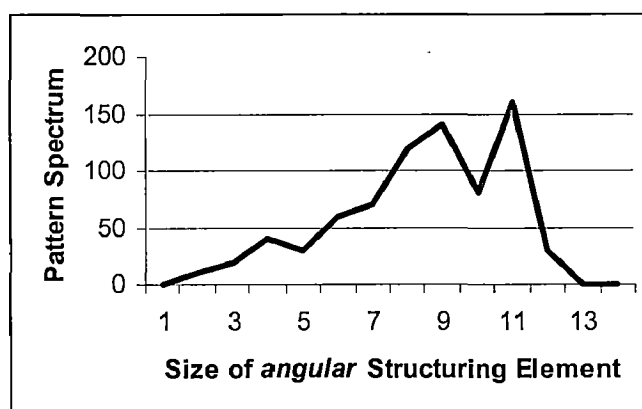


Figure 3.9: Pattern Spectrum of Figure 3.8 opened with 45° linear Structuring Element

From the pattern spectrum of Figure 3.9, it is observed that certain information regarding the shape of the image can be obtained. The maximum length of the image in the direction of 45° is 13. Also most of the image in that direction is of length 11.

It can be clearly seen that pattern spectrum acts as a shape descriptor when a structuring element of suitable shape and size are applied.

3.5.1 Pattern Spectrum Comparison

As the pattern spectrum describes the shape and size descriptor of an image, It can be used for identifying the similarity in the images. Let us consider two images with different shapes as shown Figure 3.10(a) and 3.10(b).

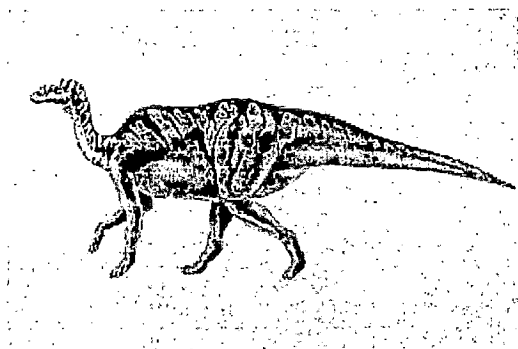


Figure 3.10(a): Dinosaur Image



Figure 3.10(b): Elephant Image

It can be clearly seen from the Figure 3.10(a) and Figure 3.10(b) that there is a considerable change in the shape of the two image. This change in the shape of the object can be captured using the pattern spectrum. The pattern spectrum of the above two images is shown in the form of a graph in Figure 3.11(a) and 3.11(b).

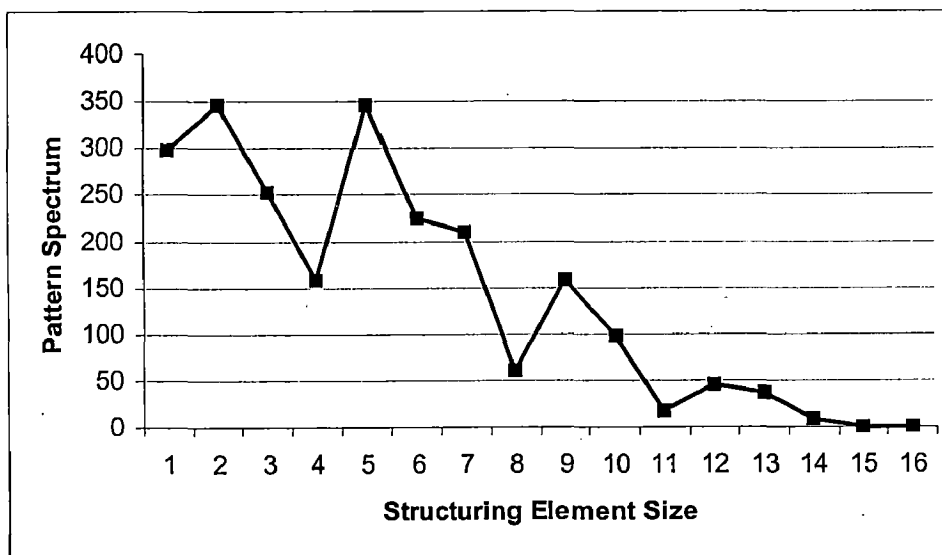


Figure 3.11(a): Pattern Spectrum of the Figure 3.10(a) opened with *diamond* Structuring Element

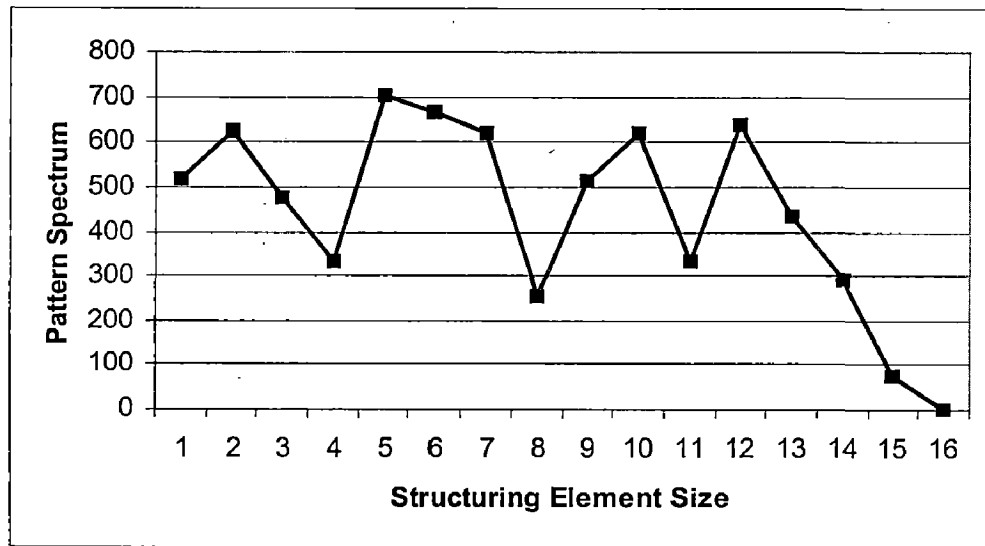


Figure 3.11(b): Pattern Spectrum of the Figure 3.10(b) opened with *diamond* Structuring Element

It is observed from the above graphs that there is a significant change, which is perceived by the normal eye, is also perceived using the pattern spectrum to some extent. This can be observed from the graph. This change helps in differentiating one expression from the other.

Let us Consider images with similar shapes as shown in Figure 3.12(a) and 3.12(b) with similar shapes.

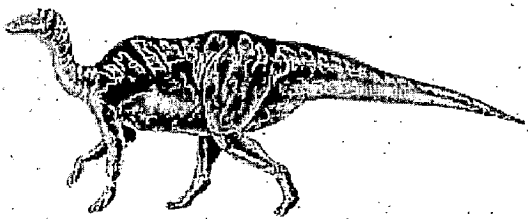


Figure 3.12(a): Dyanosaur Image 1

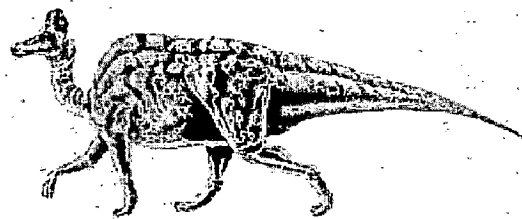


Figure 3.12(b): Dynosaur Image 2

It can be clearly seen from the Figure 3.12(a) and Figure 3.12(b) that there is a considerable change in the shape of the two image. This change in the shape of the object

can be captured using the pattern spectrum. The pattern spectrum of the above two images is shown in the form of a graph in Figure 3.13(a) and 3.13(b).

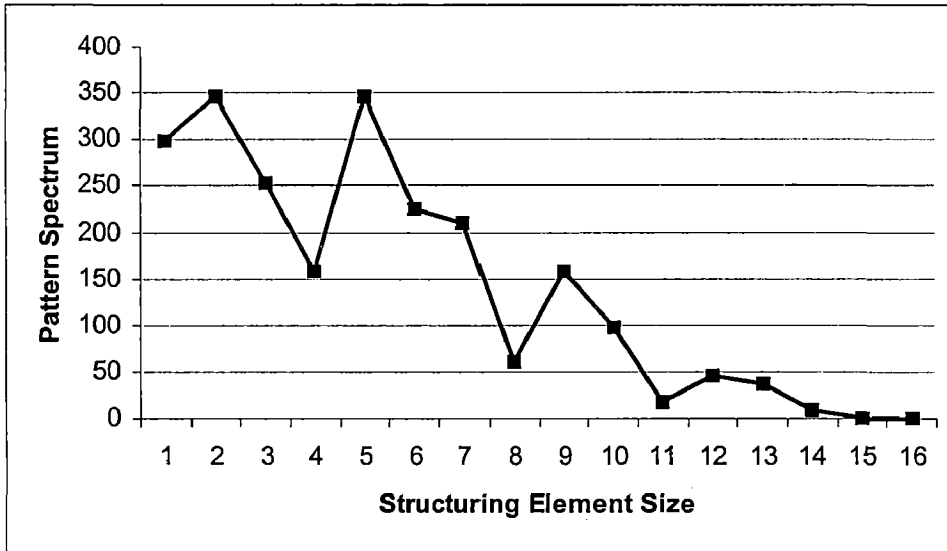


Figure 3.13(a): Pattern Spectrum of the Figure 3.12(a) opened with *diamond* Structuring Element

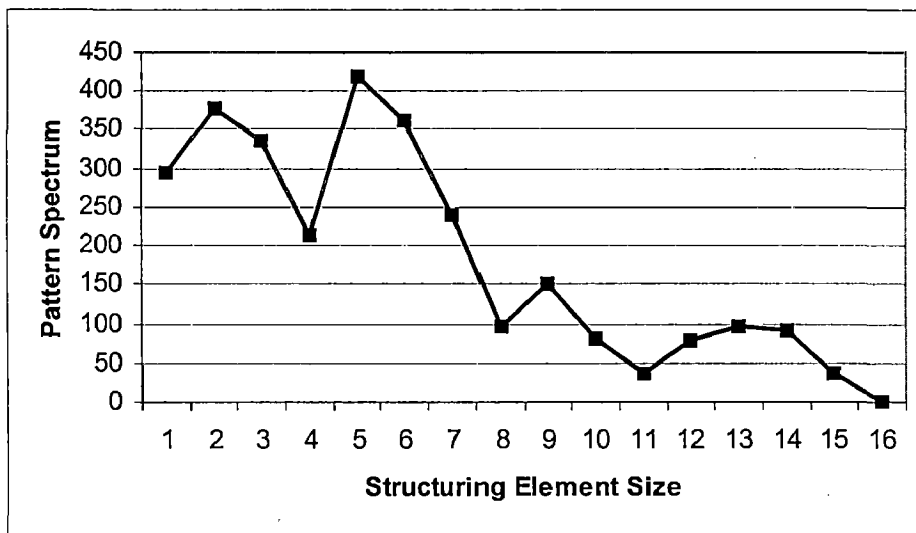


Figure 3.13(b): Pattern Spectrum of the Figure 3.12(b) opened with *diamond* Structuring Element

From the Figures 3.13(a) and 3.13(b) it is observed that the two images are similar, which is perceived by the normal eye and is also perceived using the pattern spectrum to some extent. This can be observed from the graph which helps in finding out the similarity between the images.

3.5.2 Handling Sequences

For the similarity search of images with various shapes in the system developed, the search is based on the comparison of pattern spectra, which is a sequence of numbers. This section describes a scheme for fast similarity search in large sequence databases.

A novel indexing scheme is followed for faster retrieval of similar sequences. The idea is to represent an entire sequence of numbers (pattern spectrum) with a single index called, the BIN number. In order to compare two pattern spectra, this approach allows us to compare only the corresponding BIN numbers instead of comparing all the numbers in the sequence. Every image in the database has its corresponding BIN number. Method of representing the sequence in terms of BIN number is shown below.

3.5.3 BitString Calculation

The pattern spectrum produced from the image is used to find the *bitstring*. To find the bitstring, for any two consecutive pair of pattern spectrum sequence numbers, if the value is increasing then we assign 1 and else, we assign 0. Consider, for example, the pattern spectrum $PS(I)$ of an image given in Figure 3. Thus when there is a rise in the sequence then it is indicated as 1, otherwise it is indicated as 0 as shown in Figure 3.10(a).

293	→	378	→	335	→	214	→	420	→	361	→	97	→	152	→	81	→	37	→	77	→	95	→	35	→	0
1		0		0		1		0		0		1		0		0		1		1		0		0		0

Figure 3.14: Method to Find Bitstring for given Pattern Spectrum

3.5.4 Pseudo code for BitString Calculation

Let n be the length of the pattern spectrum obtained from the image shown in Figure 3.10(a). Now the bitstring $bs(I)$ for any k is calculated as

$$\begin{aligned} bs_k(I) &= 1, & (PS_k - PS_{k+1}) > 0 \\ &= 0, & \text{Otherwise} \end{aligned}$$

Algorithms 3.1: Pseudo code to find the Bitstring

In Algorithm 6.1 PS_k and PS_{k+1} represents the pattern spectrum of k^{th} and $k+1^{th}$ sequences respectively. Where $bs_k(I)$ signifies the bitstring obtained for the k^{th} index.

3.5.5 BitString Index

To build the bitstring index, it is necessary to decide on the length of the queries that are most likely to be encountered. The queries can be longer or shorter or equal in length to the pattern spectra of the images in the database. If the queries are longer or shorter, some additional processing is required.

If the lengths of bitstrings of the query and the image in the database with which it is compared are not equal then the shorter one is appended with 0's until the lengths become equal. The bitstrings are then converted into their corresponding BIN numbers. The pseudo code for handling longer and shorter queries is given in Algorithm 3.2.

bs (q): bitstring of the query image

bs (I) : bitstring of the image in the database

concatenate(s, n): concatenates string s with n

|s|: indicates the length of string s

if(|bs(q)| > |bs(I)|) then

begin

 while (bs (I) < bs (q))

 bs (I) = concatenate (bs(I) ,0)

```

        end while
    end

    else(|bs(q)| < |bs(I)| ) then
    begin
        while (bs(q) < bs(I))

            bs(q) = concatenate (bs(q),0)
        end while
    end
end

```

Algorithm 3.2: Handling longer and shorter queries

In the Algorithms 3.2, if the length of the bitstring query $bs(q)$ less than the length of the bitstring image in the database $bs(I)$ the query image is appended with the 0's else if the length of the bitstring query $bs(q)$ greater than the length of the bitstring image in the database $bs(I)$ the the length of image in the database is appended with the 0's.

3.6 Summary

This Chapter discussed the fundamental definitions of the Mathematical Morphology like dilation, erosion, open and close. Morphological operator Pattern Spectrum was stated of the extraction of shape based features. Concept of *bins* was given for the fast and efficient image retrieval.

CLUSTERING TECHNIQUES FOR CONTENT BASED IMAGE RETRIEVAL

4.1 Introduction

Clustering is a useful technique for discovery of data distribution and patterns in the underlying data [28]. The goal of clustering is to discover dense and sparse regions in a data set. Previous approaches for similarity search, for example comparison of BIN numbers as discussed in the Chapter 3, do not adequately consider the case that the data set can be too large to fit in the main memory. In particular, they do not recognize that the problem must be viewed in terms of how to work with limited resources. In other words, it is to do clustering with accuracy as high as possible while keeping the I/O costs low. It is to be noted that the basic principle of clustering hinges on a concept of distance metric or similarity metric.

In the current system, the idea behind doing clustering is that the images belonging to the same cluster are similar or relevant to each other when compared to images belonging to different cluster. Instead of comparing the *bin* number of the query image with the *bin* numbers of all the images in the database, the images in the database are clustered, the representative *bin* number of each cluster is found and the query image is compared with only the cluster representatives. For clustering the images, the *bitstrings* discussed in Chapter 3, that represent the images are clustered. Each bitstring is considered as an object in the database.

4.2 Applications of Clustering Techniques

Clustering has wide applications in the areas like

Pattern Recognition: To identify the pattern of the given data and to find the relationships between the objects.

Spatial Data Analysis: To create maps in GIS by clustering feature spaces and to detect spatial clusters and explain them in spatial data mining.

Image Processing: In the area of image processing to process the images for different needs and application.

Marketing: Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs.

Insurance: Identifying groups of motor insurance policy holders with a high average claim cost.

Earthquake studies: Observed earthquake epicenters should be clustered along continent faults.

4.3 Classification of Clustering Techniques

Clustering techniques [28] are classified into two categories: *Partitional* and *Hierarchical*. Their definitions are as follows

4.3.1 Partitional Algorithms

Given a database of n objects, a partitional clustering algorithm constructs k partitions of the data, where each cluster optimizes a clustering criterion, such as the minimization of the *sum of squared distance from the mean* within each cluster.

One of the issues with such algorithms is their high complexity, as some of them exhaustively enumerate all possible groupings and try to find the global optimum. Even for a small number of objects, the number of partitions is huge. So, common solutions start with an initial, usually random partition and proceed with its refinement. A better practice would be to run the partitional algorithm for different sets of initial points (considered as representatives) and investigate whether all solutions lead to the same final partition. Partitional Clustering algorithms try to locally improve a certain criterion. First, they compute the values of the similarity or distance, they order the results, and pick the

one that optimizes the criterion. Hence, the majority of them could be considered as greedy-like algorithms.

4.3.2 Hierarchical Algorithms [28]

Hierarchical algorithms create a hierarchical decomposition of the objects. They are classified into either *agglomerative (bottom-up)* or *divisive (top-down)*:

- **Bottom-up:** It starts with each object being a separate cluster itself, and successively merges groups according to a distance measure. The clustering may stop when all objects are in a single group or at any other point the user wants. These methods generally follow a greedy-like bottom-up merging.
- **Top-down:** It follows the opposite strategy. They start with one group of all objects and successively split groups into smaller ones, until each object falls in one cluster, or as desired. Divisive approaches divide the data objects in disjoint groups at every step, and follow the same pattern until all objects fall into a separate cluster. This is similar to the approach followed by divide-and-conquer algorithms. Most of the times, both approaches suffer from the fact that once a merge or a split is committed, it cannot be undone or refined.

Partitional and hierarchical methods can be integrated. This would mean that a result given by a hierarchical method could be improved via a partitional step, which refines the result via iterative relocation of points. Other classes of clustering algorithms are given in the next subsection.

Apart from the two main categories of partitional and hierarchical clustering algorithms, many other methods have emerged in cluster analysis, and are mainly focused on specific problems or specific data sets available. Those are like *Density-based algorithms* based on connectivity and density functions. *Grid-based algorithms* based on a multiple-level granularity structure. *Model-based* is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other. *Categorical Data Clustering* is specifically developed for data, where Euclidean, or other numerical-oriented, distance measures cannot be applied. In the literature [28], we find the different clustering algorithms, which implements these properties.

But what makes a clustering algorithm efficient and effective? The answer is not clear. A specific method can perform well on one data set, but very poorly on another, depending on the size and dimensionality of the data as well as the objective function and structures used. Regardless of the method, researchers agree that characteristics of a good clustering technique are: *Scalability, Analyze mixture of attribute types, Find arbitrary shaped clusters, Minimum requirements for input parameters, Handling of noise, Sensitivity to the order of input records, High dimensionality of data.*

4.4 Proposed Clustering Techniques

In the present proposed system checks the applicability of two of the clustering techniques like Robust hierarchial Clustering with linKs (ROCK) and Clustering Using REpresentatives (CURE) which works well for the pattern recognition(discussed in Chpater 3) and integrates the partitional and hierarchical methods. This would mean that a result given by a hierarchical method can be improved via a partitional step, which refines the result via iterative relocation of points.

4.4.1 ROCK (RObust hierarchical Clustering with linKs)[29]

ROCK (RObust hierarchical Clustering with linKs)[29,31], a robust hierarchical-clustering algorithm is an adaptation of an agglomerative hierarchical clustering algorithm. It heuristically optimizes a criterion function defined in terms of the number of “links” between tuples. Informally, the number of links between two tuples is the number of common neighbors they have in the dataset. Starting with each tuple in its own cluster, they repeatedly merge the two closest clusters till the required number of clusters remains. Since the complexity of the algorithm is cubic in the number of tuples in the dataset, they cluster a sample randomly drawn from the dataset, and then partition the entire dataset based on the clusters from the sample.

Definitions:

Let O be an object representing an image in the database. Since a binary string represents each image in the database, the similarity metric used in the system is

$$sim(O_i, O_j) = |O_i \cap O_j| / |O_i \cup O_j| \quad \text{-----(4.1)}$$

where in equation 4.1, $sim(O_i, O_j)$ be a similarity function that is normalized and captures the closeness between the pair of objects O_i and O_j . The similarity function sim assumes values between 0 and 1.

Neighbourhood: Two images are said to be neighbours if

$$sim(O_i, O_j) \geq \theta \quad \text{-----(4.2)}$$

In equation 4.2, θ is defined as the threshold between 0 and 1.

Link: The link between two images is the number of common neighbours between the two images.

The first k objects are used as the sample data set. After the initial clustering, the goodness of each non-sample object with the cluster records is calculated. The object is assigned to the cluster with maximum goodness. The algorithm stops when all the objects in the data set are assigned to some cluster. The pseudo code of the ROCK[31] algorithm is given in Algorithm 4.1(a) and 4.1(b).

Input

Sample: Set of n sampled objects to be clustered

k : Number of clusters required

Data Structures

$q[s]$: for each object s in Sample $q[s]$ maintains a local heap structure of objects linked to s in decreasing order

Q : indicates the number of clusters that are created so far out of Sample (initially every object in the Sample is a cluster. So the size of Q is n)

Algorithm

Similarity = computeSimilarity (List, n)

Neighbour = computeNeighbours(Similarity, n)

Link = computeLinks(Neighbour, n)

for each s in Sample do

$q[s]$ = storeLinks(Link, n)

end

while (size(Q) > k) do

select a cluster u from Q and select the nearest element v from $q[u]$

delete v from Q

```

w = merge(u ,v)
for each x  $\in$  q[u]  $\cup$  q[v] do
    link[u, v] = link[x ,u] + link[x ,v]
    delete u from q[x]
    delete v from q[x]
    insert w to q[x] and compute g(x ,w)
    insert x to q[w] and g(x ,w)
    update (Q ,x, q[x])
end
insert(Q, w, q[w])
end

```

Algorithm 4.1(a): Pseudo code for ROCK Algorithm

Procedure for computing links

```

begin
    compute nbrlist[Oi] for every point i in S
    set link[Oi ,Oj] to zero for all i,j

    for i = 1 to n do
        N = nbrlist[Oi]
        for j = 1 to |N| - 1 do
            for l = j + 1 to |N| do
                link[N[Oj],N[Ol]] = link[N[Oj],N[Ol]] + 1
            end
        end
    end
end
end

```

Algorithms 4.1(b): Contd. Psuedo code to find links in ROCK algorithm

Initially all the objects are considered as separate clusters. A cluster is randomly selected. To this cluster all the images that are in its neighborhood are added. This procedure is repeated for each image in that neighborhood in turn. The above procedure is continued iteratively till there are no more images to fall into this cluster. Again a randomly selected image from the remaining images is selected to start a new cluster and the above procedure is repeated. This process is repeated till all the images fall in one or the other cluster.

The main drawback of the ROCK algorithm is that there exists a single cluster, which contains as high as 80% of images of the collection. This is because of the linear growth of the clusters. Linear growth is primarily due to the relaxed condition that existence of even a single image in a cluster in the neighbourhood list of the image under consideration is sufficient to join this new image into the cluster.

4.4.2 CURE (Clustering Using Representatives)[30]

In Clustering Using Representatives (CURE)[30], a constant number c of well scattered points in a cluster are first chosen. The scattered points capture the shape and extent of the cluster. The chosen scattered points are next shrunk towards the centroid of the cluster by a fraction c_r . These scattered points after shrinking are used as representatives of the cluster. The clusters with the closest pair of representative points are the clusters that are merged at each step of CURE's hierarchical clustering algorithm. The scattered points approach employed by CURE alleviates the shortcomings of both the all-points as well as the centroid-based approaches. It enables CURE is less sensitive to outliers since shrinking the scattered points toward the mean dampens the adverse effects due to outliers outliers are typically further away from the mean and are thus shifted a larger distance due to the shrinking. Multiple scattered points also enable CURE to discover non-spherical clusters like the elongated clusters shown in the centroid-based algorithm.

Clusters identified can be tuned by varying α : between 0 and 1. CURE reduces to the centroid based algorithm if $\alpha = 1$, while for $\alpha = 0$, it becomes similar to the all-points approach. CURE's hierarchial clustering algorithm uses space that is linear in the

input size n and has a worst-case time complexity of $O(n^2 \log n)$. For two dimensions, the complexity can be shown to further reduce to $O(n)$. Thus, the time complexity of CURE is no worse than that of the centroid based hierarchical algorithm. Instead of preclustering with all the data points, CURE begins by drawing a random sample from the database. We show, both analytically and experimentally, that random samples of moderate sizes preserve information about the geometry of clusters fairly accurately, thus enabling CURE to correctly cluster the input. In particular, assuming that each cluster has a certain minimum size. In order to further speed up clustering, CURE first partitions the random sample and partially clusters the data points in each partition. After eliminating outliers, the preclustered data in each partition is then clustered in a final pass to generate. The procedure to form clusters is shown in Algorithm 4.2(a).

```

procedure cluster(S, k)
begin
    T := buildkdtree(S)
    Q := buildheap(S)
    while size(Q) > k
    do {
        u := extractmin(Q)
        v := uclosest
        delete(Q, v)
        w := merge(u, v)
        deleterep(T, u); deleterep(T, v); insertrep(T, w)
        w.closest := x /* x is an arbitrary cluster in Q */
        for each x ∈ Q
        do {
            if dist(w, x) < dist(w, w.closest)
                w.closest := x
            if x.closest is either u or v
                {
                    if dist(x, x.closest) < dist(x, w)
                        x.closest := closestcluster(T, x, dist(x, w))
                    else x.closest := w
                    relocate(Q, x)
                }
            else if dist(x, x.closest) > dist(x, w)
                {
                    x.closest := w
                    relocate(Q, x)
                }
        }
        insert(Q, w)
    }
end

```

Algorithm 4.2(a): Pseudo code for CURE Clustering algorithm

After the clusters were formed the closed clusters to be merged together. The pseudo code for merging the clusters formed is shown in Algorithm 4.2(b).

```

procedure merge( $u, v$ )
begin
   $w := u \cup v$ 
   $w.mean = u \cup v$ 
  tmpSet := 0
  for  $i := 1$  to  $c$ 
  do {
    maxDist := 0
    foreach point  $p$  in cluster  $w$ 
    do {
      if  $i=1$ 
        minDist := dist( $p, w.mean$ )
      else
        minDist := min{dist( $p, q$ ) :  $q \in tmpSet$ }
      if (minDist  $\geq$  maxDist)
      {
        maxDist := minDist
        maxPoint :=  $p$ 
      }
    }
    tmpSet := tmpSet  $\cup$  {maxPoint}
  }
  for each point  $p$  in tmpSet do
     $w.rep := w.rep \cup \{ p + cw*(w.mean-p) \}$ 
return  $w$ 
end

```

Algorithm 4.2(b): Procedure for merging clusters

This CURE algorithm works well for large size of the databases. Though ROCK works efficiently for image retrieval .It slows down when the database image size is very large. The performance evaluations of these clustering algorithms were calculated.

4.5 Summary

This chapter undertaken a brief overview of different clustering techniques and checks the applicability of the clustering techniques to the the present proposed system for the Content Based Imaged Retrieval.The performance evaluation details of two clustering techniques like ROCK(RObust hierarchical-Clustering with linKs) and CURE(Clustering Using Representatives) are also included.

DESIGN AND IMPLEMENTATION OF CONTENT BASED IMAGE RETRIEVAL SYSTEM

5.1 Introduction

The present chapter deals with the design and development of the overall system. It starts with the different phases of the system and continues with the details of the working system. It discusses the reason behind choosing a particular programming language and the different classes, different data structures. It then discusses the approach taken to design the system and the functionality of each of the components of the system. It also discusses the software and hardware requirements of the system with which it can perform optimally.

5.2 Different phases of Proposed Retrieval System

The developed proposed system is broadly categorized into three phases which can be viewed in Figure 5.1

- Extraction of feature vectors from the images in the database
- Cluster the images in the database
- Retrieving the cluster of images closest to the query image

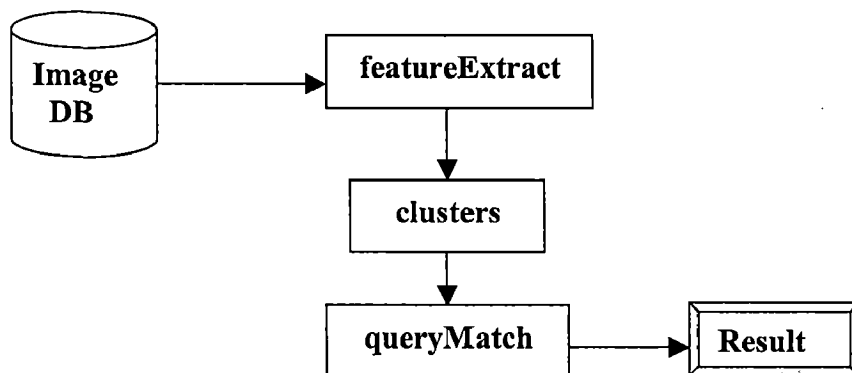


Figure 5.1: Different Phases of the Retrieval System

In the first phase, that is, feature vectors extraction is done using Mathematical Morphological operations *open distribution* and *pattern spectrum* discussed in chapter 3. In the second phase, that is, clustering the images is done using the Data Mining clustering algorithms ROCK and CURE algorithms are used, discussed in chapter 4. The third phase, that is, query image processing is implemented using pattern spectrum and the *bins* concepts discussed in chapter 3 is used for efficient image retrieval.

5.3 Design and Working of Proposed Retrieval System

The details for the design and working of the present system is shown in Figure 4.2.

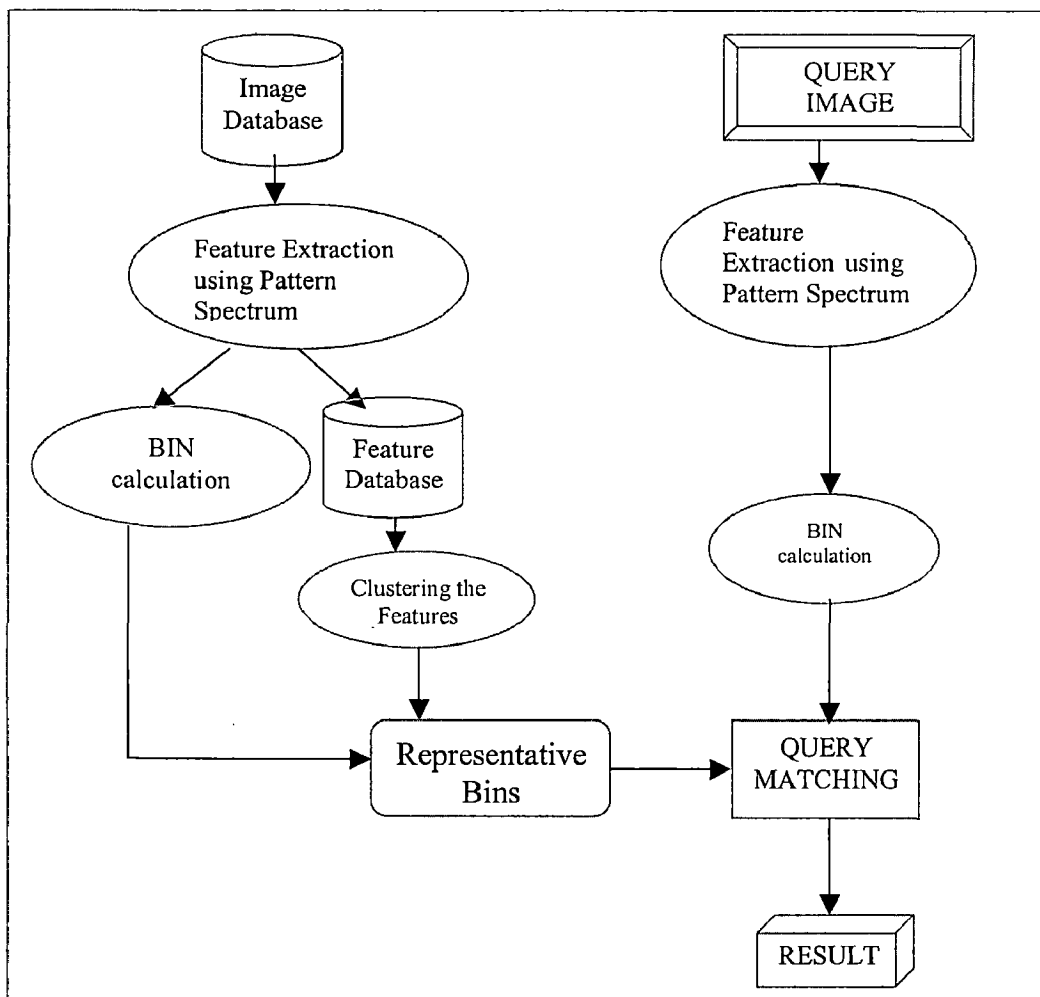


Figure 5.2: Design and Working of Proposed Retrieval System

The components and the working of the proposed retrieval system shown in Figure 5.2 is discussed in detail in section 5.3.1.

5.3.1 Extraction of feature vector

The feature vector of an image I is calculated using the pattern spectrum $PS(I)$. Initially the image is represented as a set consisting of 1's and 0's, say A . The gray-scale images are converted into binary using Thresholding concept, as discussed in chapter 3.

Using a particular structuring element B with initial size r the opening distribution of A is found. The opening distribution for a particular size s is nothing but the area of A when it is opened with B of size $s(s \geq r)$. The pattern spectrum $PS(I)$ is calculated by using the formula:

$$PS(I) = Area(A \circ_n B) - Area(A \circ_{(n+1)} B) \text{ for } n \geq r \geq 0 \quad \text{-----}(5.1)$$

From the equation 5.1 it is clear that the process of calculating *pattern spectrum* is stopped when the *Area* of the image becomes zero for a particular size of the structuring element.

Using the above procedure, the *pattern spectrum* of all the images in the database and also of the query image is found. The *pattern spectrum* of the query image is then compared with the *pattern spectrum* of all the images in the database and the images with most similar shapes are retrieved.

In order to fasten the comparison process the feature vector is then extracted from the pattern spectrum using the indexing scheme described in chapter 3. The bitstrings that are produced from the pattern spectra are the feature vectors and are stored in a table called *feature vector table*. Also the BIN numbers of all the images in the database are generated and stored for further use in the query matching process.

5.3.2 Clustering the images in the database

The clustering of the images is obtained indirectly by clustering the feature vectors of the images. The *feature vector table (fvt)* is used for this purpose. The data mining clustering algorithms discussed above are used for clustering the *fvt*. The goal of this phase is to form the clusters of images in the database. It is assumed that each record in the *fvt* as one image.

The clustering module outputs the user-required number of clusters with similar images in each cluster. In other words, the clustering algorithms are applied such that images with similar shapes fall under same cluster. ROCK and CURE algorithms are used as discussed in the previous chapter.

5.3.3 Query Matching

When a query image with a particular emotion is given, the system has to retrieve those images in the database that are more similar in shape to the query image. That is, the cluster whose features are closer to the query image feature vector is retrieved.

Given the query image, the feature vector is found using pattern spectrum. The bitstring that represents the image is converted into its corresponding BIN number. The concept of handling longer and shorter queries is also considered at this point. The representative bins of all the clusters formed are found. The representative BIN number of a cluster is the result of bit wise-OR ing the bin numbers of all the images in the cluster. The cluster of images whose representative bin is more nearer to the query bin is retrieved.

It is observed that the retrieved cluster is the one that contains images with similar shapes as that of the query image.

The pseudo code of the overall system is given in Algorithm 5.1.

Input

imgdatabase : image collection used for the purpose

Data Structures

image : two-dimensional array that contains the image as a binary set

PS: dynamic array that contains the pattern spectrum

fname : list containing the files of the images in the image collection

bs: dynamic array that contains the bitstring

bins: list containing the bin numbers of the image collection

clusters: dynamic array containing the clusters of images

query: contains the user specified query image

fvt: feature vector table

Functions

extractImage() : converts an image into a binary set

computeSpectrum(): computes the pattern spectrum of an image

featureVector(): extracts the feature vector of an image

computebin() : calculates the bin number of an image

ROCK(): implements the ROCK algorithm

CURE: implements the CURE algorithm

findrepresent: finds the representative of a cluster

compare: compares two bin numbers

findlargest: finds the index with largest value in an array

Algorithm

begin

 for each *i* in the *imgdatabase* do

image = *extractImage*(*fname*(*i*))

PS = *computeSpectrum*(*image*)

bs = *featureVector*(*PS*)

 insert *bs* into *fvt*

bins(*i*) = *computebin*(*bs*)

 end

```

clusters = ROCK(fvt)
clusters = CURE(clusters,fvt)
image = extractImage(query)
PS = computeSpectrum(fname(query))
bs = featureVector(PS)
querybin = computebin(bs)
for each I in clusters do
    repbin = findrepresent(clusters(i),bins)
    similarity(i) = compare(repbin,querybin)
end
finalmatch = findlargest(similarity)
end

```

Algorithm 5.1: Pseudo code for the Proposed Retrieval System

5.4 Implementation Details

This section gives the implementation details of the Retrieval System by stating the different classes and data structures used to implement the system and the choice of the programming language used for the implementation

5.4.1 Choice of Programming Language

Most of the Content Based Retrieval systems both commercial and academic are generally coded in 'C', 'C++', JAVA or Perl. The reason behind this may be the flexibility and the ease, which these languages provide while dealing with images. The present retrieval system is implemented in Matlab and Java language.

The reason behind choosing Matlab for feature extraction of images using Mathematical Morphology is the availability of the Image Processing Toolbox. Here emphasis was on the logic of the program and not on learning a new language or an environment. The JAVA language is used for implementing clustering algorithms and to retrieve the relevant images from the image database. The reason behind using JAVA[32] is simply the in-built methods, data structures. Other features provided by Java are

simplicity, security and portability. It captures the essence of Object-oriented programming and provides the required security to data.

5.4.2 Different Classes and Data-Structures

There are mainly three classes that correspond to the various phases of the system designed (described in Figure 5.1): *featureExtract*, *clusters*, *queryMatch*.

The class *featureExtract* is used to extract feature vector from an image. The extraction of feature vector is done using Matlab functions for query image processing using Mathematical Morphology was given below:

read(): The function reads image from graphics file. If the image is a binary image, it returns the set of 1's and 0's equivalent to the image. If the image is a gray-scale image, it displays the histogram of the image and returns the image set after performing Thresholding.

show(): The function *show(I)* displays the intensity image *I* with *N* discrete levels of gray image .

strel(): The function create morphological structuring element. *strel('SE',size)* creates a structuring element with the specified *SE*. *size* is a matrix containing 1's and 0's.

strel('square',W): It creates a square structuring element whose width is *W* pixels. *W* must be a nonnegative integer scalar.

strel('rectangle',MN):It creates a flat, rectangle-shaped structuring element, where *MN* specifies the size.*MN* must be a two-element vector of nonnegative integers. The first element of *MN* is the number of rows in the structuring element neighborhood; the second element is the number of columns.

strel('diamond',R): It creates a flat, diamond-shaped structuring element, where *R* specifies the distance from the structuring element origin to the points of the diamond. *R* must be a nonnegative integer scalar.

strel('disk',R): It creates a disk-shaped structuring element, where *R* specifies the radius. *R* must be a nonnegative integer.

strel('line',LEN,DEG): It creates a flat linear structuring element with length *LEN* and *DEG* specifies the angle (in degrees) of the line as measured in a counterclockwise

direction from the horizontal axis. *LEN* is approximately the distance between the centers of the structuring element members at opposite ends of the line.

strel('octagon',R): It creates a flat octagonal structuring element. with the specified size, *R*. Where *R* is the distance from the structuring element origin to the sides of the octagon, as measured along the horizontal and vertical axes.

dilate(): The function dilates the given image. *dilate (I,SE)* dilates the grayscale, binary image *I*, returning the dilated image. *SE* is a structuring element object, or array of structuring element objects, returned by the *strel* function.

erode(): The function erodes the give image. *erode(I,SE)* erodes the grayscale, binary image, returning the eroded image. *SE* is a structuring element object, or array of structuring element objects, returned by the *strel* function.

open():The function opens the image. *open(I,SE)* performs morphological opening on the grayscale or binary image with the structuring element *SE*. *SE* must be a single structuring element object, as opposed to an array of objects.

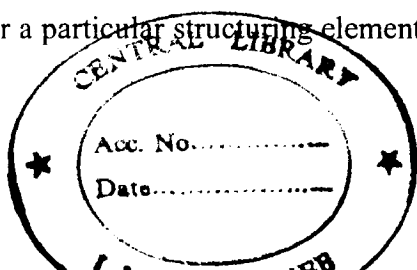
close(): The function *close(I,SE)* performs morphological closing on the grayscale or binary image with the structuring element *SE*. *SE* must be a single structuring element object, as opposed to an array of objects.

area(): The image set *A* is opened with a particular structuring element starting from size until the area of the resulting image after opening is zero. The opening is done using Minkowski algebra [18]. The various Structuring elements used are: Square, Linear, and Diagonal. For each structuring element, a separate function is defined. Finally the function returns the opening distribution of an image with a particular structuring element.

pattern (): The function takes the opening distribution of the image as input. It calculates the pattern spectrum of the image and finds the feature vector *fv* from the pattern spectrum as described in Chapter 5. The feature vector is the bitstring equivalent to the pattern spectrum.

createbins (): The function takes the feature vector of an image as input. From the bitstring, it calculates the equivalent bin number of the image.

The class creates a representative file to store the pattern spectra of all the images for a particular structuring element. Using this file, the feature vectors of all the images in



the database are found. Finally a *feature vector table* is created which consists of the bitstrings of all the images in the image database. Each record in the feature vector table represents an image in the database. Also the bin numbers of all the images are stored in a data structure called *bins*, which is a dynamic Linked List.

The class *clusters* is used to collect the images in the database. The major functions of this class are:

algoRock (): The function takes the sample set to be clustered from the feature vector table, the number of clusters required as input.

The threshold value for the similarity function is chosen apriori so that images with similar emotions fall onto the same cluster. The choice completely depends on the nature and size of the image database considered for which much experimentation is necessary. The function returns the clusters of images formed out of sample data set.

goodness (): The function takes the records from the feature vector table which are not in the sample data set as input. It categorizes the records into the clusters already formed based on the goodness criteria. The function returns the clusters of all the images in the database.

algoCure() : The function takes the clusters formed by ROCK algorithm as input. The epsilon and the threshold values are chosen apriori such that the error introduced by ROCK algorithm in clustering is overcome and images with more similar emotions fall under the same cluster. The function returns the clusters of all the images in the database. The class finally stores all the clusters formed in a data structure called *clusters*, which is a dynamically created two-dimensional array.

The class *queryMatch* is used to retrieve images that are similar to the query image. The major functions of this class are:

representbins (): The function takes the data structures *clusters* and *bins* as input. It calculates the representative bin numbers of all the clusters. The representative bin of a cluster is obtained by bit wise-OR ing the bin numbers of all the images in the cluster.

querybin (): The function takes the query image as input. It then calculates the feature vector of the query image using the *featureExtract* class. The bin number corresponding

to the feature vector is found using the *createbins()* function of the same class. The function returns the bin number of the query image.

match (): The function takes the representative bins of the clusters formed, the bin number of the query image as input. It then finds the closest representative bin by using the similarity metric, intersection. The cluster with images of similar emotion to that of query image is retrieved.

5.4.3 System Requirements

The proposed system requires the following minimum hardware and software requirements.

- Intel Pentium, 1500 MHz
- Windows NT/2000
- SQL Server 2000
- 256 MB RAM
- Microsoft Java VM
- JDK 1.3

Internet Explorer 5 is required because of the Microsoft Java virtual Machine (VM). An older Microsoft Java VM will significantly reduce the performance of the software. It is highly recommended that searches be conducted on the highest specification machine available. SQL Server 2000 is used as a back-end for storing the feature vector table.

The images in the collection should be 256 color-Bitmap images or in Graphic Interchange Format in Gray Scale Format. The preferred size of the images is 256*256(width and height). This requirement is just a temporary one and in further development of the project, this requirement can be removed.

5.5 Summary

This chapter gives the pseudo code of the implementation details of the different phases of the proposed system like feature extraction and clustering algorithms used and it also discusses about the system requirements.

EXPERIMENTAL RESULTS

6.1 Introduction

This chapter shows various experimental results of the retrieval system. The process, which the system follows in the retrieval, is discussed for a particular image. Finally the results given by the retrieval system when the user gives various queries are seen.

6.1.1 Test Case 1

This section discusses the process, which the system follows in retrieving an image (Fig. 9 Chapter 4). First the feature vectors for every image in the database are calculated. The extraction of the feature vector of an image in the database is as follows.

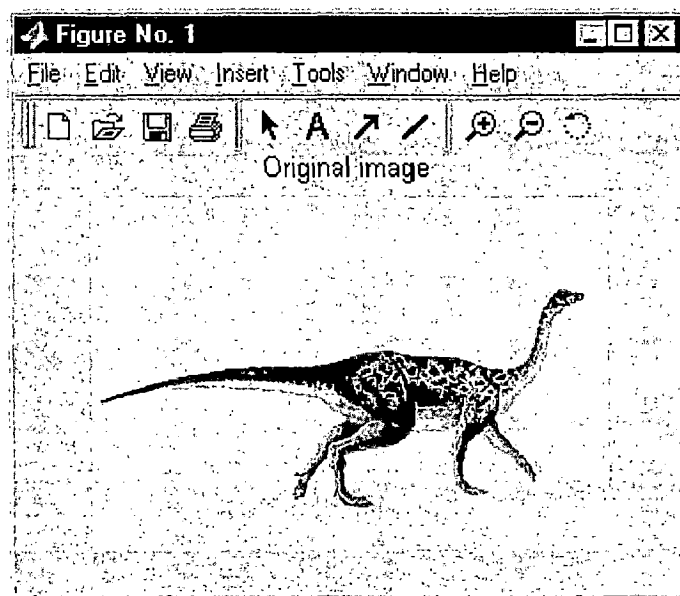


Figure 6.1: Original Query Image 1

In Figure 6.1 shows the query image is used to retrieve the relevant images from the database based on the shape as a derived feature. The process of retrieving the feature vectors for the query image using Mathematical Morphology is as follows.

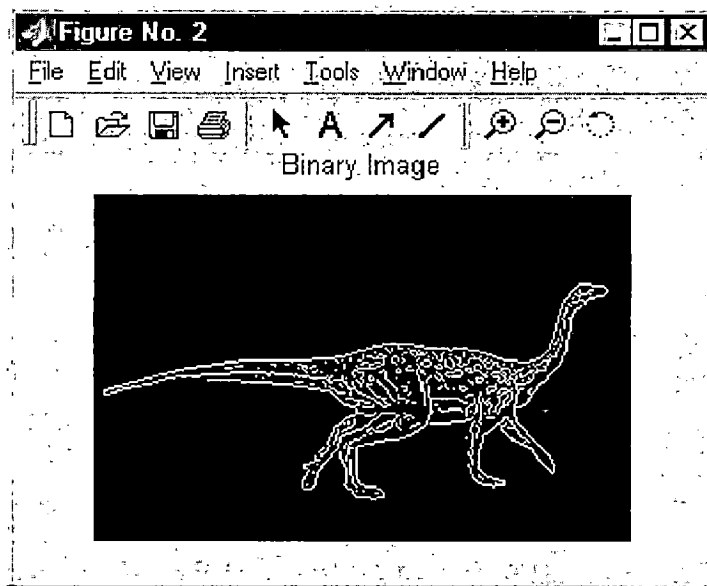


Figure 6.2: Binary Image for the given Query Image 1

Figure 6.2 shows the binary image obtained from the query image for the process of retrieving the feature vectors using Mathematical Morphology.

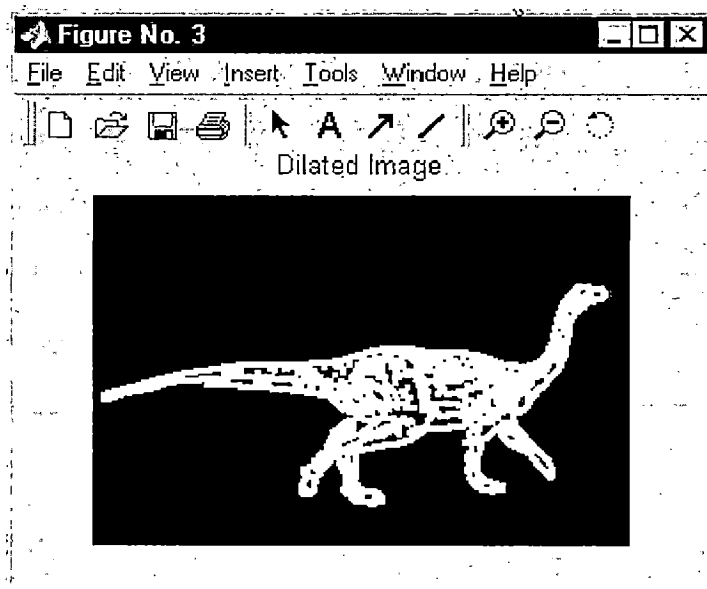


Figure 6.3: Dilated Image for the given Query Image 1

In Figure 6.3 Image obtained after recursively applying the *dilation* operation (discussed in chapter 2) on the query image by opening the size of the structuring element from 1,2,3...till the area of the image becomes zero.

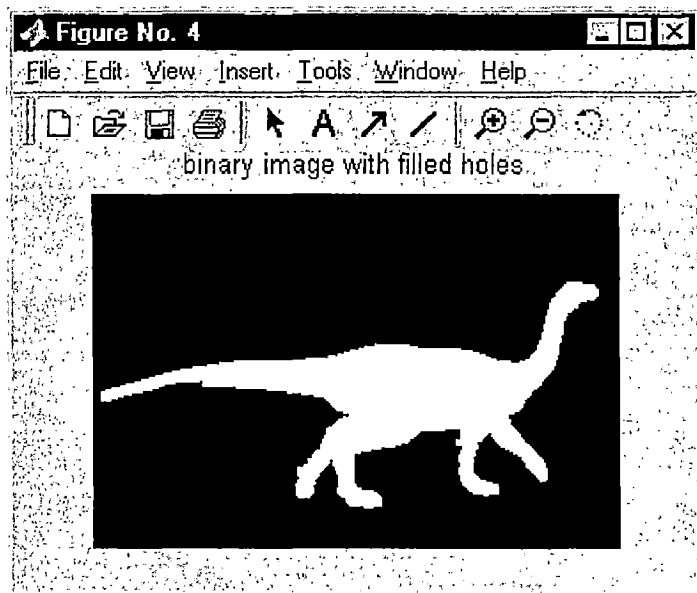


Figure 6.4: Binary Image after holes filled for the given Query Image 1

After applying Dilation operation on the above query image holes or blobs obtained in the image will be filled by applying *fillholes* operation which is shown in the Figure 6.4.

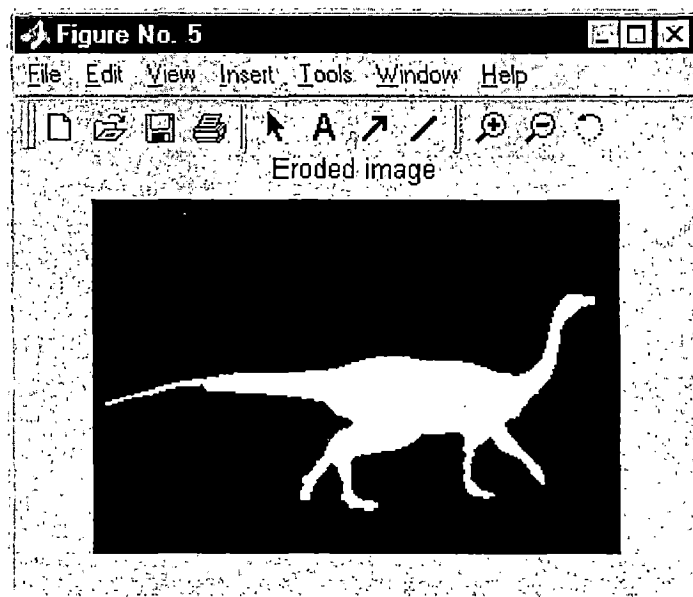


Figure 6.5: Erosion operation on the Query Image 1

Figure 6.5 shows the Eroded Image obtained after applying Mathematical operation *erosion* for the image in Figure 6.4

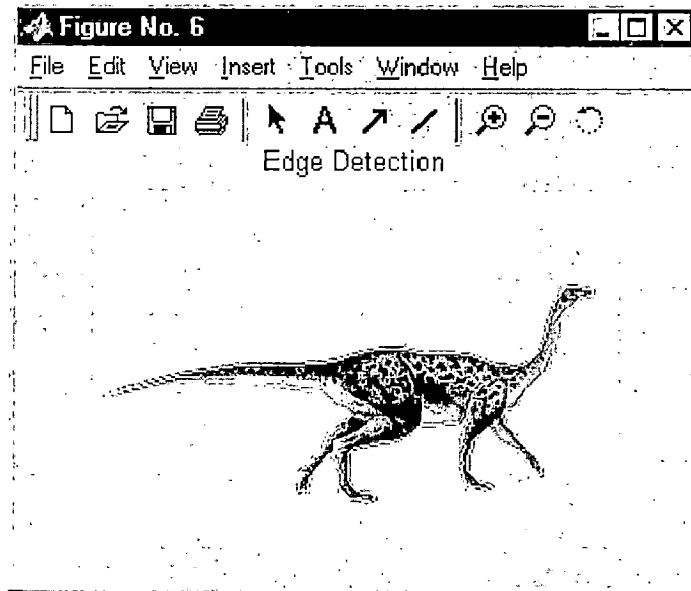


Figure 6.6: Edge Detection technique to identify the object

After a sequence of applications of dilation and erosion the object in the image will be identified which is shown in Figure 6.6

After identifying the object the pattern spectrum of the identified object will be obtained by applying the by recursively applying Mathematical Morphology operation *open* on the object and incrementing the size of the square structuring element from 1,2 ,3...till the area of the object identified becomes zero (discussed in chapter 2).

6.1.2 Pattern spectrum

The pattern spectrum obtained for the above image is shown below. This is the feature vector obtained from the query, which is used for the similarity search for retrieving the relevant images from the image database.

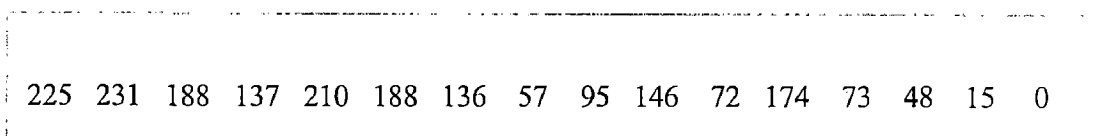


Figure 6.7: Pattern spectrum for the given query image 1

6.1.3 Bitstring

The pattern spectrum shown in Figure 6.7 (discussed in chapter 3) is used for obtaining the Bitstring shown in Figure 6.8. Bitstring is used for making the search simpler in retrieving the relevant images from the image database.

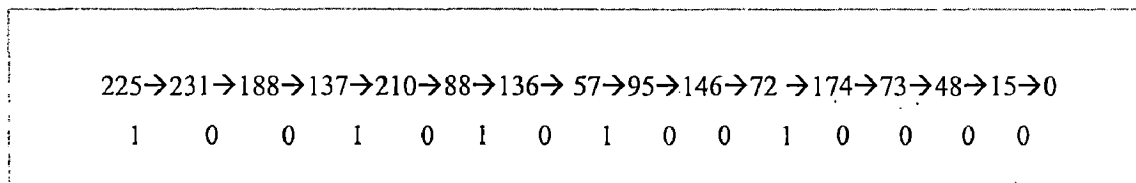


Figure 6.8: Bitstring for the Pattern Spectrum of Query Image 1

6.1.4 Output 1

The pattern spectrum of the query image obtained from the above is compared with the Pattern spectrum of the images stored in the image database and the relevant images are retrieved which are shown in Figure 6.8

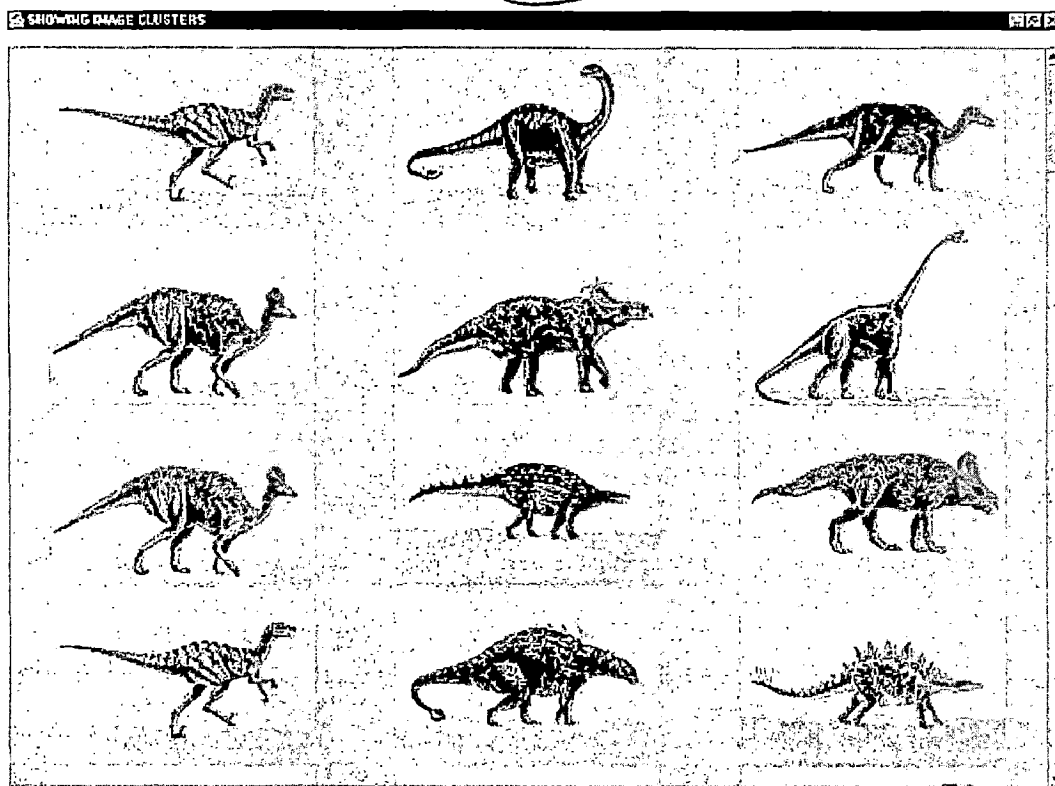


Figure 6.9: Similar Images Retrieved for the given Query Image 1

Figure 6.9 shows the relevant images obtained for the given query. The ROCK-clustering algorithm calculates the similarities of the images.

6.2 Test Case 2

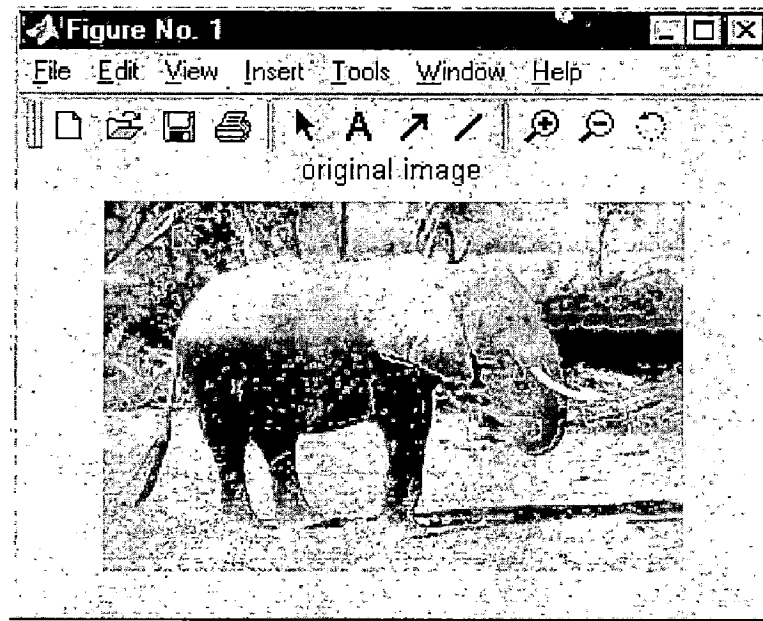


Figure 6.10: Original Query Image 2

In Figure 6.10 shows the query image is used to retrieve the relevant images from the database based on the basis of shape as a derived feature. The procedure shown in *Test case 1* in repeated here and the pattern spectrum obtained for the query image shown in Figure 6.11 is given below.

6.2.1 Pattern spectrum

The pattern spectrum obtained for the above image is shown below. The feature vector obtained from the query image is used for the similarity search for retrieving the relevant images from the image database.

43	717	587	389	793	762	692	295	598	583	321	607	510	362	220	0
----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	---

Figure 6.11: Pattern Spectrum for the given Query Image 2

6.2.2 Bitstring

The pattern spectrum shown in Figure 6.10 is used for obtaining the Bitstring. Obtaining Bitstring from the pattern spectrum is shown in Figure 6.12. Bitstring is used for making the search simpler in retrieving the relevant images from the image database.

430	→	717	→	587	→	389	→	793	→	762	→	692	→	295	→	598	→	583	→	321	→	607	→	510	→	362	→	220	→	0
1		0		0		1		0		0		0		1		0		0		1		0		0		0		0		0

Figure 6.12: Bitstring for the pattern spectrum of the given Query Image 2

6.2.3 Output 2

The Bitstring of the um of the query image obtained from the pattern spectrum shown in Figure 6.12 is used for comparing the Bitstrings of the images stored in the image database. Retrieval of relevant images (shown in Figure 6.13) from the database is obtained by applying ROCK-clustering algorithm.

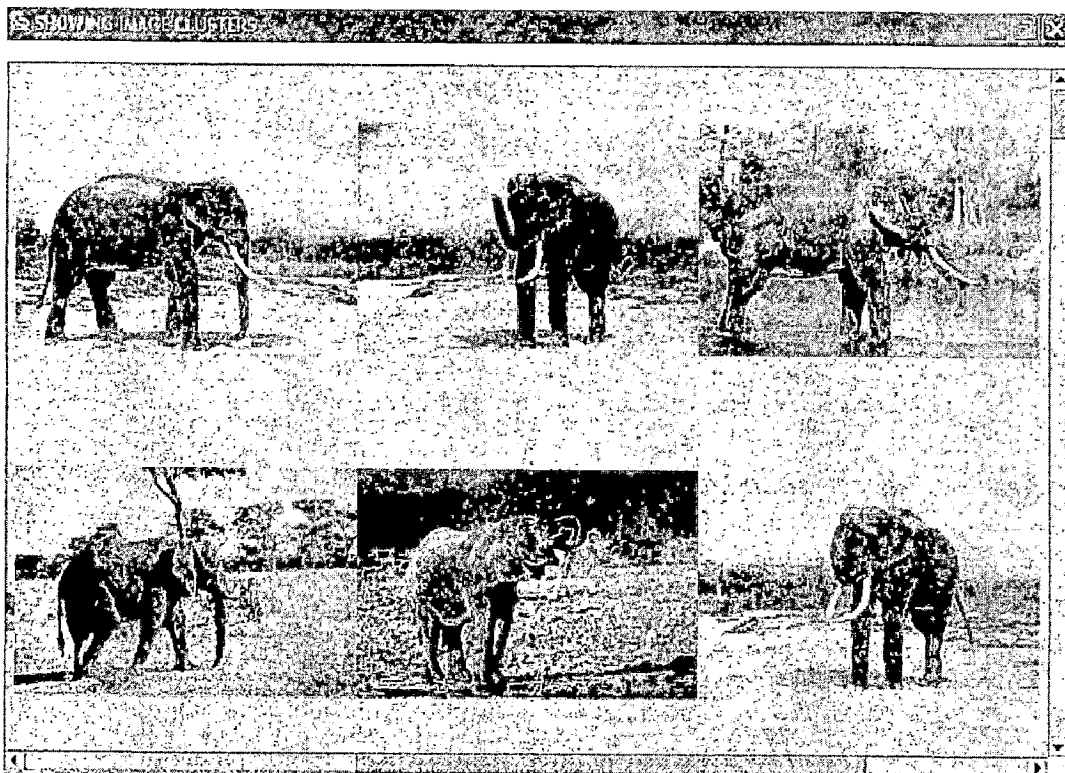


Figure 6.13: Similar Images Retrieved for the given Query Image 2

6.3 Performance Evaluation of Clustering Techniques

Two of the clustering techniques like Robust hierarchical Clustering with links (ROCK) and Clustering Using Representatives were (CURE) used individually to retrieve the relevant images and the performance evaluation of these two clustering techniques were used for image retrieval. The performance evaluation of these two algorithms were calculated in terms of time vs. domain size

6.3.1 Test case 1

Here performance evaluation of the ROCK and CURE were calculated by executing these two algorithms individually with different domain sizes and the time taken for these two algorithms is compared which is shown in Figure 6.14 (a) and Figure 6.14(b).

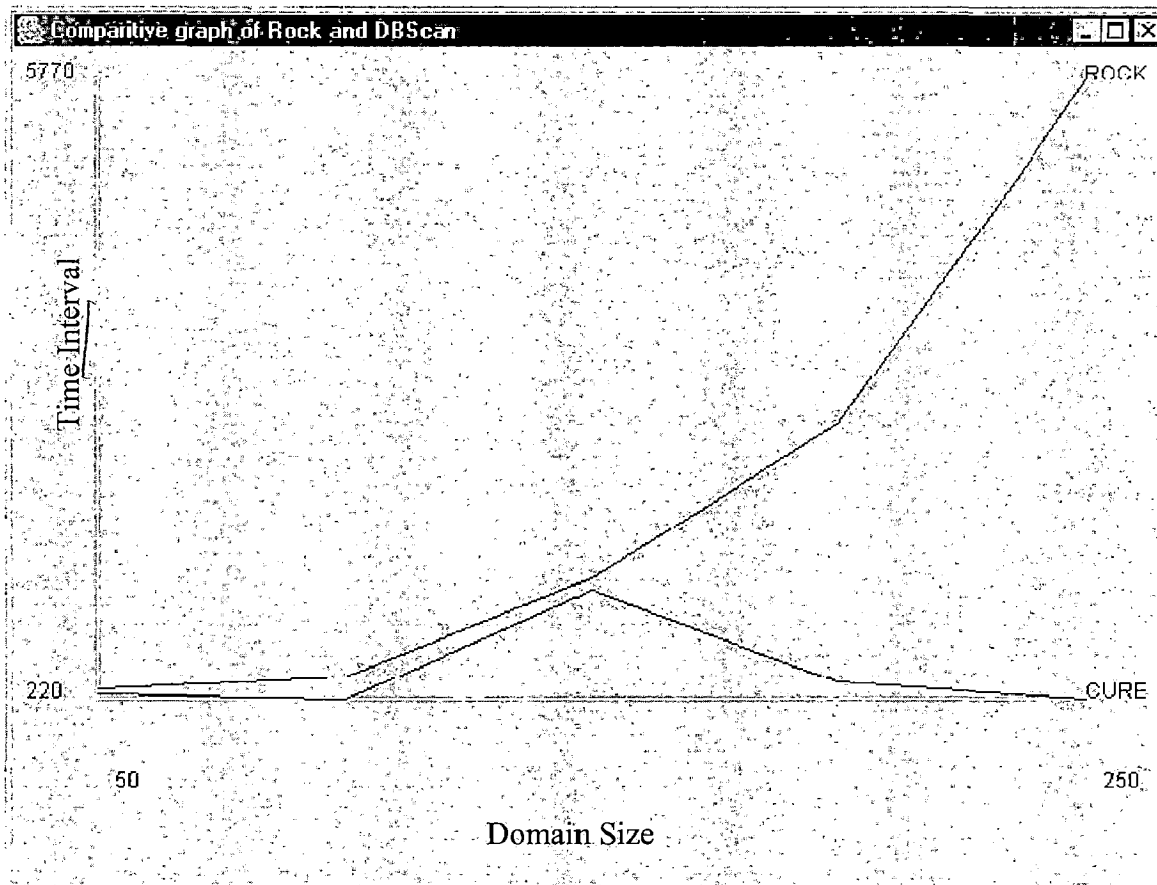


Figure 6.14(a): Performance evaluation of ROCK and CURE algorithms with Domain size on X axis and Time interval on Y axis

6.3.2 Test case 2

ROCK and CURE algorithms compared with domain sizes of 50 to 500 on X axis and with time interval of the corresponding on y axis is shown in Figure 6.14(b).

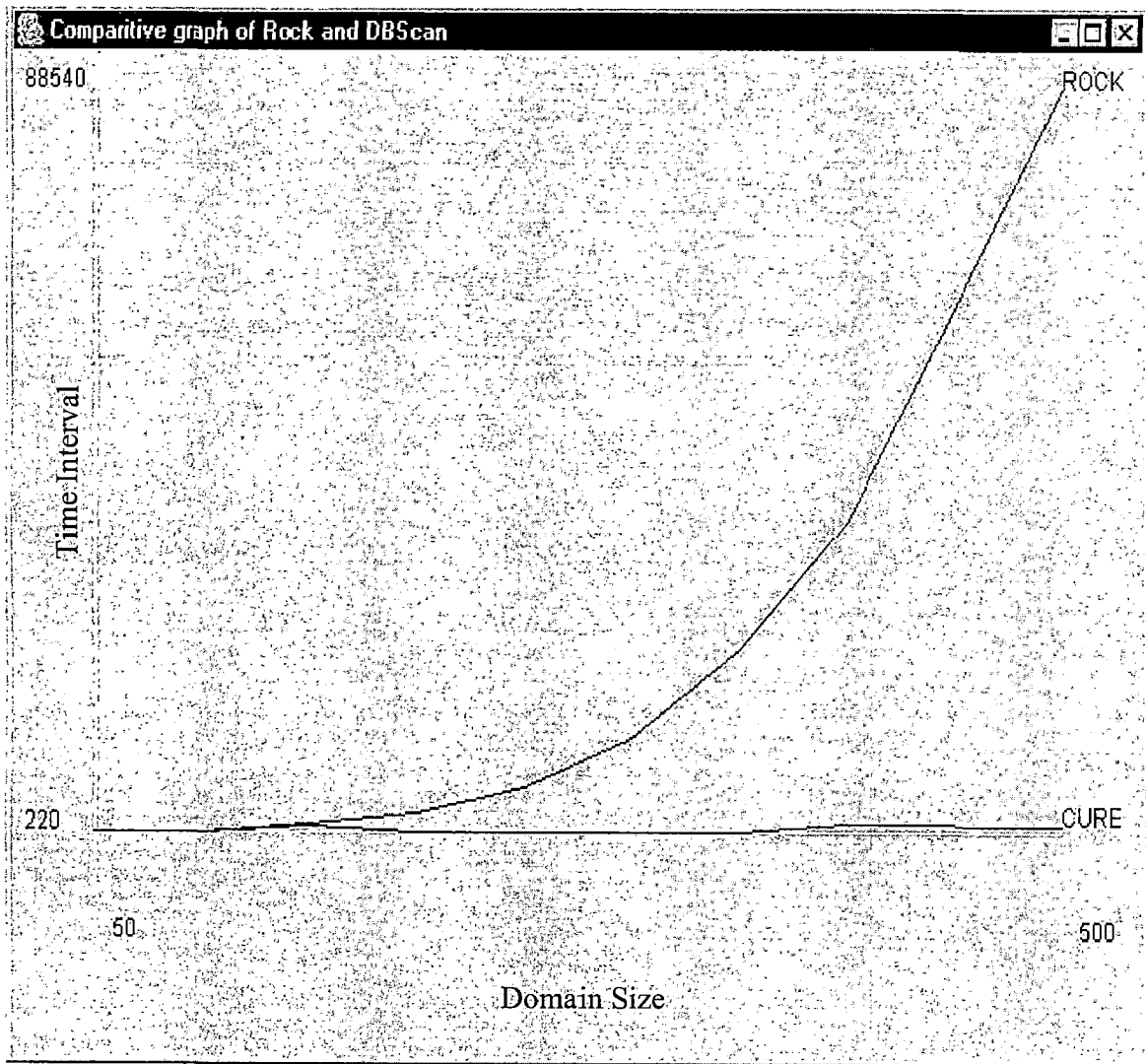


Figure 6.14(b): Performance evaluation of ROCK and CURE algorithms with Domain size on X axis and Time interval on Y axis

Form the Figure 6.14(a) and Figure 6.14(b) it can be deduced that CURE works well for the large data base sizes. The present system is able to retrieve relevant images from the image database based on the shape of the image. This system will not work when the objects in the images are with different orientations.

6.4 Summary

This Chapter discussed the experimental results of the Proposed Content Based Image Retrieval System by applying the Mathematical Morphological operations on the images for the feature extraction and performance evaluation of the data mining clustering algorithms like RObust hierarchial Clustering with linKs(ROCK) and the Clustering Using REpresentatives (CURE) were calculated in terms of sample size and expected time taken were calculated and proved that CURE works faster with large database sizes.

CONCLUSION AND FUTURE WORK

In this dissertation work an effort has been made to retrieve the relevant images from the image database based on the shape as the feature vector of the query image using Mathematical Morphology and the performance evaluation of the clustering techniques like RObust hierarchial Clustering with linKs (ROCK) and the Clustering Using Representatives (CURE) are compared in terms of Domain size and Time. The experimental results prove that the proposed retrieval system works well for the extraction of the features using Mathematical Morphology.

This Retrieval System can be useful in applications where content based retrieval is used to extract queries like – ‘*find a particular objects with a particular shape*’, that is, retrieve the images based on content specified with the derived property – *shape*.

Future Scope

There is a lot of scope for enhancements in the system developed. The present system uses the concept of Mathematical Morphology operation Pattern Spectrum to extract feature of the images but this is sensitive to certain factors like differences between images in foreground and background illuminations, the orientation of objects etc., Uniform conditions for all the images in the database can be maintained to get more accurate results.

The present system considers some predefined shapes of the structuring elements. More complex structuring elements like circle, rhombus, ellipse and 3 dimensional figures can be used for experimentation. Also some part of the shapes of the image objects like shape of the given query image can also be used as structuring elements for the extraction of features.

Apart from the RObust hierarchial Clustering with linKs (ROCK) and Clustering Using Representatives (CURE) clustering algorithms, there are a number of clustering algorithms using Data-Mining techniques, which can be experimented with, to increase the efficiency of the retrieval system.

Another important enhancement that can be done with the present system is including the concept of Relevance Feedback Mechanism. Whenever the system presents to the user a set of images considered to be similar to the provided query, the user can pick among them the images he or she considers most relevant to the submitted query.

REFERENCES

- [1] Eakins John and Graham Margaret, "A Review of Content-Based Image Retrieval Systems," *Institute for Image Data Research, University of Northumbria at Newcastle*, <http://www.unn.ac.uk/iidr/research/cbir/report.html>, January 1999.
- [2] R.M.Haralick, S.R.Sternberg and X. Zhuang, "Image Analysis Using Mathematical Morphology," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, pp.532-550, July 1987.
- [3] S. S. Wilson, "Theory of Matrix Morphology," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.14, pp.78-85, June 1992.
- [4] J.Song and E.J.Delp, "The Analysis of Morphological Filters with multiple structuring elements," *Computer Vision Graphics and Image Processing*, vol.50, pp.71-85, June 1990.
- [5] P.Maragos, "Pattern Spectrum and Multiscale Shape Representation," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.11, pp.701-716, July 1989.
- [6] Van den Boomgard, Rein, and Richard van Balen, "Methods for Fast Morphological Image Transforms Using Bitmapped Images," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, Vol. 54, pp. 252-254, May 1992.
- [7] F. Meyer, "Automatic Screening of Cytological Specimens," *Computer Vision, Graphics and Image Processing*, vol. 35, pp. September 1986.
- [8] Michael M Skolnick, "Application Of Morphological Transformation to the analysis of Two-dimensional Electrophoretic Gels of Biological Materials,"

Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing, vol.35, pp. 306-332, May 1986.

- [9] H.Boerner,“Feature Extraction by Grayscale Morphological Openings – A Comparison to DOG Filters,” International Workshop on Industrial Applications of Machine Intelligence and Vision (MVI-89), Tokyo, April 1989.
- [10] C. Bhagvati, M.Skolnick, and A.Grivas, “Morphological Analysis of Pavement Surface Condition,” *Technical Report No.91-8*,pp.1-15, April 1991.
- [11] H.J.A.M. Heijmans and C. Ronse, “The Algebraic Basis of Mathematical Morphology I. Dilations and Erosions,” *Computer Vision, Graphics, and Image Processing*, vol.50, June 1990.
- [12] Manjunath and Ma, “Texture Features for Browsing and Retrieval of Image Data”, *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.18, pp.837-842, August 1996.
- [13] Eamonn J.keogh and Michael J.Pazzani, ”An indexing Scheme for Fast Similarity Search in Large Time Databases,” *University of California,Proceedings of the 11th Internaional Conference on Scientific and Statistical Database Management*,pp.56-65,June 1999.
- [14] Niblac, Sethi, and Jain, “The QBIC project: querying images by color, texture and shape,” in *Storage and Retrieval for Image and Video Databases VI*, Proc SPIE 3312, pp.150-161 ,January1998.
- [15] Gupta, “The Virage image search engine: an open framework for image management,” in *Storage and Retrieval for Image and Video Databases IV*, Proc SPIE 2670, pp.76-87,May 1996.

- [16] Pentland, "Photobook: tools for content-based manipulation of image databases," *International Journal of Computer Vision* , pp.233-254, Year 1996.
- [17] Ogle, V E and Stonebraker, "Chabot: retrieval from a relational database of images" *IEEE Computer*, pp.40-48, March 1995.
- [18] Ma W Y and Manjunath, "Netra: a toolbox for navigating large image databases" *Proc IEEE International Conference on Image Processing (ICIP97)*, pp.568-571, Year 1997.
- [19] Beigi, Sethi, I K and Jain, "MetaSEEk: a content-based meta-search engine for images" in *Storage and Retrieval for Image and Video Databases VI Proc SPIE 3312*, pp.118-128, February 1998.
- [20] Smith J R and Chang S F, "Querying by color regions using the VisualSEEk content-based visual query system," *Intelligent Multimedia Information Retrieval*, AAAI Press, Menlo Park, CA, pp.23-41, April 1997.
- [21] Markkula, M and Sormunen, "Searching for photos – journalists' practices in pictorial IR", presented at *The Challenge of Image Retrieval* research workshop, Newcastle upon Tyne, 5 February 1998 .
- [22] Beckmann, "The R*-tree: an efficient and robust access method for points and rectangles" *ACM SIGMOD Record* , pp.322-331, June 1990.
- [23] Jacobs, C. E, "Fast Multiresolution Image Querying" *Proceedings of SIGGRAPH 95, Los Angeles*, ACM SIGGRAPH Annual Conference Series, pp.277-286, January 1995.

- [24] Jain, A K and Vailaya, "Image retrieval using color and shape," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp.1233-1244, May 1996.
- [25] Manjunath, B S and Ma, "Texture features for browsing and retrieval of large image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp.837-842, August 1998.
- [26] Wang, Wiederhold, "Wavelet-Based Image Indexing Techniques with Partial Sketch Retrieval Capability," *Proceedings of the Fourth Forum on Research and Technology Advances in Digital Libraries*, pp.633-648, May 1998.
- [27] Scassellati, Niblack, W R and Jain, "Retrieving images by 2-D shape: a comparison of computation methods with human perceptual judgements" in *Storage and Retrieval for Image and Video Databases II*, Proc SPIE 2185, pp.2-14, June 1994.
- [28] Sudipto Guha, Rajeev Rastogi, "Data Clustering Techniques", *Qualifying Oral Examination Paper Periklis, University of Toronto, Department of Computer Science*, March 11, 2002.
- [29] Sudipto Guha, Rajeev Rastogi, "ROCK: A Robust Clustering Algorithm for Categorical Attributes", *In Proceedings of the 15th International Conference on Data Engineering, IEEE Press, Sydney, Australia*, pp. 512-521, March 1999.
- [30] Sudipto Guha, Rajeev Rastogi and Kyuseok Shim, "CURE: An Efficient Clustering Algorithm for Large Databases", *In Proceedings of the International Conference on Management of Data, (SIGMOD), ACM Press, volume 27(2) of SIGMOD Record*, pp.73-84, Seattle, WA, USA, pp.1-4, June 1998.
- [31] Pujari Arun K: "Data Mining Techniques," *Second Edition, Universities Press (India) Limited*, Year 2001.

- [32] Schildt Herbert, Patrick Naughton, "The Complete Reference: Java," *Fourth Edition*, Tata McGraw-Hill Publishing Company Limited, Year 2001.
- [33] P.Maragos, "Pattern Spectrum and Multiscale Shape Representation," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol.11, pp.701-716, July 1989.
- [34] Carson C S , "Region-based image querying" in *Proceedings of IEEE Workshop on Content-Based Access of Image and Video Libraries* , San Juan, Puerto Rico, pp.42-49, June 1997.
- [35] Gudivada V N and Raghavan,"Design and evaluation of algorithms for image retrieval by spatial similarity," *ACM Transactions on Information Systems*,pp.115-144 ,May 1995.