A CONTENT-BASED IMAGE RETRIEVAL SYSTEM FOR TEXTURE AND COLOR QUERIES

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Eyüp Sabri Konak

August, 2002

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Uğur Güdükbay (Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. İbrahim Körpeoğlu

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Dr. Veysi İşler

Approved for the Institute of Engineering and Science:

Prof. Dr. Mehmet B. Baray
Director of the Institute Engineering and Science

ABSTRACT

A CONTENT-BASED IMAGE RETRIEVAL SYSTEM FOR TEXTURE AND COLOR QUERIES

Eyüp Sabri Konak
M.S. in Computer Engineering
Supervisors: Assist. Prof. Dr. Uğur Güdükbay and
Assoc. Prof. Dr. Özgür Ulusoy
August, 2002

In recent years, very large collections of images and videos have grown rapidly. In parallel with this growth, content-based retrieval and querying the indexed collections are required to access visual information. Two of the main components of the visual information are texture and color. In this thesis, a content-based image retrieval system is presented that computes texture and color similarity among images. The underlying technique is based on the adaptation of a statistical approach to texture analysis. An optimal set of five second-order texture statistics are extracted from the Spatial Grey Level Dependency Matrix of each image, so as to render the feature vector for each image maximally informative, and yet to obtain a low vector dimensionality for efficiency in computation. The method for color analysis is the color histograms, and the information captured within histograms is extracted after a pre-processing phase that performs color transformation, quantization, and filtering. The features thus extracted and stored within feature vectors are later compared with an intersection-based method. The system is also extended for pre-processing images to segment regions with different textural quality, rather than operating globally over the whole image. The system also includes a framework for object-based color and texture querying, which might be useful for reducing the similarity error while comparing rectangular regions as objects. It is shown through experimental results and precision-recall analysis that the content-based retrieval system is effective in terms of retrieval and scalability.

Keywords: Texture Analysis, Color Histograms, Texture Similarity Measurement, Content-Based Image Retrieval, Image Databases.

ÖZET

RENK VE DOKU ARAMALARI İÇİN İÇERİK-TABANLI BİR GÖRÜNTÜ ERİŞİM SİSTEMİ

Eyüp Sabri Konak
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticileri: Yard. Doç. Dr. Uğur Güdükbay ve
Doç. Dr. Özgür Ulusoy
Ağustos, 2002

Son yıllarda, çok büyük resim ve video veritabanları oluşagelmiştir. Bu büyümeye paralel olarak, görsel bilgiye erişebilmek için içerik-tabanlı erişim ve indekslenmiş veritabanları üzerinde arama yapabilme ihtiyaçları doğmaktadır. Görsel bilginin ana bileşenlerinden ikisi renk ve dokudur. Bu tezde, resimler arasında renk ve doku benzerliğini hesaplayan bir içerik-tabanlı erişim sistemi sunulmaktadır. Kullanılan teknik, doku analizinde istatistiksel yaklaşıma dayanmaktadır. Her resim için özellikler vektörünü, aynı anda hem mümkün olduğunca bilgilendirici kılmak, hem de düşük vektör boyutlarının sağlayacağı etkin hesaplama imkanından yararlanabilmek için, her resmin Uzaysal Gri Düzey Bağıntı Matrisinden beş adet optimal ikinci-seviyeden doku istatistiği hesaplanmaktadır. Renk analizinde kullanılan yöntem renk histogramları olup bunlarda bulunan bilginin elde edilmesi, renk dönüşümü, basamaklandırma ve filtrelemeyi içeren bir önişleme fazı sonucunda olmuştur. Böylelikle elde edilen ve özellik vektörlerinde saklanan özellikler daha sonra bir kesişim yöntemiyle karşılaştırılmıştır. Sistem ayrıca, görüntünün tümü üzerinde işlem yapmak yerine resmi önce farklı doku özelliğindeki bölgelerine ayıran başka bir önişleme fazıyla genişletilmiştir. Sisteme dahil edilen bir diğer özellik de, dikdörtgensel bölgeleri nesne olarak ele almanın yol açabileceği benzerlik hatalarını azaltabilmeye yönelik nesne-bazlı renk ve doku arama için bir yön haritası sunmasıdır. Deney sonuçları ve analizler bu içerik-tabanlı erişim sisteminin erişim ve ölçeklenebilirlik bakımından etkin olduğunu göstermektedir.

Anahtar sözcükler: Doku Analizi, Renk Histogramları, Doku Benzerlik Ölçümü, İçerik-Tabanlı Görüntü Erişimi, Görüntü Veritabanları.

Acknowledgement

I would like to express my gratitude to my supervisors Assist. Prof. Dr. Uğur Güdükbay and Assoc. Prof. Dr. Özgür Ulusoy for their instructive comments in the supervision of the thesis. I am also grateful to the members of the examining committee, Assist. Prof. Dr. İbrahim Körpeoğlu and Assoc. Prof. Dr. Veysi İşler, for their keen interest and comments.

To my parents and sisters...

Contents

1 Introduction **1**

1.1 Motivation 2

1.2 Organization of the Thesis 3

2 Querying by Texture and Color Content **4**

2.1 Query-by-Texture Content 4

2.1.1 Feature Extraction Methods for Texture Analysis 5

2.1.2 A Statistical Approach to Texture Feature Extraction . . . 7

2.2 Query-by-Color Content 12

2.2.1 Preliminaries 13

2.2.2 Color Content Extraction 14

2.3 Related Work 16

3 Content-Based Image Retrieval System **19**

3.1 Overview of BilVideo System 20

3.2 The Graphical User Interface 22

3.3 Feature Extraction from Images 24

3.3.1 Histogram Intersection Method for Similarity Calculation . 27

3.4 A Framework for Object-Based Texture Querying 28

3.4.1 Object Extraction Process 29

3.4.2 Framework Architecture 30

3.4.3 Capturing Motion within Video Data 30

4 Performance Experiments **33**

4.1 Evaluating Effectiveness 33

4.2 Query Examples 34

5 Conclusions and Future Work **42**

Bibliography **43**

A The Object Extractor Tool **48**

A.1 Design Principles 48

A.2 The Object Extraction Algorithm 50

B Brodatz Texture Database **54**

B.1 Brodatz Texture Index 54

B.2 Brodatz Texture Images 58

List of Figures

2.1 Computation of W-SGLDM and H-SGLDM for P(1, 0) 8

2.2 Transformation, Quantization and Color Median Filtering of *tiger* Image. (a) Original image. (b) Image produced by applying RGB to HSV color transformation and quantization. (c) Image produced after applying color median filtering. 13

2.3 A Collection of Color Median Filtering Types. (a) 3×3 box (b) 5×5 box (c) 5×5 octagonal (d) 7×7 box. 14

2.4 The Color Histogram of *tiger* Image in HSV Color Space. 15

3.1 BilVideo System Architecture 21

3.2 The Graphical User Interface of the Content-Based Image Retrieval System. 23

3.3 Feature Extraction Alternatives, (a) Fully Automatic Extraction, and (b) Semi-Automatic Extraction. 25

3.4 Energy and Entropy Functions, (a) Energy Low and Entropy High, and (b) Energy High and Entropy Low. 26

3.5 Contrast and Inverse Difference Moment Functions, (a) Contrast High and Inverse Difference Low, and (b) Contrast Low and Inverse Difference Moment High. 26

3.6 The Circular Position Vector with Radius 2. 27

3.7 Timeline of Object-Based Querying with Object Extractor. 30

3.8 A Set of Frames from `Cat.avi`. 32

4.1 Interpolated Precision Recall Graph with `D23.gif` as Query Texture Image. 35

4.2 Interpolated Precision Recall Graph with `D39.gif` as Query Texture Image. 35

4.3 Interpolated Precision Recall Graph with `D52.gif` as Query Texture Image. 36

4.4 Interpolated Precision Recall Graph with `D55.gif` as Query Texture Image. 36

4.5 Interpolated Precision Recall Graph with `D58.gif` as Query Texture Image. 37

4.6 Interpolated Precision Recall Graph with `D68.gif` as Query Texture Image. 37

4.7 Interpolated Precision Recall Graph with `D94.gif` as Query Texture Image. 38

4.8 Interpolated Precision Recall Graph with `D110.gif` as Query Texture Image. 38

4.9 Sample Queries: Brodatz Album Codes and Similarity Scores for Each Query Result. (a) Query 1, (b) Query 2, (b) Query 3, (d) Query 4. 39

4.10 Query Execution Times 40

A.1 The Overall Architecture of The *Object Extractor* 49

A.2 The Graphical User Interface of The *Object Extractor*. 50

A.3 The Snapshots of The *Object Extractor*. 52

A.4 The Snapshots of The *Object Extractor*. 53

B.1 Brodatz Texture Database Images, D1–D12. 59

B.2 Brodatz Texture Database Images, D13–D24. 60

B.3 Brodatz Texture Database Images, D25–D36. 61

B.4 Brodatz Texture Database Images, D37–D48. 62

B.5 Brodatz Texture Database Images, D49–D60. 63

B.6 Brodatz Texture Database Images, D61–D72. 64

B.7 Brodatz Texture Database Images, D73–D84. 65

B.8 Brodatz Texture Database Images, D85–D96. 66

B.9 Brodatz Texture Database Images, D97–D108. 67

B.10 Brodatz Texture Database Images, D109–D112. 68

List of Tables

3.1 Similarity Table for the Frame Sequence of `Cat.avi`. 31

4.1 Query Execution Times for Eight Brodatz Texture Images. 41

Chapter 1

Introduction

In recent years, very large collections of images and videos have grown rapidly. In parallel with this growth, content-based retrieval and querying the indexed collections are required to access visual information. As a powerful technique, content-based retrieval systems have to provide easy-to-index data structures as well as faster query execution facilities. In order to index and answer the queries that the users pose to seek visual information, the *content* of the images and videos must be extracted.

The visual content, or generally content, of images and video frames can be categorized as follows: spatial, semantic, and low-level. Since video data has a time dimension, the spatio-temporal content of a video data is also considered. However, extracting spatio-temporal content requires sophisticated techniques, thus do not included in the categorization. The spatial content of an image is the relative positioning of the objects residing in the image. The semantic content is the actual meaning of the image that a user captures when he/she looks at the image. The low-level content is formed by low-level features such as color, shape, and texture. These three features are considered important underlying primitives in human visual perceptions of the real world. Various methods exist in the literature for indexing the images based on these low-level features.

1.1 Motivation

Texture is one of the crucial primitives in human vision and texture features have been used to identify contents of images. Examples are identifying crop fields and mountains from aerial image domain. Moreover, texture can be used to describe contents of images, such as clouds, bricks, hair, etc. Both identifying and describing characteristics of texture are accelerated when texture is integrated with color, hence the details of the important features of image objects for human vision can be provided. One crucial distinction between color and texture features is that color is a point, or pixel, property, whereas texture is a local-neighborhood property. As a result, it does not make sense to discuss the texture content at pixel level without considering the neighborhood.

In this thesis, a content-based retrieval system is presented that supports querying with respect to texture and color low-level features. The main motivation for using texture is the identifying and describing characteristics of texture feature. Since the power of texture increases when combined with color, the content-based retrieval system provides techniques for querying with respect to texture and color in an integrated manner.

For texture feature, an optimal set of second order texture statistics functions are used, which are well-known and widely used in most of the systems due to their expressive power and less computational complexity. Besides, color histograms are used for color feature with a proper pre-processing phase. This pre-processing phase includes color space transformation, color quantization and color neighborhood ranking to smooth the color distribution and reduce the dimension of the color histogram. Color median filtering is employed, which is a well-known neighborhood ranking method, in the system.

The content-based image retrieval system proposed in this thesis includes the following unique features:

- In addition to texture and color extraction from images as a whole, semi-automatic and fully automatic feature extraction methods are included.

For the former, the user is provided a drawing facility that is used for the specification of regions of interest. For the latter, the system tries to capture a rectangular region that represents the texture content of the image.

- A new type of position vector, Circular Position Vector, is designed and implemented to empower the expressive power of the second order texture statistics functions.
- Histogram Intersection method is employed for similarity calculations as a result of texture vector and color histogram comparisons between database images and query image.
- A framework for texture and color querying with respect to the object regions in images is designed. This framework uses similar concepts of object extraction tools to determine object regions in images, and performs the computations for the pixels in the object regions. This type of querying is not included in most of the systems, hence our system provides more options in query specification.

1.2 Organization of the Thesis

The thesis is organized as follows: Chapter 2 presents the techniques that are used for querying by texture and color content of images. A survey on some of the existing content-based retrieval systems is also provided. In Chapter 3, the content-based retrieval system that is developed in the scope of the thesis is presented. Not only the implemented features but also a general framework for proper extensions of the system is also discussed. The performance experiments of the content-based retrieval system are given in Chapter 4 and finally, Chapter 5 concludes the thesis. Appendix A is dedicated to a detailed explanation of an object extraction tool, and Appendix B discusses Brodatz Texture Database.

Chapter 2

Querying by Texture and Color Content

Due to its broad fields of application in medical imaging, satellite photography, remote sensing, industrial quality inspection, etc., texture has been a popular research area within image processing since early 70's.

Being of paramount importance in human visual perception, together with color, texture constitutes one of the two basic features on which content-based image retrieval is expected to operate. This chapter presents a content-based image retrieval system that can perform similarity matches on the texture feature of images.

2.1 Query-by-Texture Content

Albeit our intuitive grasp of the concept, there is no formal definition of *texture* in the literature. This lack of a satisfactory formal definition is due to the nature of the term, evading such a definition. Nevertheless, the problem of retrieving similar textures falls within the category of statistical pattern recognition, and

as idiosyncratic of problems within this category, any content-based image retrieval system can be considered as roughly consisting of two major subsystems: a *Feature Extraction subsystem* and a *Similarity Measurement subsystem* [8].

2.1.1 Feature Extraction Methods for Texture Analysis

In image processing literature, there exist three main approaches to the task of texture feature extraction: *spectral approach*, *structural (or syntactic) approach* and *statistical approach*.

2.1.1.1 Spectral Approach

The spectral approach to texture analysis deals with images in the frequency domain. Therefore, this approach requires Fourier transform to be carried out on the original images to acquire their corresponding representations in the frequency space.

The two-dimensional power spectrum of an image reveals much about the periodicity and directionality of its texture. For instance, an image of coarse texture would have a tendency towards low frequency components in its power spectrum, whereas another image with finer texture would have higher frequency components. Stripes in one direction would cause the power spectrum to concentrate near the line through the origin and perpendicular to the direction.

Fourier transform based methods usually perform well on textures showing strong periodicity, however their performance deteriorates as the periodicity of textures weakens [6].

Given such performance problems and the high computational complexity of the Fourier transform, the spectral approach is neither a very popular approach among researchers dealing with texture analysis, nor seems to be promising. In fact, Haralick [14], to whom we owe the early classification of approaches in textual analysis, does not even mention the spectral approach, but sticks to the

classification of all methods among the two other approaches: structural and statistical.

2.1.1.2 Structural Approach

The structural approach is based on the theory of formal languages: A textured image is considered as a sentence in a language, of which the alphabet is a set of texture primitives called *texton*s, constructed in accordance with a certain grammar determining the layout of such texture primitives within a pattern. Although the structural approach is very fruitful as long as it deals with deterministic patterns, the vast majority of textures found in the universe are not of such strict geometry but exhibit a level of uncertain random behavior [3, 21].

2.1.1.3 Statistical Approach

From the statistical point of view, an image is a complicated pattern on which statistics can be obtained to characterize these patterns. The techniques used within the family of statistical approaches make use of the intensity values of each pixel in an image, and apply various statistical formulae to the pixels in order to calculate feature descriptors.

Texture feature descriptors, extracted through the use of statistical methods, can be classified into two categories according to the order of the statistical function that is utilized: *First-Order Texture Features* and *Second Order Texture Features* [15].

First Order Texture Features are extracted exclusively from the information provided by the intensity histograms, thus yield no information about the locations of the pixels. Another term used for First-Order Texture Features is *Grey Level Distribution Moments*.

In contrast, *Second-Order Texture Features* take the specific position of a pixel relative to another into account. The most popularly used of second-order

methods is the *Spatial Grey Level Dependency Matrix (SGLDM)* method. The method roughly consists of constructing matrices by counting the number of occurrences of pixel pairs of given intensities at a given displacement.

2.1.2 A Statistical Approach to Texture Feature Extraction

2.1.2.1 Spatial Grey Level Dependency Matrices (SGLDM)

To extract location-based statistical values, it is necessary to devise a means of describing the location of each pixel, and its relative position to pixels of a certain intensity more accurately. Though known under various names, *Spatial Grey Level Dependency Matrix*, as called by Haralick [15], is a matrix comparing the intensities of all pixels.

SGLDM has the same size as the number of grey levels in an application. In a case where there exist 64 distinct grey levels, ***SGLDM*** shall be 64x64 matrix.

In addition to the ***SGLDM*** matrix, a position operator P needs to be defined. The operator P is now passed over the image. For each image pixel, the position operator needs to be evaluated. If the pixel intensity is i and the pixel intensity to which the operator points is j, the matrix element c_{ij} of ***SGLDM*** will be incremented by one. Hence ***SGLDM*** is a function of P and P is a function of distance d and angle θ, where the angle could be one specific direction, or a set of directions. For example:

- $P(1, 45)$ is the operator which points one to the right and one below (d=1, $\theta = 45$).
- $P(5, 90)$ is the operator which points five below (d=5, $\theta = 90$).
- $P(1, 0, 45, 90, 135)$ is the sum of the results with $P(1, 0)$, $P(1, 45)$, $P(1, 90)$ and $P(1, 135)$ (d=1, $\theta = 0$, 45, 90, 135).

Image:

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

P(1, 0) :

W-SGLDM

i \ j	0	1	2	3
0	2	0	0	0
1	2	2	0	0
2	1	0	3	0
3	0	0	1	1

H-SGLDM

i \ j	0	1	2	3
0	4	2	1	0
1	2	4	0	0
2	1	0	6	1
3	0	0	1	2

Figure 2.1: Computation of W-SGLDM and H-SGLDM for P(1, 0)

An example of how to calculate the ***SGLDM*** is given in Figure 2.1.

There exists a little inconsistency in the literature on how to calculate the ***SGLDM***. The above definition produces a ***SGLDM*** which is not normally symmetric and if the orientation of an object is important this type of calculation must be used. As Weszka et al. [33] is the most important paper using this non symmetric matrix, it will be called ***W-SGLDM***. If the orientation of the object cannot be controlled, or is of no interest, the way of definition can be slightly modified to produce a symmetric matrix. The only difference to the ***W-SGLDM*** is that when the position operator $P(d, \theta)$ is passed over the image the operator $P(d, 180 + \theta)$ is simultaneously passed over the image. To this matrix we shall refer as ***H-SGLDM***, since Haralick et al. always use the symmetric ***SGLDM*** [15]. It follows that:

$$H - SGLDM = W - SGLDM + W - SGLDM^T. \tag{2.1}$$

When the term ***SGLDM*** is used it applies to both ***SGLDM***'s. In order to continue with the extraction of statistics, we need to be able to treat the ***SGLDM*** matrix as a probability density function. Hence ***SGLDM*** should be normalized before proceeding with statistical analysis by dividing each entry in the matrix by the summation of all entries of the matrix.

2.1.2.2 Second Order Statistical Features

Although the preference for second-order texture features is generally of heuristic nature, its validity is also legitimized by *Julesz' Conjecture* which states that human eye is incapable of discriminating between textures that differ only in third or higher order statistics [20].

Haralick et al. define 14 second-order statistical functions that can be calculated on a ***SGLDM*** [15]. Where n is the number of grey levels in the image, and w is the width of the image in pixels, these statistical functions are:

1. Angular Second Moment (Energy)

$$\sum_{i,j} p(i,j)^2. \tag{2.2}$$

2. Contrast (Momentum)

$$\sum_{n} n^2 \sum_{i,j:|i-j|=n} p(i,j). \tag{2.3}$$

3. Correlation

$$\frac{\sum_{i,j} ijp(i,j) - \mu_x\mu_y}{\sigma_x\sigma_y}, \tag{2.4}$$

where μ_x, μ_y, σ_x, and σ_y are the means and standard deviations of $p_x(i) = \sum_k p(i,k)$ and $p_y(j) = \sum_k p(k,j)$.

4. Variance (Sum of squares)

$$\sum_{i,j} (i-\mu)^2 p(i,j), \tag{2.5}$$

where μ is the mean of the density function $p(i,j)$.

5. Inverse Difference Moment

$$\sum_{i,j} \frac{1}{1+(i-j)^2} p(i,j). \tag{2.6}$$

6. Sum Average

$$\sum_{i=2}^{2w} i p_{x+y}(i), \tag{2.7}$$

where $p_{x+y}(i) = \sum_{j,k;j+k=i} p(j,k)$.

7. Sum Variance

$$\sum_{i=2}^{2w} (i + \sum_{j=2}^{w} p_{x+y}(j) \log(p_{x+y}(j)))^2 p_{x+y}(i). \tag{2.8}$$

8. Sum Entropy

$$-\sum_{i=2}^{w} p_{x+y}(i) \log(p_{x+y}(i)). \tag{2.9}$$

9. Entropy

$$-\sum_{i,j} p(i,j) \log(p(i,j)). \tag{2.10}$$

10. Difference Variance

$$variance\ of\ p_{x-y}. \tag{2.11}$$

11. Difference Entropy

$$-\sum_{i=0}^{w-1} p_{x-y}(i)\log(p_{x-y}(i)). \tag{2.12}$$

12. Information Measure of Correlation

$$\frac{HXY - HXY1}{max(HX,\ HY)}, \tag{2.13}$$

where HX and HY are entropies of p_x and p_y, and

$$HXY = -\sum_{i,j} p(i,j)\log(p(i,j)), \tag{2.14}$$

$$HXY1 = -\sum_{i,j} p(i,j)\log(p_x(i)p_y(j)), \tag{2.15}$$

$$HXY2 = -\sum_{i,j} p_x(i)p_y(j)\log(p_x(i)p_y(j)). \tag{2.16}$$

13. Another Information Measure of Correlation

$$\sqrt{1 - e^{-2(HXY2 - HXY)}}. \tag{2.17}$$

14. Maximal Correlation Coefficient

$$\sqrt{second\ largest\ eigenvalue\ of\ Q}, \tag{2.18}$$

where

$$Q(i,j) = \sum_{k} \frac{p(i,k)p(j,k)}{p_x(i)p_y(k)}. \tag{2.19}$$

In many applications, an appropriate subset of these 14 second-order statistical functions might be enough for an adequate representation of the neighborhood information of the pixels. Basically, ***SGLDM*** stores the neighborhood information with respect to position vectors $P(d,\theta)$. Even if the power of using such accumulator matrices to decode textural content of the images is adequate, their power can be increased by special position vectors. In this thesis, a special position vector is developed for the sake of increasing the power of the representation of the textural content of the images, thus to end up with a more powerful content-based image retrieval system. On the other hand, a theoretical comparison of texture feature extraction algorithms is presented in [7].

2.2 Query-by-Color Content

Similar to texture, color is one of the most important features of objects in image and video data. Each pixel in an image has a three-dimensional color vector and different color space approaches exist to represent color information. One of these color space models is the hardware-oriented *Red-Green-Blue Model (RGB)*, where the color vector of a pixel p is the compound of red, green and blue channels $v_p = (r,g,b)$. Another color space model is the *Hue-Saturation-Value Model (HSV)* that is based on color descriptions rather than individual color components $v_p = (h,s,v)$. The RGB model has a major drawback: it is not perceptually uniform. Therefore, most of the systems use color space models other than RGB, such as HSV [16].

2.2.1 Preliminaries

2.2.1.1 Transformation and Quantization

The color regions are perceptually distinguishable to some extent. The human eye cannot detect small color differences and may perceive these very similar colors as the same color. This leads to the *quantization* of color, which means that some pre-specified colors will be present on the image and each color is mapped to some of these pre-specified colors. One obvious consequence of this is that each color space may require different levels of quantized colors, which is nothing but a different quantization scheme. In Figure 2.2, the effect of color quantization is illustrated. Figure 2.2 (a) is the original image with RGB color space and (b) is the image produced after transformation into HSV color space and quantization. A detailed explanation of color space transformations (from RGB into HSV) and quantization can be found in [30].

(a) (b) (c)

Figure 2.2: Transformation, Quantization and Color Median Filtering of *tiger* Image. (a) Original image. (b) Image produced by applying RGB to HSV color transformation and quantization. (c) Image produced after applying color median filtering.

2.2.1.2 Color Median Filtering

Not all the colors in an image are dominant. Dominance is in the sense that some of the colors may reside in a region relatively small than the others. The *color median filtering* technique [24], a famous method for neighborhood ranking, eliminates these non-dominant colors and produces a filtered image (Figure 2.2(c)).

This technique facilitates the object extraction process because it also eliminates the noise of the color on the object boundaries to some extent.

In order to achieve the best filtering, the color median filter procedure may be applied successively. For most types of images, color median filtering gives a proper view when applied 3–5 times. Moreover, there exist different types of color median filters. The basic color median filtering types are 3×3 box, 5×5 box, 5×5 octagonal, 7×7 box and 7×7 octagonal filters. Basically, the smoothness on the edges in the image corresponding to the object boundaries vary among these color median filtering types. Figure 2.3 illustrates the color median filtering types. In each type, the neighbors of the black square is ranked to determine the color of the pixel. A detailed explanation on the effects of color median filtering can be found in [25].

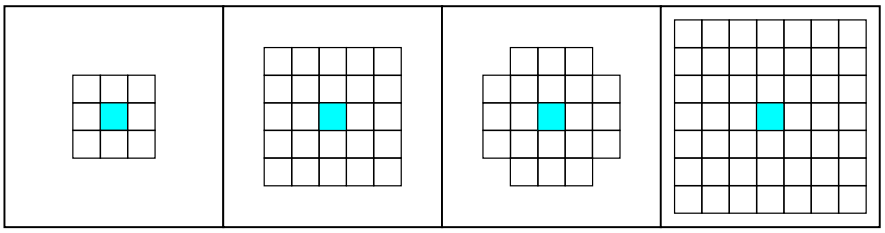


Figure 2.3: A Collection of Color Median Filtering Types. (a) 3×3 box (b) 5×5 box (c) 5×5 octagonal (d) 7×7 box.

2.2.2 Color Content Extraction

One of the widely used methods for querying and retrieval by color content is *color histograms*. The *color histograms* [12, 31] are used to represent the color distribution in an image or a video frame. Mainly, the color histogram approach counts the number of occurrences of each unique color on a sample image. Since an image is composed of pixels and each pixel has a color, the color histogram of an image can be computed easily by visiting every pixel once. By examining the color histogram of an image, the colors existing on the image can be identified

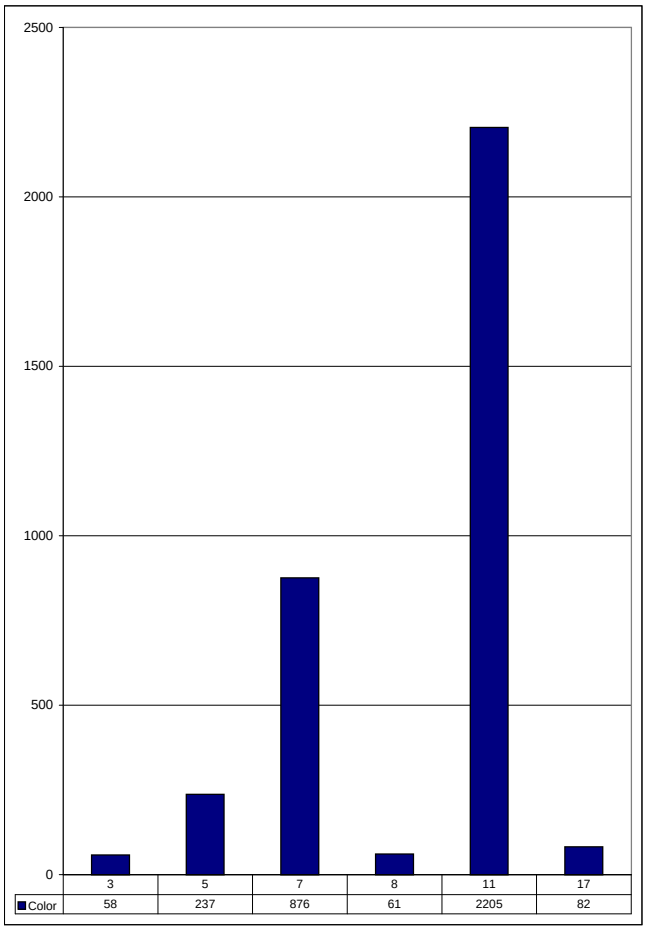


Figure 2.4: The Color Histogram of *tiger* Image in HSV Color Space.

with their corresponding areas as the number of pixels. One possible way of storing the color information is to use three different color histograms for each color channel. Another possible method is to have a single color histogram for all of the color channels. In the latter approach, the color histogram is simply a compact combination of three histograms and the empty slots can be discarded easily. The histogram approach is commonly used in most of the existing systems supporting query-by-color content. Figure 2.4 shows the color histogram of *tiger* image, which is first transformed into HSV color space and quantized. Each row is designated for a distinct color and corresponding number of pixels information is presented. A supplementary information about color for an image is the average color and it may be stored along with the color histogram for the sake of efficiency since it can be computed from the color histogram in one pass.

In [30], Smith and Chang proposed *colorsets* as an opponent to color histograms. The colorsets are binary masks on color histograms and they store the presence of colors as 1 without considering their amounts. For the absent colors, the colorsets store 0 in the corresponding bins. The colorsets reduce the computational complexity of the distance between two images. Besides, by employing colorsets region-based color queries are possible to some extent. On the other hand, processing regions with more than two or three colors is quite complex.

Another image content storage and indexing mechanism is *color correlograms* [17]. It involves an easy-to-compute method and includes not only the spatial correlation of color regions but also the global distribution of local spatial correlation of colors. In fact, a color correlogram is a table each row of which is for a specific color pair of an image. The k-th entry in a row for color pair (i, j) is the probability of finding a pixel of color j at a distance k from a pixel of color i. The method resolves the drawbacks of the pure local and pure global color indexing methods since it includes local spatial color information as well as the global distribution of color information.

2.3 Related Work

There are many content-based retrieval systems capable of querying by texture feature for images and video frames. Some examples of such systems are QBIC [12], VisualSEEk [29], VideoQ [5], Photobook [23], Blobworld [4], SurfImage [22], and ImageRover [28].

In QBIC system [12], one of the milestones in content-based retrieval systems, the users are allowed to query the image and video databases based on color, shape and texture. For textural information, the mathematical representations of coarseness, contrast, and directionality are used. The query is specified by selecting a texture from a texture sampler, which is a set of pre-stored texture images.

In VisualSEEk [29] and VideoQ [5], querying by texture feature is supported

for images and video frames. The three Tamura [32] texture measures, *coarseness*, *contrast* and *orientation*, are computed as a textural measure of the texture content of the objects residing in images and video frames. In both of the systems, the images are uniformly quantized in HSV color space and Brodatz [2] texture set is used for assigning the textural attributes to the objects.

Photobook system [23], which is a set of interactive tools for browsing and searching images and image sequences, includes tools for querying by appearance, shape and texture. The texture content is based on Wold decomposition for regular stationary stochastic processes in images. The Wold decomposition transforms textures into three orthogonal components: *harmonic*, *evanescent*, and *random*. Qualitative speaking, these components appear as *periodicity*, *directionality*, and *randomness*, respectively.

Blobworld [4] is another system for image retrieval that supports region-based queries for images instead of querying by just the whole image. The image regions are coherent and correspond to objects in images. The image is first segmented into regions, called *blob*s, by fitting a mixture of Gaussians to the pixel distribution in a joint color-texture-position feature space. At the end, each blob represents a unique region based on color and texture descriptors. The image segmentation based on color and texture is performed by an Expectation-Maximization algorithm [1]. In the system, querying is based on the user's attribute specifications for at least one or two regions of interest in terms of blobs, rather than a description of the entire image.

Surfimage [22] includes texture-based querying as a part of querying by low-level signatures such as color, shape and texture. The system includes color, orientation and texture histograms, and co-occurrence matrices. The system is tested with various datasets and the retrieval of the system is evaluated in terms of precision and recall.

In ImageRover [28], the texture direction distribution is considered for textural querying. The texture direction is calculated using steerable pyramids [13]. In the system, a steerable pyramid of 4 levels is found to be adequate. The algorithm computes texture direction and orientation using the outputs of steerable

pyramids for each pixel at each of the 4 levels.

The content-based retrieval system presented in this thesis differs from the mentioned systems in mainly texture query specification and execution steps. Most of the systems process only orientation, direction, and coarseness whereas our system computes statistical functions to encode texture feature. One unique feature of our system is that texture content of the object regions can be queried. In the existing system, texture is extracted from the whole image or a rectangular region of interest. However, our system behaves similarly for color-based retrieval. Almost all of the systems employ color histograms, however our system employ a pre-processing phase composed of transformation and quantization (like VisualSEEk). What makes our system unique is the fact that the color median filtering module is embedded in the pre-processing phase for all of the images. This helps to eliminate the effect of color regions that are very small, and this smoothing increases the retrieval of the queries in the system.

Chapter 3

Content-Based Image Retrieval System

Within the framework of a statistical approach to the problem of texture analysis, a content-based image retrieval system has been implemented to query upon images of both homogeneous texture and heterogeneous texture. Images of homogeneous texture have the same textural information and do not require segmentation, whereas images of heterogeneous texture have very distinctive set of textural regions, hence require segmentation for extracting textural content. Besides its functionality as an image retrieval mechanism, the underlying query engine is ready to be integrated to our *BilVideo* video database management system [9], specifically the query-by-feature subsystem that supports low-level queries, to enable the retrieval of video frames and shots including textures similar to a given example. In this chapter, an overview of *BilVideo* is discussed before the design principles of the content-based image retrieval system.

3.1 Overview of BilVideo System

BilVideo is a video database management system the architecture of which provides full support for spatio-temporal queries that contain any combination of spatial, temporal, object-appearance, external-predicate, trajectory-projection and similarity-based object-trajectory conditions by a rule-based system utilizing a knowledge-base while using an object-relational database to respond to semantic (keyword, event/activity and category-based) and low-level (color, shape and texture) video queries. The knowledge-base of *BilVideo* contains a fact-base and a comprehensive set of rules implemented in Prolog. The rules in the knowledge-base significantly reduce the number of facts that need to be stored for spatio-temporal querying of video data: the storage space saving was about 50% for some real video data. Moreover, the system's response time for different types of spatio-temporal queries posed on the same data was less than a second when queries were given to the system as Prolog predicates [11]. Query processor interacts with both of the knowledge-base and object-relational database to respond to user queries that contain a combination of spatio-temporal, semantic and low-level video queries. Intermediate query results returned from these two system components are integrated seamlessly by the query processor and final results are sent to Web clients. *BilVideo* has a simple, yet very powerful SQL-like textual query language for spatio-temporal queries on video data [10]. For novice users, there is also a visual query interface (visual query language) provided. Both languages are currently being extended to support semantic and low-level video queries.

BilVideo is built over a client-server architecture as it is illustrated in Figure 3.1. The system is accessed on the Internet through its visual query interface developed as a Java Applet [25]. Users may query the system with sketches and a visual query is formed by a collection of objects with some conditions, such as object trajectories with similarity measures, spatio-temporal orderings of objects, annotations and events. Object motion is specified as an arbitrary trajectory for each salient object of interest and annotations can be used for keyword-based video search. Users are able to browse the video collection before posing complex

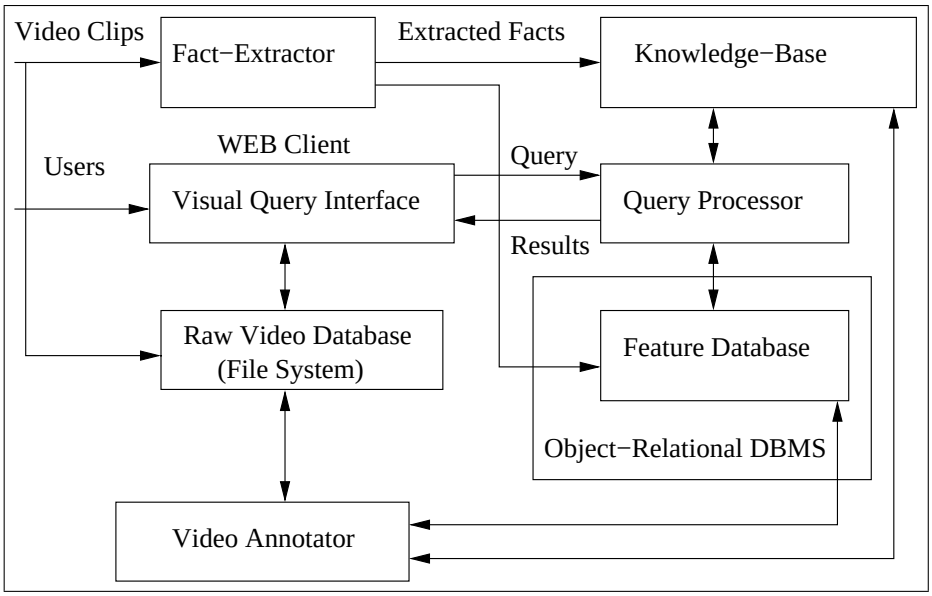


Figure 3.1: BilVideo System Architecture

and specific queries. Furthermore, an SQL-like textual query language is also available for the users. The visual query language forms an interface to the textual query language because queries constructed by the visual language are first translated to their equivalent textual query language statements before being sent to the query server. In the heart of the system lies the query processor, which is responsible for processing and responding to user queries in a multi-threaded environment. The query processor communicates with a knowledge-base and an object-relational database. The knowledge-base stores fact-based meta data used for spatio-temporal queries whereas semantic and low-level meta data is stored in the feature database maintained by the object-relational database. Raw video data and video data features are stored separately. Meta data stored in the feature database is generated and updated by the Video-Annotator tool and the facts-base is populated by the Fact-Extractor tool, both developed as a Java application. A detailed information on *BilVideo* system can be found in [9].

Moreover within the *BilVideo* system, a histogram-based approach is located to query salient objects by their shape content [27]. The histogram-based querying approach is intended to use a similarity measure between images much like

the human vision system does. Thus, the interrelation among pixels is very important and should be taken into account for object-based similarity. This is because each pixel provides a piece of information about objects and should be considered in the shape and content. To store shape information, *distance* and *angle* histograms are employed and the calculations are performed with respect to the center of masses of salient objects.

The content-based retrieval system presented in this thesis is going to be used as the query-by-feature sub-system of the *BilVideo* system together with the histogram-based approach. Having completed the integration of modules for shape, texture and color features, the users will be allowed to pose queries in a broader sense.

3.2 The Graphical User Interface

The content-based image retrieval system runs texture and color queries for images where texture feature extraction is based on an optimal set of five of the previously-stated second order texture statistics, and the color content is stored in color histogram after a proper color space transformation and quantization scheme. The color quantization parameter is set to 4 so that the number of distinct color levels on an image is 64, instead of 256 without quantization.

As seen in Figure 3.2, our system consists of two image input panels and a search results display panel, in terms of the graphical user interface. One of the input panels is used for the purpose of database population, i.e., adding new images to the existing set of images in the data store, and extracting their texture feature vectors in the interim. The other input panel is for entering query instances. When the program is prompted by the user to search within the present database, a texture feature vector is also extracted for the image in the query instance and then through a comparison of the feature vectors of all images, images with a similar texture to that of the query instance are retrieved. The query results are displayed in the output panel in descending order of similarity,

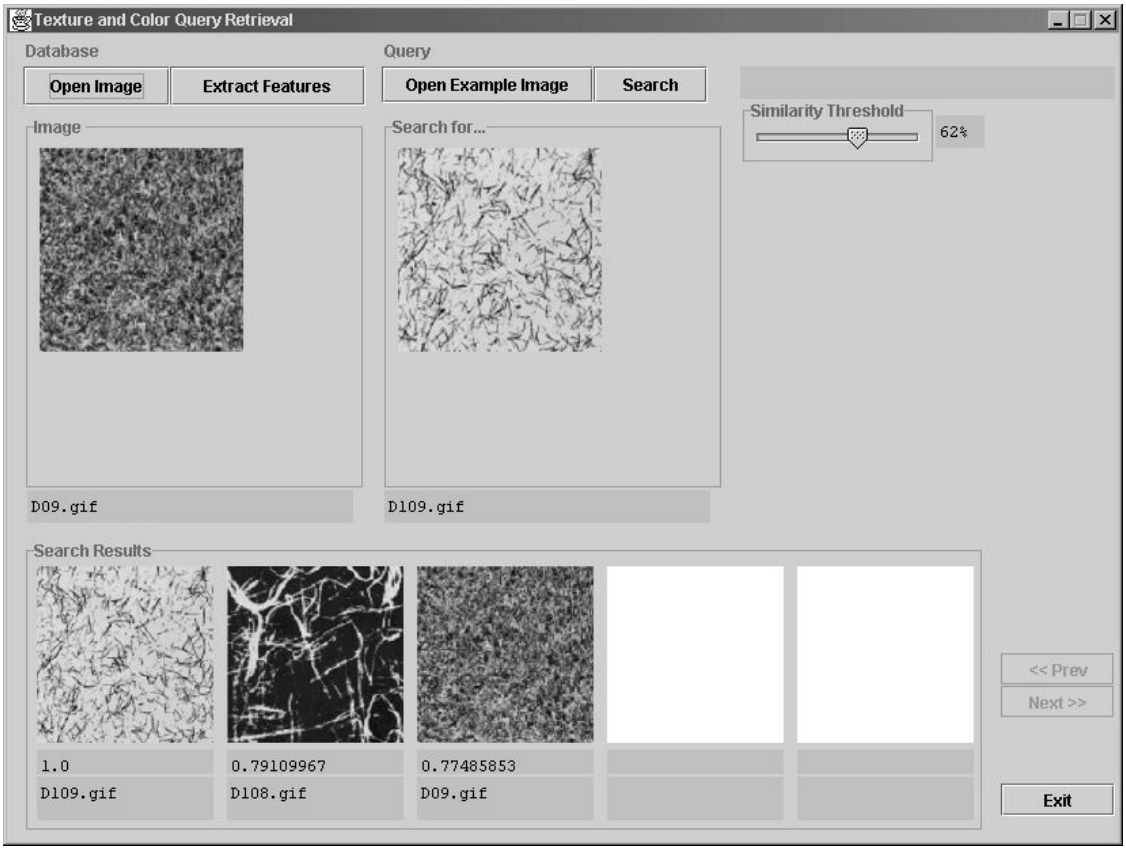


Figure 3.2: The Graphical User Interface of the Content-Based Image Retrieval System.

as thumbnails.

The user can adjust the range of similarity, within which the resultant images should be. By increasing the similarity threshold, the user can view a larger set of pictures, or through more restricted limits he/she can eliminate more pictures. In the system, the default similarity value is set to 80%. For the sake of usability and to achieve user satisfiability, the results of the queries are presented in rows of five images where the users can go forwards and backwards between the rows.

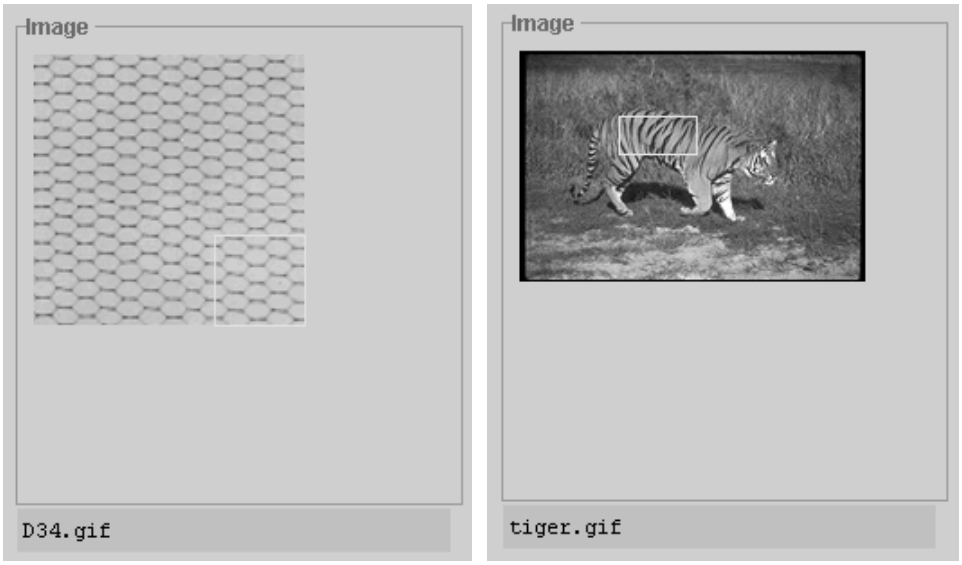
3.3 Feature Extraction from Images

The extraction of the texture and color content of the images take place both during the database population phase and querying phase. Depending on the user's intention, the texture feature extraction can be performed in three different ways:

- **Fully Automatic Texture Feature Extraction:** The system is capable of determining a rectangular region on the image representing the texture characteristics of the image. Since this region is relatively smaller than the whole image and it is a good representation, dealing with the automatically segmented region provides two things: the feature extraction time decreases, and the query processing phase is accelerated (cf., Figure 3.3 (a)).
- **Semi-Automatic Texture Feature Extraction:** In most of the applications, the users are not interested in the texture of the whole image but a specific region-of-interest. Since the user is provided drawing facilities on the loaded image, the region-of-interest is determined simply by dragging and dropping the mouse on the image. Similar to the fully automatic case, processing the region-of-interests fastens the system (cf., Figure 3.3 (b)).
- **Texture Feature Extraction of Whole Image:** The texture feature extraction for the whole image is the default case, and is meaningful when the whole image is of interest (e.g., for Brodatz Texture Images [2]).

As mentioned earlier, 14 second-order statistical functions can be used for the textural content of images. However, in many applications, an appropriate subset of these functions are enough. The compact use of these statistical functions provide efficiency in terms of computational complexity. Thus, the following five statistical functions are extracted in the system:

- **Energy:** It is also called *uniformity* and has the following characteristics: when all the matrix elements are almost equal, i.e., when gray level intensities are very close to each other, the value of the energy is small. Thus,



(a) (b)

Figure 3.3: Feature Extraction Alternatives, (a) Fully Automatic Extraction, and (b) Semi-Automatic Extraction.

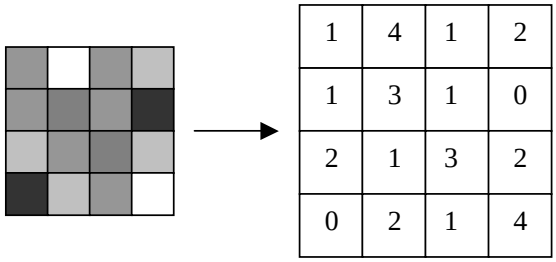
the higher the value of the energy, the more irregular the SGDLM (cf., Figure 3.4).

- **Entropy:** It is the opposite of energy, thus it has a lower value when the SGDLM is irregular. It has its highest peak when the SGDLM is uniform (cf., Figure 3.4).

- **Contrast:** It is also called *inertia* and measures the difference moment of the SGDLM. The value will be high if the image has high local variation (cf., Figure 3.5).

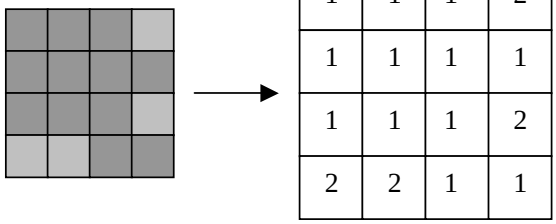
- **Inverse Difference Moment:** It is also called *local homogeneity* and it is the opposite of contrast. If the SGDLM has high values at the diagonal, the value of the function is high. The value is also high when similar gray levels are next to each other (cf., Figure 3.5).

- **Correlation:** The correlation measures the linear dependency of the gray level values in the SGDLM. A high or a low correlation value leads to no immediate conclusion about the image.



1	4	1	2
1	3	1	0
2	1	3	2
0	2	1	4

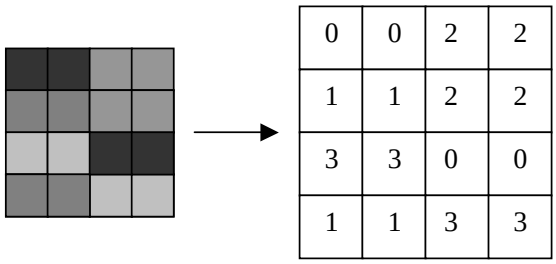
(a)



1	1	1	2
1	1	1	1
1	1	1	2
2	2	1	1

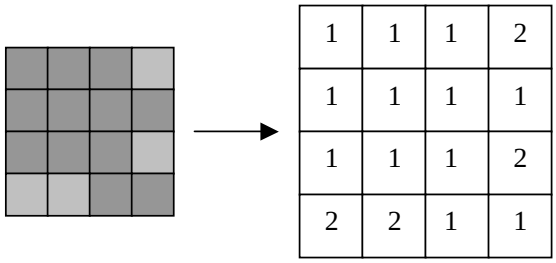
(b)

Figure 3.4: Energy and Entropy Functions, (a) Energy Low and Entropy High, and (b) Energy High and Entropy Low.



0	0	2	2
1	1	2	2
3	3	0	0
1	1	3	3

(a)



1	1	1	2
1	1	1	1
1	1	1	2
2	2	1	1

(b)

Figure 3.5: Contrast and Inverse Difference Moment Functions, (a) Contrast High and Inverse Difference Low, and (b) Contrast Low and Inverse Difference Moment High.

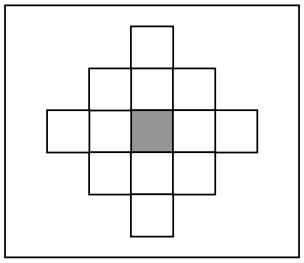


Figure 3.6: The Circular Position Vector with Radius 2.

In order to increase the expressiveness of the SGDLMs, powerful position vectors have to be developed to decode the neighborhood information among the pixels. In the system, a special position vector, called *Circular Position Vector* is designed. As its name implies, it accumulates the neighborhood information of a pixel for a pre-specified radius r as a parameter ($CPV(r)$). For example, if r is set to 2, the effect of $CPV(2)$ is equal to the total effects of $P(1, 0, 45, 90, 135, 180, 225, 270, 315)$ and $P(2, 0, 90, 180, 270)$ (cf., Figure 3.6). This position vector is powerful since it reflects the human visual system more to capture the neighborhood information among pixels.

3.3.1 Histogram Intersection Method for Similarity Calculation

In the *Histogram Intersection* technique, two normalized histograms are intersected as a whole, as the name of the technique implies. The similarity between the histograms is a floating point number between 0 and 1. Equivalence is designated with similarity value 1 and the similarity between two histograms decreases when the similarity value approaches to 0. Both of the histograms must be of the same size to have a valid similarity value. This method is used for color in [31] and for shape in [18, 27].

Let $H_1[1..n]$ and $H_2[1..n]$ denote two histograms of size n, and S_{H_1,H_2} denote the similarity value between H_1 and H_2. Then, S_{H_1,H_2} can be expressed by the

distance between the histograms H_1 and H_2 as:

$$S_{H_1,H_2} = \frac{\sum_i^n \min(H_1[i], H_2[i])}{\min(|H_1|, |H_2|)}. \tag{3.1}$$

In the system, this technique is employed for similarity calculations as a result of texture vector and color histogram comparisons between database images and query image.

3.4 A Framework for Object-Based Texture Querying

As mentioned before, the current image retrieval system supports three types of texture queries: querying by the whole image, querying by a region of interest specified by the user, and querying by a region that automatically captures a texture region. However, since the images and video frames contain salient objects, users may want to query the image database with respect to the textural content of the salient objects. In the current system, such queries can be expressed via specifying rectangular regions containing salient objects semi-automatically. This specification for each salient object is a tedious process for many users, thus a framework for this specification is needed. Another reason for developing a new framework is the fact that via specifying regions of interest, the user can only draw rectangular regions, which may mislead the textural content of the salient objects.

In order to develop a framework supporting object-based texture querying, the system can be integrated with existing object extraction tools. The *Object Extractor* [26] is an object extraction tool that extracts the salient object pixels from an image. Since only the pixels belonging to the object is extracted, the textural content of the object is retained.

Having extracted the textural content of an object region in one frame of a video, a facility to compare this extracted textural content with the database objects is required. Fortunately, since Object Extractor provides object extraction

independent from object location in the frame, this comparison facility seems to be simpler than expected. The crucial thing is to compute the texture statistics functions for the object regions, both for query object at querying time and for the database objects at database population time, and compute their similarity with Histogram Intersection method.

This type of querying provides object-based querying by texture and eases some object identification and tracking applications. In this section, an application of capturing motion of a salient object within a video is discussed. This framework provides a generic way, and application-specific assumptions might be needed for different domains.

3.4.1 Object Extraction Process

The Object Extractor tool [26] employs Flood Fill for Extraction (FFE) algorithm, an improved version of the flood fill algorithm for polygon filling [16]. The FFE algorithm is initiated with a user-clicked pixel on the object to be extracted and recursively checks the neighbors of the initiative pixel.

The Object Extractor has an easy-to-use user interface, in which all of the user's actions are handled with simple mouse actions. The loaded image is processed via the tool with the current color difference threshold and color median filtering type. In order to achieve better smoothness, color median filtering can be applied as many times as desired. The user extracts color regions separately by initiating a separate execution of FFE algorithm. When the user is satisfied with the current extracted form of the object, he/she can press the 'done' button and the extracted color regions form the extracted object. Appendix A presents detailed information for Object Extractor.

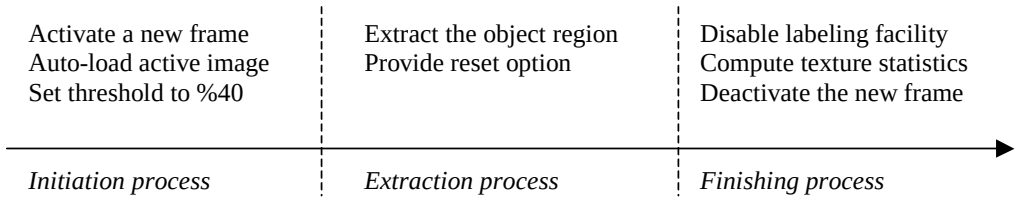


Figure 3.7: Timeline of Object-Based Querying with Object Extractor.

3.4.2 Framework Architecture

The need for object-based querying is inevitable for especially heterogeneous images, when there is no uniform texture. This situation is very common for most of the image domains and videos, hence a proper framework to fulfill this gap is to be developed.

Due to the design of the framework, the user interface of the system is extended with a new frame mimicking the user interface of Object Extractor (cf., Figure A.2). This new frame is activated by a button that is dedicated to object-based querying. This new frame is initiated with the active texture image already loaded in the system interface, to allow the algorithmic operations of Object Extractor. The default parameter for color difference threshold is retained at %40. Following the extraction operation, the 'done' button is pressed to complete the process. In this framework, this done button is overloaded to compute the texture statistics and the texture feature vector for the object region (most likely non-rectangular) and deactivate the frame. The timeline of the whole process is presented in Figure 3.7.

3.4.3 Capturing Motion within Video Data

The framework has several applications: one is capturing motion of a salient object within video data. An object moving in a video frame sequence can be identified by the help of this framework as follows:

Consider two consecutive frames of a video data. If the textural content of the images are similar enough (possibly more than %99,5), then it can be concluded

Table 3.1: Similarity Table for the Frame Sequence of `Cat.avi`.

	frame 01	frame 02	frame 03	frame 04	frame 05	frame 06
frame 01	1	0.9977	0.9981	0.9982	0.9978	0.9997
frame 02	0.9977	1	0.9983	0.9984	0.9998	0.9978
frame 03	0.9981	0.9983	1	0.9997	0.9982	0.9980
frame 04	0.9982	0.9984	0.9997	1	0.9983	0.9980
frame 05	0.9978	0.9998	0.9982	0.9983	1	0.9977
frame 06	0.9997	0.9978	0.9980	0.9980	0.9977	1

that they are sharing at least one salient object. To simplify the application and to increase the validity of the conclusion, it can be assumed that the frame sequence contains exactly one salient object. Based on this extracted information, if the location of the object in one frame is known, the location of the same object in the other frame can be found easier than extracting the object in the frame. To elaborate on this application, Figure 3.8 shows six frames of a video, named `Cat.avi`, and Table 3.1 presents the mutual similarities of the frames. In this figure, the camera moves right for the first three frames while the cat is eating its meal, and right after the third frame the camera turns back and moves left for the rest of the frames. As expected, the similarity values complies with the intuitions, such that frame 01 is almost equivalent to frame 06, so as frame 02 and frame 05, and frame 03 and frame 04.

Figure 3.8: A Set of Frames from `Cat.avi`.

Chapter 4

Performance Experiments

To test the performance of our content-based image retrieval system on texture similarity queries, we have used images scanned from the ubiquitous Brodatz texture album [2]. Appendix B includes the index of this album. We digitized all 112 textures of the Brodatz dataset to 256 grey levels. To increase the size of the texture database, first the existing texture images is rotated about random angles. Another database extension is carried out by clipping at least two texture images to form a new texture image.

4.1 Evaluating Effectiveness

Our content-based image retrieval system is first evaluated in terms of retrieval effectiveness. In order to evaluate effectiveness of retrieval systems, two well-known metrics, *precision* and *recall* [19], are used:

$$Precision = \frac{the\ number\ of\ retrieved\ images\ that\ are\ relevant}{the\ number\ of\ retrieved\ images}. \tag{4.1}$$

$$Recall = \frac{the\ number\ of\ retrieved\ images\ that\ are\ relevant}{the\ total\ number\ of\ relevant\ images}. \tag{4.2}$$

In the experiments, the query image is randomly picked from the texture images. Based on the Brodatz Texture Index, the relevance degrees for the picked query image are used in the analysis. Based on this index, two images are either relevant or irrelevant to each other (i.e., relevant=1, irrelevant=0). In order to improve the evaluation, the retrieval process is performed several times for different randomly picked query objects. Then, the effectiveness is evaluated as the average of the results calculated for each query separately. Basically, precision and recall are set-based measures, in other words they evaluate the quality of an unordered set of retrieved images. To evaluate ranked lists, precision can be plotted against recall after each retrieved image. Since different queries may lead to different precision and recall values, the computation of average effectiveness. To ease this computation, the individual precision values are interpolated to a set of 11 standard recall levels (0,0.1,0.2,...,1) in order to facilitate the computation of average of precision and recall values [19].

In this chapter, the interpolated precision-recall graphs for eight texture images are given: `D23.gif`, `D39.gif`, `D52.gif`, `D55.gif`, `D58.gif`, `D68.gif`, `D94.gif`, and `D110.gif` (cf., Figures 4.1–4.8). There is no specific reason for presenting these eight query images, but they provide a comprehensive way to evaluate the retrieval effectiveness of the content-based image retrieval system. To conclude that the system is effective, the basic expectation from the interpolated precision-recall graphs is the fact that they have to be non-increasing. In all of these graphs, the precision value is 1 for the first a few standard recall levels, and while standard recall levels are increasing, the precision values continue in a non-increasing manner.

4.2 Query Examples

In Figure 4.9 are given the result sets for four example queries that were run on this dataset. The first texture in each query example is the query instance itself, thus yielding the highest possible similarity score (i.e., 1.000). Figure 4.9 merely depicts the top four matches for each query instance.

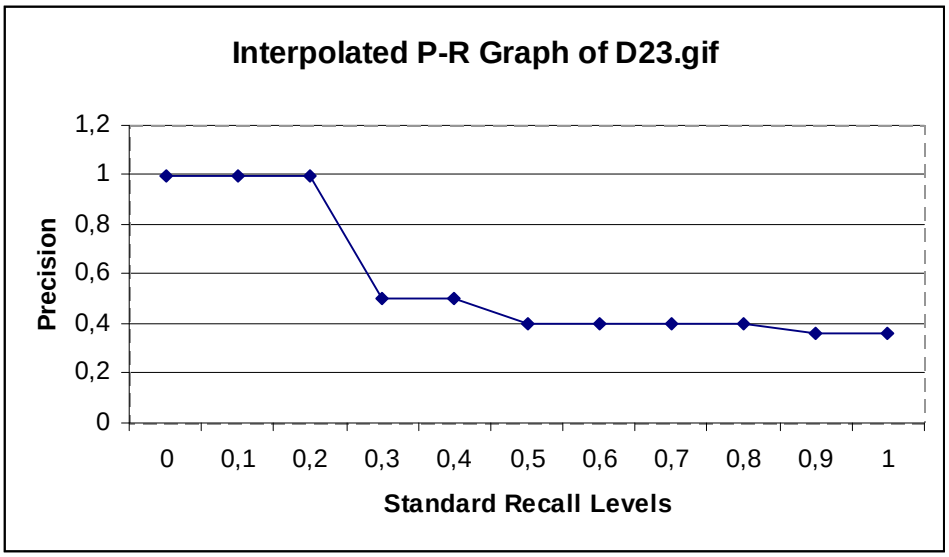


Figure 4.1: Interpolated Precision Recall Graph with `D23.gif` as Query Texture Image.

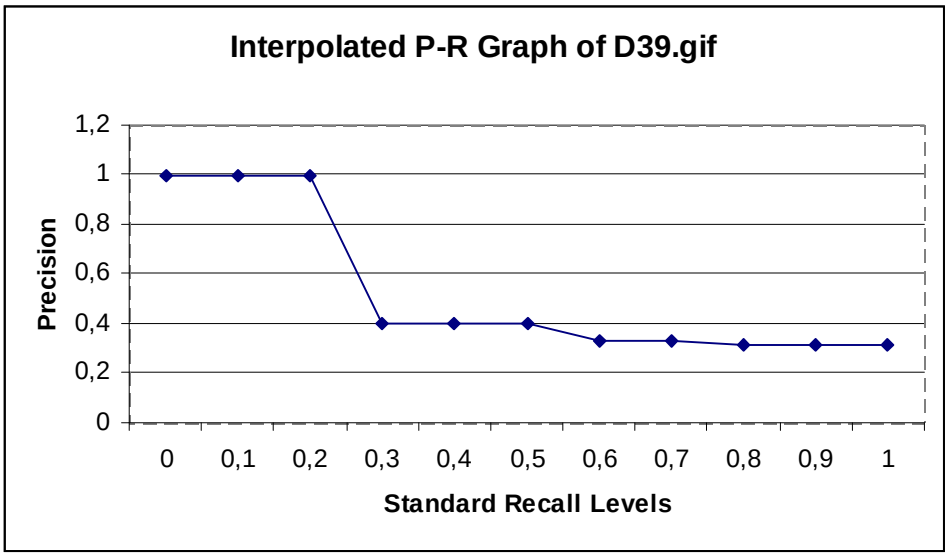


Figure 4.2: Interpolated Precision Recall Graph with `D39.gif` as Query Texture Image.

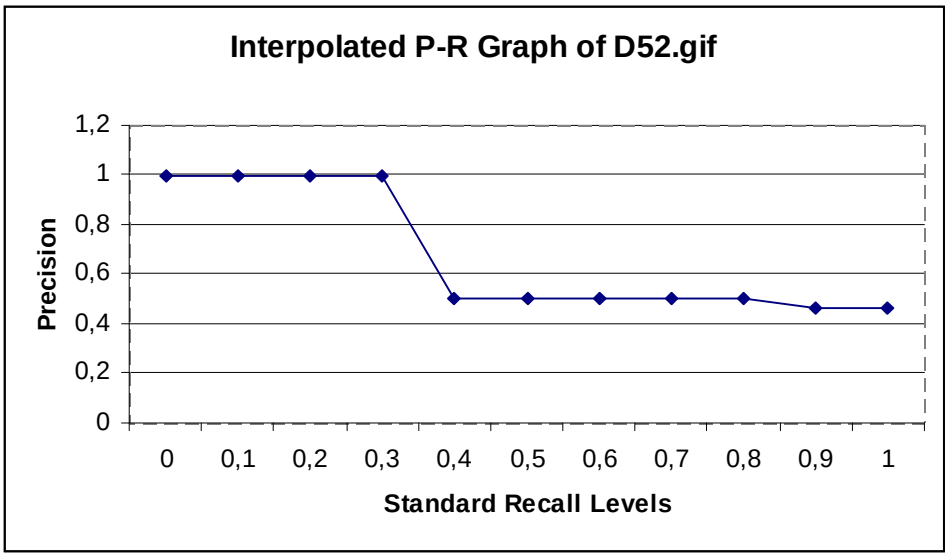


Figure 4.3: Interpolated Precision Recall Graph with `D52.gif` as Query Texture Image.

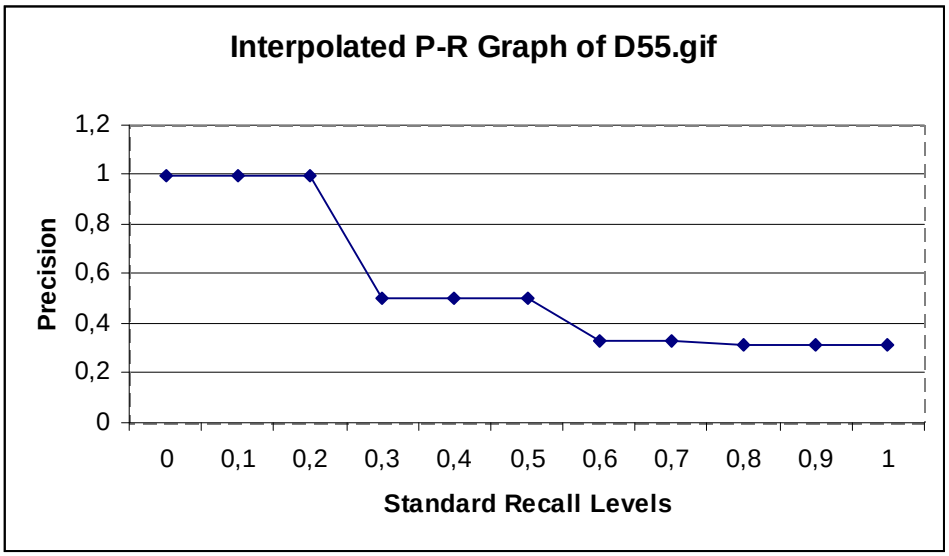


Figure 4.4: Interpolated Precision Recall Graph with `D55.gif` as Query Texture Image.

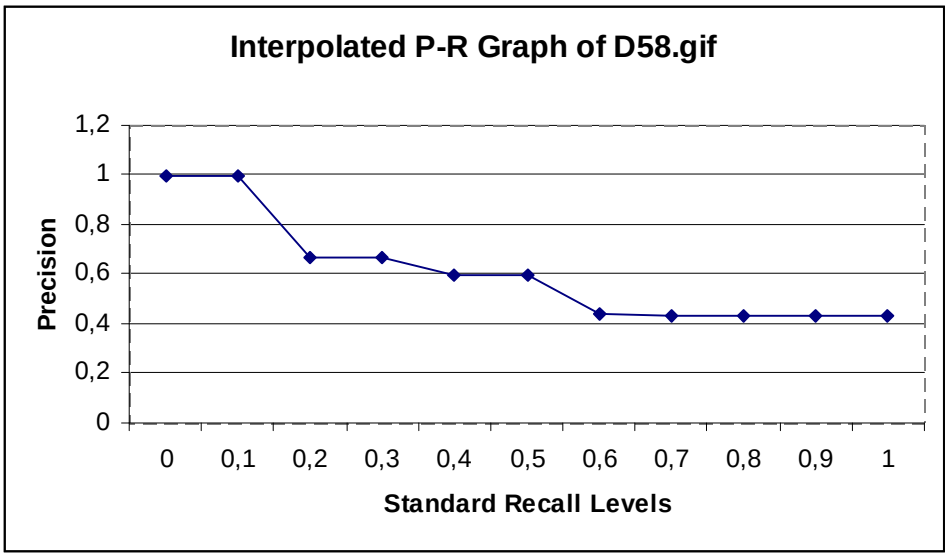


Figure 4.5: Interpolated Precision Recall Graph with `D58.gif` as Query Texture Image.

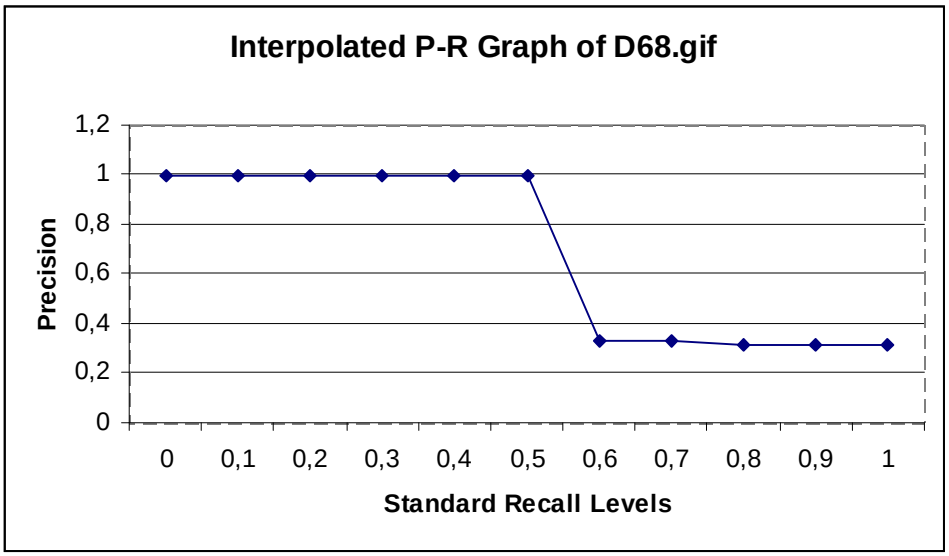


Figure 4.6: Interpolated Precision Recall Graph with `D68.gif` as Query Texture Image.

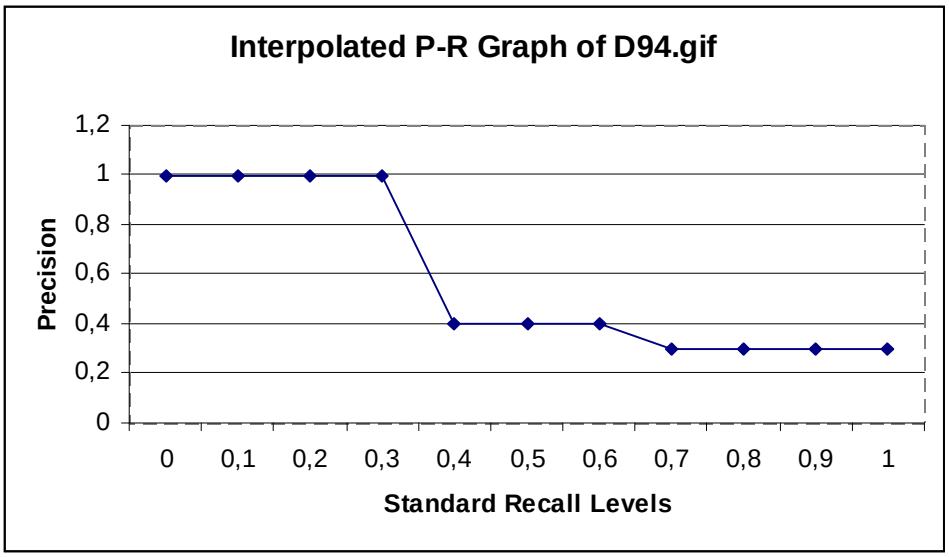


Figure 4.7: Interpolated Precision Recall Graph with `D94.gif` as Query Texture Image.

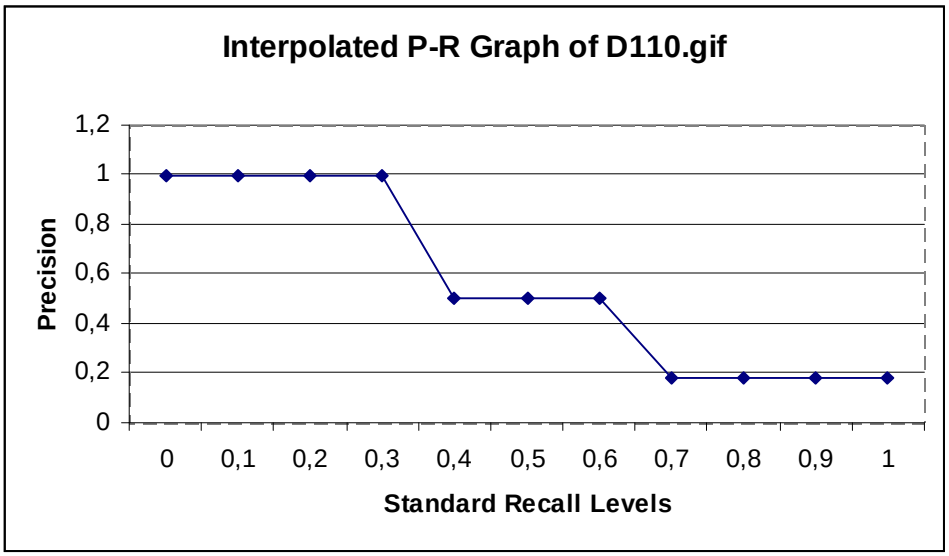
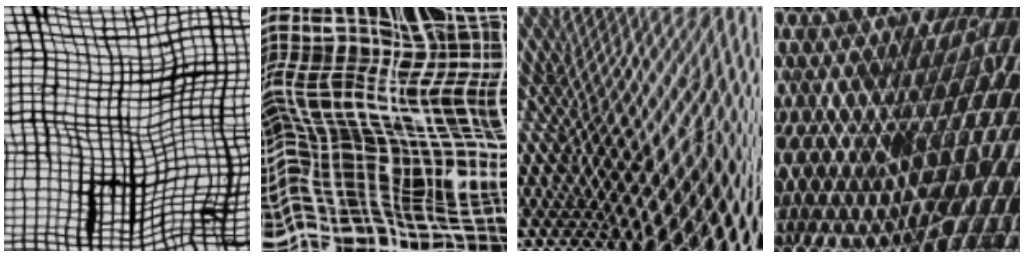
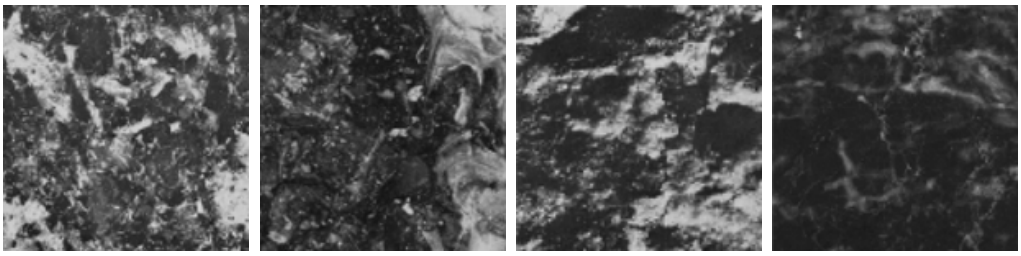


Figure 4.8: Interpolated Precision Recall Graph with `D110.gif` as Query Texture Image.



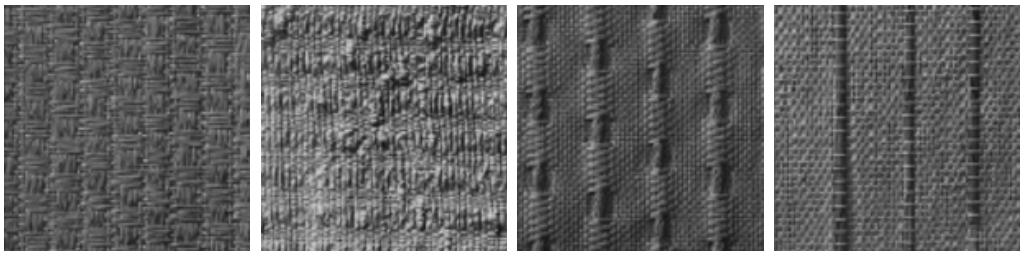
D 103 1.000 | D 104 0.927 | D 36 0.851 | D 22 0.813

(a)



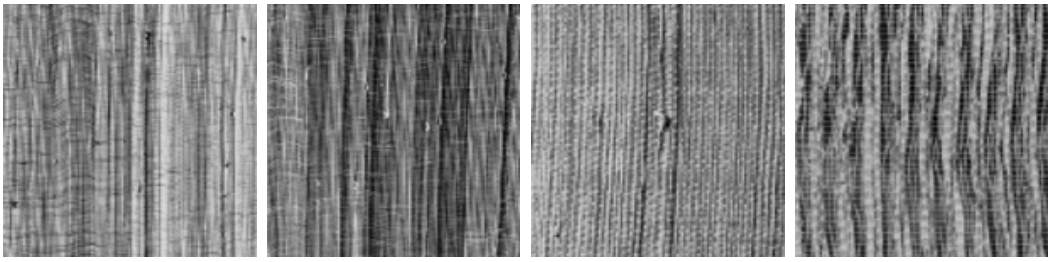
D 60 1.000 | D 58 0.950 | D 7 0.893 | D 63 0.824

(b)



D 82 1.000 | D 80 0.973 | D 83 0.946 | D 85 0.898

(c)



D 105 1.000 | D 106 0.982 | D 79 0.945 | D 76 0.875

(d)

Figure 4.9: Sample Queries: Brodatz Album Codes and Similarity Scores for Each Query Result. (a) Query 1, (b) Query 2, (b) Query 3, (d) Query 4.

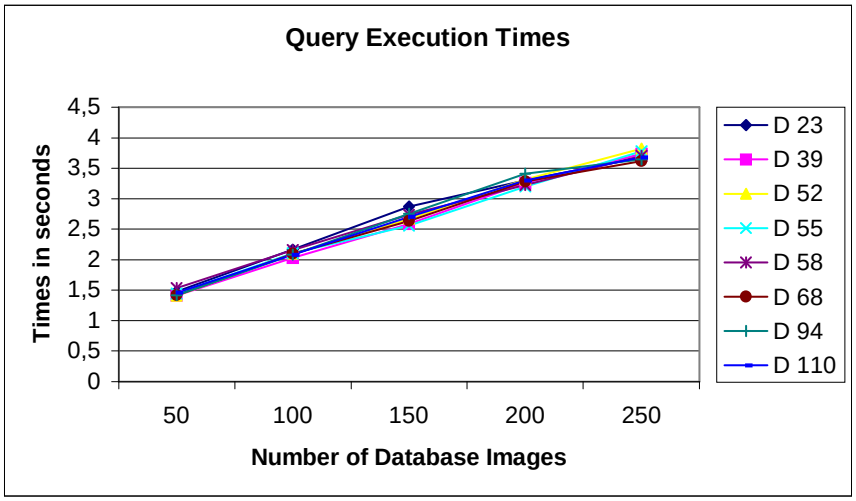


Figure 4.10: Query Execution Times

In Query 1, when loose burlap tissue (D 103) was given as the query instance, among the most similar matches were another example of burlap (D 104), and also two reptile skin textures (D 36 and D 22).

A European marble texture (D 60) was the query instance in Query 2, and resulted in the retrieval of other examples of marble texture (D 58 and D 63), as well as a fieldstone texture of similar quality (D 7).

Query 3 probed for textures similar to that of woven Oriental straw (D 82) and successfully retrieved several woven straw and matting textures (D 80, D 83 and D 85).

Cloth samples were again the subject of Query 4, which upon the submission of a cheesecloth texture (D 105), retrieved another cheesecloth texture (D 106) and two textures of grass fiber cloth (D 79 and D 76).

With these examples illustrating the quality of the query results, there remains quantitative measures of the performance of our content-based image retrieval engine such as query response time. Whereas the size of the test dataset (Brodatz album) is not adequate to estimate the response time for queries on realistic datasets, Table 4.1 presents the query execution times and Figure 4.10 shows the corresponding graph on a single processor 600 MHz Pentium PC. One

Table 4.1: Query Execution Times for Eight Brodatz Texture Images.

	Number of Database Images				
Query Instance (Brodatz Number)	50	100	150	200	250
D 23	1,48	2,17	2,87	3,30	3,68
D 39	1,41	2,03	2,59	3,26	3,74
D 52	1,41	2,10	2,67	3,32	3,82
D 55	1,44	2,11	2,56	3,20	3,78
D 58	1,54	2,16	2,74	3,22	3,70
D 68	1,42	2,10	2,64	3,28	3,62
D 94	1,42	2,08	2,75	3,41	3,64
D 110	1,46	2,08	2,70	3,30	3,68

straightforward observation is that the query time increases in parallel with the number of extracted images in the database due to query processing time spent to compare texture feature vectors and color histograms. This graph shows that the query times are scalable with the number of images in the database.

Chapter 5

Conclusions and Future Work

We have designed and implemented a content-based image retrieval system that evaluates the similarity of each image in its data store to a query image in terms of textural and color characteristics, and returns the images within a desired range of similarity.

From among the existing approaches to texture analysis within the domain of image processing, we have adopted the statistical approach to extract texture features from both the query images and the images of the data store. *Energy*, *entropy*, *inertia*, *correlation* and *local homogeneity* have been selected as an optimal subset of the set of second order statistical features that can be extracted from *Spatial Grey Level Dependency Matrices*. Histogram intersection method has been used as the similarity measure between two feature vectors.

For the color content extraction, a well-known and powerful technique, color histograms, are used. The expressiveness of this technique is accelerated via color space transformation and quantization, and the images are smoothed by the help of color median filtering, a famous method for neighborhood ranking. Hence, it also complies with texture extraction due to being related with the neighborhood property of pixels.

Our system has been tested on images scanned from the commonly used Brodatz texture album and shown to be an efficient tool for image retrieval.

As a result of our incremental approach to systems development, the initial version of the image retrieval system extracted feature statistics from over the whole image, thus treating the image as if it is composed of one sole texture. For image databases of restricted variety, such as the Brodatz set, which merely includes monochrome photographs of virtually homogeneous patterns taken under studio lighting at an angle parallel to the film plane, this approach yielded satisfactory results. Nonetheless, more realistic image databases, embracing color images, with segments of different textures, produced under varying lighting conditions, indisputably necessitated the extension of the capabilities of our search engine through a preliminary process of supervised and unsupervised image segmentation. Thus by providing fully-automatic and semi-automatic texture feature extraction capabilities, our system was enhanced so as to be able to operate on rectangular regions of interest, rather than the whole image.

A crucial future work to be done on our system is to further enhance its capability of operating on image segments by integrating the forementioned Object Extractor into the system. Thus in both input panels of the content-based image retrieval system, when a new image is opened, the Object Extractor shall be activated with the opened image automatically loaded. The user shall then be able to extract salient object through simple clicks, and when the object extraction is completed, texture feature vector shall be calculated only for the selected object. This shall certainly yield more accurate descriptions of regions of interest.

A promising application of such an integration, that needs further research, would be a texture-based tool to monitor motion of salient objects in a series of images such as a video.

Another task to be completed in the future is the integration of our texture and color based image retrieval system with the existing query-by-shape feature of *BilVideo* video database management system so as to equip *BilVideo* with a complete set of low-level query features within its query-by-feature subsystem.

Bibliography

[1] S. Belongie, C. Carson, H. Greenspan, and J. Malik. Color- and texture-based image segmentation using EM and its application to content-based image retrieval. In *Proceedings of the Sixth International Conference on Computer Vision*, 1998.

[2] P. Brodatz. *Textures - A Photographic Album for Artists and Designers*. Dover Publications, New York, 1966.

[3] L. Carlucci. A formal system for texture languages. In *Pattern Recognition*, pages 53–72, 1972.

[4] C. Carson, M. Thomas, S. Belongie, J.M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *Third International Conference on Visual Information Systems*. Springer, 1999.

[5] S. Chang, W. Chen, H.J. Meng, H. Sundaram, and D. Zhong. VideoQ: An automated content-nased video search system using visual cues. In *ACM Multimedia 97*, pages 313–324, Seattle, Washington, USA, 1997.

[6] C. H. Chen. A study of texture classification using spectral features. In *Proceedings of the 6th International Conference on Pattern Recognition*, pages 1074–1077, 1982.

[7] R. Conners and C. Harlow. A theoretical comparison of texture algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(3):204–222, 1980.

[8] M. N. Do and M. Vetterli. Texture similarity measurement using Kullback-Leibler distance on wavelet subbands. In *Proceedings of IEEE International Conference on Image Processing*, 2000.

[9] M.E. Dönderler, E. Şaykol, U. Arslan, Ö. Ulusoy, and U. Güdükbay. BilVideo: A video database management system. *submitted journal paper*, 2002.

[10] M.E. Dönderler, Ö. Ulusoy, and U. Güdükbay. Rule-based spatio-temporal query processing for video databases. *submitted journal paper*, 2002.

[11] M.E. Dönderler, Ö. Ulusoy, and U. Güdükbay. A rule-based video database system architecture. *Information Sciences*, 143(1-4):13–45, June 2002.

[12] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani abd J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The QBIC system. *IEEE Computer*, pages 23–32, September 1995.

[13] W. Freeman and E.H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(9):891–906, 1991.

[14] R. M. Haralick. Statistical and structural approaches to texture. In *Proceedings of the IEEE*, volume 67, pages 786–804, 1979.

[15] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man and Cybernetics*, 3:610–621, 1973.

[16] D. Hearn and M.P. Baker. *Computer Graphics*. Prentice Hall, Inc., 1994.

[17] J. Huang, S.R. Kumar, M. Mitra, W.J. Zhu, and R. Zabih. Image indexing using color correlograms. In *Proceedings of IEEE Comp. Soc. Conf. Comp. Vis. and Patt. Rec.*, pages 762–768, 1997.

[18] A.K. Jain and A. Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29(8):1233–1244, August 1996.

[19] K.S. Jones. *Information Retrieval Experiment*. Butterworth and Co., 1981.

[20] B. Julesz. Visual pattern discrimination. In *IRE Transactions on Information Theory*, pages 407–419, 1992.

[21] S. Y. Lu and K. S. Fu. A syntactic approach to texture analysis. In *Computer Graphics and Image Processing*, pages 303–330, 1978.

[22] C. Nastar, M. Mitschke, C. Meilhac, and N. Boujemaa. Surfimage: a flexible content-based image retrieval system. In *ACM Multimedia 98*, pages 339–344, Bristol, UK, 1998.

[23] A. Pentland, R.W. Picard, and S. Sclaroff. Photobook: Content-based manipulation of image databases. *Int. J. Computer Vision*, 18(3):233–254, 1993.

[24] J. Russ. *The Image Processing Handbook*. CRC Press, in cooperation with IEEE Press, 1999.

[25] E. Şaykol. Web-based user interface for query specification in a video database system. M.S. Thesis, Department of Computer Engineering, Bilkent University, Ankara, Turkey, September 2001.

[26] E. Şaykol, U. Güdükbay, and Ö. Ulusoy. A semi-automatic object extraction tool for querying in multimedia databases. In S. Adali and S. Tripathi, editors, *7th Workshop on Multimedia Information Systems MIS'01, Capri, Italy*, pages 11–20, 2001.

[27] E. Şaykol, U. Güdükbay, and Ö. Ulusoy. A histogram-based approach for object-based query-by-shape-and-color in multimedia databases. *submitted journal paper and also available as Bilkent University Tech. Rep. BU-CE-0201, Computer Engineering Department*, January 2002.

[28] S. Sclaroff, L. Taycher, and M.L. Cascia. Imagerover: A content-based image browser for the world wide web. In *IEEE Workshop on Content-Based Access of Image and Video Libraries*, pages 2–9, San Juan, Puerto Rico, June 1997.

[29] J.R. Smith and S-F. Chang. VisualSEEk: A fully automated content-based image query system. In *ACM Multimedia Conference*, pages 87–98, Boston, MA, November 1996.

[30] J.R. Smith and S.F. Chang. Tools and techniques for color image retrieval. In I.K. Jain and R.C., editors, *Proceedings of Storage and Retrieval for Image and Video Databases IV, IS&T/SPIE*, volume 2670, pages 426–437, 1996.

[31] M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[32] H. Tamura and S. Mori. Textural features corresponding to visual perception. *IEEE Transactions on Systems, Man, and Cybernetics*, 8(6), June 1978.

[33] J. Weszka, C. Dyer, and A. Rosenfeld. A comparative study of texture measures for terrain classification. *IEEE Transactions on Systems, Man and Cybernetics*, 6(4):269–285, 1976.

Appendix A

The Object Extractor Tool

A.1 Design Principles

The *Object Extractor* tool extracts objects from images and videos semi-automatically with the help of the improved version of the flood fill algorithm. The overall architecture of the tool is shown in Figure A.1. Videos are segmented with *Fact Extractor* in the system and keyframes of the videos are processed as images. An image is passed through quantization and color median filtering steps and then the final image, where the object is to be extracted, is produced. This filtered final image is processed with *Flood Fill for Extraction Algorithm* to extract the objects along with their features. Since we deal with realistic images and videos in the system, automatic extraction of objects and features would be inadequate, so that the user intervention becomes inevitable. The last step in the process is storing the features of the extracted objects in the object feature database. Thus, *Object Extractor* is one of the basic tools that cooperate with the query-by-feature sub-system of our video database and querying system [11].

The image whose objects to be extracted passes through a color space conversion step and the color vectors of all of the pixels are transformed from RGB color space into HSV color space. Then, this data is used in the quantization step yielding 18 hue, 3 saturation, 3 value levels and 4 gray levels. After completing

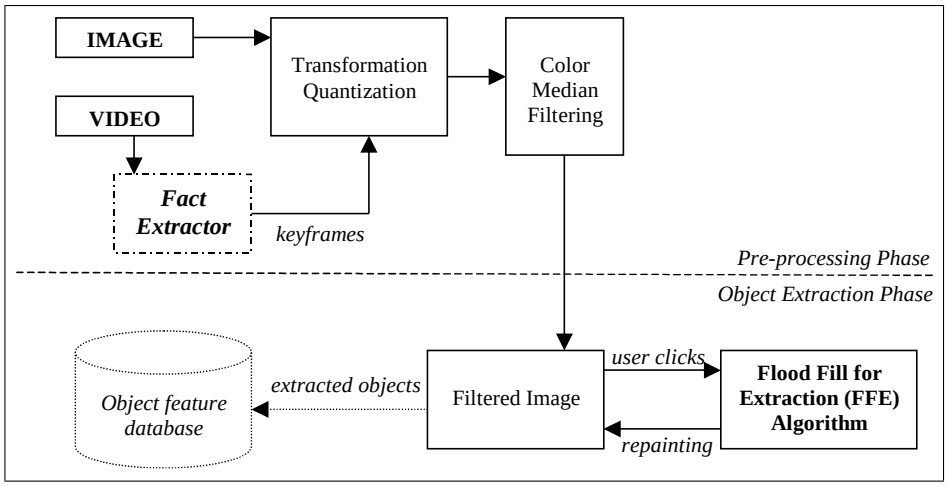


Figure A.1: The Overall Architecture of The *Object Extractor*

this step, the median filtering algorithm is applied to eliminate the non-dominant color regions. The *Flood Fill for Extraction (FFE)* algorithm, an improved version of the flood fill algorithm, is designed to specify object regions. The color of the user-clicked pixel initiates the process and it forks into four branches corresponding to the four neighboring pixels (north, east, south and west). As soon as the difference between the processed pixel's color and the initiative pixel's color exceeds a pre-specified threshold, the execution stops. The user may continue to specify new initiative pixels as necessary to extract the object.

In the pre-processing phase of the *Object Extractor*, color space conversion and color quantization are applied to the input, thus the FFE algorithm performs better. This yields an increase in the effectiveness of the technique. Moreover, since the objects are extracted separately, images containing more than one image are handled successfully. The *Object Extractor* provides an environment where a wide range of images and/or videos are handled effectively by the help of the adapted and improved techniques.

In the current implementation of the tool, the user clicks on a pixel p on the image and the FFE algorithm is initiated with the pixel p and the current color difference threshold t. During this execution, the pixels in the *t-neighborhood* of p are repainted. However, if the object bounds unprocessed pixels, the user may

click onto another pixel for extracting other parts of the object. Having satisfied with the extracted object region, the user labels the object.

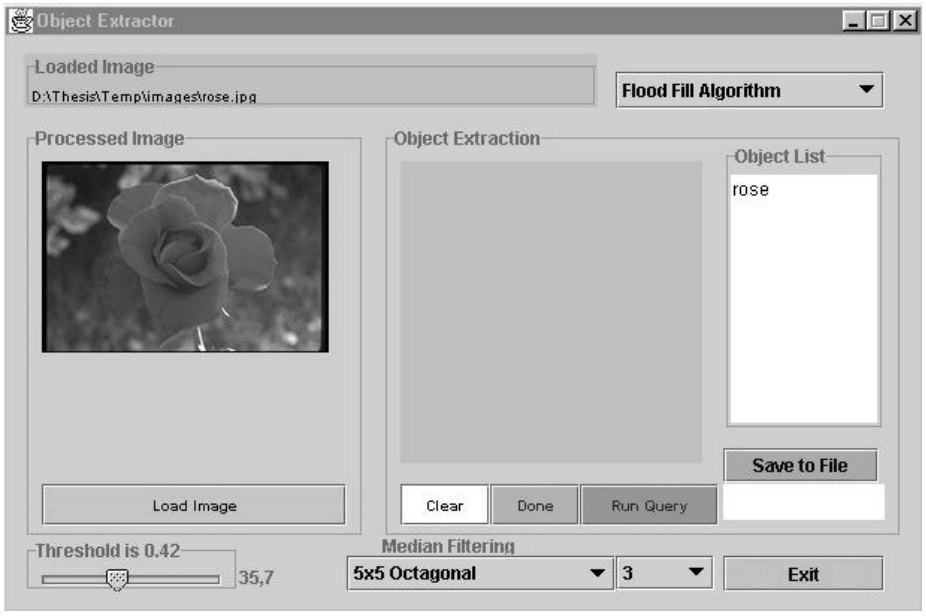


Figure A.2: The Graphical User Interface of The *Object Extractor*.

The graphical user interface of *Object Extractor* has been developed in *Java* programming language and it is shown in Figure A.2. The main usage of the tool is to extract objects for the query-by-feature sub-system of the rule-based video querying system that we develop [11]. 'Run Query' button activates this querying operation with the previously extracted objects. The image to be queried is also segmented with this tool and the objects are extracted. Since all of the extracted objects are handled with a proper indexing mechanism, the extracted objects of the query image can be queried with the existing objects.

A.2 The Object Extraction Algorithm

The *Object Extractor* tool processes both images and/or video frames. The method it employs for both types of data is very similar since each video frame can be treated as a single image. The *Fact Extractor* tool inside the video database

system handles video data and produces video keyframes. Thus, videos can be processed in the *Object Extractor* tool through their keyframes. *Flood Fill for Extraction (FFE)* algorithm works with a pre-processed (i.e., transformed, quantized and median filtered) image. When the algorithm finishes, it repaints some of the pixels on the image and the user may continue the extraction as many times as he/she wants. Figures A.3 and A.4 present snapshots of *Object Extractor* tool with four images.

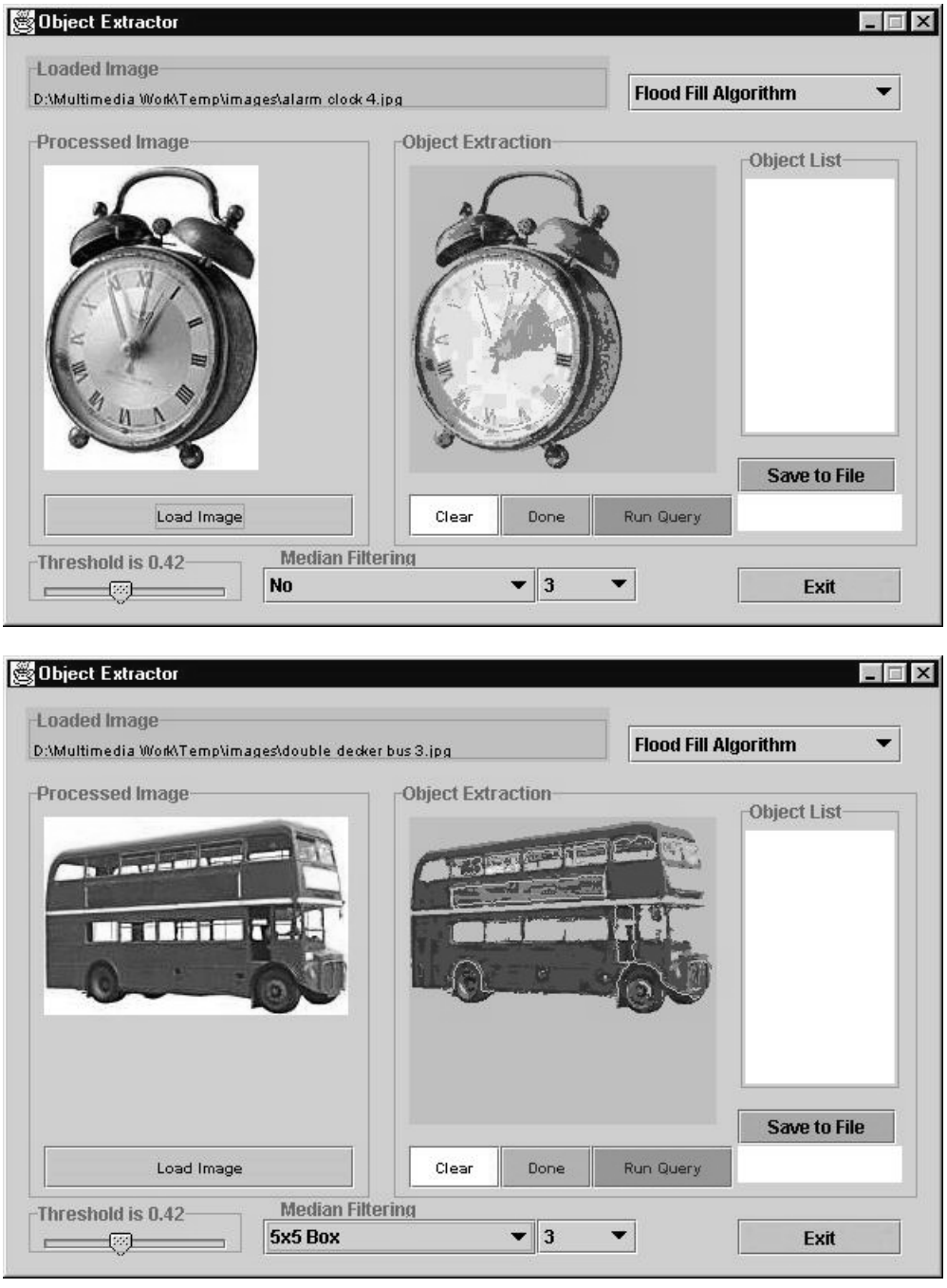


Figure A.3: The Snapshots of The *Object Extractor*.

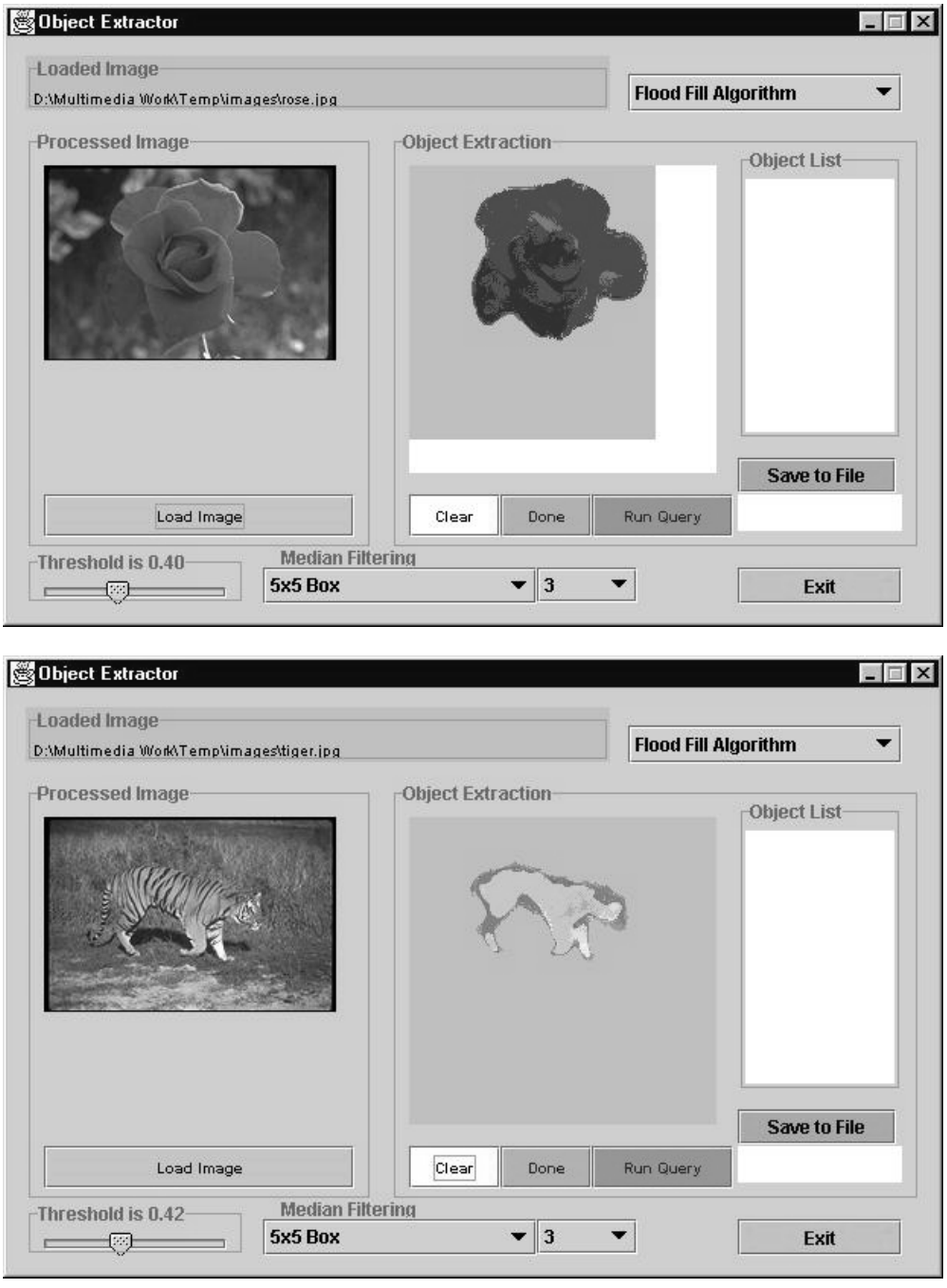


Figure A.4: The Snapshots of The *Object Extractor*.

Appendix B

Brodatz Texture Database

B.1 Brodatz Texture Index

Bark of tree, D12, D13
Beach pebbles, D23, D27, D30, D31, D54
    see also Stone
Beach sand, D28, D29
    see also Stone
Beans, coffee, D74, D75
Brass mesh, woven, D46, D47
Brick wall, D94, D95, D96
    ceramic-coated, D25, D26
Bubbles
    plastic, D111, D112
    soap, D73

Cane, D101, D102
Calf, unborn, hide of, D93
Canvas
    cotton, D77
    French, D20, D21
Cloth
    burlap, loose, D103, D104

cheese-, D105, D106
herringbone weave, D16, D17
homespun woolen, D11
Oriental grass fiber, D76, D79
Oriental straw, D52, D53, D78, D80, D81, D82
woolen, loosely woven, D19
see also Lace
Clouds, D90, D91
Coffee beans, D74, D75
Cork, pressed, D4, D32, D33
Crocodile skin, D10
Crystals, ice, D100

Fieldstone, D2, D7
see also Beach pebbles, Mica, Quartz, Stone
Flowers, dried hop, D88, D89
Fur, calf, D93

Grain, wood, D68, D69, D70, D71
Grass fiber cloth, Oriental, D76, D79
Grass lawn, D9

Hop flowers, dried, D88, D89

Ice crystals, D100

Lace, D39, D40, D41, D42
see also Cloth
Leather
pigskin, D92
pressed calf, D24
see also Skin
Lights, swinging, D43, D44, D45
Lizard skin, D35, D36

Marble, European, D58, D59, D60, D61, D62, D63
see also Stone
Masonite panel, perforated, D48

Matting, straw, D55, D56, D83, D85
Mesh, brass, D46, D47
Mica, expanded, D5
    see also Quartz, Stone

Netting, D34

Paper
    handmade, D57, D109, D110
    Japanese rice, D107, D108
Pebbles, D23, D27, D30, D31, D54
Plastic bubbles, D111, D112
Plastic pellets, D66, D67

Quartz, rose, D98, D99
    see also Mica, Stone

Raffia, D19, D50, D51, D84
    see also Straw
Rattan, handwoven Oriental, D64, D65
    see also Straw
Reptile Skin, D3, D22

Sand, D28, D29
Screening
    straw, D49
    woven aluminium, D1, D6, D14
See fan, fossilized, D87
Skin
    calf, D93
    crocodile, D10
    lizard, D35, D36
    reptile, D3, D22
    see also Leather
Soap bubbles, D73
Stone
    fieldstone, D2, D6
    marble, European, D58, D59, D60, D61, D62, D63

see also Beach pebbles, Mica, Quartz
Straw
cloth, Oriental, D52, D53, D78, D80, D81, D82
matting, D55, D56, D85
North Shore beach, L.I., D15
raffia weave, D18, D84
raffia with cotton, D50, D51
screening, D49

Tree, bark of, D12, D13
Tree stump, D72, D97
Tile, ceiling, D86

Wall, brick, D25, D26, D94, D95, D96
Water, D37, D38
Weaves, cloth
burlap, D103, D104
canvas, D20, D21, D77
cheesecloth, D105
grass fiber, D79
herringbone, D16, D17
lace, D39, D40, D41, D42
woolen, D11, D19
Weaves, metal
brass mesh, D46, D47
wire, abstract illusion, D8
wire, woven aluminium, D1, D6, D144
Weaves, paper
handmade, D57, D109, D110
Japanese rice, D107, D108
Weaves, straw
cloth, D52, D53
matting, D55, D56, D83, D85
Oriental, D78, D80, D81, D82
Oriental grass fiber, D76, D79
raffia, D18, D50, D51, D84
rattan, D64, D65
screening, D49

Wire
woven, abstract illusion of, D8
woven aluminium, D1, D6, D14
woven brass mesh, D46, D47
Wood
grain, D68, D69, D70, D71
tree, bark of, D12, D13
tree stump, D72, D97

B.2 Brodatz Texture Images

In this section, Brodatz texture database images are presented. In the database, there are 112 texture images 12 of which are given in each page separately. Within a page, the order of the images are from left to right, and then from top to bottom.

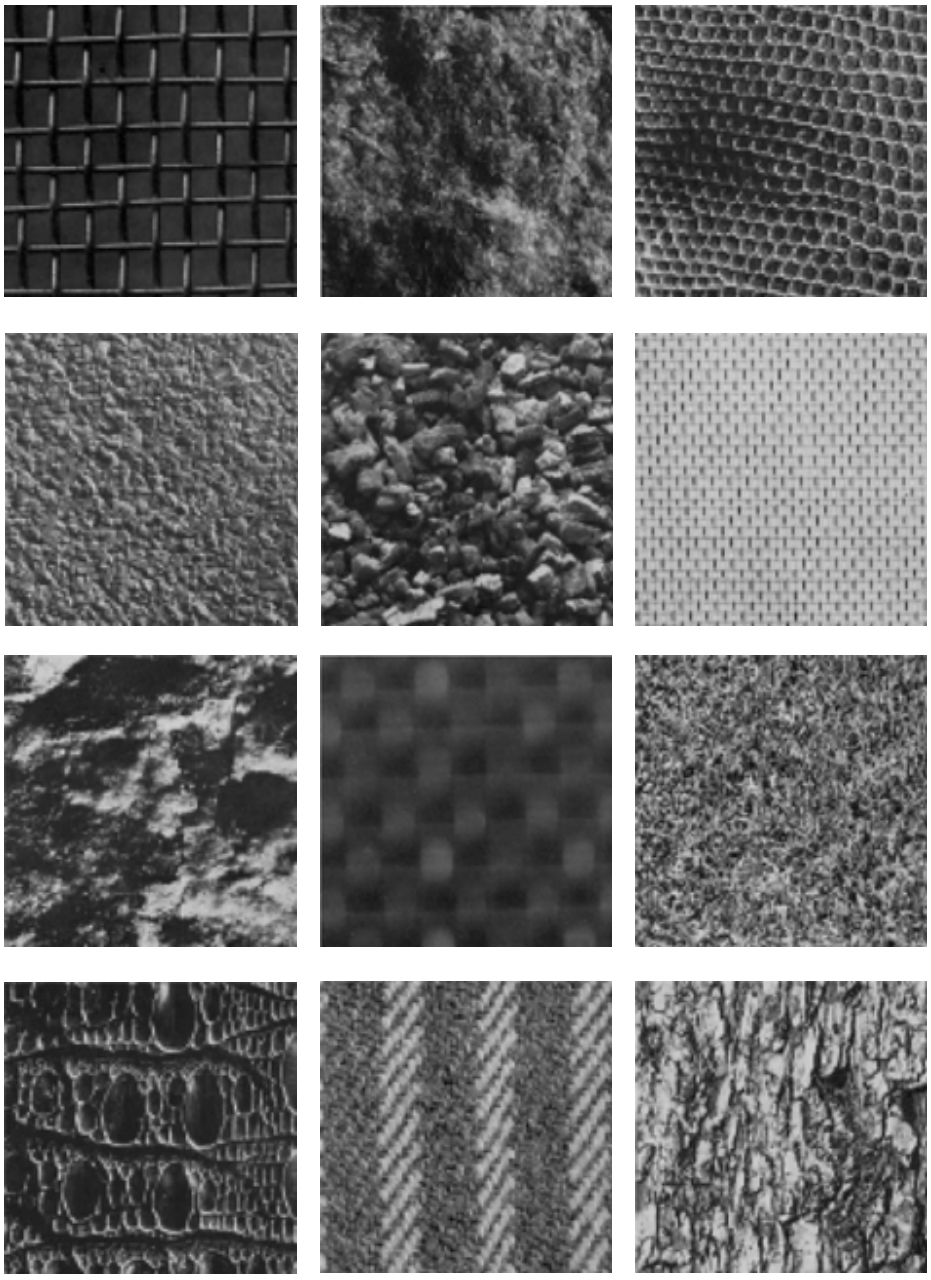


Figure B.1: Brodatz Texture Database Images, D1–D12.

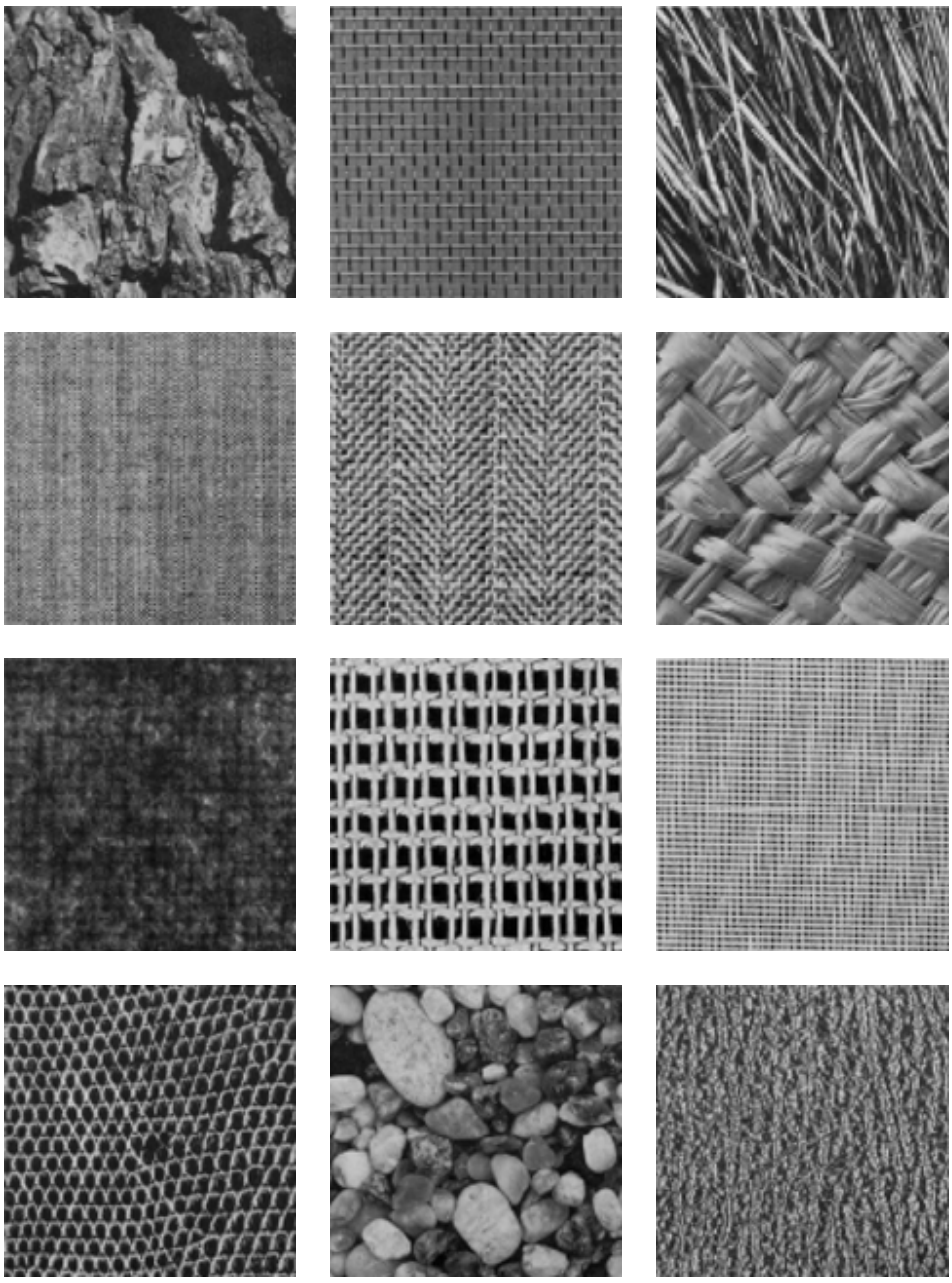


Figure B.2: Brodatz Texture Database Images, D13–D24.

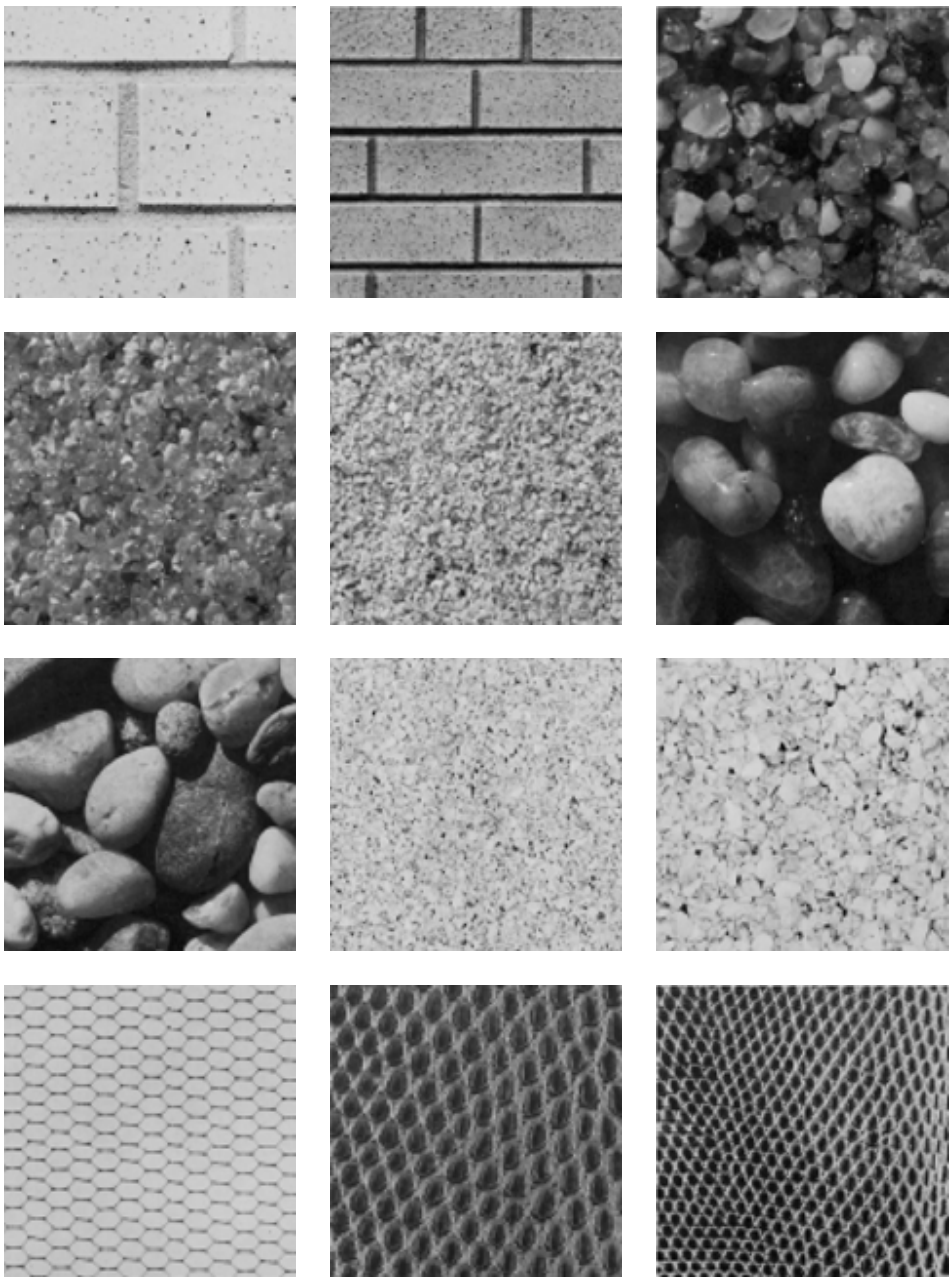


Figure B.3: Brodatz Texture Database Images, D25–D36.

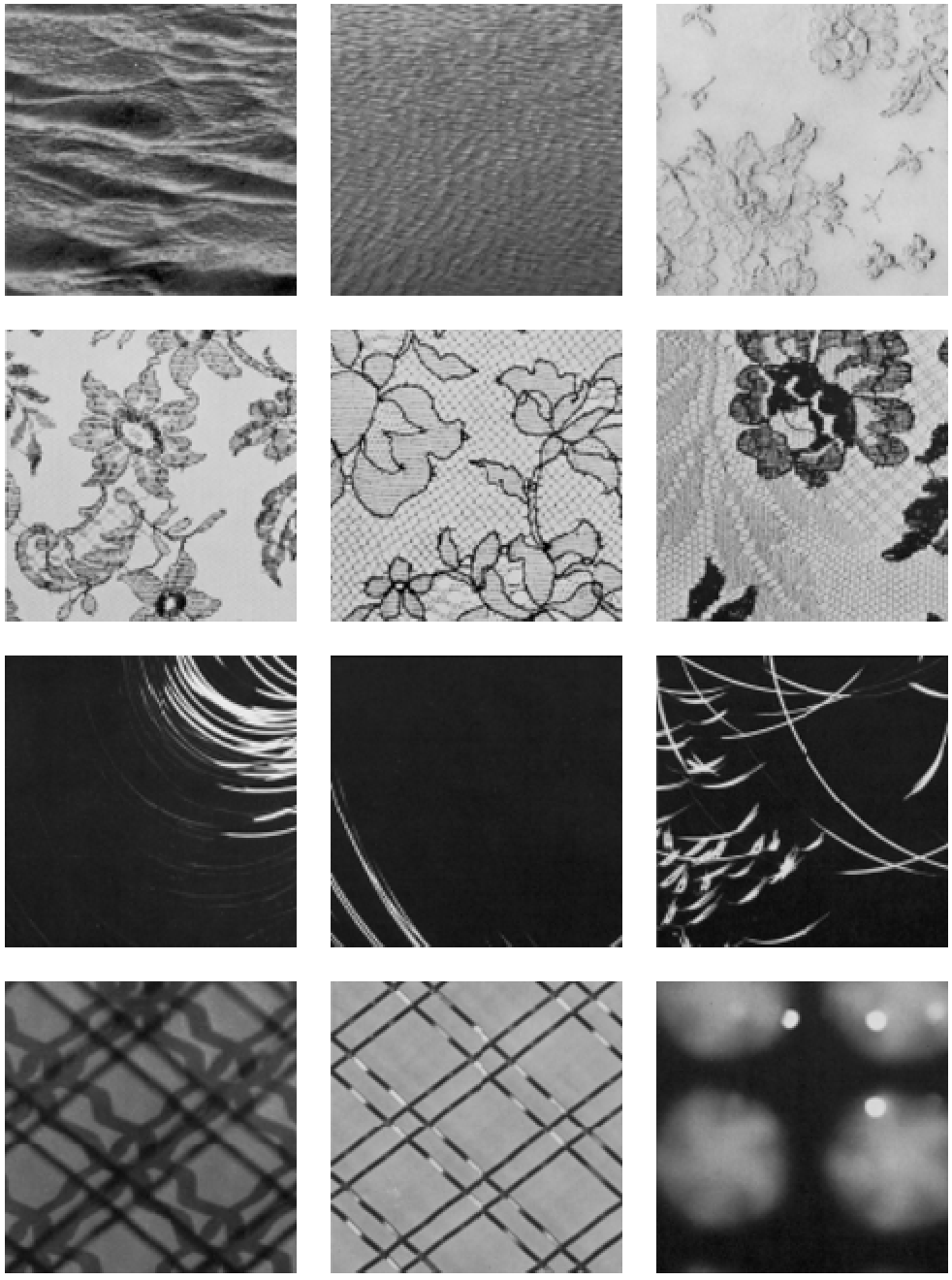


Figure B.4: Brodatz Texture Database Images, D37–D48.

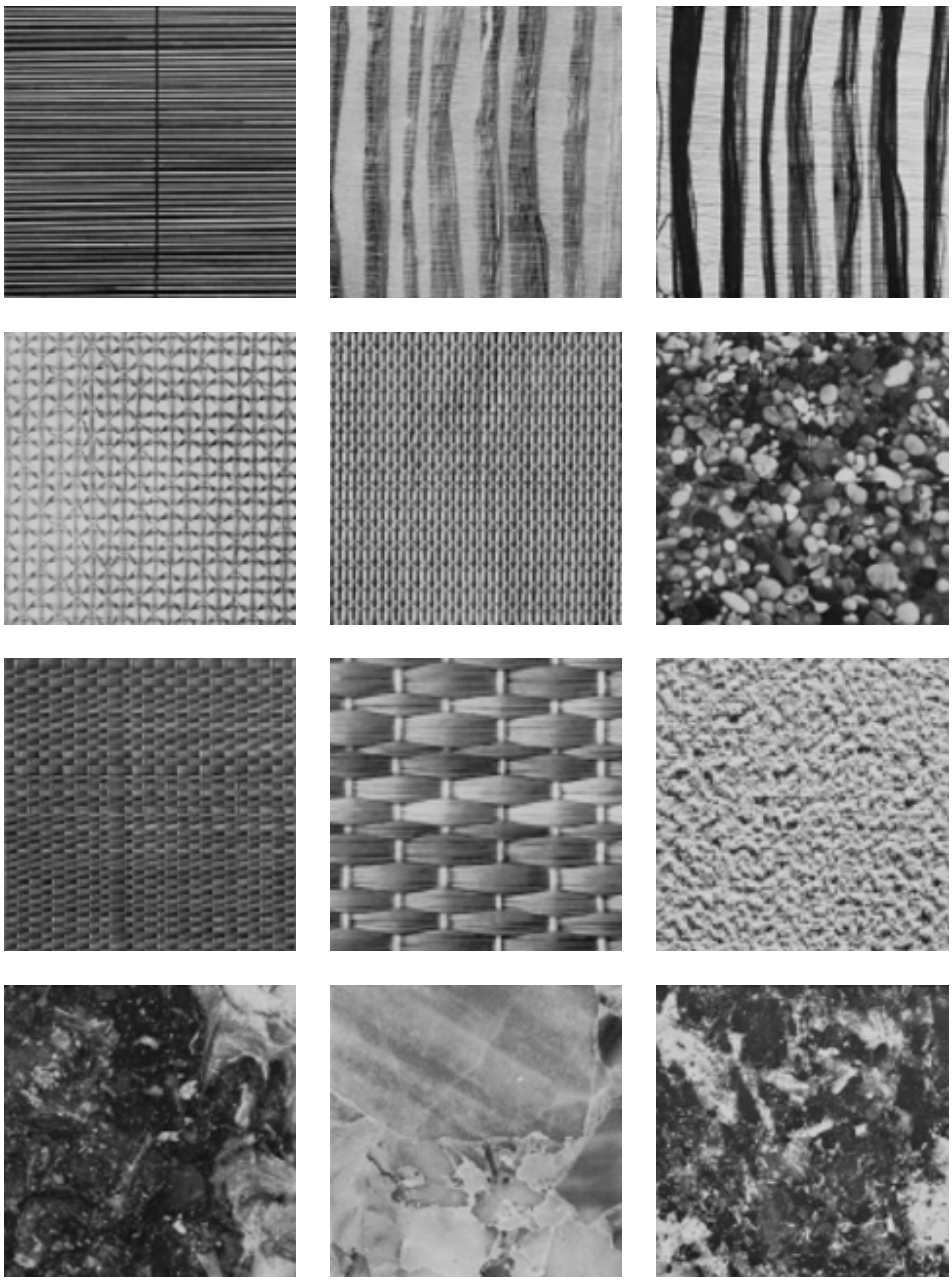


Figure B.5: Brodatz Texture Database Images, D49–D60.

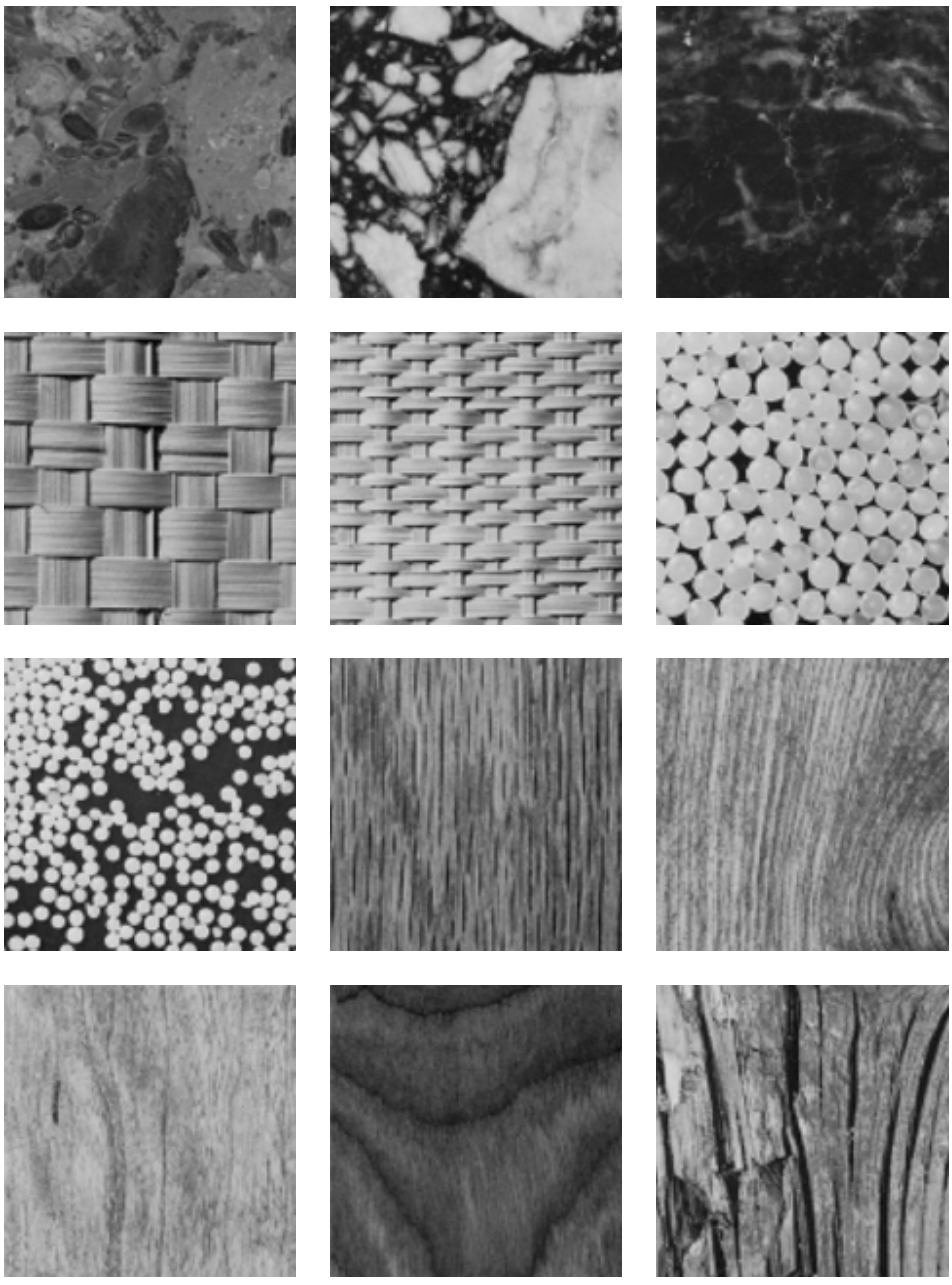


Figure B.6: Brodatz Texture Database Images, D61–D72.

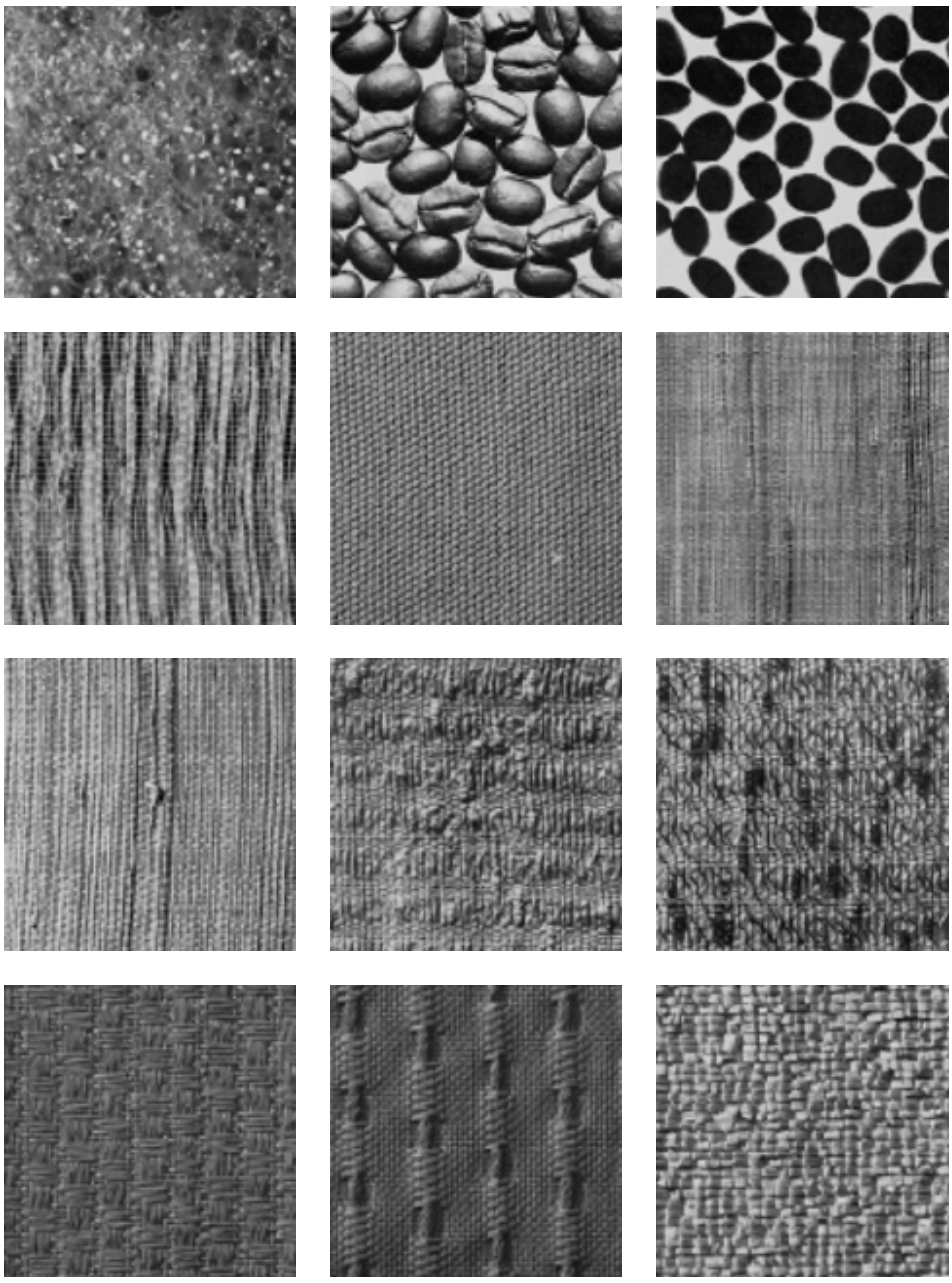


Figure B.7: Brodatz Texture Database Images, D73–D84.

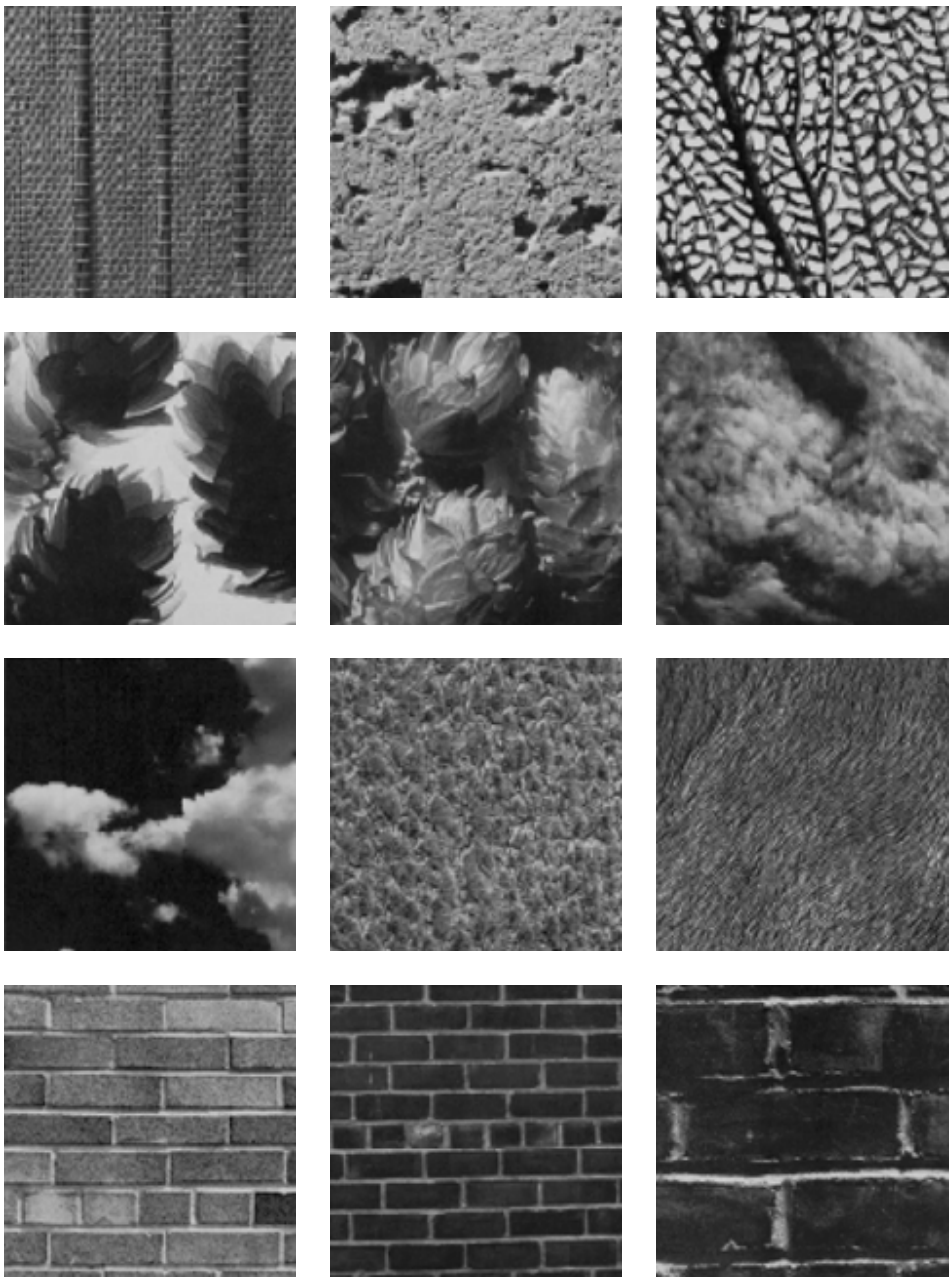


Figure B.8: Brodatz Texture Database Images, D85–D96.

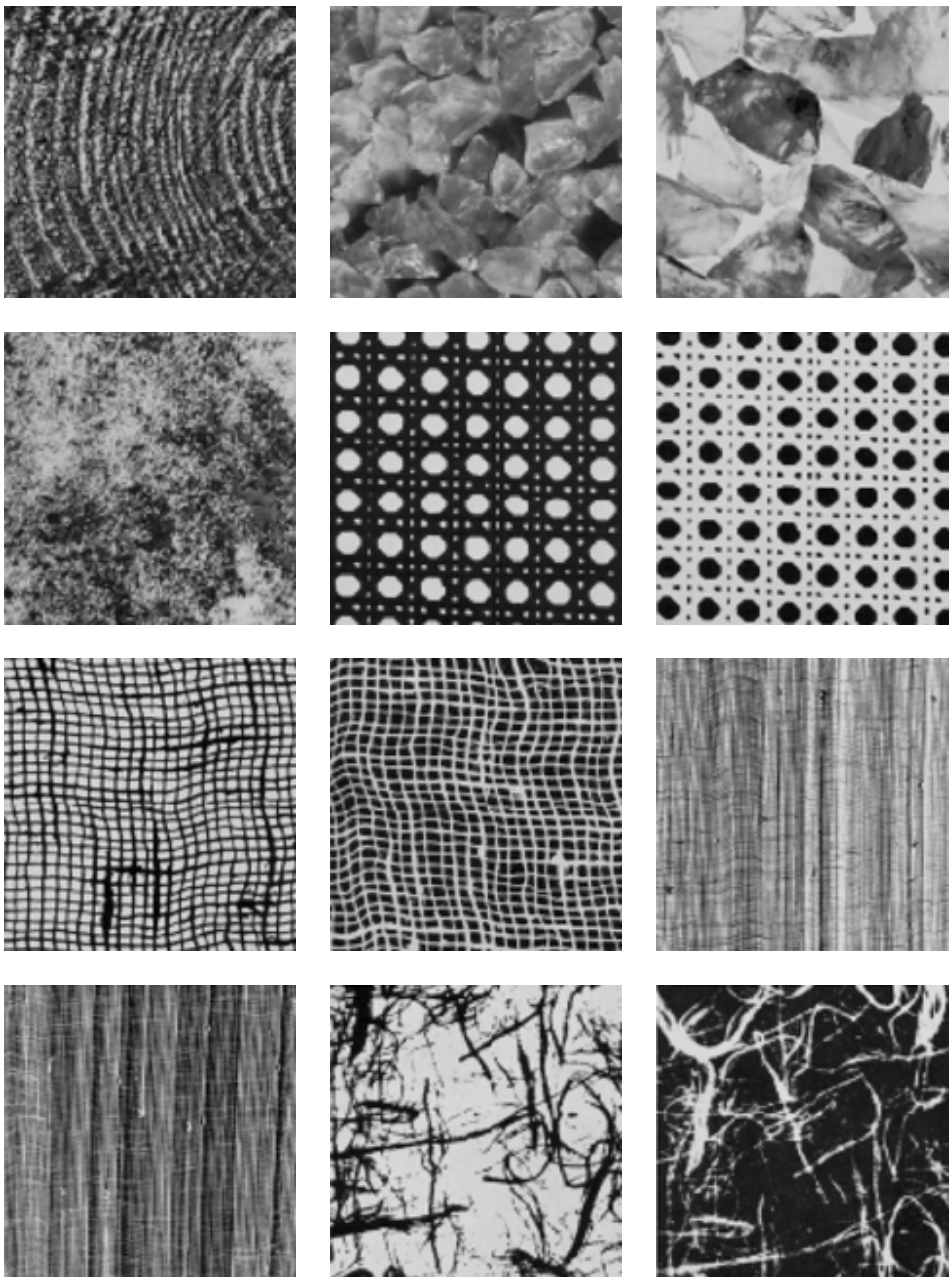


Figure B.9: Brodatz Texture Database Images, D97–D108.

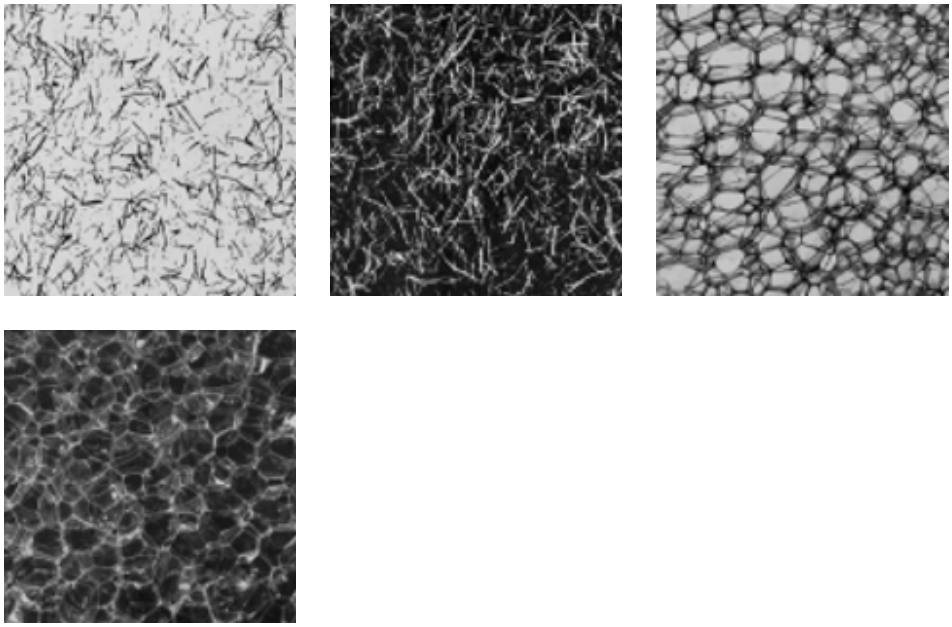


Figure B.10: Brodatz Texture Database Images, D109–D112.