Image Retrieval Using Discrete Curvelet Transform

Ishrat Jahan Sumana

A dissertation submitted in fulfillment of the requirement for the degree of

Master of Information Technology

Gippsland School of Information Technology

Monash University, Australia

November, 2008

Declaration

I, Ishrat Jahan Sumana, affirm that this thesis contains no material which has been accepted for the award of any other degree or diploma in any university or other institution and affirm that to the best of my knowledge, the thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Ishrat Jahan Sumana

Date:

Dedicated to my parents, brother and my beloved husband

Acknowledgement

Writing thesis is a difficult task. It could not be accomplished without the proper directions and guidance of my supervisors. I wish to express my gratitude and appreciation to my supervisors Professor Dr. Guojun Lu and Dr. Dengsheng Zhang for their patient and kind support throughout the research and thesis writing process. Their vision, keen technical expertise, and enthusiasm have provided a boost to this research.

Lots of thanks to my friends and colleagues in Gippsland School of Information Technology for their cooperation. During the whole period of research, I spent a big part of each day in the 7N research centre, so 7N has become a part of my precious memories.

I would like to thank all the staff in Gippsland School of Information Technology for their prompt administrative support. The generous research facility provided by Monash University is beyond my expectation. I am really grateful to this School for providing me such a splendid opportunity to use the resources throughout the research.

My heartiest gratitude to my beloved parents, my younger brother, and my husband for their unconditional support in every aspect of my life. They extended their helpful hands whenever I needed. Without their patience and inspiration it would not be possible for me to start and continue my research.

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

Abstract

Finding specific digital images from large resources has become an area of wide interest nowadays. Among image retrieval approaches, text based retrieval is widely used as it has been commercialized already. But it is not effective as it involves time consuming text annotation process. Also there is difference in understanding of image content which affects image labeling process. Content based image retrieval (CBIR) is another method of retrieving images from large image resources, which has been found to be very effective. CBIR involves the use of low-level image features, like, color, texture, shape, and spatial location, etc. to represent images in terms of their features. To improve existing CBIR performance, it is very important to find effective and efficient feature extraction mechanisms. This research aims to improve the performance of CBIR using texture features.

Texture is one of the most important and prominent properties of an image. It is the surface pattern of objects in the image or the whole image. Texture features effectively describe the distinguishing characteristics between images. After extensively studying existing texture feature descriptors, we have proposed a wrapping based discrete curvelet texture descriptor for future use in CBIR. Discrete curvelet transform is one of the most powerful approaches in capturing edge curves in an image. Related works on curvelet features are also investigated.

In this research, we generate a texture features descriptor using wrapping based discrete curvelet transform. This descriptor is used to represent images in a large database in terms of their features and to measure the similarity between images. The retrieval outcome shows, the proposed curvelet texture feature descriptor outperforms the Gabor filters in both retrieval accuracy and efficiency. The optimal level of curvelet decomposition is also investigated to obtain the highest retrieval outcome in terms of effectiveness and efficiency. To observe the scale distortion tolerance of the curvelet features, CBIR tests are performed using a large database containing scale distorted images along with the original images. From the experimental results, we find that curvelet texture feature is robust to a reasonable scale distortion. We compare curvelet retrieval to Gabor filters retrieval performed on the scale distorted data and find that curvelet is more robust to scale distortions than Gabot filters.

List of Abbreviations

1. CBIR: Content Based Image Retrieval
2. QBIC: Query By Image Content
3. MR: Multiresolution
4. MR-SAR: Multiresolution Simultaneous Auto-Regressive
5. PWT: Pyramid structured Wavelet Transform
6. TWT: Tree structured Wavelet Transform
7. DWT: Discrete Wavelet Transform
8. 2-D: 2 Dimensional
9. HSV: Hue, Saturation, Value
10. HSL: Hue, Saturation, Lightness
11. HSB: Hue, Saturation, Brightness
12. RGB: Red, Green, Blue color space
13. DCT: Discrete Cosine Transform
14. FT: Fourier Transform
15. STFT: Short Time Fourier Transform
16. OWT: Orthogonal Wavelet Transform
17. BWT: Bi-orthogonal Wavelet Transform
18. USFFT: Unequally-Spaced Fast Fourier Transform
19. IFFT: Inverse Fast Fourier Transform
20. CSF: Curvelet Statistical Features
21. CCF: Curvelet Co-occurrence Features
22. FFT: Fast Fourier Transform

Publication Based on This Research

I. Sumana, M. Islam, D. S. Zhang and G. Lu, "Content Based Image Retrieval Using Curvelet Transform", In Proc. of IEEE International Workshop on Multimedia Signal Processing (MMSP08), Cairns, Queensland, Australia, ISBN 978-1-4244-2295-1, pp.11-16, October 8-10, 2008.

List of Figures

List of Tables

Table of Contents

Introduction

1.1 Overview

There are many large resources on the web sites which people can use to create and store images. This has created the need for a means to manage and search these images. Therefore, finding efficient image retrieval mechanisms has become a wide area of interest to researchers. Image retrieval method is a system for searching and retrieving images from a large database of digital images. For the last few decades, researchers have been working on image retrieval processes and two types of promising techniques have been developed such as, text based image retrieval and content based image retrieval (CBIR).

In text based image retrieval method, users use keyword or description to the images as query so that they can use the retrieved images, which are relevant to the keyword. Text based retrieval has several disadvantages. First of all, there is inconsistency in labeling by different annotators due to different understanding about image contents. For example, an image consisting of grass and flowers might be labeled as either ‘grass’ or ‘flower’ or ‘nature’ by different people. Second, it consumes a lot of time to annotate each image in a large database and makes the process subjective [1]. Third, there is a high probability of error occurrence during the image tagging process when the database is

large. As a result, text based image retrieval cannot achieve high level of efficiency and effectiveness.

Yahoo web based image searching is an example which uses text based image retrieval. In most cases, we find only the first few retrieved images are relevant to the query. For instance, the first 2 pages of retrieved images for the key word 'ginger' are shown in Fig. 1.1. Most images are irrelevant among the retrieved images in this figure.

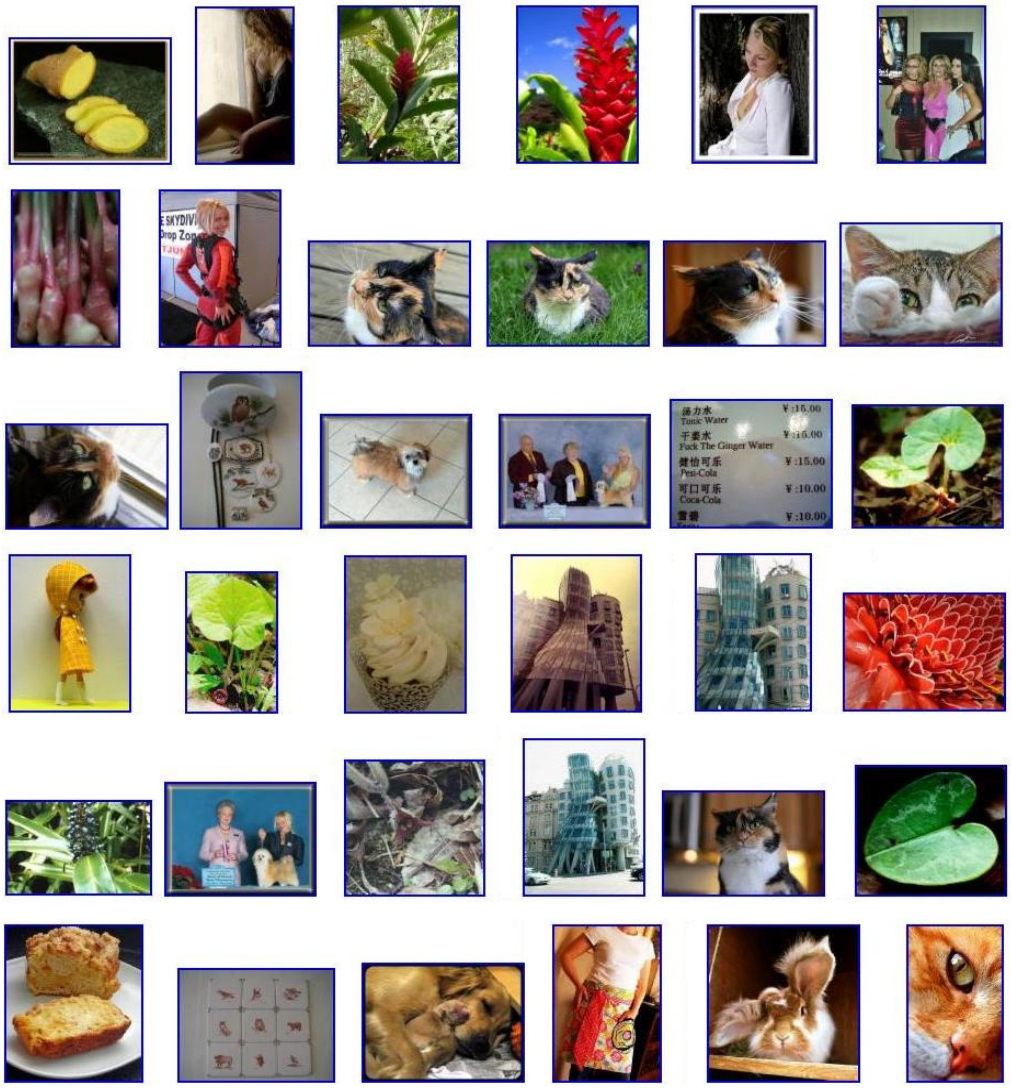


Fig. 1.1: The First 2 pages of retrieval using the key word 'ginger' from Yahoo [2].

Content based image retrieval is also known as Query By Image Content (QBIC) [3]. The term CBIR originated in the early 1990's [1]. It is an automated technique that takes

an image as query and returns a set of images similar to the query. Low-level image features like texture, color, and shape are extracted from the images of the database to define them in terms of their features. Images of the same category are expected to have similar characteristics. Therefore, when similarity measurement is performed on the basis of image features, the output set achieves a high level of retrieval performance. CBIR has several advantages over the traditional text based retrieval. Due to using the visual contents of the query image in CBIR, it is a more efficient and effective way at finding relevant images than searching based on text annotations. Also CBIR does not consume the time wasted in manual annotation process of text based approach. These advantages have motivated us to employ a CBIR technique for our research.

Among the low level image features, texture has been shown to be effective and objective in CBIR. A variety of techniques have been developed for extracting texture features, broadly classified into the spatial and spectral methods. The spatial approaches mostly rely on statistical calculations on the image. Unfortunately, these statistic techniques are sensitive to image noise and have insufficient number of features [4]. On the other hand, spectral methods of texture analysis for image retrieval are robust to noise. The spectral methods include the use of discrete cosine transform [5], multiresolution (MR) methods such as, Gabor filters [6-8] and wavelet transform [9-11] for texture representation. The drawback with these spectral methods is that they do not capture the edge information of an image effectively. This is the reason for finding better multiresolution spectral approaches, which can capture the edge and orientation information of an image effectively.

1.2 Background

We have provided the overview of CBIR system and the problems in this method. Now we briefly describe the background of CBIR.

Among the spectral approaches of texture feature representation, multiresolution simultaneous auto-regressive model (MR-SAR), wavelet transform, Gabor filters and Wold decomposition have been found to be used in CBIR successfully [1]. Wavelet transforms have been used most widely in many aspects of image processing. A wide range of wavelet-based tools and ideas have been proposed and studied for noise removal from images, image compression, image reconstruction, and image retrieval. The multiresolution wavelet transform has been employed to retrieve images in [11]. The wavelet features do not achieve high level of retrieval accuracy. Therefore, various methods have been developed to achieve higher level of retrieval accuracy using wavelet

transform. Wavelet features computed from discrete wavelet coefficients are assigned weights to increase effectiveness in CBIR [10]. In this approach, features from lower resolutions are assigned higher weights as low resolution coefficients are likely to contain major energy portions of an image.

Wavelets perform better for one dimensional signal as it is good in representing point discontinuities. All singularities in a one dimensional signal are point singularities, so wavelets have certain universality there. However, in higher dimensions, more types of singularities may exist. In these cases wavelets lose their universality [12, 13] and perform merely good enough in capturing the edge discontinuities in 2-D space, which is important in texture representation.

Another multiresolution approach, the Gabor filters, consists of a group of wavelets each of which capturing energy at a specific resolution and orientation. Therefore, Gabor filters are able to capture the local energy of the entire signal or image. Daugman discovered that Gabor filters provide optimal Heisenberg joint resolution in visual space and spatial frequency [14]. The spatial responses of a Gabor filters is very similar to the receptive field profiles in mammalian vision [14]. For this reason, Gabor filters have been successfully employed in many applications including image coding, texture segmentation, retina identification, document analysis, target detection, fractal dimension measurement, line characterization, edge detection, image representation, and others. Gabor filters are also applied to solve the problem of retrieving images with rotation [15]. Based on the experimental results of [16], it has been found that the Gabor filters are the most promising method among tree structured wavelet (TWT), pyramid structured wavelet (PWT), Tamura and MR-SAR features in retrieving images from a large and standard database like Brodatz album. However, Gabor filters have half bandwidth problem [7]; i.e. to avoid redundancy in the filtered image, half-peak magnitude support of the frequency responses is considered in the spectral domain.

To overcome the problems in using the discrete wavelet and Gabor filters transform, a new multiresolution approach named discrete curvelet transform has recently been developed by E. J. Candès and D. L. Donoho [12]. Curvelets take the form of basis elements, which have elongated effective support; i.e. length > width [17]. Therefore, curvelets can capture anisotropic elements such as the edges of an image effectively [18]. Furthermore, curvelet spectra cover the frequency plane of an image completely. For these important properties, curvelet transform can be used as a powerful image feature capturing tool in CBIR. So far, curvelet transform has only been used for image denoising [19], character recognition [20], and the classification of hand written manuscripts [21]. There is no systematic analysis and representation of discrete curvelet transform in texture based CBIR.

Texture feature representation and its use in CBIR is an important research issue. Though many works on texture classification and representation have already been done, it is still an open issue. Using discrete curvelet texture descriptor is a new and promising direction in image retrieval. In order to understand the effectiveness of the curvelet features for CBIR, we make a systematic analysis, application and evaluation of this feature in this research. In the next section, we present the aim and contribution of our research.

1.3 Aim and Contribution of Thesis

As mentioned above, texture representation for CBIR requires both efficiency and effectiveness. Several methods have been described in recent literature using both the spatial and spectral approaches. Most of them are not capable of effective texture representation. Recent researches on the multiresolution analysis, especially curvelet transform, provide a good opportunity to extract more accurate texture features suitable for use in image retrieval. Curvelet transform provides more detailed information of the image in the spectral domain using more orientation information at each scale. Therefore, our main objective is to employ curvelet texture features for content based image retrieval.

Aim and contribution of this thesis are summarized as following.

1. Our first objective is to investigate multiresolution spectral features used in CBIR. Extensive studies are made to find out a suitable spectral approach of texture features representation. From the CBIR background, we analyze and examine the advantages and disadvantages of several spectral methods. Therefore, we study the spectral methods used in CBIR and try to investigate the reasons of the drawbacks. We then investigate the spectral approaches which can overcome these problems in CBIR. Multiresolution discrete curvelet transform [12] is then found as an effective approach to represent image textures.
2. Second, we study the existing methods of curvelet transform. Based on this study we find wrapping based fast discrete curvelet transform is the most efficient and effective among the existing curvelet transforms [18]. We then study and analyze this method to understand its construction and its advantages in representing texture of an image. Related works using the discrete curvelet transform are also investigated.
3. Third objective of the research is to apply and evaluate discrete curvelet texture features for image retrieval. For CBIR, it is necessary to index all the images in a

large database in terms of their features. Therefore, wrapping based discrete curvelet transform is used to represent database images using low order statistics for retrieval. To study the robustness of this feature, curvelet texture features are used to retrieve images from a large database consisting of original and scale distorted images.

4. Finally, to complete the systematic analysis, application and presentation of curvelet texture features in CBIR, we compare the retrieval performance of wrapping based discrete curvelet, Gabor filters and discrete wavelet texture features. This performance comparison is also necessary to measure the effectiveness and efficiency of curvelet texture features in CBIR. Based on this comparison, it can be decided whether this approach can be chosen as a standard for future CBIR works. Published paper related to this part of research is in the 'Publication Based on This Research' on page viii.

1.4 Organization of thesis

Chapter 1 provides an overview of the research problem and a brief background. The objectives of the research are also described in this chapter.

In the first part of **Chapter 2,** the key issues that affect the performance of the CBIR system including the selection of database, image features, similarity measurement, and performance evaluation method are described. In the second part of this chapter, several texture representation mechanisms in the spectral domain, e.g., the discrete cosine transform, the Fourier transform, multiresolution filtering such as discrete wavelet and Gabor filters are described and discussed with corresponding pros and cons.

Chapter 3 discusses the concept of multiresolution curvelet transform. This includes the representation of curvelets to show the effectiveness of curvelet transform in capturing texture property of an image. Related works on curvelet transform are also discussed at the last part of this chapter.

In **Chapter 4,** we describe the method to compute a curvelet texture descriptor for image databases. In addition, the image retrieval process based on this texture descriptor is described. To compare curvelet texture features with that of the Gabor filters, we provide the description of Gabor filters texture representation for the database we use in this CBIR research.

Chapter 5 analyzes the effectiveness and efficiency of curvelet texture features on both normal and scaled image retrieval. The outcome is compared with two well established texture representations, i.e., discrete wavelet and Gabor filters. The retrieval

performances of discrete curvelet, discrete wavelet and Gabor filters features in CBIR are analyzed in this chapter.

Chapter 6 concludes the thesis with a summary of the findings and contributions in this research. Several possible future directions of this research are also provided in this chapter.

A Brief Review of CBIR

2.1 Introduction

Content based image retrieval depends on several factors, such as, feature extraction method, suitable features to use in CBIR, similarity measurement method, mathematical transform chosen to calculate effective features, user feedback, etc. All these factors are important in CBIR. Since an improvement to any of these influencing factors can result in a more effective retrieval mechanism. For this purpose, we first provide a brief review of the factors that can affect CBIR.

From the discussion on CBIR in Chapter 1, we find color, texture, shape, these low-level image features can be used directly to gather information from an image for retrieval. The aim of this research is to find and use the most effective texture features in CBIR. Therefore, we study all the existing approaches of representing image texture features in recent literature and discuss several spectral approaches of texture features extraction as those are robust to noise. In this discussion, we try to address what are the shortcomings of one spectral approach and how another approach provides the solutions, and which one is the most effective in texture features representation.

2.2 Key Factors Affecting CBIR

We have already described the basic concept of content based image retrieval in Chapter 1. We now describe the important issues of content based image retrieval system, which are [1]

1. Selection of image database,
2. Similarity measurement,
3. Performance evaluation of the retrieval process and
4. Low-level image features extraction.

In this section, we describe the use and effects of these factors on CBIR with reference to recent research.

2.2.1 Image Database Selection

To ensure the quality of research it is necessary to select a proper dataset as it plays an important role in the CBIR performance. Based on the research objectives, choosing a proper database is necessary to establish a relevant analysis of various CBIR techniques. For any database, it is important to determine the ground truth, on the basis of which retrieval is performed and performance is measured. Size and variety are other two properties of a database, which also affects the retrieval outcome. If the database size is large consisting of multivariate images, a good retrieval result ensures the acceptance of the database as well as the implied method as a standard. Retrieval result varies significantly by selecting different databases as the ground truth definition, size, and variety of the databases are different. From recent literature, we find a subset of Corel database [22] has been used in most of the color image retrieval systems and the Brodatz database [23] has been taken as a standard of texture analysis and texture studies [7, 24-27]. For the Brodatz album, the ground truth is, there are 112 categories of textures each of which consists of 16 similar but non-overlapping textures. This database contains both man made and natural textures, so the large collection of diverse varieties of textures is a good source of texture based retrieval study. Brodatz database has already been used in many texture based image retrieval [7, 15, 25, 28, 29] to benchmark the CBIR performances. For all these reasons Brodatz database is appropriate for texture based CBIR and texture analysis.

2.2.2 Similarity Measurement

Once features are extracted, from all the database images and the query image, the similarity measurement becomes the crucial issue in content based image retrieval. Similarity measurement is the process of finding the difference or similarity between the database images and the query image using their features. The database image list is then sorted according to the ascending order of distance to the query image and images are retrieved from the database according to that order. There are various methods of calculating this distance, such as the Minkowski-Form distance, quadratic form distance, Mahalanobis distance, Kullback-Leibler divergence, and Jeffrey-Divergence [1].

The Euclidean distance, also known as L_2 distance, is one variety of the Minkowski-Form distance [1]. It has been used in many content based image retrieval approaches. It is applicable when the image feature vector elements are equally important and the feature vectors are independent of one another. The Euclidean distance has been used in shape-based CBIR [30], retrieval using wavelet transform [11], color and texture image retrieval using weighted wavelet descriptor [10], Netra, a region based image retrieval system using color, texture, shape and spatial location information [31], etc. Therefore, Euclidean distance has been taken as a standard to similarity measurement in CBIR process.

2.2.3 Performance Evaluation Methods of CBIR

The level of retrieval accuracy achieved by a system is important to establish its performance. If the outcome is satisfactory and promising, it can be used as a standard in future research works. In CBIR, precision-recall is the most widely used measurement method to evaluate the retrieval accuracy. We have found some recent literature [6, 32-34] use this pair to measure the retrieval performance. Precision P is defined as the ratio of the number of retrieved relevant images r to the total number of retrieved images n, i.e., $P = r/n$ [1]. Precision measures the accuracy of the retrieval.

$$P = \frac{No.\ of\ relevant\ images\ retrieved}{Total\ no.\ of\ images\ retrieved} = \frac{r}{n} \tag{2.1}$$

Recall is defined by R and is defined as the ratio of the number of retrieved relevant images r to the total number m of relevant images in the whole database, i.e., $R = r/m$ [1]. Recall measures the robustness of the retrieval.

$$R = \frac{No.\ of\ relevant\ images\ retrieved}{Total\ no.\ of\ relevant\ images\ in\ DB} = \frac{r}{m} \tag{2.2}$$

Generally, precision values fall with increases in the recall values. A retrieval system can be called 'ideal' if both the precision and recall values remain high. However, in reality, no such image retrieval system has yet been found. Precision-recall pair is a good standard of performance evaluation. It provides meaningful result when the database type is known and has been effectively used in some earlier research. For other data sets, especially those which have been created by collecting user generated images, the result may vary due to different human concepts of image classification. The precision-recall pair behavior of our experiment is analyzed in detail in Chapter 5.

2.2.4 Low-level Features in CBIR

A number of low-level image features can be extracted from an image. Detailed study on image features are presented in [1, 35]. Some commonly used low-level image features in recent literature includes the application of color, texture, shape, spatial location, etc. Some CBIR approaches use a combination of more than one low-level feature to improve retrieval performance. In this section we briefly describe the features used in recent CBIR researches and their impacts.

Color is one of the most prominent visible properties of an image and each pixel of image contains a different value for color. As human vision perception can easily distinguish different colors, application of color features has widely been accepted in numerous CBIR applications. Before generation of a color descriptor, it is necessary to define a suitable color space. From the recent literature, we find HSV or HSL or HSB, YCrCb, CIE-L*u*v*, CIE-L*a*b* are popularly used in CBIR [10, 16, 32, 36-38]. Various color spaces have already been developed and used for different purposes in image processing. In some retrieval approaches, color features are combined with texture features to obtain a better performance [10, 36-38]. For convenience in color feature extraction process, color space conversion processes have been introduced. The transformation from RGB to HSV, HSB or HSL space is described in [39] whereas RGB to CIE-L*u*v* or CIE-L*a*b* conversion is shown in [40]. Among the color spaces, HSV is more useful in measuring perceptual similarity. Commonly used color descriptors include the use of the color histogram, color moments, the color coherence vector, and the color correlogram. Sometimes more than one color descriptors is used for image

retrieval, e.g., in [37] all the color descriptors mentioned are used to retrieve cervicographic images to detect cancer cell patterns. Generally, different objects of the same color are expected to have different texture patterns. Therefore, color feature alone is not enough to differentiate images. Color feature in combination with other features such as texture and shape provides more accurate CBIR performance.

Another low-level feature that can easily be identified by human being is **shape**. Often, using color and texture features combinedly make it harder to distinguish the shape of separate objects in an image. Eventually, it becomes difficult to define the shape clearly. Shape descriptions can be of two categories, boundary based and region based [1]. Boundary based shape description, which considers the shape of the outer boundary only, includes the application of Fourier shape descriptors, finite element models, polygonal approximation and rectilinear shapes. Region-based shape description, which considers the entire region of a defined shape, includes moment invariants to translation, rotation and scale [1]. Fourier descriptors have been used in 2-D shape classification and have performed better than autoregressive modeling based shape descriptor [41]. Usually the shape features are useful after segmenting the image. Due to the difficulties related to using robust and accurate segmentation algorithm, only limited applications of shape features are found in CBIR.

Spatial location helps to resolve the ambiguity in defining objects in an image or defining difference between images due to color and texture similarity [1]. Sometimes blue sky and sea may possess the same texture and color histogram, thereby making it difficult to distinguish them. The ambiguity can be resolved when the word ‘top’ is used for sky and ‘bottom’ is used for sea to specify their spatial location.

Texture is an important and prominent visual property of an image. Low-level texture features play a vital role in CBIR and texture classification. Defining texture is difficult. Almost all the natural and real world images are composed of different kinds of objects. If we observe carefully, we will find these objects have different surface patterns within the image. In a simpler way, it can be said that the surface pattern of an object in the image or of the whole image is known as the texture. For example, some objects may have repetition of similar patterns (brick built wall), some objects may look directional (a grass field) and some may contain no regular patterns (sky, soil, and fractal objects). Similar images are expected to have similar texture patterns, so texture features are important for content based image retrieval.

Basically, there exist two types of texture features extraction approaches, such as spatial domain and spectral domain methods. The classification of texture feature extraction methods is shown in Fig. 2.1 [73].

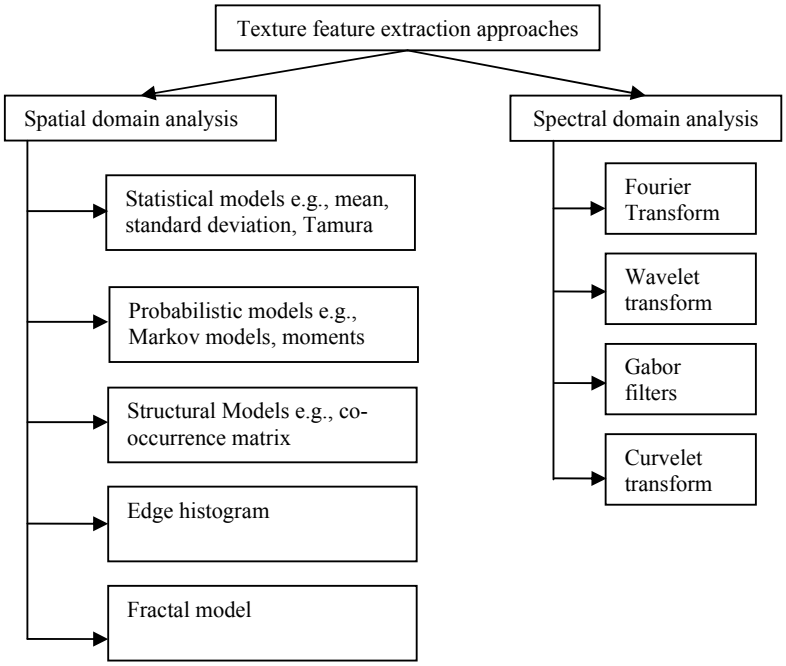


Fig. 2.1: Classification of texture features extraction methods.

The spatial domain approaches are susceptible to noise. As a result, small changes in the image due to noise affect the total feature extraction process. When the same image with and without noise is compared, the values of their texture features vary significantly (shown in Fig.2.2). As a result, similar but noisy images may not be retrieved in a CBIR method using spatial features.

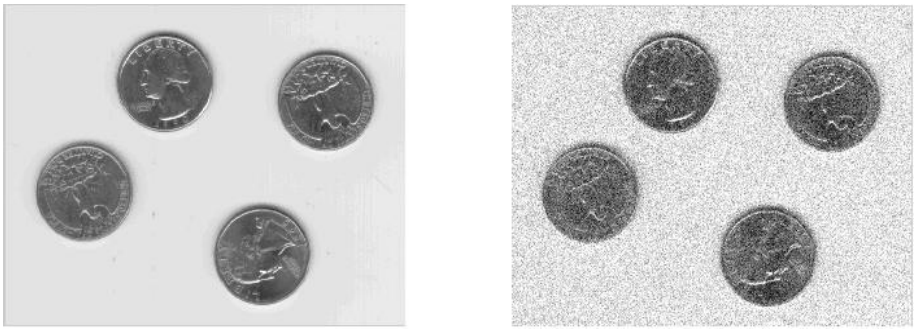


Fig. 2.2: Right side image is created by adding Gaussian noise (mean = 0, variance = 0.01) to the image in the left.

In Fig. 2.2, right side image is generated by adding Gaussian noise to the left side image. Mean and standard deviation are then calculated from their spatial domain coefficients. Left side image has a mean and standard deviation of 198.26 and 52.84 respectively. However, these values are 196.88 and 56.97 respectively, for the right hand side image. If retrieval is done using the spatial texture method, they are likely to be treated as different images because their features have large difference.

Tamura texture features are typical spatial domain features. Tamura texture features consist of 6 components, e.g., coarseness, contrast, directionality, likeliness, regularity, and roughness [1]. Among these, coarseness, contrast and directionality are considered to be more important [1]. The Tamura feature descriptor is not effective when used to represent deformed images because it is sensitive to scale and orientation [26].

On the other hand, spectral domain methods like multiresolution wavelet [42], Gabor filters [14, 43], discrete cosine transform [5], and the multiresolution simultaneous auto-regressive model [44] have the advantage of being insensitive to noise. Therefore, these transforms have widely been used to represent image textures. A new multiresolution method, discrete curvelet transform, has been developed by Candès and Donoho [12], which is effective in representing the edges in an image. Discrete curvelet transform has been successfully used to denoise images [19]. Due to the advantages of spectral domain texture features over spatial methods, they have been widely used in content based image retrieval.

In the next section, we describe and discuss the most popular spectral domain approaches of texture features extraction. This is to highlight the evolution of these approaches in texture features extraction.

2.3 Spectral Features

In the following sections, we describe some spectral approaches which have already been used in texture feature extraction for content based image retrieval. A human can distinguish two different images at a glance but when a machine tries to perform the same job, a lot of image discriminatory information needs to be pre-processed and stored to make the system automated. The prominent texture characteristics of an image can be found with the mathematical transforms used in various spectral approaches. Now we discuss the effectiveness of several well known transforms in representing image texture features and their acceptance in CBIR techniques.

2.3.1 Fourier Transform

The purpose of Fourier transform (FT) is to convert a time-domain signal into the frequency-domain. FT uses Fourier analysis to measure the frequency components of the signal. The discrete Fourier transform for a 2-D image $f(x, y)$ can be represented as [45]:

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M + vy/N)} \tag{2.3}$$

where $x = 0,1,2,...,M-1$ and $y = 0,1,2,...,N-1$ denote an $M \times N$ image.

Fourier transform provides the pattern information of an image that is collected from its frequency components as shown in Fig. 2.3.

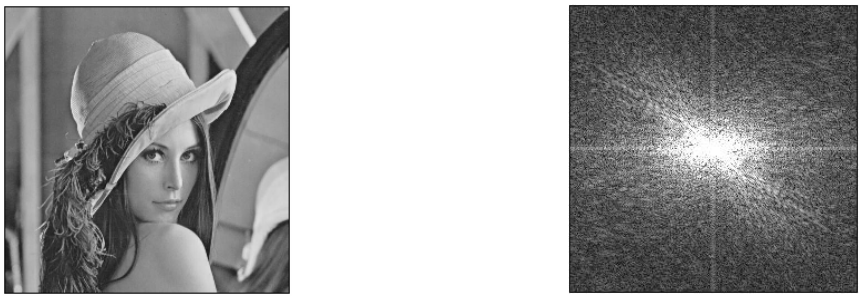


Fig. 2.3: Original Lena image (512×512) (left) and its Fourier transform (right).

The frequency components at specific locations of an image are used to represent the texture features of that image. Texture features computed from high frequency components are the main distinguishing factors between images which are used in CBIR. Therefore, frequency information at specific locations is required to distinguish images in CBIR. However, the disadvantage of Fourier transform is it captures only the global spectral features but does not provide any information about the exact location of these features [46].

Very often, Fourier transform in 2-D space fails to provide texture pattern discrimination information properly. Two completely different images may have similar patterns in their Fourier domain (shown in Fig. 2.4). The original images in Fig. 2.4 (a) and (b) look different, however, their Fourier spectra have similar patterns. Based on this spectral pattern, these images may be considered as similar in a CBIR process but they can easily be differentiated by human perception.

(a) (b)

Fig. 2.4: Both the images (a) and (b) contains original image at the left and its Fourier transform at the right.

Therefore, Fourier transform is functional only when spectral features of the signal are taken into consideration but not their exact location of occurrence. It has been used to extract the shape features for the purpose of image retrieval [41]. However, in using texture for content based image retrieval, the frequency components and the exact location of the different frequencies are equally important in distinguishing images. Thus, Fourier transform is not applicable directly in texture based CBIR.

Discrete cosine transform (DCT) is similar to the discrete Fourier transform, it transforms a signal or image in the spatial domain to the frequency domain and obtains DCT coefficients which can be used in various image processing purpose. Given an image $f(x,y)$ of size $n \times n$, its 2-D discrete cosine transform can be defined as [5]:

$$C(u,v)=\alpha(u,v)\sum_{x=0}^{n-1}\sum_{y=0}^{n-1} f(x,y)\cos[\frac{(2x+1)u\pi}{2n}]\cos[\frac{(2y+1)v\pi}{2n}] \quad (2.4)$$

where $u,v = 0,1,2,\ldots\ldots,n\text{-}1$.

Discrete cosine transform (DCT) has been adopted as an effective technique for image and video compression. It possesses the property of preserving most of the image energy in the low-frequency DCT coefficients which makes it so popular for data compression [47]. Different approaches for shape, texture, and color feature extraction and indexing using DCT can be found in [48]. Similar to FT, DCT texture features can only capture global features while ignoring local details. Therefore, it is not suitable for CBIR.

2.3.2 Short Time Fourier Transform

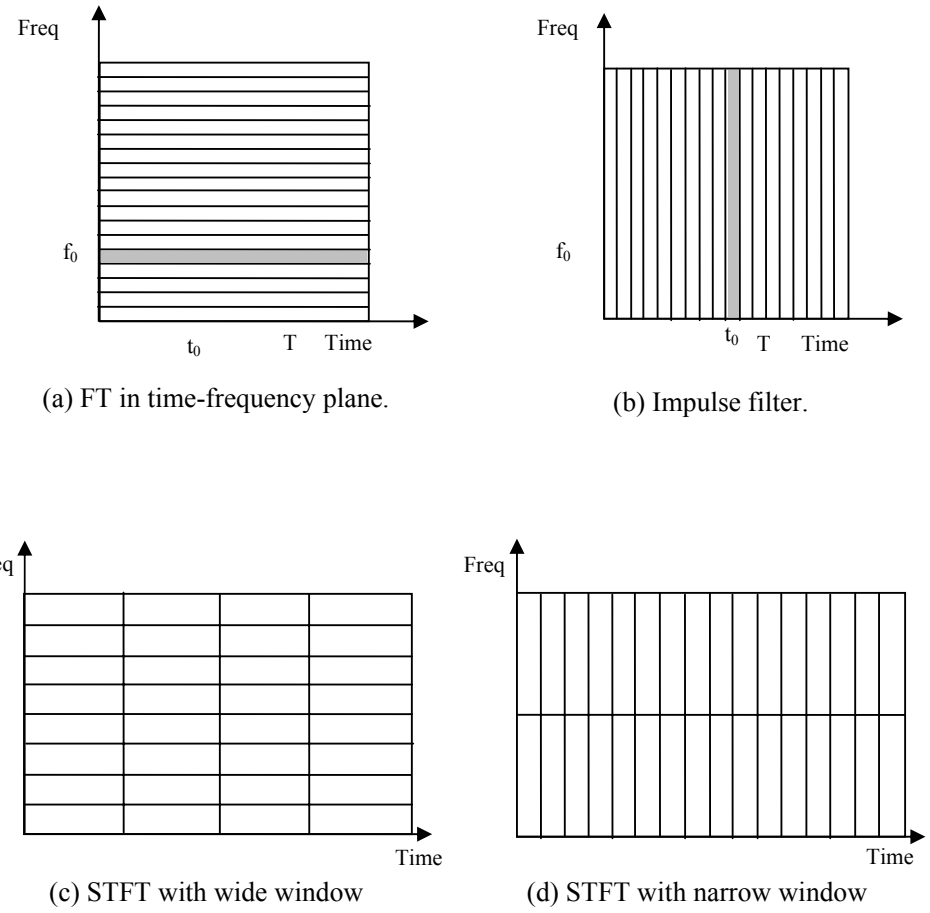
Short time Fourier transform (STFT) provides a time-frequency representation that is not possible in FT. In STFT, a window function is chosen in such a way that the portion of a

non-stationary signal which is covered by the window function seems stationary. This window function is then convolved with the original signal so that the signal or data part covered by the window is selected only. FT is then applied to the newly generated stationary signal [49]. The window is then moved to the next slot of signal and the previously mentioned steps are applied repeatedly until the whole signal is completely analyzed. This time-frequency representation is necessary for CBIR. For an image $f(x, y)$, its STFT can be defined as [50]:

$$X(\tau_1, \tau_2, \omega_1, \omega_2) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) W^*(x - \tau_1, y - \tau_2) e^{-j(\omega_1 x + \omega_2 y)} dx dy \tag{2.5}$$

where $W(x, y)$ is the selected 2-D window to convolve the 2-D signal, τ_1, τ_2 represent the spatial positions of the window and ω_1, ω_2 represents the spatial frequency parameters.

Compared to FT, STFT can show where the frequency components are. Fig. 2.5 illustrates this. Impulse filter and FT can capture location information either in spatial or in spectral domain. FT coefficients provide the information about which frequency components exists in the signal but it gives no time information in the time domain (Fig. 2.5 (a)). While using impulse filter (the narrowest window analysis), it can only capture location in time domain (Fig. 2.5(b)). STFT mainly shows the presence of some specific frequency components in a range of location but not the existence of frequency components at any specific location. The main drawback with this approach is the difficulty in determining the size of the window. The frequency representation in the transformed domain is controlled by the width of the chosen window. When the window is wide, it provides a better frequency resolution (close frequency components can be separated) but a poor time resolution (Fig. 2.5 (c)), and vice versa when the window is narrow (Fig. 2.5 (d)) [51]. For this reason, a window size may be appropriate for one image but not for another. Therefore, STFT is not an ideal tool to extract texture features from images as the exact locations of spectral components cannot be obtained with it.



(a) FT in time-frequency plane.

(b) Impulse filter.

(c) STFT with wide window

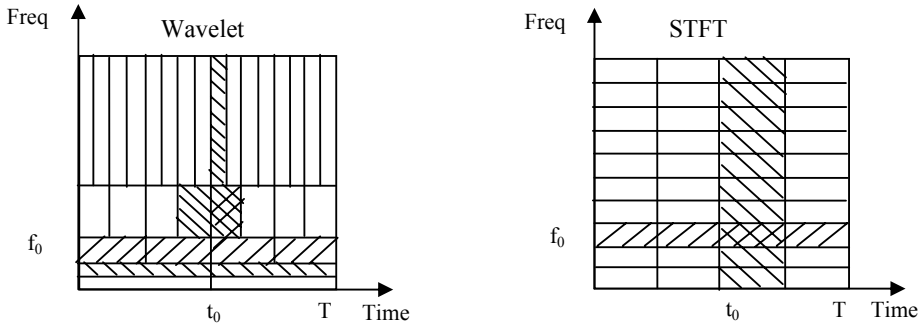
(d) STFT with narrow window

Fig. 2.5: Time-frequency illustration for FT and STFT
(reproduced from [52] and [51] respectively).

2.3.3 Mutiresolution Texture Feature Extraction

Multiresolution methods or hierarchical approaches attempt to find a specific frequency at a specific location, which is the main shortcoming of FT and STFT. However, it is not possible to find a specific frequency at a specific location simultaneously. Therefore, as a trade off between time-frequency representations, multiresolution methods are created. Multiresolution methods are designed to obtain a good time resolution but less accurate frequency resolution at high frequencies and a good frequency resolution but less accurate time resolution at low frequencies (Fig. 2.6 (a)) [53]. This approach is useful when the signal contains high frequency components for short durations and low frequency components for long duration [53]. Usually, 2-D images follow this frequency pattern. This effectively overcomes the window size problem of STFT (Fig. 2.6(b)).

Therefore, multiresolution approaches are more effective in image analysis and they overcome the limitations of frequency and location resolutions found in FT and STFT.



(a) (b)

Fig. 2.6: Time-frequency tiling using wavelet and STFT (reproduced from [52]).

A set of coefficients are obtained from a multiresolution transform. These coefficients corresponds to the frequency information at a different resolution, location [54] and sometimes the orientation of the image. In multiresolution approaches, such as discrete wavelet, Gabor filters and discrete curvelet transforms, the frequency information at different scales, orientations and locations are obtained. The multiresolution method is similar to image zooming process. When the image is zoomed out, we get a global view of the image. Whereas, a detailed view of the image is obtained when it is zoomed in. Using the multiresolution approach, we can get a complete picture of the image consisting of its low frequency components [53]. Meanwhile, high frequency components of the image at low scales provide the detailed and discriminatory structures of the image, which is important when using content based image retrieval based on texture features.

In the following, we describe the two major multiresolution approaches from the literature, namely, Gabor filters transform and the discrete wavelet transform.

2.3.3.1 Gabor Filters Transform

Gabor filters transform is a good multiresolution approach that represents the edges of image in an effective way using multiple orientations and scales. Gabor filters have a spatial property that is similar to mammalian perceptual vision, thereby providing researchers a good opportunity to use it in image processing. Gabor filters transform creates a filter bank consisting of Gabor filters with various scales and orientations. Then

the filters are convolved with the image. The Gabor filters transform can be represented as [7]:

$$G_{m,n}(x,y) = \sum_s \sum_t f(x_1, y_1) g_{m,n} * (x - x_1, y - y_1) \tag{2.6}$$

where s and t are the filter mask size variables, $x_1 = x - s$ and $y_1 = y - t$. m and n represent the scale and orientation of a Gabor wavelet, respectively. The mother Gabor wavelet $g(x,y)$ and its Fourier transform $G(u,v)$ are defined as [7]:

$$g(x,y) = (\frac{1}{2\pi\sigma_x\sigma_y}) \exp[-\frac{1}{2}(\frac{x^2}{\sigma_x{}^2} + \frac{y^2}{\sigma_y{}^2}) + 2\pi jWx], \tag{2.7}$$

$$G(u,v) = \exp\{-\frac{1}{2}[\frac{(u-W)^2}{\sigma_u{}^2} + \frac{v^2}{\sigma_v{}^2}]\} \tag{2.8}$$

where $\sigma_u = 1/2\pi\sigma_x$, $\sigma_v = 1/2\pi\sigma_y$ and W is the modulation frequency. Similar Gabor filters are then obtained from the dilation and rotation of the mother Gabor wavelet [7]:

$$\begin{aligned} &g_{m,n}(x,y) = a^{-m} G(x', y'), \\ &x' = a^{-m}(x\cos\theta + y\sin\theta) \text{ and } y' = a^{-m}(-x\sin\theta + y\cos\theta) \end{aligned} \tag{2.9}$$

where $a > 1, \theta = n\pi / K$, K is the total number of orientations. The values of the parameters a, σ_u and σ_v are given as [7]:

$$\begin{aligned} &a = (U_h / U_l)^{-\frac{1}{S-1}}, \\ &\sigma_u = \frac{(a-1)U_h}{(a+1)\sqrt{2\ln 2}}, \\ &\sigma_v = \tan(\frac{\pi}{2k})[U_h - 2\ln(\frac{\sigma_u^2}{U_h})][2\ln 2 - \frac{(2\ln 2)^2\sigma_u^2}{U_h{}^2}]. \end{aligned} \tag{2.10}$$

Here S is the number of scales, $W = U_h$ and U_h and U_l specify the upper and lower centre of frequencies. Gabor filtered images for the Lena picture are shown in Fig. 2.7.

This is generated using 5 levels and 6 orientations in 'Simplegabor - Multiresolution Gabor Feature Toolbox' version-1.0.0 of [55].

Gabor filters use multiple window size at different level of resolutions whereas STFT uses only one window. STFT obtains either a good frequency or good location resolution information but Gabor filters obtain both by applying multiresolution filter banks.

From a study of recent literature, we find that Gabor filters have been widely used in texture representation. Manjunath and Ma [7] presented a comparison of the performance of image retrieval with Gabor filters, TWT, PWT and MR-SAR model where Gabor filters texture features are found to be the most promising and robust. Zhang et al. [15] proposed a computationally efficient Gabor filters texture features representation that overcomes the problem of retrieving similar but rotated images from an image database. Lai et al. [56] employed Gabor filters for scaled and rotated natural image retrieval where they find it performing better than scale and rotation non-normalized Gabor filters. The Gabor filters have been popularly used in texture classification and segmentation [43], finger print analysis [57] and medical image classification [36, 37]. Image retrieval performance of several different spectral approaches have been shown in [28] where Gabor filters outperform other wavelet transforms, including the orthogonal wavelet (OWT), bi-orthogonal wavelet (BWT) and TWT at the cost of high computational expense. Gabor filters texture features have been found useful in CBIR of biomedical images like cervicographic images for cancer detection [36, 37]. It is effective at distinguishing strongly ordered textures whereas MR-SAR texture features are useful for retrieving only weakly ordered or random textures. Therefore, from the recent literature, the Gabor filters seem to be the most promising for texture representation as well as image retrieval based on texture features.

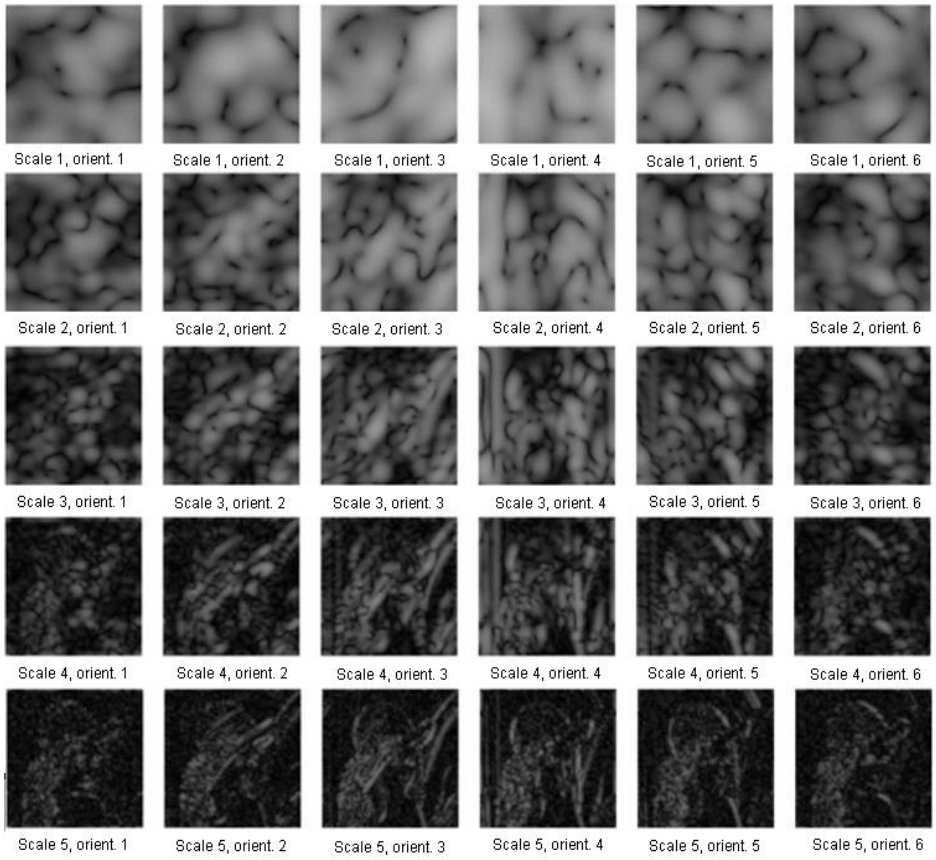


Fig. 2.7: Gabor filtered subbands for a 512×512 Lena image.

Though Gabor filters texture features have been found to be the best over all, they have some limitations as they use wavelets as a filter bank. Wavelets are not so effective in representing edge discontinuities in images [12, 13]. Moreover, Gabor spectral cover is not complete, to avoid overlap in the spectral domain, Gabor filters only use half peak-magnitudes in the frequency domain. As a consequence, information loss results in the spectral domain (Fig. 2.8). Gabor filters tiling of frequency ranges between $U_l = 0.05$ and $U_h = 0.4$ is shown in the Fig. 2.8 where the half peak-magnitude filters only touch each other. Consequently, the high frequency components, which are considered to be the most important in characterizing image textures, are not effectively captured. In addition, lower scales Gabor spectra capture little texture information, rather they only provide the energy information of the global image (Fig. 2.7). Redundant information also exists in transformed images as Gabor filters do not involve image down sampling. These limitations need to be improved in texture based CBIR.

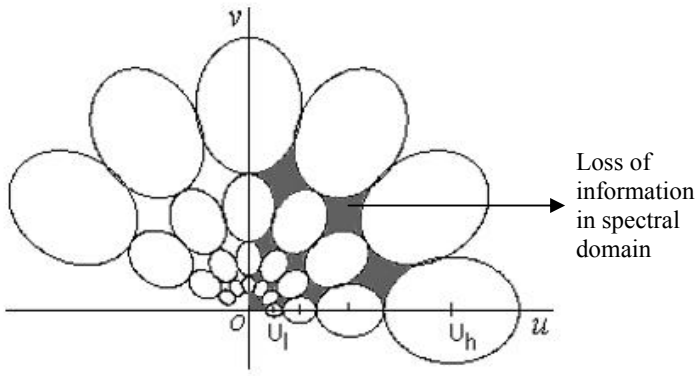


Fig. 2.8: Frequency tiling of half frequency plan by Gabor filters, the ovals are the covered spectrum [7]. U_h and U_l specify the upper and lower centre of frequencies.

Discrete wavelet transforms use down sampling of images when extracts texture features. Next we describe how this transform addresses the problems found in FT and STFT.

2.3.3.2 Discrete Wavelet Transform

Wavelet transform is introduced with the advancement in multiresolution transform research. Discrete wavelet transform is one of the most promising multiresolution approaches used in CBIR. It has the advantage of a time-frequency representation of signals where Fourier transform is only frequency localized. The location, at which a frequency component of an image exists, is important as it draws the discrimination line between images. Given an image $f(x,y)$, its continuous wavelet transform is given by [58]:

$$WT_{\psi}(a_1,a_2,b_1,b_2)=\int_R\int_R f(x,y)\overline{\psi_{a_1,a_2,b_1,b_2}(x,y)}dxdy \tag{2.11}$$

where a wavelet with scale parameter a_1, a_2 and position parameter b_1, b_2 can be described as follows:

$$\psi_{a_1,a_2,b_1,b_2}(x,y)=a_1^{-\frac{1}{2}}a_2^{-\frac{1}{2}}\psi(\frac{x-b_1}{a_1})\psi(\frac{y-b_2}{a_2}) \tag{2.12}$$

Unlike the FT and STFT, the window size varies at each resolution level when the wavelet transform is applied to an image. In discrete wavelet transform, the original image is high-pass filtered yielding three detail images, describing the local changes in horizontal, vertical and diagonal direction of the original image. The image is then low-pass filtered yielding an approximation image which is again filtered in the same manner to generate high and low frequency subbands at the next lower resolution level (Fig. 2.9). This process is continued until the whole image is processed or a level is determined as the lowest to stop decomposition. This continuing decomposition process is known as down sampling and shown in Fig. 2.9.

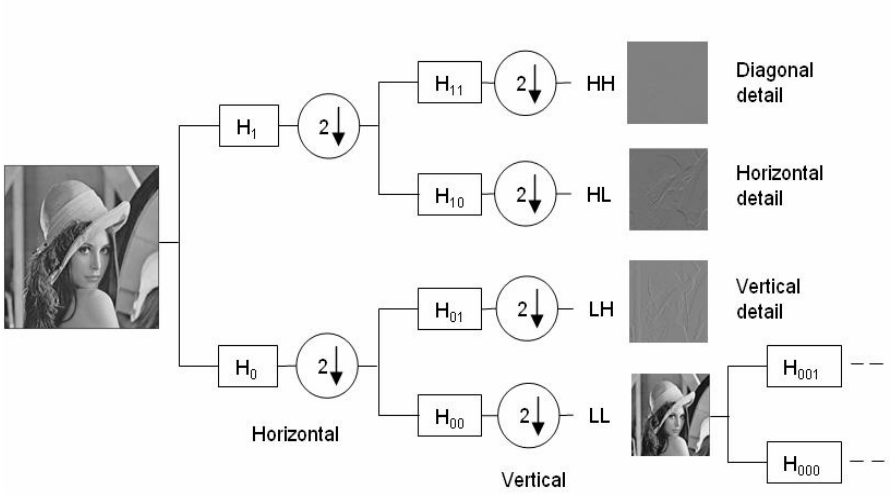


Fig. 2.9: DWT decomposition tree (reproduced from [52]).

The whole decomposition process provides us with an array of DWT coefficients obtained from each subbands at each scale. These coefficients can then be used to analyze the texture patterns of an image. Wavelet subbands obtained from the Lena image using 4 decomposition levels are shown in Fig. 2.10.

Due to its good image texture representation capability, wavelet transform has been used in many image processing applications, e.g., texture analysis, CBIR methods [10, 11, 59-61], texture classification [59] and image deconvolution [62]. The wavelet transform in [59] yielded a much higher texture classification rate and retrieval accuracy than discrete cosine transform or spatial partitioning. Furthermore, Ma et al. applied and compared the retrieval performance of different wavelet and Gabor filters texture representations in [28] where Gabor filters perform better than other wavelet methods. Yuan et al. proposed a mixed Gaussian statistical model in [63] to represent the wavelet features and implemented it in Brodatz texture retrieval. This model has been introduced

to extract texture features from each subband at each scale of the wavelet spectral domain. The outcome of this texture representation has been compared with other representations, including PWT, TWT, and the Gabor filters. From this comparison, it is found that discrete wavelet feature extraction time and descriptor size are less than that of Gabor filters but the retrieval result are not as good as Gabor filters. A weighted standard deviation wavelet texture descriptor has been presented in [10]. This texture descriptor has been combined with color features to retrieve color images. For texture image retrieval, Sitaram and Kapil [10] applied discrete wavelet transform to all the database images including the query image. Then a $3l+2$ dimensional features vector is generated from each image where l is the number of levels of wavelet decomposition. Higher weights are assigned to lower scale subbands under the assumption that high frequency components at low levels are expected to contain more texture discriminating information. This weighted texture features vector scheme has outperformed the retrieval using TWT and PWT scheme. During the same experiment, it is discovered that the image retrieval rate becomes slightly higher (70.30% with 5 levels and 70.24% with 4 levels of decomposition) when more decomposition levels are used in Haar wavelet transform. N. Suematsu et al. proposed a wavelet based texture representation for the region of interest in an image [11]. Instead of the whole image, their approach has concentrated on a specific region.

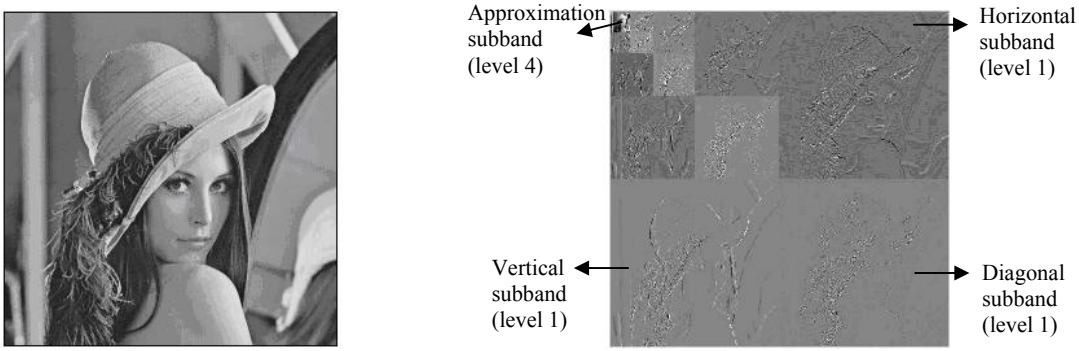


Fig. 2.10: A 512×512 Lena image (left) and its DWT transform (right). Wavelet subbands generated by horizontal, vertical, and diagonal wavelets are shown at all frequency level. The approximation image is shown at the lowest frequency level.

Though wavelet transform has been widely accepted, it has several problems which results in a poor outcome for content based image retrieval. In 2-D space, wavelets can not capture highly anisotropic elements like the curves of an image effectively as wavelets are not effective at representing line singularities. Images with a dense

composition of highly anisotropic elements such as curves may not be well represented using wavelet texture representation [12, 13]. Besides, discrete wavelet transform only uses 3 directional wavelets; horizontal, vertical and diagonal to capture the image texture information. Images containing a high level of directionality will not be well represented by wavelet spectral domain. Because of the above mentioned flaws of discrete wavelet transform, researchers have been trying to introduce spectral approaches which involve more directional information in an image for texture representation. Discrete curvelet transform is the result of this endeavor.

Discrete curvelet transform consists of more scales and orientations in the frequency domain than the Gabor filters and completely covers the spectral plane. The Gabor filters transform has less number of orientations at every scale whereas, in the curvelet transform, the number of orientations increases as the level of resolution increases so that more directional information from high frequency components can be captured. We describe discrete curvelet transform concept and its applications in the next chapter.

2.4 Summary

In this chapter, we discussed the key issues involved in content based image retrieval, such as, image database selection, low-level features, i.e., color, texture, shape, spatial location representation, image similarity measurement methods, performance evaluation and the spectral approaches used in texture based image retrieval. From the recent literature, we find the texture features of an image are effective due to their fine discriminatory property. We also find that a combination of image features gives a boost to the outcome of CBIR. To emphasis our discussion on texture, we discuss the spatial and the spectral approaches of texture features analysis in this chapter and find that the spectral texture feature representation is superior to the spatial approaches. Therefore, we find spectral texture features are more suitable for content based image retrieval.

In addition to the discussion on the main factors of CBIR, we made an effort to provide a clear overview of the spectral texture representations used in recent literature. These texture representations are presented with their corresponding advantages and disadvantages. In parallel, we described the application and performance of these spectral approaches in various image retrieval and texture representations. From this discussion, multiresolution approaches are found to be the most effective in texture features representation. We also find that the limitations of Gabor filters and wavelet transform leaves room for improvement in texture based image retrieval.

Curvelet Transform

3.1 Introduction

Curvelet transform has been developed to overcome the limitations of wavelet and Gabor filters. Though wavelet transform has been explored widely in various branches of image processing, it fails to represent objects containing randomly oriented edges and curves as it is not good at representing line singularities. Gabor filters are found to perform better than wavelet transform in representing textures and retrieving images due to its multiple orientation approach. However, due to the loss of spectral information in Gabor filters they cannot effectively represent images. This affects the CBIR performance. Consequently, a more robust mechanism is necessary to improve CBIR performance. To achieve a complete coverage of the spectral domain and to capture more orientation information, curvelet transform has been developed.

The initial approach of curvelet transform implements the concept of discrete ridgelet transform [64]. Since its creation in 1999 [12], ridgelet based curvelet transform has been successfully used as an effective tool in image denoising [19], image decomposition [61], texture classification [65], image deconvolution [62], astronomical imaging [66] and contrast enhancement [67], etc. But ridgelet based curvelet transform is not efficient as it uses complex ridgelet transform [17]. In 2005, Candès et al. proposed two new forms of curvelet transform based on different operations of Fourier samples [18], namely,

unequally-spaced fast Fourier transform (USFFT) and wrapping based fast curvelet transform. Wrapping based curvelet transform is faster in computation time and more robust than ridgelet and USFFT based curvelet transform [17]. To our knowledge, wrapping based curvelet transform has not been used in CBIR and there is no work on a systematic evaluation of curvelet in CBIR.

In the following section, we first describe the curvelet transform approaches and their advantages in texture representation over other spectral approaches. Then, we provide a brief description of the related works already done using curvelet transform.

3.2 Discrete Curvelet Transform

Basically, curvelet transform extends the ridgelet transform to multiple scale analysis. Therefore, we start from the definition of ridgelet transform. Given an image $f(x, y)$, the continuous ridgelet coefficients are expressed as [19]:

$$\Re_f(a,b,\theta) = \iint \psi_{a,b,\theta}(x,y) f(x,y) dxdy. \tag{3.1}$$

Here, a is the scale parameter where $a > 0$, $b \in R$ is the translation parameter and $\theta \in [0, 2\pi)$ is the orientation parameter. Exact reconstruction is possible from these coefficients. A ridgelet can be defined as [19]:

$$\psi_{a,b,\theta}(x,y) = a^{-\frac{1}{2}} \psi\left(\frac{x\cos\theta + y\sin\theta - b}{a}\right) \tag{3.2}$$

where θ is the orientation of the ridgelet. Ridgelets are constant along the lines $x\cos\theta + y\sin\theta = const$ and transverse to these ridges are wavelets [19]. If we compare "equation 2.12" with "equation 3.2", we find that the point parameters of wavelet (b_1, b_2) are replaced by line and orientation parameters (b, θ) in the case of a ridgelet. This means that ridgelets can be tuned to different orientations and different scales to create the curvelets (Fig.3.1). Ridgelets take the form of a basis element and obtain a high anisotropy. Therefore, it captures the edge information more effectively. A ridgelet is linear in its edge direction and is much sharper than a conventional sinusoidal wavelet [21, 58] (Fig. 3.1).

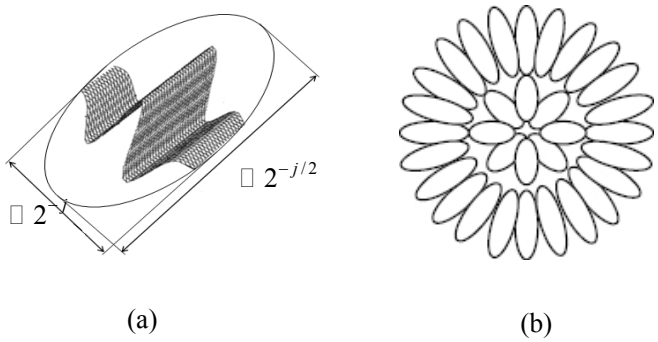


Fig. 3.1: (a) A single curvelet with width ≈ 2^{-j} and length ≈ $2^{-j/2}$ and (b) several curvelets tuned to 2 scales and different orientations (right).

The contrast between wavelet and ridgelet on capturing edge information is shown in Fig. 3.2. It can be observed that the curvelets, at all scales, capture the edge information more accurately and tightly than wavelets.

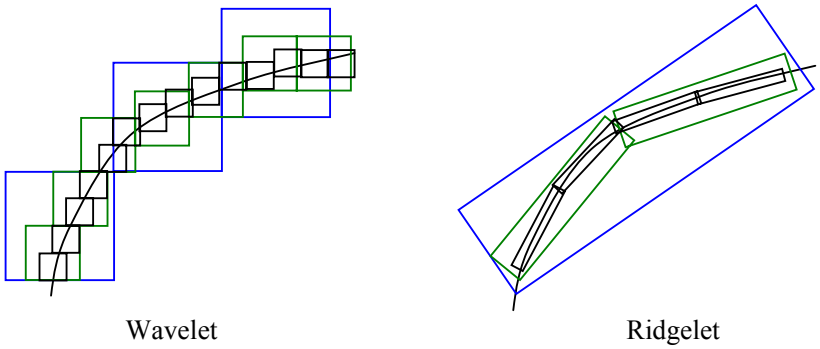


Fig. 3.2: Edge representation using wavelet and ridgelet (reproduced from [68]).

The ridgelet based curvelet transform is a combination of the à trous wavelet transform and the Radon transform. In this curvelet approach, input image is first decomposed into a set of subbands each of which is then partitioned into several blocks for ridgelet analysis. The ridgelet transform is implemented using the Radon transform and the 1-D wavelet transform [19]. During the ridgelet transform, one of the processes is the spatial partitioning which involves overlapping of windows to avoid blocking effects. It results in a large amount of redundancy. Moreover, this process is very time consuming, which makes it less feasible for texture features analysis in a large database [17].

Fast discrete curvelet transform based on the wrapping of Fourier samples has less computational complexity as it uses fast Fourier transform instead of complex ridgelet transform. In this approach, a tight frame has been introduced as the curvelet support to reduce the data redundancy in the frequency domain [17]. Normally, rideglets have a fixed length that is equal to the image size and a variable width, whereas curvelets have both variable width and length and represent more anisotropy. Therefore, the wrapping based curvelet transform is simpler, less redundant and faster in computation [17] than ridgelet based curvelet transform. We now discuss discrete curvelet transform based on wrapping Fourier samples [18]. As it is the most promising approach of curvelet so far, we intend to use it for texture representation in our CBIR research.

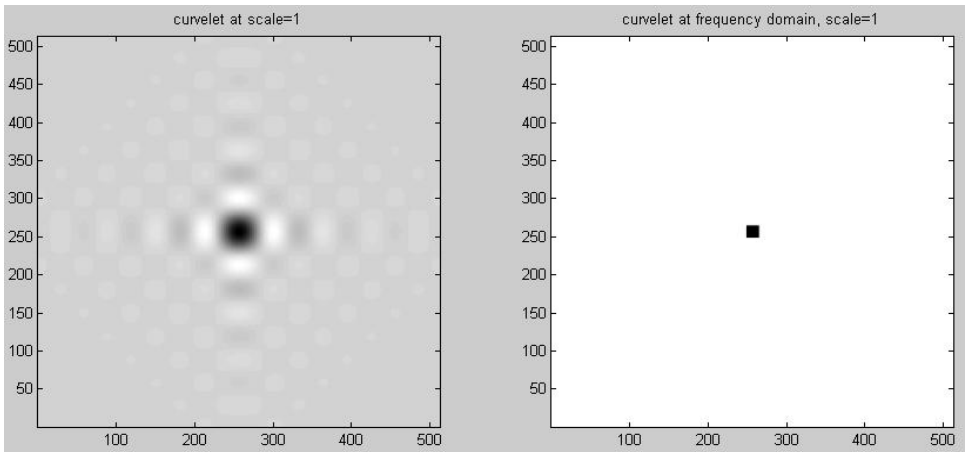
Curvelet transform based on wrapping of Fourier samples takes a 2-D image as input in the form of a Cartesian array $f[m,n]$ such that $0 \le m < M, 0 \le n < N$ and generates a number of curvelet coefficients indexed by a scale j, an orientation l and two spatial location parameters (k_1, k_2) as output. To form the curvelet texture descriptor, statistical operations are applied to these coefficients. Discrete curvelet coefficients can be defined by [18]:

$$C^D(j,l,k_1,k_2) = \sum_{\substack{0 \le m < M \\ 0 \le n < N}} f[m,n]\, \varphi^D_{j,l,k_1,k_2}[m,n]. \tag{3.3}$$

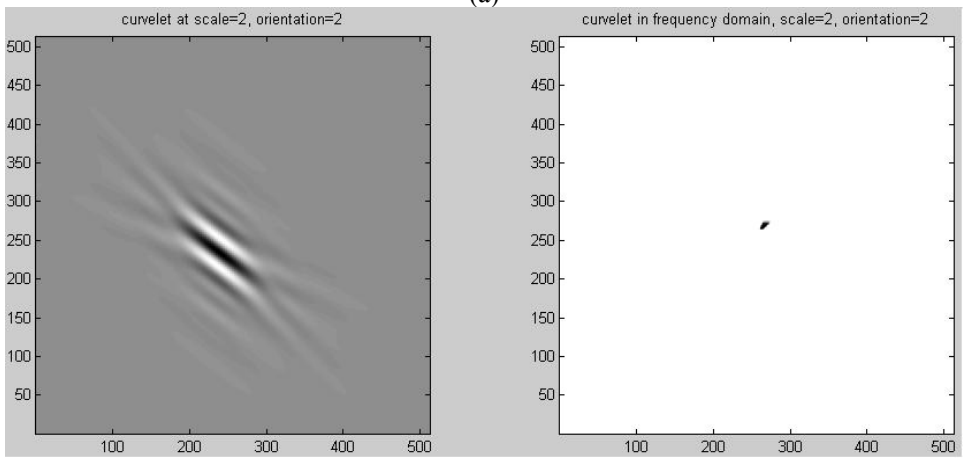
Here, each $\varphi^D_{j,l,k_1,k_2}[m,n]$ is a digital curvelet waveform. This curvelet approach implements the effective parabolic scaling law on the subbands in the frequency domain to capture curved edges within an image more effectively. Curvelets exhibit an oscillating behavior in the direction perpendicular to their orientation in frequency domain.

Basically, wrapping based curvelet transform is a multiscale transform with a pyramid structure consisting of many orientations at each scale. This pyramid structure consists of several subbands at different scales in the frequency domain. Subbands at high and low frequency levels have different orientations and positions. At high scales, the curvelet waveform becomes so fine that it looks like a needle shaped element (left images of Fig. 3.3 (e), (f)). Whereas, the curvelet is non directional at the coarsest scale (left image of Fig. 3.3(a)) [18]. With increase in the resolution level the curvelet becomes finer and smaller in the spatial domain and shows more sensitivity to curved edges which enables it to effectively capture the curves in an image (Fig. 3.3). As a consequence, curved singularities can be well-approximated with few coefficients. High frequency

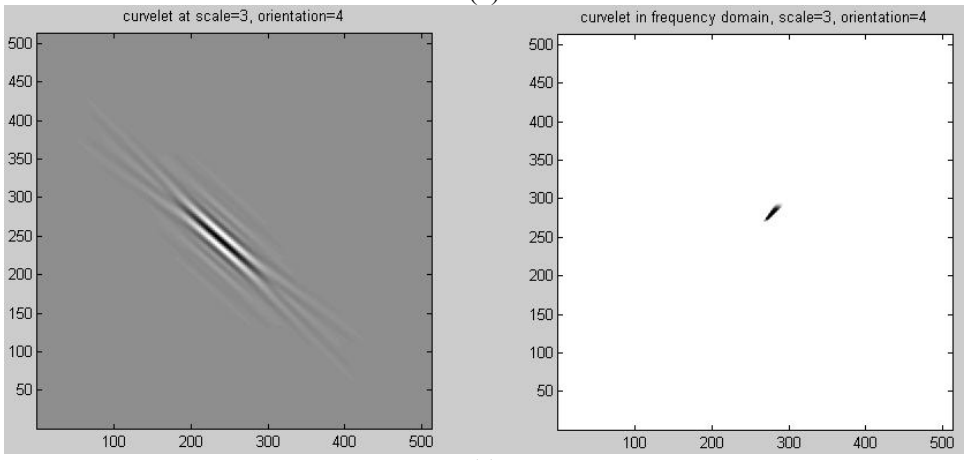
components of an image play a vital role in finding distinction between images. Curvelets at fine scales effectively represent edges by using texture features computed from the curvelet coefficients. Curvelets at different scales and their frequency responses are shown in Fig. 3.3. These are generated using Curvelab-2.1.2 of [69].



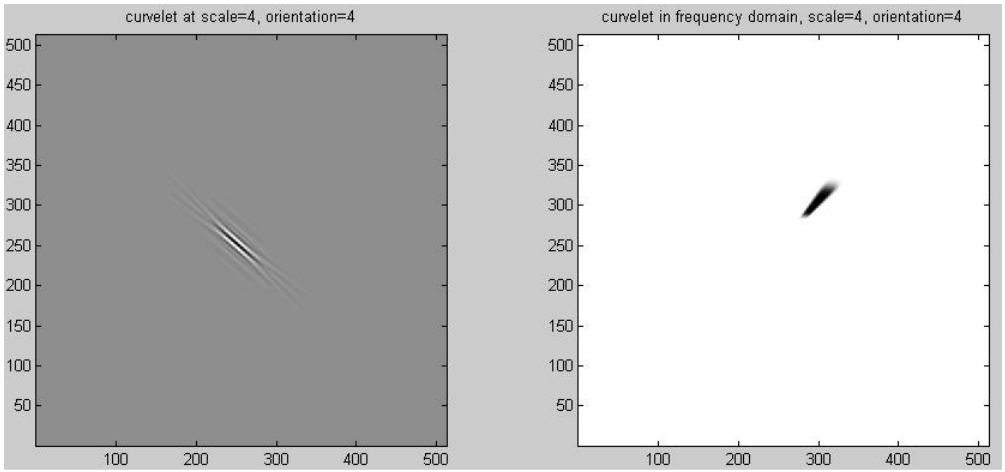
(a)



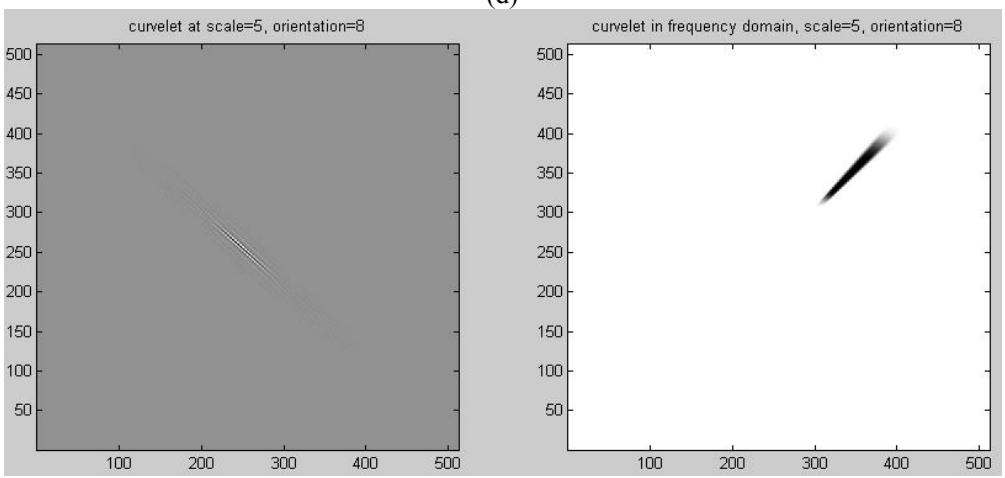
(b)



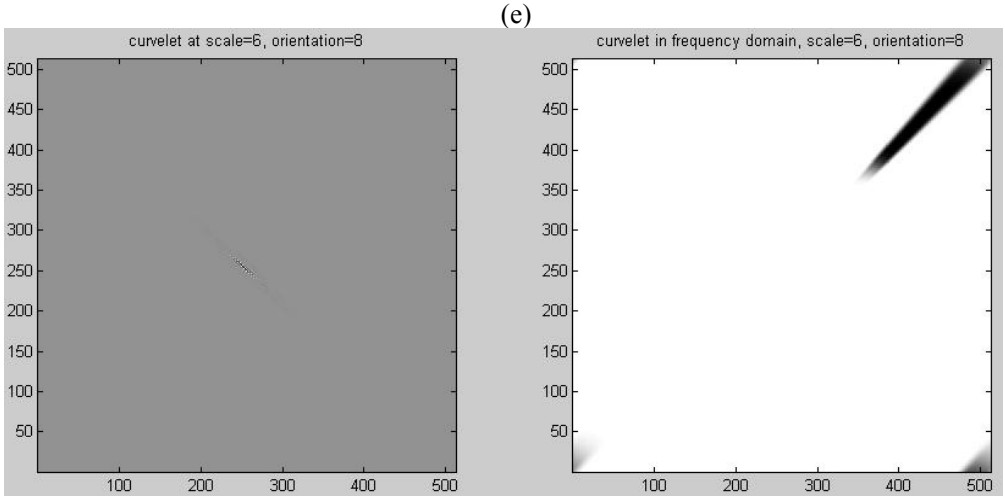
(c)



(d)



(e)



(f)

Fig. 3.3: Curvelets (absolute value) at scales different scales at a single direction are shown in the spatial domain (left) and in the frequency domain (right).

If we combine the frequency responses of curvelets at different scales and orientations, we get a rectangular frequency tiling that covers the whole image in the spectral domain (Fig. 3.4). Thus, the curvelet spectra completely cover the frequency plane and there is no loss of spectral information like the Gabor filters.

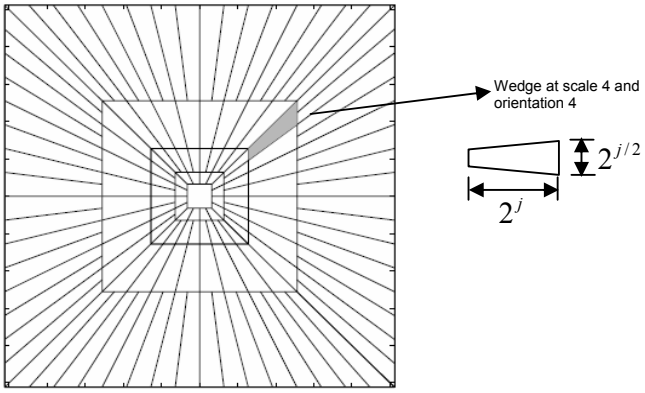


Fig. 3.4: Rectangular frequency tiling of an image with 5 level curvelets.

To achieve higher level of efficiency, curvelet transform is usually implemented in the frequency domain. That is, both the curvelet and the image are transformed and are then multiplied in the Fourier frequency domain. The product is then inverse Fourier transformed to obtain the curvelet coefficients. The process can be described as $\text{Curvelet transform} = \text{IFFT}\,[\,\text{FFT(Curvelet)} \times \text{FFT(Image)}]$ and the product from the multiplication is a wedge.

The trapezoidal wedge in the spectral domain is not suitable for use with the inverse Fourier transform which is the next step in collecting the curvelet coefficients using IFFT. The wedge data cannot be accommodated directly into a rectangle of size $2^{j} \times 2^{j/2}$. To overcome this problem, Candes et al. have formulated a wedge wrapping procedure [18] where a parallelogram with sides 2^{j} and $2^{j/2}$ is chosen as a support to the wedge data. The wrapping is done by periodic tiling of the spectrum inside the wedge and then collecting the rectangular coefficient area in the center. The center rectangle of size $2^{j} \times 2^{j/2}$ successfully collects all the information in that parallelogram (Fig. 3.5).

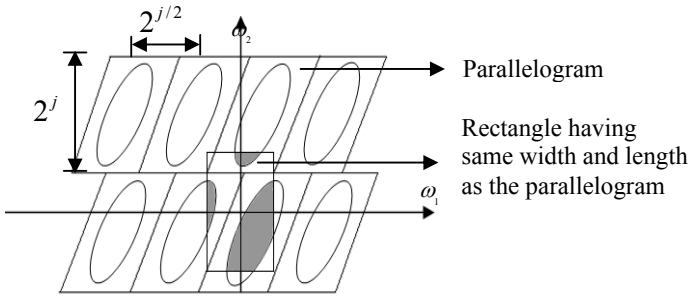


Fig. 3.5: Wrapping wedge around the origin by periodic tiling of the wedge data. The angle θ is in the range $(\pi/4, 3\pi/4)$.

Thus we obtain the discrete curvelet coefficients by applying 2-D inverse Fourier transform to this wrapped wedge data. Fadili et al. [17] have shown that wrapping based fast discrete curvelet transform is much more efficient and provides better transform result than ridgelet based curvelet transform.

The curvelet properties and advantages have been discussed in detail above. Principles of wrapping based discrete curvelet transform have also been described. Next, we describe several related works using different approaches of discrete curvelet transform.

3.3 Related Works on Curvelet Transform

Majumder has described a method to automate Bangla basic character recognition using ridgelet based curvelet transform [20]. There are fifty characters in Bangla language and all the existing Bangla fonts use all these characters. Majumder has changed each character morphologically by thinning and thickening twice the original characters for his experiment. In training phase, the curvelet coefficients have been extracted from all these characters to generate texture features descriptors and 5 sets of classifiers have been created from each character. The characters are altered to capture the changes in characters of different fonts by slightly varying their edge positions. Curvelet texture features of the query character are then compared with the training sets to find the same characters. He has done the experiment on only twenty well known Bangla fonts. Therefore, there is no guarantee that this application will recognize all characters in complex format as well. Feature descriptor size and its computation time are not mentioned in this paper. Therefore, it is not possible to measure how efficient the system is. The outcome is not compared to any well known character recognition method.

Joutel et al. have created a convenient assistance tool for the identification of ancient handwritten manuscripts [21, 70] using ridgelet based curvelet transform. The curvature and orientation of handwritten scripts are the two main morphological shape properties used to generate discrete curvelet features. Joutel et al. have focused on characterizing handwritings and classifying them into visual writers' families. Problems regarding historical manuscript classification include a difficulty in segmenting lines, words, non-linear text size differences, irregular handwritten shapes, difficulty in the recognition of spaces or edges due to lack of pen pressure, unpredictable page layouts, etc. Moreover, backgrounds of many ancient documents have noisy texture patterns. Although the classification and writer recognition tests computed on two separate databases obtain a high level of accuracy, this approach has some shortcomings. One orientation representation and one curvature representation have been generated in this approach from each script, which is not enough to classify and characterize all ancient handwritten scripts. Texture patterns of image are not represented in this approach, so it will not be effective for natural image retrieval.

Ni et al. [71] have proposed an image retrieval method using discrete curvelet transform. From each ridgelet block, a pair of features has been extracted where the first one represents the edge strength and the other represents the angle difference between the significant edges. This method is not effective enough in representing images with complex contents and also has some notable problems. First of all, it is not clearly mentioned which coefficients corresponds to which scale in this paper. Second, no standard database has been chosen for retrieval purpose. Third, the size of the images and the number of levels of curvelet decomposition are not mentioned. Finally, the similarity measurement technique is not described and it is not compared with any other well established spectral approach of CBIR.

In [65], texture classification by statistical and co-occurrence features using discrete curvelet based on ridgelet is presented. In this work, texture classification has been shown on the basis of three different feature descriptors. The first consists of curvelet statistical features (CSF), i.e., mean and standard deviation. The second consists of curvelet co-occurrence features (CCF), i.e., contrast, cluster shade, cluster prominence, and local homogeneity. The last one involves the combination of the CSF and CCF descriptors. The authors use 2520 regions created by subdividing 30 texture images of size 512×512 from VisTex database. Curvelet features are found to obtain a higher degree of accuracy than wavelet features on the average. But the database they used has only small number of categories (30). Another shortcoming of this approach is the use of large feature descriptor. Therefore, these are not efficient features for CBIR.

Above, we have described several related works on curvelet transform, all of them use ridgelet based curvelet transform. So far, we find only one application of wrapping based curvelet transform in texture classification method for analyzing medical images gathered from computed tomography [72]. This method crops a 512×512 image into a 32×32 image so that it becomes a single human organ tissue image and extracts the texture features of that region. Then a texture classifier is used to distinguish the different types of tissue images. The main problems in this approach are the existence of large number of similar images in the database and small image dimension. Because the database has only human tissue images, therefore, it has less variation in its domain compared to the natural image databases. Tissue texture of same human organ is expected to have a negligible difference in such a small image (32×32), thereby making the classification method simple. Natural images are quite different in nature. Therefore, this process will not be effective for CBIR in a large database with large natural images.

3.4 Summary

In the beginning of this chapter, we have described and discussed the basics of curvelet transform. We described discrete curvelet transform based on ridgelet filtering. From the recent literature, we find ridgelet based curvelet transform has some drawbacks. Newer approaches of curvelet transform, USFFT based and wrapping based fast discrete curvelet transform have several advantages over ridgelets as well as the ridgelet based curvelet transform. Among these new approaches, wrapping based discrete curvelet transform provides additional features such as robustness, simplicity, and good edge capturing capability in image texture representation. We also described how this wrapping based curvelet transform works by providing details on curvelet structures in spatial and spectral domain and how these curvelets provide better texture discriminatory property in representing edges.

In the second part of this chapter, we have provided information on recent works related to texture representation and image classification using curvelet transform. From this discussion, we found curvelet transform based on ridgelet filtering has been used in most researches and those applications require improvement. To our knowledge, wrapping based fast discrete curvelet transform has not been widely used. Therefore, CBIR experiments using wrapping based curvelet transform needs more attention and investigation. From the literature, we find it to be the most effective approach in defining image discriminatory characteristics. Wrapping based curvelet transform is the best among all the curvelet approaches introduced. Therefore, we find it promising enough to

study its application in CBIR. In the next chapter, we describe the texture representation and the CBIR mechanism used in our texture retrieval system.

Feature Extraction Using Curvelet

4.1 Introduction

Texture features are important in content based image retrieval due to their ability to define the entire image characteristics effectively. We have already described the spectral approaches of texture features extraction in Chapter 2. Concepts of curvelet transform and curvelet structures at different resolutions have been described in Chapter 3. Since discrete curvelet transform has been found to represent the curved edges of images more effectively than wavelet and Gabor filters, it is expected to find the discriminatory texture patterns of an image as well. In this chapter, we describe image representation using wrapping based discrete curvelet texture features and the use of these features in CBIR process. For this purpose, first, we describe the general procedure of curvelet texture features descriptor generation from spectral domain coefficients. Second, we describe the general image indexing mechanism. Third, we provide the detail information on how the curvelet texture descriptors are used to index the images in the feature database we use. Finally, we provide the implementation of curvelet texture features in our CBIR research.

4.2 Curvelet Computation

Discrete curvelet transform is applied to an image to obtain its coefficients. These coefficients are then used to form the texture descriptor of that image. Recalling "Equation 3.3" of Chapter 3, curvelet coefficients of a 2-D Cartesian grid $f[m,n]$, $0 \le m < M, 0 \le n < N$ are expressed as:

$$C^D(j,l,k_1,k_2) = \sum_{\substack{0 \le m < M \\ 0 \le n < N}} f[m,n]\, \varphi^D_{j,l,k_1,k_2}[m,n]$$

where $\varphi^D_{j,l,k_1,k_2}[m,n]$ is the curvelet waveform. This transform generates an array of curvelet coefficients indexed by their scale j, orientation l and location parameters (k_1,k_2).

Discrete curvelet transform is implemented using the wrapping based fast discrete curvelet transform [18]. Basically, multiresolution discrete curvelet transform in the spectral domain utilizes the advantages of fast Fourier transform (FFT). During FFT, both the image and the curvelet at a given scale and orientation are transformed into the Fourier domain. The convolution of the curvelet with the image in the spatial domain then becomes their product in the Fourier domain. At the end of this computation process, we obtain a set of curvelet coefficients by applying inverse FFT to the spectral product. This set contains curvelet coefficients in ascending order of the scales and orientations. The complete feature extraction process using one single curvelet is illustrated in Fig. 4.1(a).

There is a problem in applying inverse FFT on the obtained frequency spectrum. The frequency response of a curvelet is a trapezoidal wedge which needs to be wrapped into a rectangular support to perform the inverse Fourier transform. The wrapping of this trapezoidal wedge is done by periodically tiling the spectrum inside the wedge and then collecting the rectangular coefficient area in the origin. Through this periodic tiling, the rectangular region collects the wedge's corresponding fragmented portions from the surrounding parallelograms. For this wedge wrapping process, this approach of curvelet transform is known as the 'wrapping based curvelet transform'. The wrapping is illustrated in Fig. 4.1(b) and explained as following. As shown in Fig. 4.1(b), in order to do IFFT on the FT wedge, the wedge has to be arranged as a rectangle. The idea is to replicate the wedge on a 2-D grid, so a rectangle in the center captures all the components a, b, and c of the wedge.

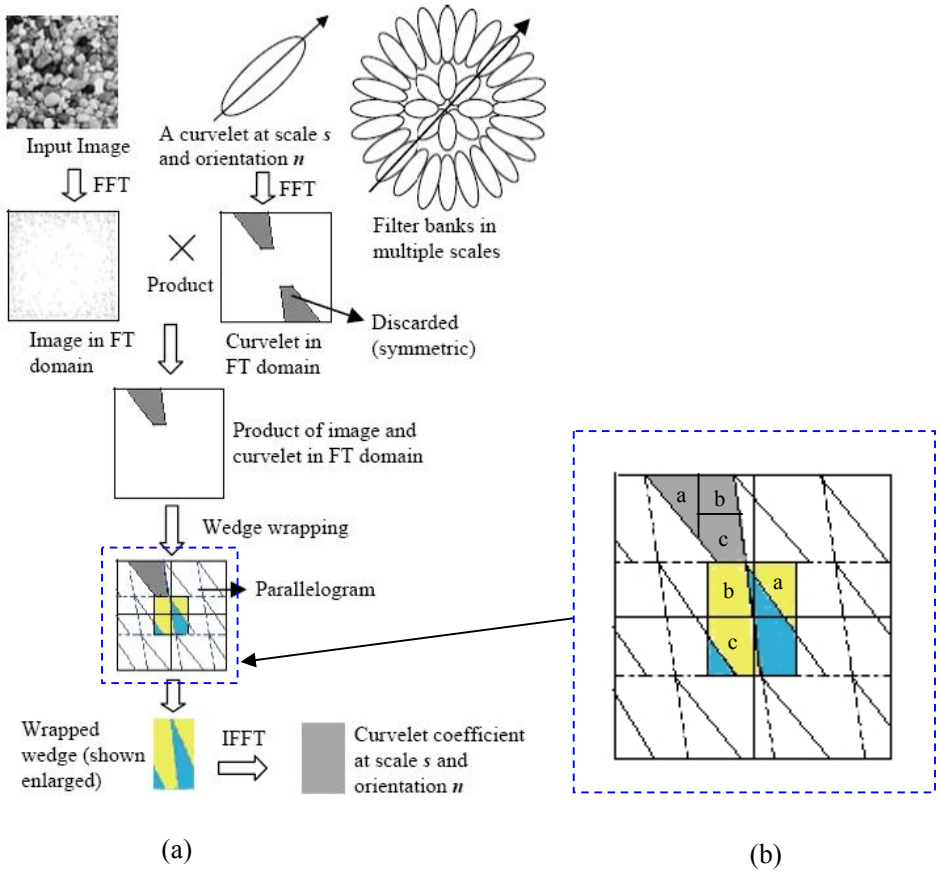
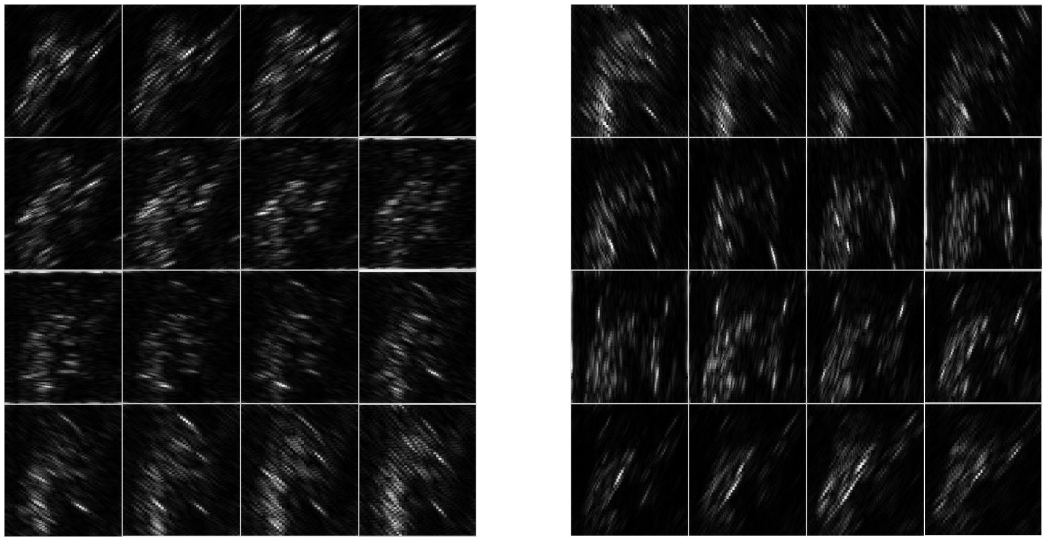


Fig. 4.1: Fast discrete curvelet transform to generate curvelet coefficients.

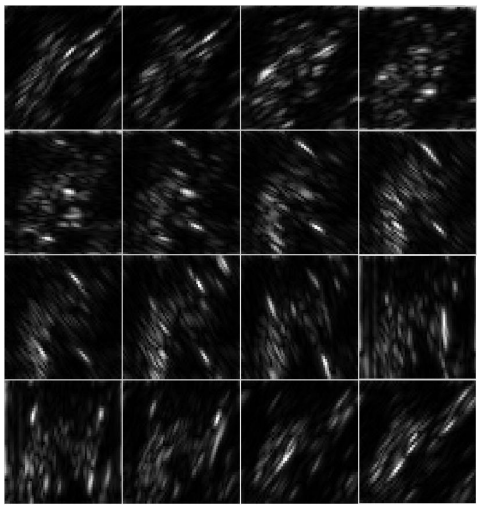
Wedge wrapping is done for all the wedges at each scale in the frequency domain, so we obtain a set of subbands or wedges at each curvelet decomposition level. These subbands are the collection of discrete curvelet coefficients.

To provide an illustration of curvelet subbands, we apply fast discrete curvelet transform to a 512×512 Lena image with 6 decomposition levels using Curvelab-2.1.2 of [69]. The subbands generated from this image are shown in Fig. 4.2. Lena image has a rich collection of multidirectional edges. From all the subband images of Lena image shown in Fig. 4.2, we find that the wrapping based discrete curvelet coefficients capture and represent the edge information more accurately than wavelet (Fig. 2.10) and Gabor filters (Fig. 2.7).

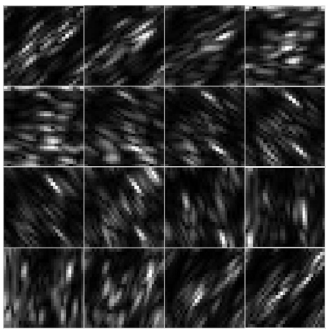
(a) Curvelet subband at scale 6.



(b) First 32 subbands (left contains first 16, right contains last 16 subbands) at scale 5.



(c) First 16 subbands at scale 4



(d) First 16 subbands at scale 3.

(e) First 8 subbands at scale 2.

(f) Subband at scale 1.

Fig. 4.2: Curvelet subbands at different scales for Lena image (512×512). Each subband captures curvelet coefficients of Lena image from each orientation.

4.3 Curvelet Texture Features Extraction and Indexing

Once the curvelet coefficients are generated and stored in each subband, the mean and standard deviation of the coefficients associated with each subband are computed. Generally, these mean and standard deviation are then used as the texture feature vector elements of the image. Thus, for each curvelet, we obtain two texture features. If n curvelets are used for the transform, $2n$ texture features are obtained. This results in a $2n$ dimensional texture feature vector which represents each image in the feature database. These feature descriptors are then used to index images in the feature database, which is also known as the image 'indexing scheme'. An internal mapping is generated to make links between images in the database to the corresponding features in the feature database. A generalized flow of image indexing mechanism is shown in Fig. 4.3.

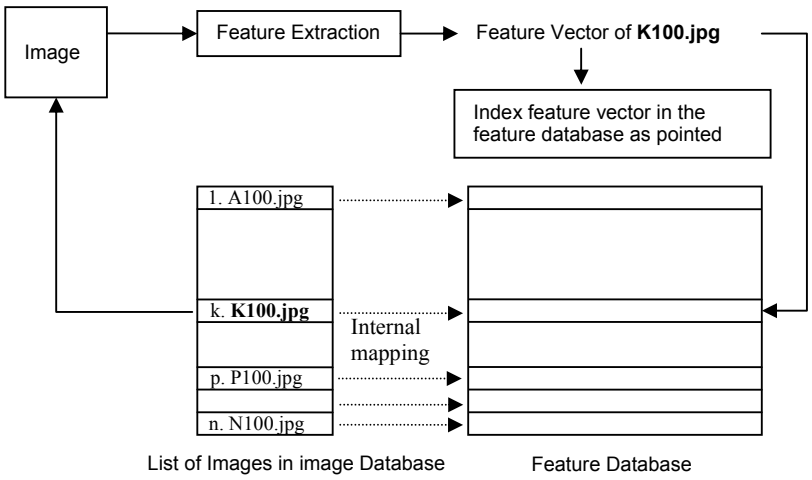


Fig. 4.3: Image indexing process

For our image retrieval tests, we use the well known Brodatz texture database [23] as our test bed. Justification behind selection of this database is presented in Chapter 2. In the following we describe the texture representation mechanism using curvelet transform which we apply to the images in this database.

Database Used: The proposed texture feature vector generation method is implemented on a database consisting of 1792 images taken from the Brodatz texture database [23]. There are 112 different categories of textures and each category contains 16 similar images. This results in a large database of 1792 texture images.

All the images in this database are in JPEG format and are 128×128 pixels in size. These images are quite diverse and each category of textures possesses different qualities. The database includes both natural and human generated textures. Hence, this database is really a good choice for a comparative retrieval performance evaluation based on texture features. Some texture images from this database are shown in Fig 4.4 each belonging to a different texture pattern.

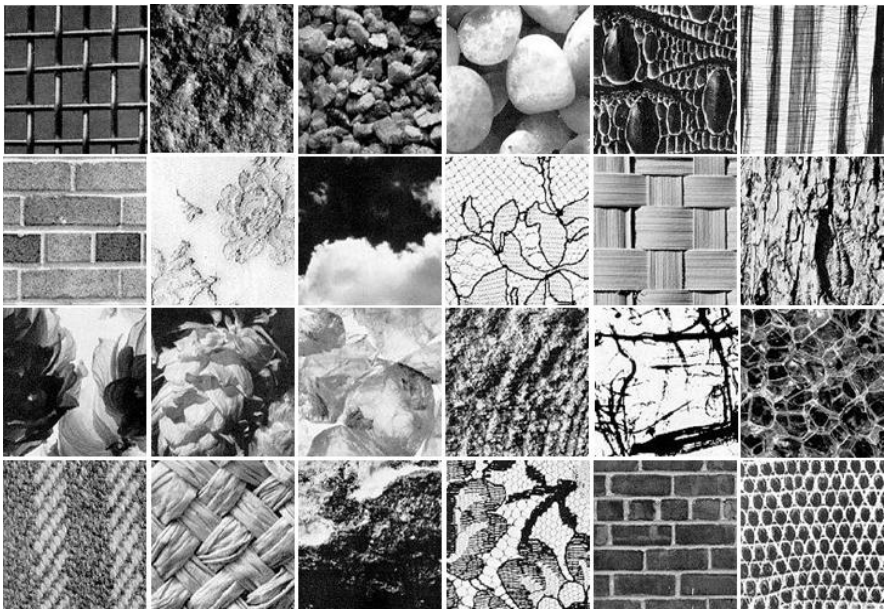


Fig. 4.4: Sample images from 24 different categories of textures from the Brodatz database.

Texture features computation using fast discrete curvelet transform employs the feature extraction and then the feature vector formation mechanism.

Feature Extraction and Descriptor Creation For the Brodatz Database: Our first step is to apply the wrapping based discrete curvelet transform using Curvelab-2.1.2 from [69] to find the coefficients and to create feature vectors for every 128×128 images in the database. The first phase of our CBIR experiment involves 4 levels of discrete curvelet decomposition. To obtain more accuracy in CBIR outcome, 5 levels of decomposition is used which is the highest level of decomposition possible for a 128×128 image using Curvelab-2.1.2. In this implementation, we choose wavelet in the finest level of curvelet transform as it reduces the redundancy factor [17]. We obtain one subband at the coarsest and one subband at the finest level of curvelet decomposition. For other levels of curvelet decomposition, we get different number of subbands at each level which is shown in Table 4.1. Next, we calculate the mean and standard deviation of the first half of the total subbands at each scale (excluding the subband at the coarsest and at the finest level). The mean and standard deviation are calculated for the subband at the coarsest and the finest scale separately. The reason why only first half of the total subbands at a resolution level are considered for feature calculation is that the curvelet at angle θ produces the same coefficients as the curvelet at angle ($\theta+\pi$) in the frequency domain. These subbands are symmetric in nature. Therefore, considering half of the total

number of subbands at each scale reduces the total computation time for the feature vector formation.

Table 4.1: Curvelet subband Distribution at Each Scale

Curvelet transform (4 level decomposition)					
Scale	1	2	3	4	
Total no. subbands	1	16	32	1	
Subbands considered for feature calculation	1	8	16	1	
Curvelet transform (5 level decomposition)					
Scale	1	2	3	4	5
Total no. subbands	1	16	32	32	1
Subbands considered for feature calculation	1	8	16	16	1

The mean of a subband at scale a and orientation θ can be expressed as:

$$\mu_{a\theta} = \frac{E(a,\theta)}{M \times N}, \tag{4.1}$$

where $M \times N$ is the size of the image and $E(a,\theta) = \sum_x \sum_y |Curvelet_{a\theta}(x,y)|$ is the energy of the curvelet transformed image at scale a and orientation θ. The standard deviation of a subband at scale a and orientation θ can be shown as:

$$\sigma_{a\theta} = \frac{\sqrt{\sum_x \sum_y (|Curvelet_{a\theta}(x,y)| - \mu_{a\theta})^2}}{M \times N}. \tag{4.2}$$

Total number of elements in the feature vector can be expressed as:

$$2 \times (2 + \sum_{j=2}^{Total\ no.\ of\ scale-1} (Total\ no.\ of\ subbands\ at\ scale\ j)/2). \tag{4.3}$$

All these elements are organized in such a way that the standard deviations remain in the first half of the feature vector and the means are inserted into the second half of the

feature vector. The following explanation illustrates the structure of the feature vector. Let a curvelet feature vector of a texture image be denoted by fc and the standard deviation and mean of the curvelet subband at scale a and orientation θ are denoted by $\sigma_{a\theta}$ and $\mu_{a\theta}$ respectively. Then the curvelet feature vector fc can be expressed as:

$$fc = \{\sigma_{11}, \sigma_{21}, .., \sigma_{2k}, \sigma_{31}, .., \sigma_{3k}, .., \sigma_{J-1k}, \sigma_{J1}, \mu_{11}, \mu_{21}, .., \mu_{2k}, \mu_{31}, .., \mu_{3k}, .., \mu_{J-1k}, \mu_{J1}\}, \tag{4.4}$$

where, J is the finest scale and k is the total number of subbands taken at scale a. This feature vector is then used to index the image in the feature database.

We obtain a total of (1+8+16+1) = 26 subbands for each of the 128×128 images in the database when using 4 levels of discrete curvelet decomposition. Hence the feature vector for each image becomes 26×2 = 52 dimensional. Using 5 levels of discrete curvelet decomposition, we obtain a total of (1+8+16+16+1) = 42 subbands from each image. As a result, the feature vector length becomes 42×2 = 84 dimensional. After the feature vectors are generated all the images in the image database are indexed by their feature vectors in the feature database. For our texture database, the feature database size is 1792×52 using 4 levels of decomposition and 1792×84 using 5 levels of decomposition. We maintain an internal mapping of which image in the Brodatz database corresponds to which feature vector in the feature database. For this internal mapping, a list is generated containing the names and indices of images in the image database. The feature vectors of each image are then inserted into the feature database according to the same sequence. We generate the curvelet feature vector for the query image using the same procedure described above.

4.4 Image Retrieval Using Curvelet Features

Once the database images are indexed in the feature database, search and retrieval is performed on the basis of these features. The query image is subjected to the feature descriptor generation process to obtain its feature vector. The database images are then compared to the query image using a similarity measurement technique on the feature elements in the features vectors. A generalized form of content based image retrieval process is shown in Fig. 4.5.

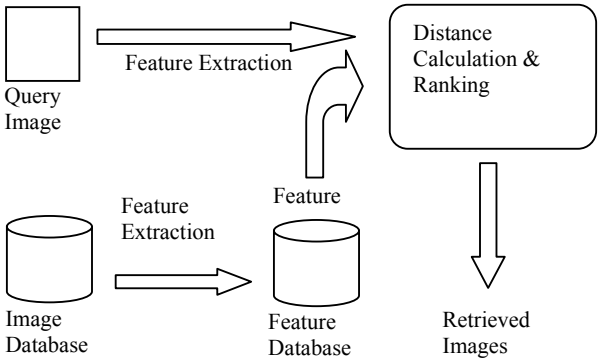


Fig. 4.5: Image retrieval mechanism.

In our retrieval experiments, texture features are first computed from the curvelet coefficients of the query image. The system uses the Euclidean distance to compare the query feature vector with all the feature vectors in the feature database. The Euclidean distance d can be expressed as:

$$d(\mathbf{Q},\mathbf{T}) = (\sum_{i=1}^{2n}(Q_i - T_i)^2)^{\frac{1}{2}}. \tag{4.5}$$

Here, $\mathbf{Q} = \{Q_1, Q_2, ..., Q_{2n}\}$ is the feature vector of the query image, and $\mathbf{T} = \{T_1, T_2, ..., T_{2n}\}$ is the feature vector of the target image in the database. The images in the database are ranked according to their distance *d* to the query image in ascending order. Then the ranked images are returned to the user.

We have described Gabor filters transform already in Chapter 2. We also find that Gabor filters texture features are so far the most effective spectral approach in CBIR. To compare the wrapping based discrete curvelet texture features to the existing Gabor filters features, we create Gabor filters features for the Brodatz images and perform retrieval. Discrete curvelet retrieval and Gabor filters retrieval are then compared. Gabor filters feature vector formation process is described in brief in the following section.

4.5 Gabor Filters Texture Feature Descriptor

For Gabor filter, we use the best combinations of parameters found by Chen et al in [6], that is, 6 scales and 4 orientations with filter size of 13x13 pixels.. After applying the

Gabor filters to each image of the Brodatz texture database we obtain 24 subbands in total from every image. Each of these energy subbands contains Gabor filters coefficients. Similar to curvelet feature, these subbands are then used to calculate Gabor filters features. The mean and standard deviation are then calculated from each subband to form the feature vector as described in "Equation 4.2". As a result, we get a feature vector of $24 \times 2=48$ dimension.

4.6 Summary

In this chapter, first we have described curvelet computation. From the curvelet texture representation of Lena image, we find curvelet transform is powerful in capturing multidirectional edges in the image. Then we have described the texture features calculation and the texture features descriptor generation process using wrapping based discrete curvelet transform. This includes the computation of the curvelet texture feature and the image indexing used in the feature database. After this general description, the complete process for curvelet texture feature descriptor generation specific to the Brodatz database is discussed. Thus we obtain texture feature representation of Brodatz images in the feature database. In order to benchmark our experiment, we have described the texture representation using Gabor filters. This chapter lays the foundation of experiments performed in the next chapter. CBIR experiments using these two features, their performance comparison and performance analysis is described in chapter 5.

Image Retrieval Using Curvelet

5.1 Introduction

In previous chapters, we described the concept of curvelet, its implementation and feature extraction process. Based on those analyses, we find that curvelet is powerful and effective in capturing edge information accurately which is a key to texture features based CBIR. In this chapter, we test the performance of image retrieval using curvelet texture features. We also test curvelet features performance on the scale distorted image retrieval. In the literature, Gabor filters texture feature is the de facto bench mark for texture image retrieval. In order to benchmark the wrapping based curvelet texture retrieval performance, we compare its performance with that of existing Gabor filters and wavelet texture features. For this purpose, we organize our experiments in the following ways:

1. First, we need to determine the level of curvelet decomposition which provides the best texture feature based CBIR performance in terms of effectiveness and efficiency. From the curvelet texture representation shown in Fig. 4.2 of Chapter 4, we find that image texture pattern is clearer at the higher levels (4, 5 and 6) than at the lower levels (1, 2 and 3). For Brodatz images we can apply at most 5 levels of curvelet decomposition when we use Curvelab-2.1.2 [69]. In chapter 4, we created curvelet texture descriptors using 4 and 5 levels of decomposition.

Now we compare the image retrieval performance of these curvelet features to determine the best performance of curvelet.

2. To benchmark the best performance of curvelet retrieval, we compare the curvelet retrieval performance with the Gabor filters and wavelet.
3. Finding the robustness of discrete curvelet texture features to scale distortions by performing image retrieval experiments on a large database that contains original and scale distorted images.
4. Compare scale distortion tolerance of discrete curvelet features with the Gabor filters features.

The following sections describe and analyze the scenario for CBIR with discrete curvelet texture features. Retrieval performances of both these features are analyzed after all the comparisons are presented.

5.2 Performance Measurement

The most commonly used performance measurement, precision-recall pair, is used to evaluate the retrieval performance. Definition of precision and recall has been given in chapter 2. Precision measures the accuracy whereas recall measures the robustness of a retrieval mechanism. The inversely proportional behavior is clearly visible from all the precision vs. recall graphs shown in this chapter.

5.3 Optimal Level of Discrete Curvelet Decomposition

The main purpose of this test is to identify the number of curvelet decomposition level which gives the best retrieval outcome. In our first test, the Brodatz database images [23] are decomposed using 4 level in discrete curvelet transform. Based on the feature vector calculation process described in Chapter 4, a 52 dimensional feature vector is generated from each of the 1792 images. A set of ranked output images are obtained from the database images applying the image retrieval mechanism described in the previous chapter.

Because the ground truth of the whole database is known, every image in the database is used as a query. For each query, the precision of the retrieval at each level of recall is obtained. These precision values are then averaged to produce the average precision-recall curve for the 1,792 database images. Using the precision and recall measure for performance evaluation, we obtain 77.65% of retrieval accuracy on the average.

In our second test, we apply 5 levels of discrete curvelet decomposition to the Brodatz texture images to generate the texture features vectors instead of 4. As described in chapter 4, we obtain a 84 dimensional feature vector from each image. Using the Euclidean distance as the similarity measurement, we obtain 79.54% average precision accuracy. This higher level of performance is achieved at the cost of larger feature descriptor dimension. The retrieval results for the two curvelet approaches are compared in Fig. 5.1.

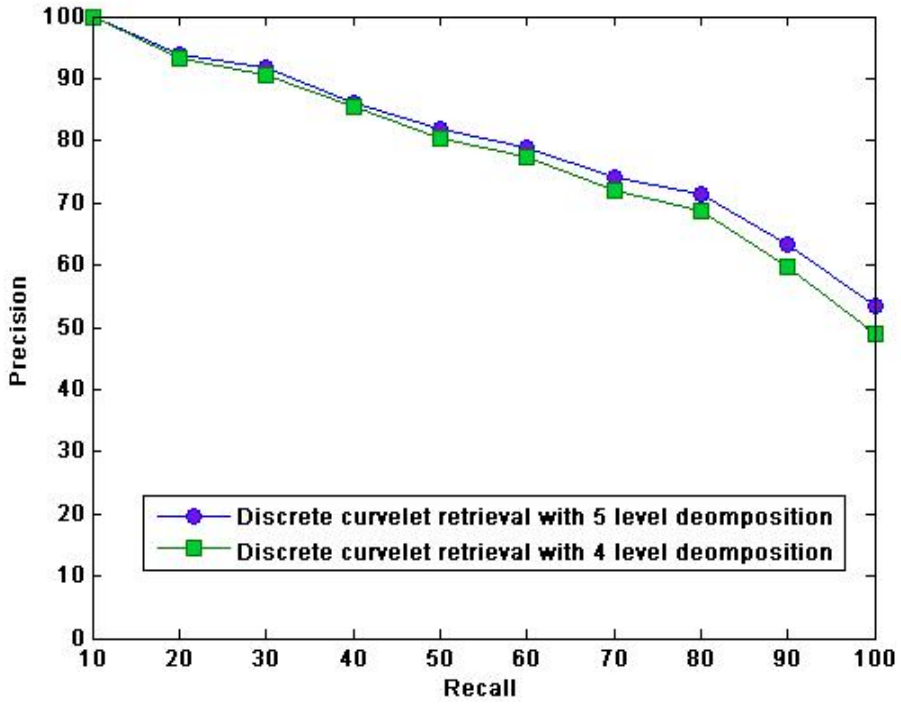
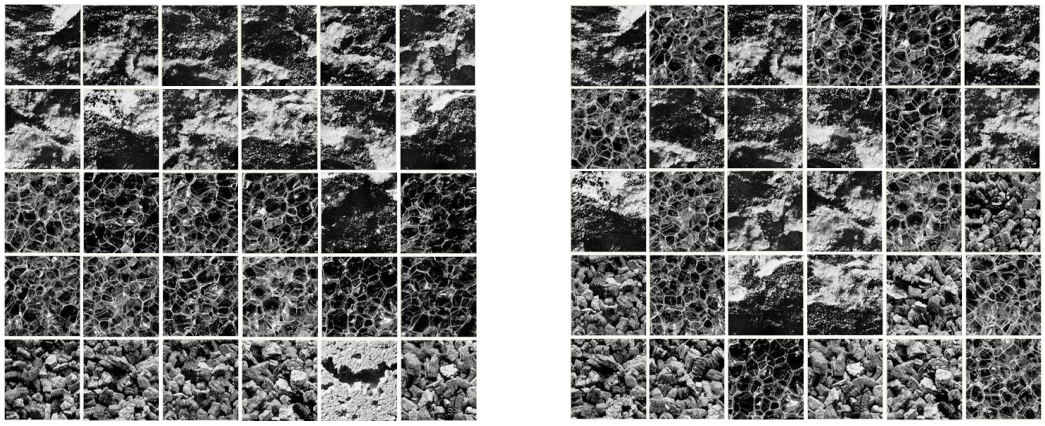


Fig. 5.1: Image retrieval performance with 4 levels and 5 levels of discrete curvelet texture features.

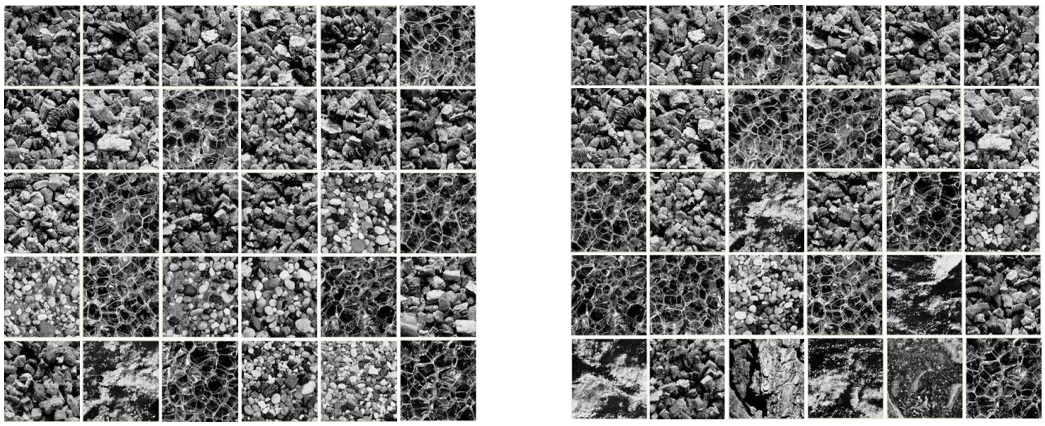
From Fig. 5.1, we see that the retrieval using curvelet texture features obtained from 5 levels of decomposition achieves a higher level of accuracy than that of 4 levels of decomposition. But the slight retrieval performance gain is at the cost of much higher dimension. In CBIR, the retrieval efficiency is more important than a slight precision gain. Therefore, for retrieval efficiency, curvelet features obtained from 4 levels of decomposition is sufficient and we use this feature for all the experiments onwards.

Examples of several retrieved images using discrete curvelet texture features with 4 and 5 levels of decomposition are shown in Fig. 5.2. Queries used for these retrievals are

d007-002.jpg and d005-000.jpg. The top left image in the retrieved pages is used as the query.



(a) D007-002.jpg



(b) D005-000.jpg

Fig. 5.2: The first 30 retrieved images. The left page corresponds to 5 level and the right page corresponds to 4 level discrete curvelet retrieval. The images are organized from left to right and top to bottom in the order of decreased similarity to the query image.

5.4 Comparison of Retrieval Performance between Curvelet, Gabor Filters and Wavelet

To benchmark the retrieval performance, we compare the curvelet texture retrieval result with the two best spectral texture features in literature, e.g., Gabor filters and wavelet texture features. As we already described discrete curvelet and Gabor filters texture feature vectors, testing image retrieval is easily done. For all the CBIR experiments, we use Matlab programming environment.

5.4.1 Comparison of Retrieval Accuracy

As the ground truth of the database is known, all the images are used as queries and the precision values at each recall level are obtained. Then the average precision values are also computed for both the curvelet and Gabor filters retrieval. The P-R curve is shown in Fig. 5.3.

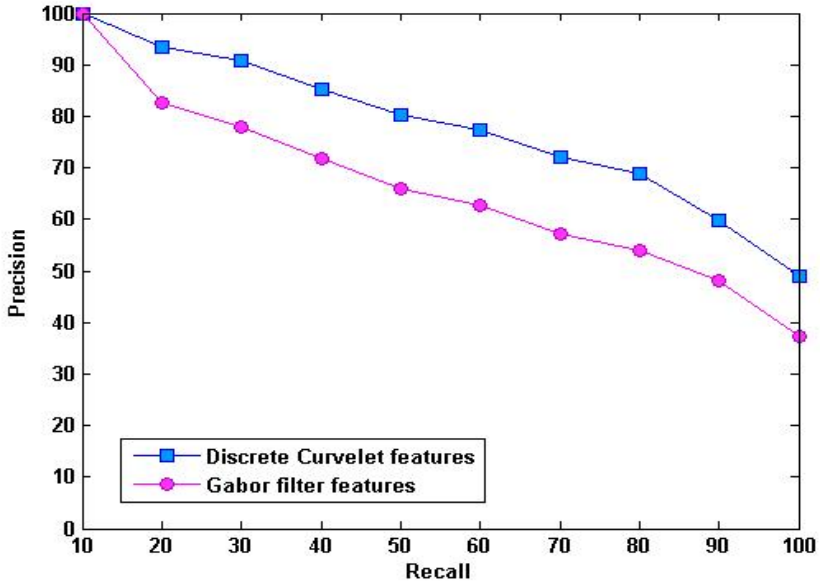


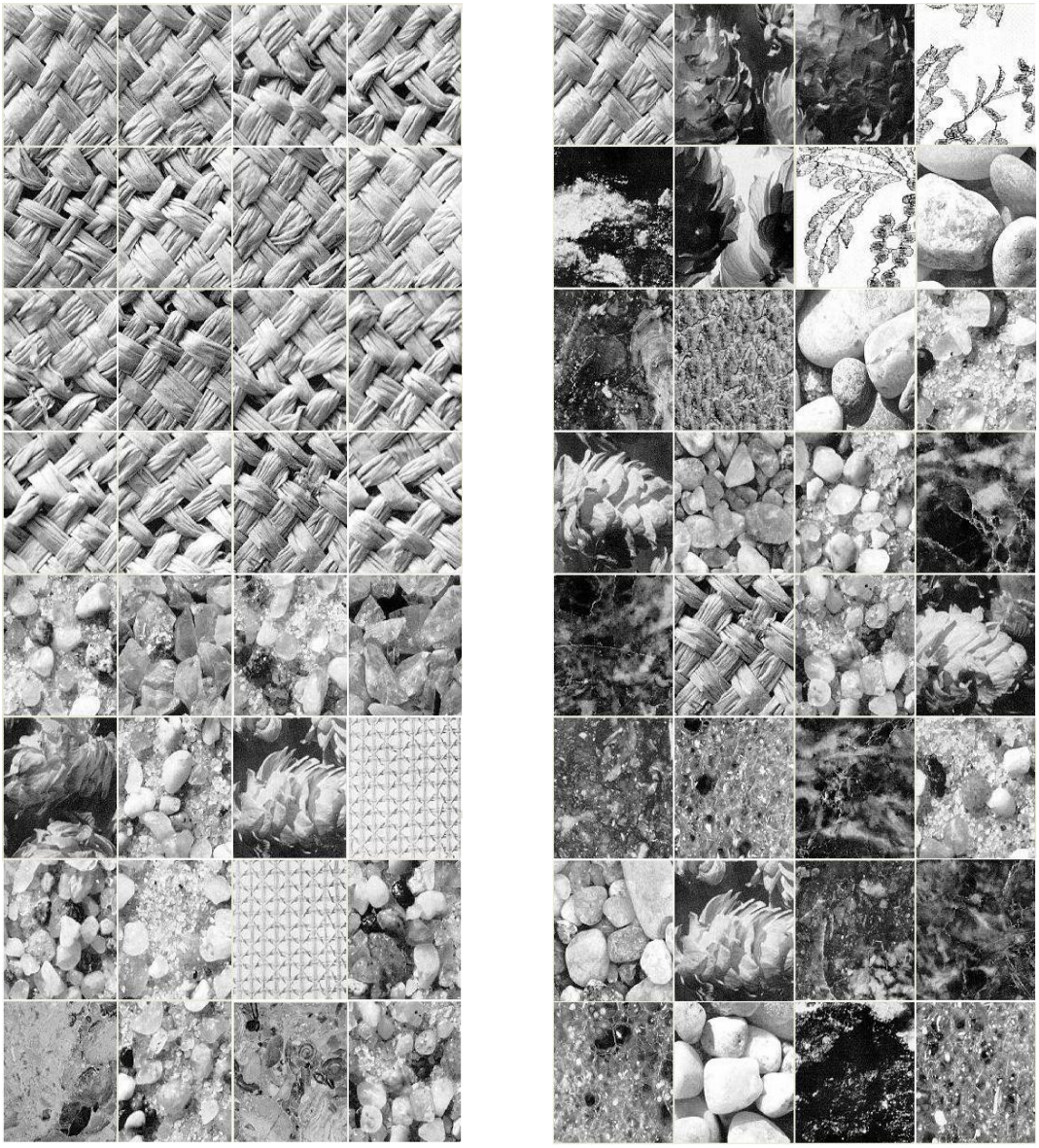
Fig. 5.3: Comparison of discrete curvelet and Gabor filters retrieval performance on Brodatz texture database.

As can be seen from Fig. 5.3, the retrieval performance of curvelet features is significantly higher than that of Gabor filters features. From our previous experiments, we see that curvelet transform with 5 levels of decomposition has even better retrieval performance than at 4 levels of decomposition. But for the sake of curvelet retrieval efficiency, we choose 4 levels and compare the retrieval outcome with Gabor filters.

From the average performance, we find Gabor filters show 65.75% retrieval precision at full recall. Curvelet, however, gives an average retrieval precision of 77.65%. According to [10], discrete wavelet texture features provide 70.30% of retrieval accuracy using the same database. Therefore, curvelet retrieval performance is significantly better than both the Gabor filters and wavelet methods.

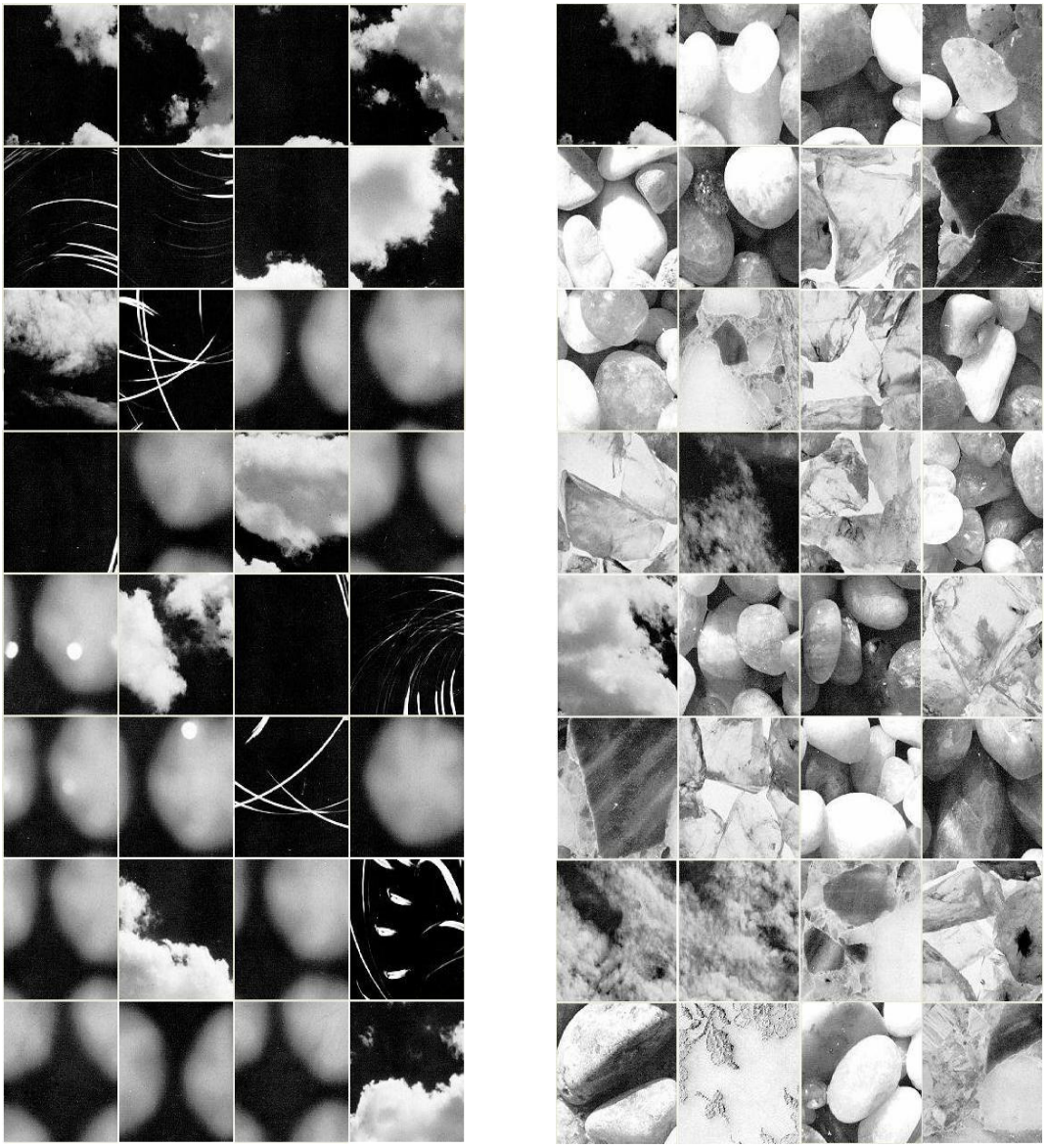
Figures 5.4 through 5.6, shows a few retrieval results using discrete curvelet and Gabor filters texture descriptor. All the query images are natural textures which are

difficult to retrieve. In all the three scenarios, curvelet texture descriptor gives better result than Gabor filters.



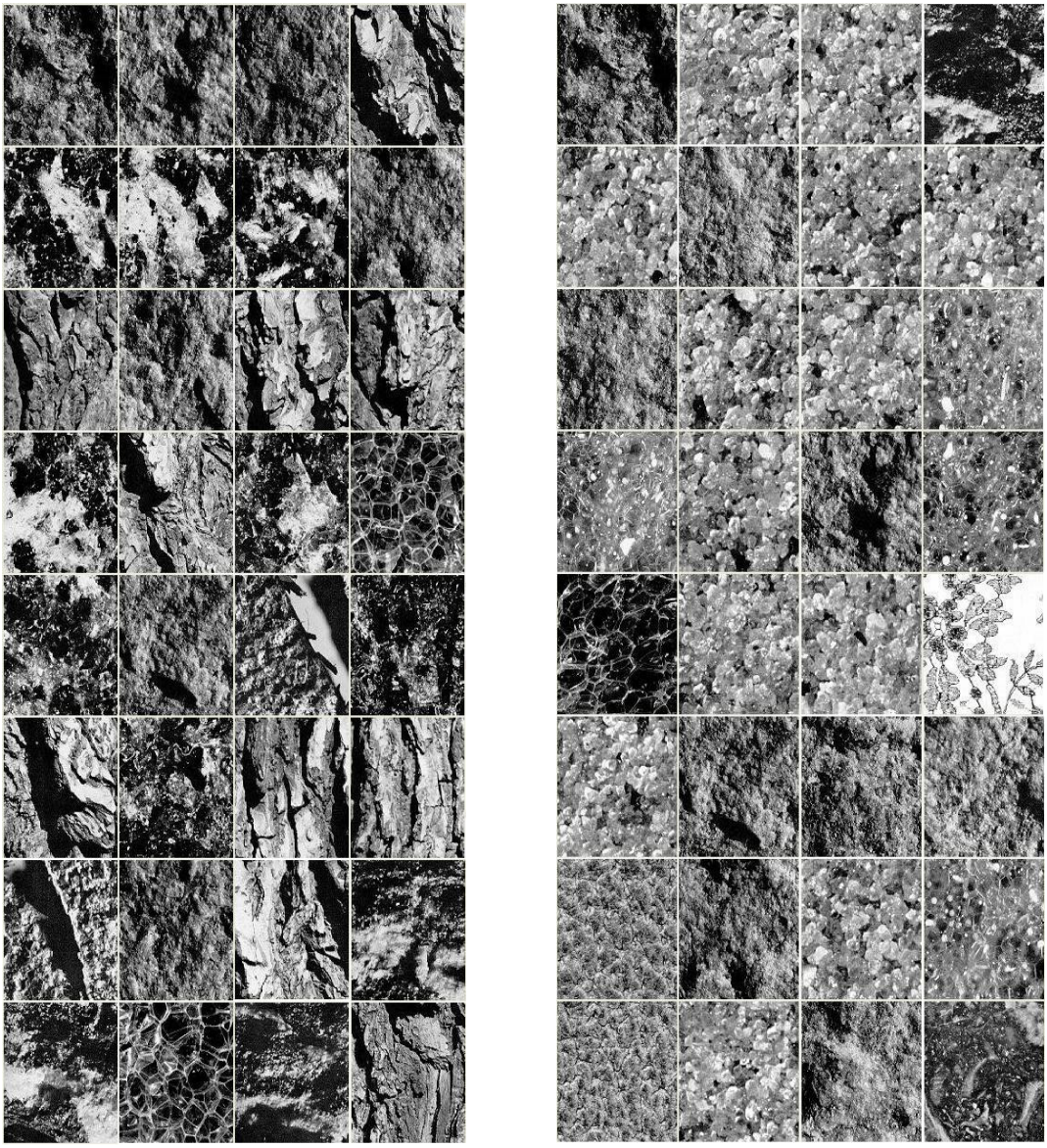
(a) Curvelet retrieval (b) Gabor filters retrieval

Fig. 5.4: The first 32 retrieved images using d018-002.jpg as query. The images in all the retrieved pages are organized from left to right and top to bottom in the order of increased feature distance (or decreased similarity) to the query image.



(a) Curvelet retrieval (b) Gabor filters retrieval

Fig. 5.5: The first 32 retrieved images using d091-003.jpg as query. The images in all the retrieved pages are organized from left to right and top to bottom in the order of increased feature distance (or decreased similarity) to the query image.



(a) Curvelet retrieval (b) Gabor retrieval

Fig. 5.6: The first 32 retrieved images using d002-002.jpg as query. The images in all the retrieved pages are organized from left to right and top to bottom in the order of increased feature distance (or decreased similarity) to the query image.

5.4.2 Comparison of Computation Time

Feature vector calculation time using two different levels of curvelet decompositions and Gabor filters are shown in table 5.1. The time reported is obtained on Windows platform of PC with 3.00 GHz Pentium(R) D processor and 2 GB memory.

For each spectral approach, first, the total time to calculate the 1792 feature vectors for images in Brodatz database is recorded. Then the average time for each feature vector formation is computed from this total time. From Table 5.1, we see the average time to create curvelet texture descriptor of dimension 84 is slightly faster than that of 48 dimensional Gabor filters texture descriptor. Curvelet texture descriptor of dimension 52 is the most efficient among these three.

Table 5.1: Texture Feature Vector Calculation Time

Spectral approach	**Total time taken for 1792 images (seconds)**	**Average time taken for each image (seconds)**
Curvelet (4 level decomposition. 52 features)	589.59	0.3290
Curvelet (5 level decomposition. 84 features)	613.68	0.3425
Gabor filters (48 features)	674.49	0.3764

We also calculate the average retrieval time for these three spectral approaches by recording the total time required to retrieve the relevant images for all the 1792 Brodatz textures. Table 5.2 shows there is negligible retrieval time difference among Gabor filters, 4 level, and 5 level curvelet.

Table 5.2: Texture Retrieval Time

Spectral Approach	**Total time taken for 1792 images (seconds)**	**Average time taken for each image (seconds)**
Curvelet (4 level decomposition)	58	0.0324
Curvelet (5 level decomposition)	58	0.0324
Gabor filters	59	0.0329

5.5 Test of Tolerance on Scale Distortions

In this section, we test the scale tolerance of curvelet and compare the performance to Gabor filters method. Tolerance to scale distortion is important as images in nature are usually captured in different scales.

5.5.1 Scaled Database

To observe the effect of discrete curvelet texture features on geometrically enlarged and distorted images, we create a new database consisting of an additional 200 scale distorted

images. We first randomly select 40 images from the Brodatz database each of which belongs to a different category of textures. We then apply 10% to 50% increase in the size of all these images. Finally, we crop the central area of these 200 images to 128×128 pixels and add them to the Brodatz texture database which we used earlier. The total number of elements in the new database is now 1992. Some examples of original and scale distorted images are shown in Fig. 5.7.

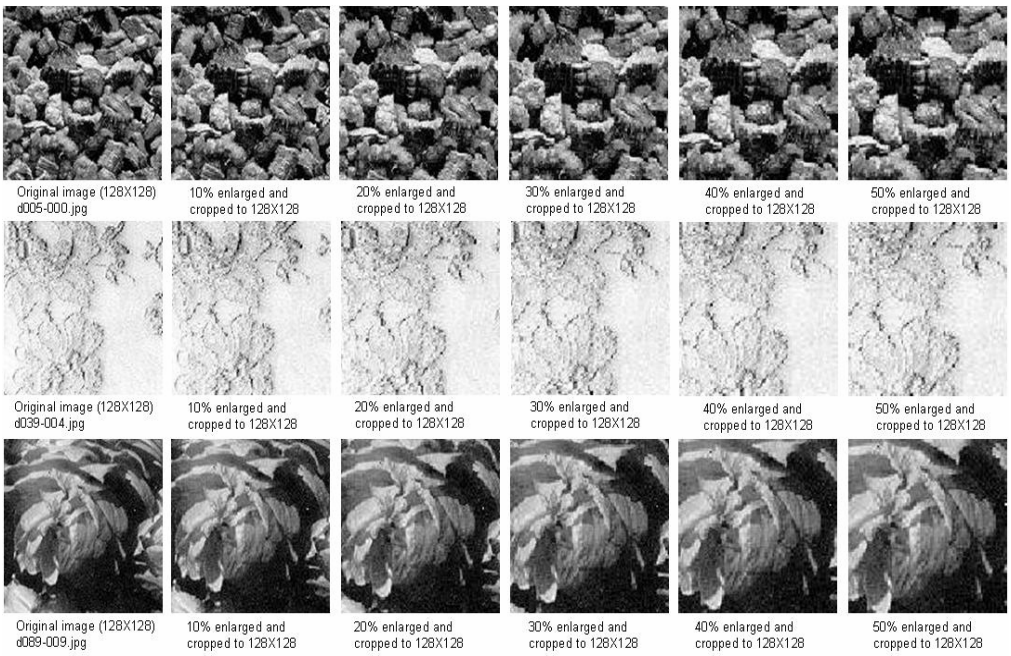


Fig. 5.7: Examples of original and scale distorted images from new texture database.

5.5.2 Experiments and Results

For the experiments in this section, all the 1992 images in the new database are indexed with the discrete curvelet (52 dimensional) and Gabor filters descriptor in two different feature databases. To test scale distortion tolerance, different sets of query images are used to identify the retrieval effects in different conditions.

5.5.2.1 Performance of Curvelet Retrieval on Scaled Images

In first phase of the experiment, we perform curvelet texture retrieval using the selected 40 categories as query, so the total number of images used as query is 40×16 (each category contains 16 original image) = 640. These will be used as the reference to the

scaled image retrieval test. In the second phase of retrieval, we combine the 200 scale distorted and the original 640 images (640+200 = 840) as queries. The main objective of this phase is to study the overall curvelet features scale distortion tolerance.

Performance of curvelet retrieval is measured in terms of precision and recall values for these two query sets. To compare the retrieval performance, precision values for the two tests are plotted against the corresponding recall levels marked from 10 to 100 percentages (Fig. 5.8).

It is evident from the figure that the retrieval accuracy is higher when original images are used as the query. When we consider the combined query set, the overall performance degrades. This is because some scale distorted images fail to effectively recognize all the similar textures in the same category. On the average, the retrieval precision is 80.64% when undistorted original images are used as queries and it drops to 75.90% when distorted and original images are considered as queries. There is an average drop of 4.74% in curvelet retrieval performance.

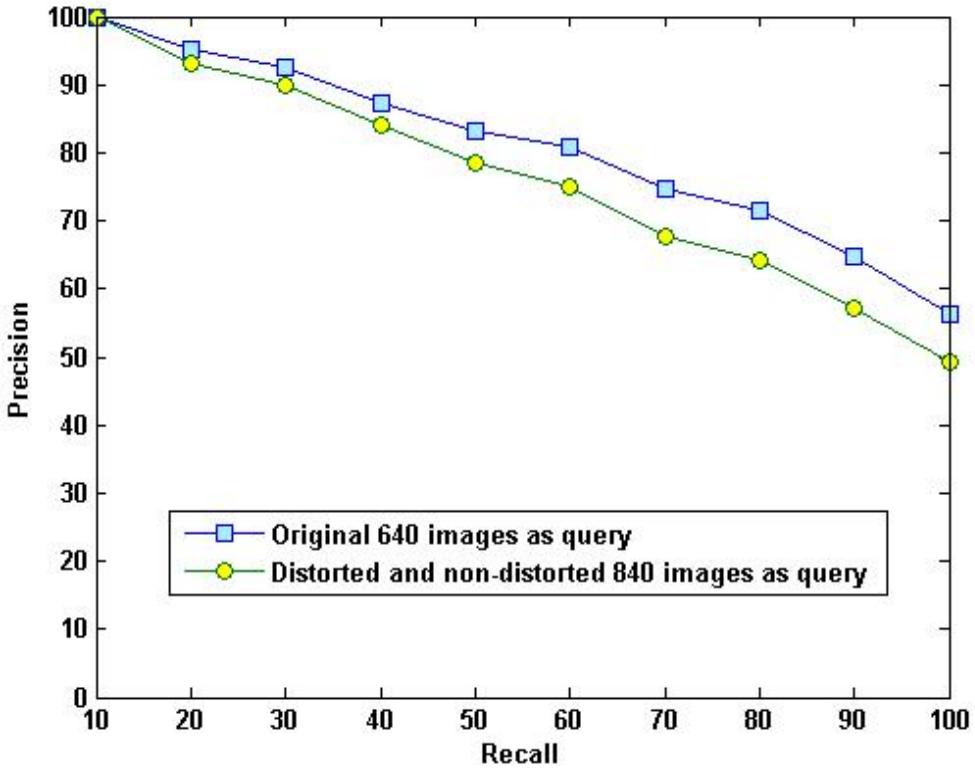


Fig. 5.8: Image retrieval performance of curvelet texture features with 640 original and 840 mixed images as query.

5.5.2.2 Curvelet vs. Gabor Retrieval Performance on Scaled Images

To compare the obtained curvelet retrieval performances with Gabor filters features, we follow the same process of retrieval using Gabor filters features with the same sets of query. Thus, we get two precision-recall curves for Gabor filters retrieval using 640 and 840 images in two different query sets (Fig. 5.9).

From the precision-recall curves shown in Fig. 5.9, we find the retrieval accuracy is higher for original images than mixed images using Gabor filters features. On the average, Gabor filters texture descriptor obtains 73.50% retrieval accuracy when the 640 original images are used as queries. However, the average precision falls to 67.87% when retrieval is performed on the 840 original and distorted images as queries. There is a 5.63% decrease in Gabor retrieval performance whereas it is 4.74% for curvelet.

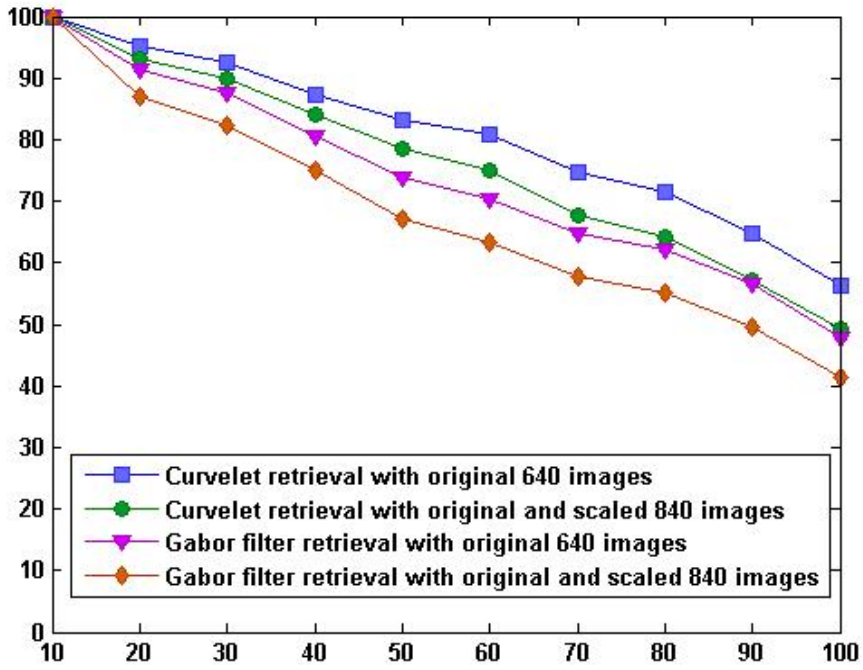
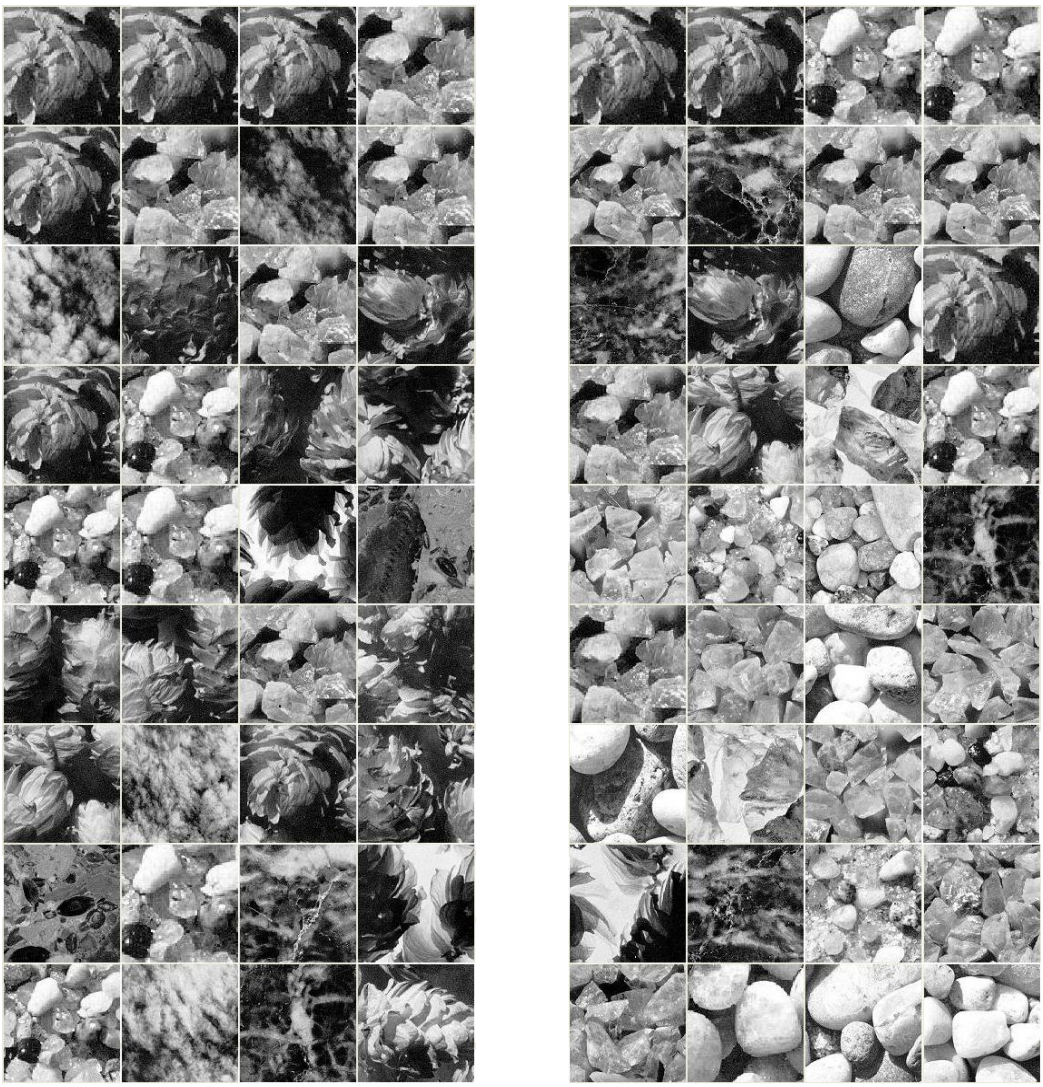


Fig. 5.9: Image retrieval performance of Gabor filters texture features with 640 original and 840 mixed images as query.

Comparing the average retrieval outcome in both the cases, we notice that there is a significant retrieval difference between the discrete curvelet and Gabor filters texture features. Using original images as query curvelet achieves 80.64% of accuracy whereas it is 73.50% in case of Gabor filters. Curvelet retrieval (75.90%) also outperforms Gabor filters (67.87%) when the 840 mixed images are used as queries. These results indicate

that curvelet texture features are more tolerant to scale distortions than Gabor filters texture features. Another observation is, while curvelet retrieval performance on the mixed query set drops gradually, Gabor retrieval performance drops consistently across all the recall levels. Again it demonstrates that Gabor texture features are more sensitive to scale distortions than curvelet. Fig. 5.10 shows an example of retrieval result using scaled distorted image as query. From this figure we find that curvelet captures scale distorted natural images more effectively than Gabor filters.



(a) Curvelet retrieval (b) Gabor filters retrieval

Fig. 5.10: The first 36 retrieved images using 50% enlarged and cropped d089-009.jpg as query. The images retrieved are organized from left to right and top to bottom in the order of increased feature distance (or decreased similarity) to the query image.

5.5.2.3 Curvelet Retrieval Performance on the Whole Database

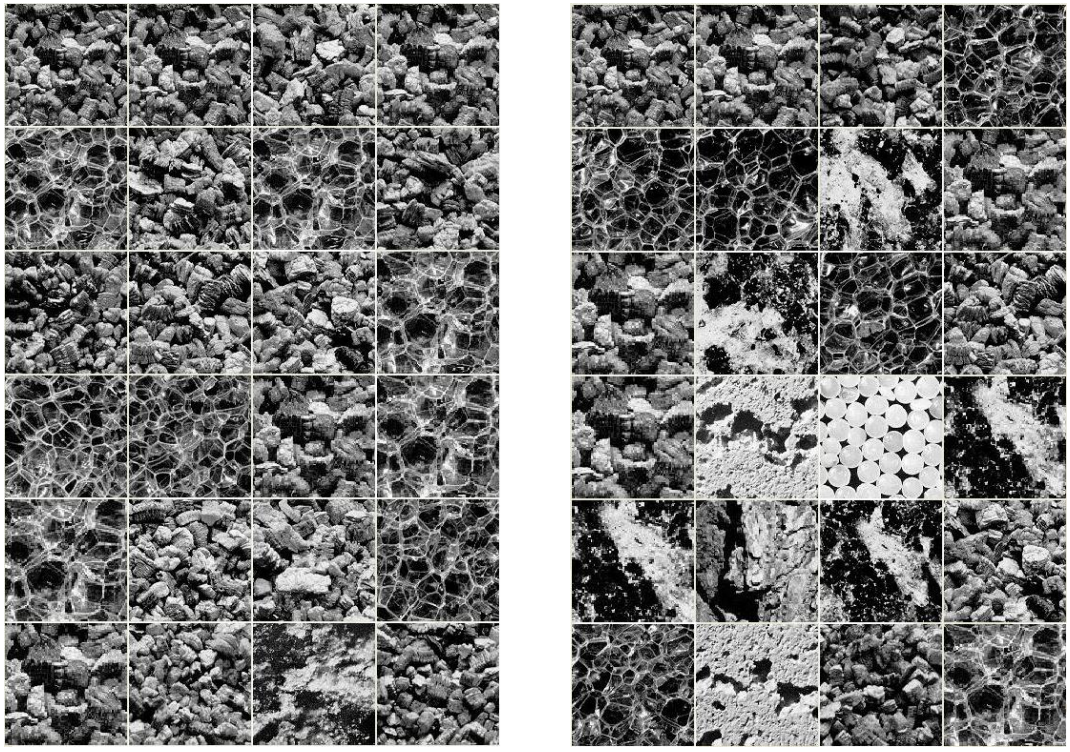
This is the second trial of curvelet scale distortion tolerance test. To study the scale distortion tolerance capability of curvelet features, at first we choose original 1792 Brodatz images as query to retrieve images from the new database of 1992 images. This is done to observe how effectively this set of curvelet texture descriptors captures scale distorted images from the whole database. Then we choose all the 1992 images (including original and scale distorted) as query to observe the overall performance and compare it to the previous result.

On the average, using 1792 original images as query, curvelet texture descriptor obtains an average retrieval accuracy of 79.08%. However, retrieval accuracy is 79.04% when all the 1992 database images are used as query. There is almost no difference in the retrieval accuracy in both the cases. Therefore, we can say that curvelet transform is robust to scale distortions, so it can effectively capture scale distorted images from a large database.

5.5.2.4 Curvelet vs. Gabor Filters Retrieval Performance on the Whole Database

To compare the overall curvelet distortion tolerance with Gabor filters, we perform the same tests as mentioned in section 5.5.2.3 with the same set of queries using Gabor filters features. On the average, the Gabor filters texture descriptor shows 67.76% and 67.18% retrieval accuracy using 1792 and 1992 images as query, respectively. The average performance drop is 0.58%.

Comparing the average retrieval outcome for both the query sets, we find discrete curvelet texture descriptor still performs better than Gabor filters. When all the images are used as query, the discrete curvelet texture descriptor obtains 79.04% whereas the Gabor filters texture features achieve 67.18% retrieval accuracy. Curvelet features show more resistance to the distortions generated by the sequence of scaling and cropping. Therefore, it can be used as a general approach where retrieval involves similar scale distorted images in the database. Fig. 5.11 shows an example of retrieval using an image of scale distorted category.



(a) Curvelet retrieval (b) Gabor filters retrieval

Fig. 5.11: The first 24 retrieved images using d005-000.jpg as query that contains 16 original and 5 distorted, total 21 images in the same category. Images in the retrieved pages are organized from left to right and top to bottom in increased feature distance (or decreased similarity) to the query image.

5.6 Results Analysis

When we used 5 levels of curvelet decomposition instead of 4 in CBIR experiments, we obtained higher retrieval accuracy using the same set as query (Fig. 5.1). The reason behind this is curvelet transform with higher levels of decomposition contains more directional information at high frequencies, thereby representing edges of an image effectively. It is known that high frequency components provide better discrimination between images. Therefore, the highest possible level of decomposition provides the best outcome in case of curvelet texture retrieval. Inspecting the texture feature computation time, we found texture feature descriptor generated using 4 levels of curvelet decomposition is more efficient. Thus, we use this for later experiments.

We have compared the curvelet feature obtained from 4 levels of decomposition with the most promising Gabor filters feature. Using all the 1792 images of 112 categories as

query we find that fast discrete curvelet texture retrieval outperforms Gabor filters significantly (Fig. 5.3). We also find that though Gabor filters texture descriptor has fewer dimensions in comparison to both the curvelet descriptors but it takes longer time to generate the texture descriptor (table 5.1).

To observe the scale distortion tolerance of the Curvelet texture descriptor, distorted images have been created and different sets of query images are defined to obtain retrieval outcome. The scale distortion results in boundary and overall information change in all the scale distorted images. Therefore, the energy distribution of an image also changes. From the experimental results, curvelet texture descriptor is found to be highly adaptive to scale distortions. In spite of the changes in the image content, curvelet texture representation recognizes the relevant images more effectively than Gabor filters.

From the experimental results obtained already, we see discrete curvelet texture feature maintains a consistent and good retrieval outcome even though the database is changed with scale distorted images. It obtains more than 75% retrieval accuracy throughout all the experiments. The experimental results indicate that the curvelet texture representation and retrieval is more promising than using Gabor filters and discrete wavelet. The reasons behind this are as follows. First, due to the half-height truncation of the filters to avoid overlapping in the spectral domain, many holes are created in the Gabor filters spectrum. Therefore, the spectrum is not completely covered by Gabor filters [7]. Consequently, much of the frequency information in an image is lost due to the incomplete spectrum cover (Fig.2.8). On the other hand, curvelet transform has a complete coverage in the spectral domain of the image (Fig. 3.4). Second, the Gabor filters use wavelet as the filter bank, which doses not represent edge information effectively [12, 13]. The curvelet transforms has a nice property of effectively representing edges at fine scales (Fig. 4.2) but Gabor filters cannot obtain this high level of accuracy (Fig. 2.7). Third, curvelets provide optimally sparse representation of objects with edges but Gabor filters cannot. Therefore, curvelet transform captures edge information or texture information more accurately than Gabor filters. Finally, Gabor filtering is not a true wavelet transform because it does not scale an image during the filtering process. Due to the scaling of the image size at different levels of transform to adapt an image to the curvelet filter bank, curvelet can tolerate more scale distortions in images. This is evident from all the retrieved image examples in this chapter.

The reason why curvelet performs better than wavelet image retrieval can be explained in two aspects. First, as mentioned earlier, curvelets capture more accurate edge information or texture information than wavelets. Second, as curvelets are tuned to different orientations and capture more accurate directional features than wavelets,

curvelet performs better than both Gabor filters and wavelet in Brodatz image retrieval. It also performs better than Gabor filters in scale distorted image retrieval.

5.7 Summary

In this chapter, we have measured the discrete curvelet texture retrieval performance by performing different experiments with three objectives. Our first objective was to determine the curvelet decomposition level that obtains the highest texture retrieval outcome. Experimental result of this part shows that curvelet texture features obtain peak retrieval accuracy with the highest level of decomposition applicable to the database images. Second, to benchmark the most effective and the most efficient curvelet texture descriptor, its retrieval performance has been compared to that of the Gabor filters and wavelet. We find that curvelet has significantly higher retrieval performance than both Gabor filters and wavelet. Third, to observe the scale distortion tolerance of the curvelet feature, it is used to retrieve images from a large database containing scaled images and is found to be more robust to scale distortion than Gabor filters. The discrete curvelet feature is also more efficient than Gabor filters feature. Therefore, we find discrete curvelet texture descriptor as a powerful means to perform CBIR.

Conclusions and Future Work

6.1 Overview

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

From the very beginning of CBIR research, texture is considered to be a primitive visual cue like the color and shape of an image. Though image retrieval using texture features is not a brand new approach, there are still scopes to enhance the retrieval accuracy with a proper representation of texture features. In this research, we aimed to obtain a high image retrieval accuracy using the multiresolution discrete curvelet texture features. The following section describes the contributions and major findings of the research.

6.2 Findings and Contributions

The main objective of this research is to investigate and evaluate an effective and robust spectral approach for texture representation and to use it in image retrieval. For this

purpose, we have investigated the texture analysis using several spectral approaches. Extensive studies are done on these spectral approaches to explain why they do not achieve satisfactory CBIR outcome. From this investigation, we find that the multiresolution is an evolving process and the time-frequency analysis is the trade off between accurate localization of information in the spatial and the spectral domain. We have described the concepts of the Fourier transform, STFT, discrete wavelet transform, and Gabor filters to provide a clearer idea about their characteristics and contrasted them to understand the evolution of MR methods. From this discussion we find that the multiresolution spectral approaches are more useful than the spatial methods in CBIR. From the investigation and study on spectral methods, we find that discrete curvelet transform represents the latest development in MR. Discrete curvelet transform has a powerful capability of representing the edge curves in an image. Among the existing methods of curvelet transform, the wrapping based discrete curvelet transform has been found to be the most effective and efficient. Therefore, we choose it to represent the image textures in this CBIR research.

Second, curvelet has been systematically described and analyzed to explore the construction of the wrapping based discrete curvelet transform. Throughout the contrast, the characteristics of the three main MR methods, i.e., discrete wavelet, Gabor filters, and discrete curvelet were discussed. Discrete curvelet transform has absorbed the advantages of both the Gabor filters and wavelet while overcomes the disadvantages of both these methods. The wrapping based discrete curvelet transform has been employed to represent the texture features of an image. From this representation, we find how effectively curvelet coefficients at each scale capture the discriminatory characteristics between images (Fig. 4.2).

Third, a better texture feature descriptor based on wrapping based curvelet transform has been created. We have employed this descriptor to represent the images in our test database. The curvelet features are then rigorously tested in a standard database. Based on the experimental results, the curvelet texture features are found to be promising. We have also performed the scale distortion tolerance test of the curvelet features on a modified database by adding distorted images to the original database. Curvelet texture features are also found to be robust to the changes in images due to scale distortion.

Finally, we compared the curvelet CBIR performance with that of the existing Gabor filters and wavelet. This is the first systematic evaluation of the image retrieval using curvelet transform. This research has found that curvelet features outperformed the existing texture features in both accuracy and efficiency. Although scale normalization has not been carried out to curvelet features, it is found that these features are robust to the reasonable scale distortions.

The main contribution of this research is the systematic analysis of curvelet and evaluation of curvelet features for CBIR. Previous applications of curvelet in CBIR are either flawed or use non-standard methodology including accuracy measurement and database selection. The result of the research is the finding of a better texture descriptor than the benchmark Gabor filters texture descriptor.

The second contribution is the systematic investigation of multiresolution spectral analysis for CBIR. The investigation gives a complete picture of the evolution of MR and shows how the MR methods differ in image content representation.

The research will help the research community to understand and implement the 3 powerful MR methods for the variety of multimedia applications. Some future plans to extend this research are discussed next.

6.3 Future Direction to Research

This research solely concentrates on using discrete curvelet texture features for CBIR. Curvelet texture features can also be used combined with color features to retrieve color images. For color features, HSV color space can be chosen as this is found to be most effective according to recent literature. Combining discrete curvelet texture features with color features may improve the retrieval outcome.

Rotation invariance is an important issue to improve the retrieval performance. Generally, it is assumed that the database images possess same dominant direction. However, in reality, an image database may contain similar but rotated images. Therefore, when we intend to use non-normalized features for retrieval, similar images with some rotation will not be captured. Finding rotation normalized curvelet features can be an important research issue.

So far, we studied the scale distortion tolerance test with scale non-normalized curvelet features. However, the effect of scale normalized curvelet features on image retrieval is not yet explored. This could be investigated to improve the CBIR performance.

We did not use any additional weights to the curvelet features. Weighted curvelet features can also be investigated to observe the CBIR performance. We found that curvelets at high frequency levels capture image edge information more effectively. Therefore, to create the feature vector, if we add some weight to the higher level curvelet texture features, the retrieval outcome may vary. This will be further investigated in future.

Curvelet Feature Extraction Code

A.1 Introduction

The curvelet features extraction code consists of two parts. The first part computes the texture features from the curvelet coefficients of each image in the Brodatz database. The second part returns the curvelet coefficients of a selected image to the first part. This part applies the wrapping based discrete curvelet transform to an image. We provide the curvelet feature extraction codes in the following sections with necessary comments to make it easier to understand.

A.2 Computation of Curvelet Texture Features

Curvelet transform returns the set of curvelet coefficients indexed by scale, orientation and location parameters. Curvelet features are computed by calculating mean and deviation of these coefficients. Code used in Matlab to generate curvelet features is shown below.

```
function create_feature_from_DB_sc4()
%This function does the following
%1. Find the names of the files from the database directory.
%2. Read each image file name from that text file and read that image.
```

```
%3. Calculate feature vector for each texture image in the database and insert those to
%another text file according to same order. Then the database images are indexed in the
%feature database.

%Setting the database path here for brodatz
pathname = 'C:\Sumana\ Brodatz_DB_mixed_scale_2\';

%Reading filename from the previously created text file and generating feature vector for
%each image
filelist = fopen('C:\Sumana\ Brodatz_DB_mixed_scale_2\Dbfilenames.txt','rt');
%previously %created list of database images.
if filelist == -1
   disp('File not opend');
   return;
end
num_of_total_files = fscanf(filelist,'%d\n',1);

fid1 = fopen ('C:\Sumana\Brodatz_DB_mixed_scale_2\Dbfeature_curvelet_4.txt','w');
% discrete curvelet feature file
for n = 1: num_of_total_files % This generates feature for each texture file
   file_name = fscanf(filelist,'%d\n',1);  %just reading to skip the first index of file
   file_name = fscanf(filelist,'%s\n',1);  %this is the filename
   imfile = strcat(pathname,file_name);  %creates the filename without single quotation.
   X = imread(imfile); %reading the file.
   C = sum_fdct_wrapping_finding_feature(double(X),0,2,4);
   %curvelet transform is done here and the coefficients are saved in the cell C.
   %Setting the finest decomposition level to wavelet (2 is the value for that), number of
   %levels to 4.

   %Finding features from the coefficients we obtain from each cell. Each cell
   corresponds
   %to the coefficients of each orientation at each scale.
   len = 2; % as the coarsest and finest scale has only one subbands, those are summed up
   for m =2: (length(C)-1)% number of times the for loop rotates, equal to length(C)-2.
      len = len + (length(C{m})/2);%calculating half of the total number of subbands in
   the
      %image. Because opposite subbands are symmetric.
   end;

   feature_vec = zeros(1,2*len); % Defining feature vector size for each image.
   avg_coeff = zeros(1, len);
   stddev_coeff = zeros(1,len);
   index = 1;

   %for scale 1 and 4 there is only one cell and we can calculate the
   %mean and standard deviation for those cells intially.
   %scale 1 C{1}{1}
   C{1}{1}(find(C{1}{1} == 0)) = eps;
   logcoeff = log(abs(C{1}{1}));
   avg = mean(logcoeff(:));
   stddev = std(logcoeff(:));
   avg_coeff(index) = avg;
   stddev_coeff(index) = stddev;

   %scale 4 C{4}{1}
```

```
    C{length(C)}{1}(find(C{length(C)}{1} == 0)) = eps;
    logcoeff = log(abs(C{length(C)}{1}));
    avg = mean(logcoeff(:));
    stddev = std(logcoeff(:));
    avg_coeff(len) = avg;
    %len is the last position where the calculated values from last cell should be saved.
    stddev_coeff(len) = stddev;

    %For other scales we have to calculate the mean and standard deviation
    %for the first half of the total cells at each scale.
    index = index+1;
    for i=2:(length(C)-1)
        for j=1:(length(C{i})/2)
            %assigning a negligible small value to the positions where C{i}{j} is absolutely
zero.
            C{i}{j}(find(C{i}{j} == 0)) = eps;
            logcoeff = log(abs(C{i}{j}));
            avg = mean(logcoeff(:));
            stddev = std(logcoeff(:));
            avg_coeff(index) = avg;
            stddev_coeff(index) = stddev;
            index = index + 1;
        end;
    end;

    feature_vec = ghty[stddev_coeff,avg_coeff ]; %all the standard deviations are put first
then
    %comes the average of each subband.
    fprintf(fid1,'%f,', feature_vec);
    fprintf(fid1,'\n');
end;
fclose(filelist);
fclose(fid1);
end
```

A.3 Computation of Curvelet Coefficients from an Image

Code used for curvelet transform in CurveLab-2.1.2 [69] is provided below.

```
function C = sum_fdct_wrapping_finding_feature(x, is_real, finest, nbscales,
nbangles_coarse)
%This function is used to calculate the coefficients at the different scales and orientations
of %an image. Basically, we call this process from the previous one to use the curvelet
%coefficients in the cells for feature extraction.

% fdct_wrapping.m - Fast Discrete Curvelet Transform via wedge wrapping - Version 1.0
%
% Inputs
%   x           M-by-N matrix
%
% Optional Inputs
%   is_real     Type of the transform
```

```
%                0: complex-valued curvelets
%                1: real-valued curvelets
%            [default set to 0]
%   finest     Chooses one of two possibilities for the coefficients at the
%            finest level:
%                1: curvelets
%                2: wavelets
%            [default set to 2]
%   nbscales    number of scales including the coarsest wavelet level
%            [default set to ceil(log2(min(M,N)) - 3)]
%   nbangles_coarse
%            number of angles at the 2nd coarsest level, minimum 8,
%            must be a multiple of 4. [default set to 16]
%
% Outputs
%   C          Cell array of curvelet coefficients.
%            C{j}{l}(k1,k2) is the coefficient at
%                - scale j: integer, from finest to coarsest scale,
%                - angle l: integer, starts at the top-left corner and
%                increases clockwise,
%                - position k1,k2: both integers, size varies with j
%                and l.
%            If is_real is 1, there are two types of curvelets,
%            'cosine' and 'sine'. For a given scale j, the 'cosine'
%            coefficients are stored in the first two quadrants (low
%            values of l), the 'sine' coefficients in the last two
%            quadrants (high values of l).
%
% See also ifdct_wrapping.m, fdct_wrapping_param.m
%
% By Laurent Demanet, 2004

X = fftshift(fft2(ifftshift(x)))/sqrt(prod(size(x)));
[N1,N2] = size(X);
if nargin < 2, is_real = 0; end;
if nargin < 3, finest = 2; end;
if nargin < 4, nbscales = ceil(log2(min(N1,N2)) - 3); end;
if nargin < 5, nbangles_coarse = 16; end;

% Initialization: data structure
nbangles = [1, nbangles_coarse .* 2.^(ceil((nbscales-(nbscales:-1:2))/2))];
if finest == 2, nbangles(nbscales) = 1; end;
C = cell(1,nbscales);
for j = 1:nbscales
   C{j} = cell(1,nbangles(j));
end;

% Loop: pyramidal scale decomposition
M1 = N1/3;
M2 = N2/3;
if finest == 1,

   % Initialization: smooth periodic extension of high frequencies
   bigN1 = 2*floor(2*M1)+1;
   bigN2 = 2*floor(2*M2)+1;
```

```
    equiv_index_1 = 1+mod(floor(N1/2)-floor(2*M1)+(1:bigN1)-1,N1);
    equiv_index_2 = 1+mod(floor(N2/2)-floor(2*M2)+(1:bigN2)-1,N2);
    X = X(equiv_index_1,equiv_index_2);
       % Invariant: equiv_index_1(floor(2*M1)+1) == (N1 + 2 - mod(N1,2))/2
       % is the center in frequency. Same for M2, N2.
    window_length_1 = floor(2*M1) - floor(M1) - 1 - (mod(N1,3)==0);
    window_length_2 = floor(2*M2) - floor(M2) - 1 - (mod(N2,3)==0);
       % Invariant: floor(M1) + floor(2*M1) == N1 - (mod(M1,3)~=0)
       % Same for M2, N2.
    coord_1 = 0:(1/window_length_1):1;
    coord_2 = 0:(1/window_length_2):1;
    [wl_1,wr_1] = fdct_wrapping_window(coord_1);
    [wl_2,wr_2] = fdct_wrapping_window(coord_2);
    lowpass_1 = [wl_1, ones(1,2*floor(M1)+1), wr_1];
    if mod(N1,3)==0, lowpass_1 = [0, lowpass_1, 0]; end;
    lowpass_2 = [wl_2, ones(1,2*floor(M2)+1), wr_2];
    if mod(N2,3)==0, lowpass_2 = [0, lowpass_2, 0]; end;
    lowpass = lowpass_1'*lowpass_2;
    Xlow = X .* lowpass;

    scales = nbscales:-1:2;

else

    M1 = M1/2;
    M2 = M2/2;
    window_length_1 = floor(2*M1) - floor(M1) - 1;
    window_length_2 = floor(2*M2) - floor(M2) - 1;
    coord_1 = 0:(1/window_length_1):1;
    coord_2 = 0:(1/window_length_2):1;
    [wl_1,wr_1] = fdct_wrapping_window(coord_1);
    [wl_2,wr_2] = fdct_wrapping_window(coord_2);
    lowpass_1 = [wl_1, ones(1,2*floor(M1)+1), wr_1];
    lowpass_2 = [wl_2, ones(1,2*floor(M2)+1), wr_2];
    lowpass = lowpass_1'*lowpass_2;
    hipass = sqrt(1 - lowpass.^2);
    Xlow_index_1 = ((-floor(2*M1)):floor(2*M1)) + ceil((N1+1)/2);
    Xlow_index_2 = ((-floor(2*M2)):floor(2*M2)) + ceil((N2+1)/2);
    Xlow = X(Xlow_index_1, Xlow_index_2) .* lowpass;
    Xhi = X;
    Xhi(Xlow_index_1, Xlow_index_2) = Xhi(Xlow_index_1, Xlow_index_2) .* hipass;
    C{nbscales}{1} = fftshift(ifft2(ifftshift(Xhi)))*sqrt(prod(size(Xhi)));

    if is_real, C{nbscales}{1} = real(C{nbscales}{1}); end;

    scales = (nbscales-1):-1:2;
end;

for j = scales,
    M1 = M1/2;
    M2 = M2/2;
    window_length_1 = floor(2*M1) - floor(M1) - 1;
    window_length_2 = floor(2*M2) - floor(M2) - 1;
    coord_1 = 0:(1/window_length_1):1;
    coord_2 = 0:(1/window_length_2):1;
```

```
[wl_1,wr_1] = fdct_wrapping_window(coord_1);
[wl_2,wr_2] = fdct_wrapping_window(coord_2);
lowpass_1 = [wl_1, ones(1,2*floor(M1)+1), wr_1];
lowpass_2 = [wl_2, ones(1,2*floor(M2)+1), wr_2];
lowpass = lowpass_1'*lowpass_2;
hipass = sqrt(1 - lowpass.^2);
Xhi = Xlow;                % size is 2*floor(4*M1)+1 - by - 2*floor(4*M2)+1
Xlow_index_1 = ((-floor(2*M1)):floor(2*M1)) + floor(4*M1) + 1;
Xlow_index_2 = ((-floor(2*M2)):floor(2*M2)) + floor(4*M2) + 1;
Xlow = Xlow(Xlow_index_1, Xlow_index_2);
Xhi(Xlow_index_1, Xlow_index_2) = Xlow .* hipass;
Xlow = Xlow .* lowpass;     % size is 2*floor(2*M1)+1 - by - 2*floor(2*M2)+1

% Loop: angular decomposition
l = 0;
nbquadrants = 2 + 2*(~is_real);
nbangles_perquad = nbangles(j)/4;

for quadrant = 1:nbquadrants
   M_horiz = M2 * (mod(quadrant,2)==1) + M1 * (mod(quadrant,2)==0);
   M_vert = M1 * (mod(quadrant,2)==1) + M2 * (mod(quadrant,2)==0);
   if mod(nbangles_perquad,2),
       wedge_ticks_left = round((0:(1/(2*nbangles_perquad)):.5)*2*floor(4*M_horiz) +
      1);
      wedge_ticks_right = 2*floor(4*M_horiz) + 2 - wedge_ticks_left;
      wedge_ticks = [wedge_ticks_left, wedge_ticks_right(end:-1:1)];
   else
      wedge_ticks_left = round((0:(1/(2*nbangles_perquad)):.5)*2*floor(4*M_horiz) +
      1);
      wedge_ticks_right = 2*floor(4*M_horiz) + 2 - wedge_ticks_left;
      wedge_ticks = [wedge_ticks_left, wedge_ticks_right((end-1):-1:1)];
   end;
   wedge_endpoints = wedge_ticks(2:2:(end-1));        % integers
   wedge_midpoints = (wedge_endpoints(1:(end-1)) + wedge_endpoints(2:end))/2;

   % Left corner wedge
   l = l+1;
   first_wedge_endpoint_vert = round(2*floor(4*M_vert)/(2*nbangles_perquad) + 1);
   length_corner_wedge = floor(4*M_vert) - floor(M_vert) +
   ceil(first_wedge_endpoint_vert/4);
   Y_corner = 1:length_corner_wedge;
   [XX,YY] = meshgrid(1:(2*floor(4*M_horiz)+1),Y_corner);
   width_wedge = wedge_endpoints(2) + wedge_endpoints(1) - 1;
   slope_wedge = (floor(4*M_horiz) + 1 - wedge_endpoints(1))/floor(4*M_vert);
   left_line = round(2 - wedge_endpoints(1) + slope_wedge*(Y_corner - 1));

   [wrapped_data, wrapped_XX, wrapped_YY] =
   deal(zeros(length_corner_wedge,width_wedge));
   first_row = floor(4*M_vert)+2-ceil((length_corner_wedge+1)/2)+...
   mod(length_corner_wedge+1,2)*(quadrant-2 == mod(quadrant-2,2));
   first_col = floor(4*M_horiz)+2-ceil((width_wedge+1)/2)+...
   mod(width_wedge+1,2)*(quadrant-3 == mod(quadrant-3,2));
         % Coordinates of the top-left corner of the wedge wrapped
         % around the origin. Some subtleties when the wedge is
         % even-sized because of the forthcoming 90 degrees rotation
```

```
for row = Y_corner   %for 3
    cols = left_line(row) + mod((0:(width_wedge-1))-(left_line(row)-
    first_col),width_wedge);
     admissible_cols = round(1/2*(cols+1+abs(cols-1)));
     new_row = 1 + mod(row - first_row, length_corner_wedge);
     wrapped_data(new_row,:) = Xhi(row,admissible_cols) .* (cols > 0);
     wrapped_XX(new_row,:) = XX(row,admissible_cols);
     wrapped_YY(new_row,:) = YY(row,admissible_cols);
end;    %end for 3
slope_wedge_right = (floor(4*M_horiz)+1 - wedge_midpoints(1))/floor(4*M_vert);
mid_line_right = wedge_midpoints(1) + slope_wedge_right*(wrapped_YY - 1);
% not integers in general
coord_right = 1/2 + floor(4*M_vert)/(wedge_endpoints(2) - wedge_endpoints(1)) *
...
(wrapped_XX - mid_line_right)./(floor(4*M_vert)+1 - wrapped_YY);
C2 = 1/(1/(2*(floor(4*M_horiz))/(wedge_endpoints(1) - 1) - 1) +
1/(2*(floor(4*M_vert))/(first_wedge_endpoint_vert - 1) - 1));
C1 = C2 / (2*(floor(4*M_vert))/(first_wedge_endpoint_vert - 1) - 1);
wrapped_XX((wrapped_XX - 1)/floor(4*M_horiz) + (wrapped_YY-
1)/floor(4*M_vert) == 2) = ...
wrapped_XX((wrapped_XX - 1)/floor(4*M_horiz) + (wrapped_YY-
1)/floor(4*M_vert) == 2) + 1;
coord_corner = C1 + C2 * ((wrapped_XX - 1)/(floor(4*M_horiz)) - (wrapped_YY -
1)/(floor(4*M_vert))) ./ ...
(2-((wrapped_XX - 1)/(floor(4*M_horiz)) + (wrapped_YY - 1)/(floor(4*M_vert))));
wl_left = fdct_wrapping_window(coord_corner);
[wl_right,wr_right] = fdct_wrapping_window(coord_right);
wrapped_data = wrapped_data .* (wl_left .* wr_right);

switch is_real
   case 0
      wrapped_data = rot90(wrapped_data,-(quadrant-1));
      C{j}{l} =
      fftshift(ifft2(ifftshift(wrapped_data)))*sqrt(prod(size(wrapped_data)));
   case 1
      wrapped_data = rot90(wrapped_data,-(quadrant-1));
      x = fftshift(ifft2(ifftshift(wrapped_data)))*sqrt(prod(size(wrapped_data)));
      C{j}{l} = sqrt(2)*real(x);
      C{j}{l+nbangles(j)/2} = sqrt(2)*imag(x);
end;

%Regular wedges
length_wedge = floor(4*M_vert) - floor(M_vert);
Y = 1:length_wedge;
first_row = floor(4*M_vert)+2-ceil((length_wedge+1)/2)+...
mod(length_wedge+1,2)*(quadrant-2 == mod(quadrant-2,2));
for subl = 2:(nbangles_perquad-1);  %for 4
   l = l+1;
   width_wedge = wedge_endpoints(subl+1) - wedge_endpoints(subl-1) + 1;
   slope_wedge = ((floor(4*M_horiz)+1) - wedge_endpoints(subl))/floor(4*M_vert);
   left_line = round(wedge_endpoints(subl-1) + slope_wedge*(Y - 1));
   [wrapped_data, wrapped_XX, wrapped_YY] =
   deal(zeros(length_wedge,width_wedge));
   first_col = floor(4*M_horiz)+2-ceil((width_wedge+1)/2)+...
   mod(width_wedge+1,2)*(quadrant-3 == mod(quadrant-3,2));
```

```
    for row = Y   %for 5
       cols = left_line(row) + mod((0:(width_wedge-1))-(left_line(row)-
       first_col),width_wedge);
       new_row = 1 + mod(row - first_row, length_wedge);
       wrapped_data(new_row,:) = Xhi(row,cols);
       wrapped_XX(new_row,:) = XX(row,cols);
       wrapped_YY(new_row,:) = YY(row,cols);
    end;    %end for 5
    slope_wedge_left = ((floor(4*M_horiz)+1) - wedge_midpoints(subl-
    1))/floor(4*M_vert);
    mid_line_left = wedge_midpoints(subl-1) + slope_wedge_left*(wrapped_YY - 1);
    coord_left = 1/2 + floor(4*M_vert)/(wedge_endpoints(subl) -
    wedge_endpoints(subl-1)) * (wrapped_XX - mid_line_left)./(floor(4*M_vert)+1 -
    wrapped_YY);
    slope_wedge_right = ((floor(4*M_horiz)+1) -
    wedge_midpoints(subl))/floor(4*M_vert);
    mid_line_right = wedge_midpoints(subl) + slope_wedge_right*(wrapped_YY -
    1);
    coord_right = 1/2 + floor(4*M_vert)/(wedge_endpoints(subl+1) -
    wedge_endpoints(subl)) * (wrapped_XX - mid_line_right)./(floor(4*M_vert)+1 -
    wrapped_YY);
    wl_left = fdct_wrapping_window(coord_left);
    [wl_right,wr_right] = fdct_wrapping_window(coord_right);
    wrapped_data = wrapped_data .* (wl_left .* wr_right);
    switch is_real
       case 0
          wrapped_data = rot90(wrapped_data,-(quadrant-1));
          C{j}{l} =
         fftshift(ifft2(ifftshift(wrapped_data)))*sqrt(prod(size(wrapped_data)));
       case 1
          wrapped_data = rot90(wrapped_data,-(quadrant-1));
          x = fftshift(ifft2(ifftshift(wrapped_data)))*sqrt(prod(size(wrapped_data)));
          C{j}{l} = sqrt(2)*real(x);
          C{j}{l+nbangles(j)/2} = sqrt(2)*imag(x);
    end;

end;

% Right corner wedge
l = l+1;
width_wedge = 4*floor(4*M_horiz) + 3 - wedge_endpoints(end) -
wedge_endpoints(end-1);
slope_wedge = ((floor(4*M_horiz)+1) - wedge_endpoints(end))/floor(4*M_vert);
left_line = round(wedge_endpoints(end-1) + slope_wedge*(Y_corner - 1));
[wrapped_data, wrapped_XX, wrapped_YY] =
deal(zeros(length_corner_wedge,width_wedge));
first_row = floor(4*M_vert)+2-ceil((length_corner_wedge+1)/2)+...
mod(length_corner_wedge+1,2)*(quadrant-2 == mod(quadrant-2,2));
first_col = floor(4*M_horiz)+2-ceil((width_wedge+1)/2)+...
mod(width_wedge+1,2)*(quadrant-3 == mod(quadrant-3,2));
for row = Y_corner   %for 6
    cols = left_line(row) + mod((0:(width_wedge-1))-(left_line(row)-
    first_col),width_wedge);
    admissible_cols = round(1/2*(cols+2*floor(4*M_horiz)+1-abs(cols-
    (2*floor(4*M_horiz)+1))));
```

```
        new_row = 1 + mod(row - first_row, length_corner_wedge);
        wrapped_data(new_row,:) = Xhi(row,admissible_cols) .* (cols <=
        (2*floor(4*M_horiz)+1));
        wrapped_XX(new_row,:) = XX(row,admissible_cols);
        wrapped_YY(new_row,:) = YY(row,admissible_cols);
      end;    %end for 6
      slope_wedge_left = ((floor(4*M_horiz)+1) -
      wedge_midpoints(end))/floor(4*M_vert);
      mid_line_left = wedge_midpoints(end) + slope_wedge_left*(wrapped_YY - 1);
      coord_left = 1/2 + floor(4*M_vert)/(wedge_endpoints(end) - wedge_endpoints(end-
      1)) * …
      (wrapped_XX - mid_line_left)./(floor(4*M_vert) + 1 - wrapped_YY);
      C2 = -1/(2*(floor(4*M_horiz))/(wedge_endpoints(end) - 1) - 1 +
      1/(2*(floor(4*M_vert))/(first_wedge_endpoint_vert - 1) - 1));
      C1 = -C2 * (2*(floor(4*M_horiz))/(wedge_endpoints(end) - 1) - 1);
      wrapped_XX((wrapped_XX - 1)/floor(4*M_horiz) == (wrapped_YY -
      1)/floor(4*M_vert)) = ...
      wrapped_XX((wrapped_XX - 1)/floor(4*M_horiz) == (wrapped_YY -
      1)/floor(4*M_vert)) - 1;
       coord_corner = C1 + C2 * (2-((wrapped_XX - 1)/(floor(4*M_horiz)) +
      (wrapped_YY - 1)/(floor(4*M_vert)))) ./ ...
      ((wrapped_XX - 1)/(floor(4*M_horiz)) - (wrapped_YY - 1)/(floor(4*M_vert)));
      wl_left = fdct_wrapping_window(coord_left);
      [wl_right,wr_right] = fdct_wrapping_window(coord_corner);

      wrapped_data = wrapped_data .* (wl_left .* wr_right);
      switch is_real
        case 0
          wrapped_data = rot90(wrapped_data,-(quadrant-1));
          C{j}{l} =
          fftshift(ifft2(ifftshift(wrapped_data)))*sqrt(prod(size(wrapped_data)));
        case 1
          wrapped_data = rot90(wrapped_data,-(quadrant-1));
          x = fftshift(ifft2(ifftshift(wrapped_data)))*sqrt(prod(size(wrapped_data)));
          C{j}{l} = sqrt(2)*real(x);
          C{j}{l+nbangles(j)/2} = sqrt(2)*imag(x);
      end;
      if quadrant < nbquadrants, Xhi = rot90(Xhi); end;

    end;
end;

% The Coarsest level
C{1}{1} = fftshift(ifft2(ifftshift(Xlow)))*sqrt(prod(size(Xlow)));

if is_real == 1,
    C{1}{1} = real(C{1}{1});
end;


function [wl,wr] = fdct_wrapping_window(x)

% fdct_wrapping_window.m - Creates the two halves of a C^inf compactly supported
%window
%
```

```
% Inputs
%   x       vector or matrix of abscissae, the relevant ones from 0 to 1
%
% Outputs
%   wl,wr   vector or matrix containing samples of the left, resp. right
%           half of the window
%
% Used at least in fdct_wrapping.m and ifdct_wrapping.m
%
% By Laurent Demanet, 2004

wr = zeros(size(x));
wl = zeros(size(x));
x(abs(x) < 2^-52) = 0;
wr((x > 0) & (x < 1)) = exp(1-1./(1-exp(1-1./x((x > 0) & (x < 1)))));
wr(x <= 0) = 1;
wl((x > 0) & (x < 1)) = exp(1-1./(1-exp(1-1./(1-x((x > 0) & (x < 1))))));
wl(x >= 1) = 1;
normalization = sqrt(wl.^2 + wr.^2);
wr = wr ./ normalization;
wl = wl ./ normalization;
```

Bibliography

[1] F. Long, H. J. Zhang, and D. D. Feng, "Fundamentals of Content-based Image Retrieval," in *Multimedia Information Retrieval and Management, D. Feng Eds, Springer*, 2003.

[2] accessed http://au.images.search.yahoo.com/ on 09 November, 2008.

[3] "Content-based image retrieval," http://en.wikipedia.org/wiki/CBIR.

[4] H. Tamura, S. Mori, and T.Yamawaki, "Texture Features Corresponding to Visual Perception," *IEEE Trans. on Systems, Man and Cybernetics,* vol. 8(6), pp. 460-473, 1978.

[5] Y. L. Huang, "A Fast Method for Textural Analysis of DCT-Based Image," *Journal of Information Science and Engineering,* vol. 21(1), pp. 181-194, 2005.

[6] L. Chen, G. Lu, and D. S. Zhang, "Effects of Different Gabor Filter Parameters on Image Retrieval by Texture," in *Proc. of IEEE 10th International Conference on Multi-Media Modelling*, Australia, 2004, pp. 273-278.

[7] B. S. Manjunath and W. Y. Ma, "Texture Features for Browsing and Retrieval of Image Data," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 18(8), pp. 837-842, 1996.

[8] B. S. Manjunath et al, "Introduction to MPEG-7," John Wiley & Son Ltd., 2002.

[9] J. Z. Wang, J. Li, and G. Wiederhold, "SIMPLIcity: Semantics-sensitive Integrated Matching for Picture Libraries," *IEEE Trans. Pattern and Machine Intelligence,* vol. 23(9), pp. 947-963, 2001.

[10] S. Bhagavathy and K. Chhabra, "A Wavelet-based Image Retrieval System," in *Technical Report—ECE278A*: Vision Research Laboratory, University of California,Santa Barbara, 2007.

[11] N. Suematsu, Y. Ishida, A. Hayashi, and T. Kanbara, "Region-Based Image Retrieval using Wavelet Transform," in *Proc. 15th International Conf. on Vision Interface*, Calgary, Canada, May, 2002, pp. 9-16.

[12] E. J. Candès and D. L. Donoho, "Curvelets - a surprisingly effective nonadaptive representation for objects with edges," in *Curve and Surface Fitting: Saint-Malo*, A. Cohen, C. Rabut, and L. L. Schumaker, Eds. Nashville, TN: Vanderbilt University Press, 1999.

[13] J.-L. Starck and M. J. Fadili, "Numerical Issues When Using Wavelets," *Encyclopedia of Complexity and System Science , in press.*

[14] J. G. Daugman, "Complete Discrete 2-D Gabor Transforms by Neural Networks for Image Analysis and Compression," *IEEE Trans. Acoust. Speech, Signal Processing,* vol. 36(7), pp. 1169-1179, July, 1988.

[15] D. Zhang, A. Wong, M. Indrawan, and G. Lu, "Content-based Image Retrieval Using Gabor Texture Features," in *Proc. of First IEEE Pacific-Rim Conference on Multimedia (PCM'00)*, Sydney, Australia, December 13-15 , 2000, pp. 1139-1142.

[16] W.-Y. Ma and H. Zhang, "Benchmarking of Image Features for Content-based Retrieval," in *Signals, Systems & Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference*, Pacific Grove, CA, USA., 1998, pp. 253-257.

[17] M. J. Fadili and J.-L. Starck, "Curvelets and Ridgelets," *Encyclopedia of Complexity and System Science , in press.,* 2007.

[18] E. J. Candès, L. Demanet, D. L. Donoho, and L. Ying, "Fast Discrete Curvelet Transforms," *Multiscale Modeling and Simulation,* vol. 5, pp. 861-899, 2005.

[19] J.-L. Starck, E. J. Candès, and D. L. Donoho, "The Curvelet Transform for Image Denoising," *IEEE Transactions on Image Processing,* vol. 11(6), pp. 670-684, 2002.

[20] A. Majumdar, "Bangla Basic Character Recognition Using Digital Curvelet Transform," *Journal of Pattern Recognition Research,* vol. 1, pp. 17-26, 2007.

[21] G. Joutel et al., "Curvelets Based Feature Extraction of handwritten shapes for ancient manuscripts classification," in *Proc. of SPIE-IS&T Electronic Imaging, SPIE*, 2007.

[22] http://www.corel.com.

[23] P. Brodatz, "Textures: A Photographic Album for Artists and Designers," Dover Publications, 1966.

[24] J. Zhang and T. Tan, "Brief Review of Invariant Texture Analysis Methods," *Pattern Recognition,* vol. 35, pp. 735–747, 2002.

[25] B. S. Manjunath, P. Wu, S. Newsam, and H. D. Shin, "A Texture Descriptor for Browsing and Similarity Retrieval," *Signal Processing: Image Communication,* vol. 16(1-2), pp. 33-43, Sep. 2000.

[26] W. K. Leow and S. Y. Lai, "Scale and Orientation-Invariant Texture Matching for Image Retrieval," in *Texture Analysis in Machine Vision*, M. K. Pietikainen, Ed.: World Scientific, 2000, pp. 151-164.

[27] C.-Y. Chiu, H.-C. Lin, and S.-N. Yang, "Texture Retrieval with Linguistic Descriptors," in *Proc. of IEEE Pacific Rim Conf. on Multimedia*, 2001, pp. 308-315.

[28] W. Y. Ma and B. S. Manjunath, "A Comparison of Wavelet Transform Features for Texture Image Annotation," in *IEEE International Conference on Image Processing (ICIP)*, 1995, pp. 256-259.

[29] K. L. Lee and L. H. Chen, "A New Method for Coarse Classification of Textures and Class Weight Estimation for Texture Retrieval," *Pattern Recognition and Image Analysis,* vol. 12, No. 4, pp. 400–410, 2002.

[30] Y. Li, M. J. Kyan, and L. Guan, "Improving Shape-Based CBIR for Natural Image Content Using a Modified GFD," in *2nd International Conference on Image Analysis and Recognition*, Toronto, Canada, 2005, pp. 593-600.

[31] W. Y. Ma and B. S. Manjunath, "Netra: A toolbox for navigating large image databases," *Multimedia Systems,* vol. 7, no. 3, pp. 184-198, 1999.

[32] J. R. Smith and S.-F. Chang, "VisualSEEk: A fully automated content-based image query system," in *Proc. of ACM Multimedia*, 1996.

[33] Y. Song, W. Wang, and A. Zhang, "Automatic Annotation and Retrieval of Images." vol. 6, no.2: World Wide Web: Internet and Web Information Systems, 2003, pp. 209-231.

[34] Z. Lu, S. Li, and H. Burkhardt, "A Content-Based Image Retrieval Scheme in JPEG Compressed Domain," *International Journal of Innovative Computing, Information and Control,* vol. 2(4), pp. 831-839, 2006.

[35] Y. Rui, T. S. Huang, and S. F. Chang, "Image Retrieval: Current Techniques, Promising Directions, and Open Issues," *Journal of Visual Communication and Image Representation,* vol. 10(4), pp. 39-62, 1999.

[36] P. S. Hiremath and J. Pujari, "Content Based Image Retrieval using Color, Texture and Shape features," in *15th International Conference on Advanced Computing and Communications (ADCOM)*, 2007, pp. 780-784.

[37] Z. Xue, S. Antani, L. R. Long, J. Jeronimo, and G. R. Thoma, "Investigating CBIR Techniques for Cervicographic Images," in *Proceedings of the 2007 Annual Symposium of the American Medical Information Association (AMIA)*, Chicago, IL, 2007, pp. 826-830.

[38] Z. Katol, T.-C. Pong, and S. G. Qiang, "Unsupervised Segmentation of Color Textured Images Using a Multi-layer MRF Model," in *Proceedings of International Conference on Image Processing (ICIP)*, September 2003, pp. 961-964.

[39] J. D. Foley, A. V. Dam, S. K. Feiner, and J. F. Hughes, "Computer graphics: principles and practice," 2nd ed, M. Reading, Addison-Wesley, Ed., 1990.

[40] A. K. Jain, "Fundamentals of Digital Image Processing," E. Cliffs, Ed.: Prentice Hall, 1989.

[41] H. Kauppinen, T. Seppanen, and M. Pietikainen, "An Experimental Comparison of Autoregressive and Fourier-Based Descriptors in 2D Shape Classification," *IEEE Transaction on Pattern Analysis and Machine Intelligence,* vol. 17, no. 2,, pp. 201-207, 1995.

[42] S. G. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 11, no. 7, pp. 674-693, July 1989.

[43] A. K. Jain and F. Farroknia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition,* vol. 24, No.12, pp. 1167-1186, 1991.

[44] J. Mao and A. K. Jain, "Texture Classification and Segmentation Using Multiresolution Simultaneous Autoregressive Models," *Pattern Recognition,* vol. 25, No. 2, pp. 173-188, 1992.

[45] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, Digital Image Processing Using MATLAB: Pearson Prentice Hall, 2004.

[46] http://web.quick.cz/elaop/Wteory/WveletTutorRobi/THE%20WAVELET%20TUTORIAL%20PART%20I%20by%20ROBI%20POLIKAR.htm.

[47] Y. Li, X. Chen, X. Fu, and S. Belkasim, "Multi-Level Discrete Cosine Transform for Content-Based Image Retrieval by Support Vector Machines," in *International Conference on Image Processing (ICIP)*, 2007, pp. 21-24.

[48] C. W. Ngo, T. C. Pong, and R. T. Chin, "Exploiting Image Indexing Techniques in DCT Domain," *Pattern Recognition,* vol. 34(9), pp. 1841-1851, September 2001.

[49] http://web.quick.cz/elaop/Wteory/WveletTutorRobi/THE%20WAVELET%20TUTORIAL%20PART%20II%20by%20ROBI%20POLIKAR.htm.

[50] S. Chikkerur, V. Govindaraju, and A. N. Cartwright, *Fingerprint Image Enhancement Using STFT Analysis* vol. 3687: Springer Berlin, 2005.

[51] "Short-time Fourier transform," http://en.wikipedia.org/wiki/Short-time_Fourier_transform.

[52] M. Vetterli, *Wavelet Transforms and Applications to Signal Processing and Communications*: AVIPAC Workshop in Faculty of Information Technology, Monash University, Melbourne Australia., March 6, 2003.

[53] "THE WAVELET TUTORIAL PART III," http://web.quick.cz/elaop/Wteory/WveletTutorRobi/THE%20WAVELET%20TUTORIAL%20PART%20III%20by%20ROBI%20POLIKAR.htm.

[54] E. J. Stollnitz, T. D. DeRose, and D. H. Salesin, "Wavelets for Computer Graphics Theory and Application," Morgan Kaufmann Publisher, Inc. San Francisco, California, 1996.

[55] "Simplegabor - Multiresolution Gabor Feature Toolbox," http://www.it.lut.fi/project/simplegabor/downloads/src/simplegabortb/.

[56] S. Y. Lai and W. K. Leow, "Invariant Texture Matching for Content-based Image Retrieval " in *Proc. Int. Conf. on Multimedia Modeling MMM '97*, Singapore, November 1997.

[57] M. U. Munir and M. Y. Javed, "Fingerprint Matching using Gabor Filters," in *National Conference on Emerging Technologies (ACM-IEEE)*, 2004.

[58] P. Lambert, S. Pires, J. Ballot, R. A. García, J.-L. Starck, and S. Turck-Chièze, "Curvelet analysis of asteroseismic data. I. Method description and application to simulated sun-like stars," *Astronomy and Astrophysics* pp. 1021-1027, 2006.

[59] J. R. Smith and S. F. Chang, "Transform Features for Texture Classification and Discrimination in Large Image Databases," in *International Conference on Image Processing (ICIP)*, 1994, pp. 407-411.

[60] K. S. Thyagarajan, T. Nguyen, and C. Persons, "A Maximum Likelihood Aproach to Texture Classification Using Wavelet Transform," in *Proc. IEEE Int. Conf. on Image Proc*, 1994.

[61] J.-L. Starck, M. Elad, and D. L. Donoho, "Image Decomposition Via The Combination of Sparse Rrepresentations and a Variational Approach," in *IEEE Transactions on Image Processing*, Oct. 2005, pp. 1570-1582.

[62] J.-L. Starck, M. K. Nguyen, and F. Murtagh. . , 2003., "Wavelets and Curvelets for Image Deconvolution: a Combined Approach," *Signal Processing,* vol. 83, pp. 2279-2283, October 2003.

[63] H. Yuan, X.-P. Zhang, and L. Guan, "A Statistical Approach For Image Feature Extraction in the Wavelet Domain," in *IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, Montreal, Canada, May 2003.

[64] E. J. Candès and D. L. Donoho, "Ridgelets: a key to higher-dimensional intermittency?," *Philosophical Transactions of the Royal Society of London. A.,* vol. 357, pp. 2495–2509, 1999.

[65] S. Arivazhagan, L. Ganesan, and T. G. S. Kumar, "Texture classification using Curvelet Statistical and Co-occurrence Features," in *The 18th International Conference on Pattern Recognition (ICPR'06)*, 2006.

[66] J.-L. Starck, D.L.Donoho, and E.J.Candes, "Astronomical Image Representation by the Curvelet Transform. . ()." *Astronomy & Astrophysics, 398,* pp. 785-800, 1999.

[67] J. Starck, F. Murtagh, E. J. Candès, and D. L. Donoho, "Gray and Color Image Contrast enhancement by the Curvelet Transform," *IEEE Transactions on Image Processing,* vol. 12(6), pp. 706-717, 2003.

[68] M. N. Do, "Directional Multiresolution Image Representations," PhD thesis, EPFL, 2001.

[69] http://www.curvelet.org/software.html.

[70] G. Joutel, V. Eglin, S. Bres, and H. Emptoz, "Curvelets based queries for CBIR Application in handwriting collections," in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, 2007, pp. 649-653.

[71] L. Ni and H. C. Leng, "Curvelet Transform and Its Application in Image Retrieval," in *3rd International Symposium on Multispectral Image Processing and Pattern Recognition Proceedings of SPIE*, 2003.

[72] L. Semler and L. Dettori, "Curvelet-based Texture Classification of Tissues in Computed Tomography," in *IEEE Proceedings of the International Conference on Image Processing (ICIP)*, Atlanta, Georgia, USA, October 2006, pp. 2165-2168.

[73] I. Sumana, M. Islam, D. S. Zhang and G. Lu, "Content Based Image Retrieval Using Curvelet Transform," a camera ready presentation In Proc. of IEEE International Workshop on Multimedia Signal Processing (MMSP08), Cairns, Queensland, Australia, ISBN 978-1-4244-2295-1, pp.11-16, October 8-10, 2008.