



Frequency Based Texture Feature Descriptors

A thesis submitted to the Auckland University of Technology,

In fulfilment of the requirements for the degree of

Doctor of Philosophy (PhD)

Xinyu Hu

*School of Engineering, Computer and Mathematical
Sciences*

2017

Primary Supervisor: Dr. Andrew Ensor

Secondary Supervisor: Associate Professor Wei Qi Yan

Table of Contents

| | |
|--|----|
| Attestation of Authorship | 1 |
| Acknowledgements | 2 |
| Abstract | 3 |
| Chapter 1 Introduction | 4 |
| Chapter 2 Literature / Past Research Review | 10 |
| 2.1 Edge Detection | 10 |
| 2.1.1 Prewitt and Sobel Operators | 10 |
| 2.1.2 Hough Transform | 11 |
| 2.1.3 Canny Edge Detection | 13 |
| 2.2 Basic Geometrical Shape Recognition | 15 |
| 2.2.1 Hough Transform for Circles Recognition | 15 |
| 2.2.2 Proportion Based Shape Recognition | 15 |
| 2.3 Contour Based Object Matching and Recognition | 16 |
| 2.3.1 Shape Context Descriptors | 16 |
| 2.3.2 Hu Moments | 17 |
| 2.3.3 Colour Based Object Recognition | 19 |
| 2.3.4 Machine Learning Approach | 20 |
| 2.4 Texture Analysis | 22 |
| 2.4.1 Gray-Level Co-occurrence Matrix (GLCM) | 22 |
| 2.4.2 Local Binary Patterns (LBP) | 25 |
| 2.4.3 Tamura's Texture Features | 25 |
| 2.4.3.1 Coarseness | 26 |
| 2.4.3.2 Contrast | 26 |
| 2.4.3.3 Directionality | 26 |
| 2.4.3.4 Linelikeness | 27 |
| 2.4.3.5 Regularity | 27 |
| 2.4.3.6 Roughness | 28 |

| | |
|---|-----------|
| 2.5 Fourier Transform ----- | 29 |
| 2.5.1 Fourier Spectrum and Fourier Phase Angle ----- | 29 |
| 2.5.2 Aliasing and Leakage ----- | 32 |
| 2.5.3 Fourier Spectrum for Variance Images ----- | 33 |
| 2.5.3.1 Direct Current of Spectrum Image----- | 33 |
| 2.5.3.2 Spectrum of Different Pattern Image ----- | 33 |
| 2.5.4 Basic Applications of Fourier Transform ----- | 33 |
| 2.5.4.1 Fourier Spectrum and Contrast ----- | 35 |
| 2.5.4.2 Fourier Spectrum and Edge Detection ----- | 35 |
| 2.5.4.3 Fourier Spectrum and Image Smoothing and Sharpening ----- | 36 |
| 2.5.4.4 Fourier Spectrum and Notch Filters ----- | 37 |
| 2.5.5 Advanced Applications of Fourier Spectrum (Blur and Deblur)----- | 38 |
| 2.5.6 Fourier Spectrum Descriptors ----- | 40 |
| Chapter 3 Image Preparation and Frequency Domain Texture Descriptors ----- | 42 |
| 3.1 Shadow Removal ----- | 42 |
| 3.2 Segmentation ----- | 46 |
| 3.2.1 Issues of Spatial Domain Segmentation ----- | 46 |
| 3.2.2 Frequency Domain Segmentation ----- | 46 |
| Chapter 4 Fourier Transform and Spectrum Pattern Enhancement ----- | 51 |
| 4.1 Windows Functions Selection ----- | 51 |
| 4.1.1 Discrete Fourier Transform Issues and Solutions ----- | 51 |
| 4.1.2 Candidate Test ----- | 53 |
| 4.1.3 Forming Two-Dimension Test ----- | 56 |
| 4.2 Enhance Spectrum Pattern ----- | 63 |
| 4.2.1 Power-Law Transformations and Cosine Function ----- | 63 |
| Chapter 5 Texture Spectrum Feature Descriptors ----- | 67 |
| 5.1 Hu Moments Descriptors ----- | 67 |
| 5.2 Tamura's Texture Features and Spectrum Texture Features ----- | 73 |
| 5.2.1 Coarseness Descriptor on Texture Pattern Size ----- | 73 |
| 5.2.2 Calculation of Texture Pattern Size ----- | 78 |

| | |
|---|------------|
| 5.2.3 Contrast ----- | 79 |
| 5.2.4 Linelikeness ----- | 81 |
| 5.2.5 Texture pattern Edge Directional Unification Value----- | 86 |
| 5.2.6 Directionality ----- | 88 |
| 5.2.7 Spectrum Bins Approach Versus Tamura Directionality ----- | 91 |
| 5.3 Texture Pattern Formation ----- | 93 |
| 5.3.1 Spectrum Pattern Texture ----- | 93 |
| 5.3.2 Finding Formation Information by Spectrum Pattern Texture ----- | 96 |
| Chapter 6 Testing and Results ----- | 105 |
| 6.1 Linelikeness Test ----- | 105 |
| 6.2 Direction Unification Value Test ----- | 112 |
| 6.3 Directionality Test ----- | 116 |
| 6.4 Texture Pattern Formation Test ----- | 124 |
| Chapter 7 Conclusion ----- | 129 |
| 7.1 Key Achievements and Limitations----- | 129 |
| 7.1.1 Texture Pattern Size----- | 129 |
| 7.1.2 Linelikeness----- | 130 |
| 7.1.3 Direction Unification----- | 131 |
| 7.1.4 Directionality----- | 132 |
| 7.1.5 Texture Pattern Formation----- | 132 |
| 7.1.6 Hu Moments----- | 133 |
| 7.1.7 Image Pre-processing----- | 133 |
| 7.1.8 Overview of Frequency-Based Approach----- | 134 |
| 7.2 Further Research----- | 136 |
| Reference List ----- | 129 |

Attestation of Authorship

“I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.”

Signed: _____

Acknowledgements

Firstly, I would like to say thank you to my primary supervisor Dr. Andrew Ensor. It would be an impossible mission for me to complete the research without his guidance, wisdom and kindness. Without his help I may still trying to reduce Fourier Transform leakage problems and be having trouble with the accuracy of the feature descriptors. I would also like to thank my secondary supervisor Associate Professor Wei Qi Yan, he suggested Tamura Texture Features as a comparative algorithm, and all the AUT staff who helped me during my research. I am grateful for receiving an AUT School of Engineering, Computer & Mathematical Sciences fees scholarship. I would like to thank my family and friends. Thanks for their support and encouraging me throughout my PhD studies.

Abstract

A texture descriptor is a collection of quantified measurements of a texture's properties. They are often used for classifying texture images and recognition of objects in an image that have repeated patterns. In the past, quantified measures have often been calculated in the spatial domain based on the texture pattern, and include Coarseness, Linelikeness and Directionality. These are the local texture features which describe the basic unit of texture that is repeated, whereas global texture features describe how the unit of texture is repeated throughout the image.

This research proposes a new approach for finding descriptors of local and global texture features by calculating them in the frequency domain. The research takes advantage of properties of the Fourier spectrum to introduce a Directional Unification descriptor, which measures how much the lines in a pattern are in the same direction and Texture Pattern Formation feature descriptors, which describe how a pattern repeats spatially within a texture.

In order to discover the advantages and the limitations of the texture feature descriptors in frequency domain, this research uses an experimental methodology. Standard testing images are used to compare between the spatially-based approaches and the frequency-based approaches. Additional images were included to facilitate the investigation of specific texture features. This testing demonstrated and that the new approach provides translation invariance to the descriptors, they are less affected by image intensity and image clarity.

Chapter 1

Introduction

Webcams, security cameras, mobile phone cameras, and vision equipment for moving vehicles have become pervasive in today's society. These cameras are connected to computers to process the captured images. The field of Computer Vision is concerned with image acquisition, processing, and analysis of images obtained by such cameras [1], and often relies on image restoration [2][3], object recognition [4][5] and motion estimation techniques in real-time [6][7].

One issue with many object recognition algorithms is their requirement that an entire object be captured within view so the algorithm can be applied to the images. In real life, especially in uncontrolled environments with some objects occluding others, it can be difficult to capture an image of an entirely unobstructed object.

However, some objects have distinctive texture patterns. These texture patterns provide some information about an object and can potentially be used as a signature of the object. More important, these texture patterns can even be recognised from partially obscured objects. Therefore texture features can be used for recognition of some objects even in uncontrolled environments. In order to describe and compare these features, numerical descriptors of the texture patterns are involved.

Most algorithms for finding numerical descriptors use spatially-based approaches, such as the Gray-level co-occurrence matrix (GLCM see more discussion in Section 2.4.1) algorithm and the Tamura Texture Features (see more discussion in Section 3.3.2.7). A significant problem with most spatially-based approaches is that they deal with the texture image pixels in the spatial domain and focus only on a texture's local features which describe the basic unit of texture that is repeated, such as coarseness. These features do not show the relation between the texture units and effect of the

texture units on the entire texture image. In other words, they are not adequate for describing global features of a texture image such as how the texture units are repeated.

Furthermore, the selection of appropriate window size can be difficult for spatially-based approaches. Tamura uses six windows with different sizes. This increases the calculation cost. The resolution technologies of cameras have been developing significantly, a 64×64 window may not even cover an entire texture pattern. Although GLCM does not apply windows, it has to go through an entire image to build a 255×255 neighborhood matrix. The neighboring pixels may not always only contain texture information, but could be heavily influenced by noise. This may affect the final results of the feature descriptors.

The dominating signal of a spectrum image is located at the center of the spectrum and gets weak toward the edge of an image. This possibly provides ability of analysis texture pattern details on different frequency levels in frequency-based multiplayer. Although the Fourier transform has been used extensively for other purposes in image processing (see Section 2.5.4 and Section 2.5.5), not a lot of literature has been found that applies it to calculate texture features especially on the global features of textured images. No literature discusses and compares the performance of obtaining local features and global features both in spatial domain and frequency domain. It could be important to investigate and compare spatial-based approach and frequency-based approach to find which is better for local texture feature descriptors and which is better for global texture feature descriptors.

This thesis utilises the Fourier transform to build the spectrum of a texture and obtains both local and global texture features descriptors. Based on the testing results, the thesis discusses the ability to get both local and global texture features descriptors in the frequency domain to compare with these feature descriptors in the spatial domain. This is important for choosing an appropriate approach to get better texture feature

descriptors. In addition, two new global features descriptors are introduced.

This research has been designed to utilise three phases: Texture Pattern Segmentation, Fourier Transform and Texture Features Recognition. Segmentation is an important task in object texture recognition. It subdivides an image into its constituent regions or objects [8]. The level of detail to subdivide depends on the problem to be solved. Existing algorithms for segmentation and object recognition have two issues as described below.

Background subtraction is a very basic and key algorithm for segmentation [9] [10]. This method of segmentation subtracts the image with both the background and the object of interest by the background image. It takes multiple photos, and then compares each corresponding pixel from two consecutive images. The pixels are explored as soon as they are different from the corresponding pixels in the previous image. These explored pixels are segmentation as a foreground object from the background. However the result includes not only the objects that have moved into view but also includes parts of the background that are in movement [8].

Colour can also be used to perform segmentation (see more details in Section 2.3.3). However if the background contains the same colour as the object of interest, the results of segmentation will be affected. Furthermore, if the foreground object contains multiple colours, then segmentation by colour will be harder to implement.

This research aims to overcome these existing issues by introducing new algorithms that utilise frequency domain image information to achieve segmentation. Frequency domain segmentation applies the Fourier Transform linearity law to decompose a Fourier spectrum into its principal spectra components. Frequency domain decomposition takes advantage of manipulating the energy in the spectrum to segment the texture and is invariant to translations of background objects. This is discussed in Section 3.2.

One of the problems of using the Fourier transform is that an image cannot be infinite and texture patterns have finite extent in images. These cause frequency leakage and extra-unexpected lines in the spectrum image (see more discussion in Section 2.5.2). In order to overcome these issues, this research investigates and compares higher-order generalized cosine window functions such as Blackman window function; generalized Hamming window function; adjustable window functions such as Kaiser window function and finite impulse response (FIR) filter designed Sinc function (domain is $[0, \pi]$). These window functions are all one-dimensional functions. Selecting an appropriate way of converting them to two-dimensional function is one of the tasks of this research. This task includes exploring and comparing three candidate conversion algorithms to find the best way of building two dimensional window functions in order to reduce the leakage problem.

The spectrum image contains both low and high frequency components. In order to find the global features of a texture, this research also will discuss and apply approaches to reduce the intensity of the high frequency signals and increase the low frequency signal intensities.

The texture descriptor algorithms developed in this thesis utilise a spectrum bin approach. The spectrum bin approach basically is designed by the polar coordinates in the spectrum image, using directionality and distance. The outcomes of the bin approach are frequency domain texture features descriptors. These descriptors include texture local features (Pattern Size, Linelikeness and Directionality) and two new global feature descriptors (pattern Directional Unification Value and Texture Pattern Formation features).

The pattern Directional Unification measures whether the linear texture patterns have a similar Directionality. This further classifies the linear texture patterns and provides measurement of how useful the Directionality values are. If a texture has low

Directional Unification value, this means the texture has varied directionalities and the directionalities may not be an important feature for the texture. Examples of directly applying this feature that could be further developed for distinguishing the difference between a railway line and highway from satellite images.

Texture pattern formation features describe how texture patterns repeat, such as whether a pattern repeats in a linear direction and if so in which direction the pattern repeats. They may provide valuable information about a texture although are not included on Tamura's Texture Features list.

There are still a lot of challenges of extracting the feature descriptors from a texture image. Due to the attributes of the spectrum images low frequency components often have high intensity and the high frequency part often has low intensity. Hu moments may not distinguish these similar spectrum images. Furthermore, most articles say that the large shape in spatial domain can have small pattern in frequency domain. However there is no verification to show that a small pattern in frequency domain must because of a large pattern in spatial domain. It can also be a small pattern in frequency domain but lower contrast. Most articles which discuss the relationship between spectrum Pattern Size and the Texture Pattern Size, are all based on the same contrast.

The experimental methodology is applied for this research. The motivation of this research is on personal interest. After reviewing past work that is particularly relevant to this thesis, some new texture feature descriptors not part of Tamura and GLCM are found, particularly global texture features. This research aims to complement those local texture features and to find suitability of a frequency domain approach. The approach has three phases. The third chapter, which is the first phase, contains two parts. In order to have texture images without a shadow effect, the first part introduces a shadow-removing algorithm. The second part introduces and compares two new frequency-based image segmentation algorithms. Chapter four, which is the second

phase, discusses and compares different window functions to reduce leakage. Chapter five describes new algorithms for calculating the texture feature descriptors in the frequency domain. Chapter six tests a set of images from a standard image database and to find how results compare between frequency domain and spatial domain approaches. In the last chapter, this research concludes which roles for frequency domain versus spatial domain approaches are more suitable for extracting local texture features or global texture features.

In further development, the potential abilities of global texture features will be further explored and investigated. Such as descriptors for the distance between each pattern unit, or the variance of pattern unit size.

Although this research focuses on texture feature descriptors, it is readily applicable to help improve texture classification and object recognition techniques. It has many potential practical applications, such as object identification, automated navigation, visually impaired assistance, and surveillance.

Chapter 2

Literature/Past Research Review

Digital images were first used in the newspaper industry and were transmitted electronically from London to New York in the 1920s [8]. Digital image processing grew rapidly with the introduction of digital signal processors in 1980. Computer vision transforms visual data, from still images or videos, to make a decision or a new representation. Object recognition is an important part of computer vision [11]. It includes edge detection, geometrical shape recognition or more general contour-based object matching and recognition, colour-based object recognition and feature-based categorization.

This research takes advantage of the Fourier transform to convert an image to the frequency domain and applies image processing techniques for textured object recognition. Edge detection, basic geometrical shape recognition and colour-based object recognition techniques provide the historical context. Contour-based object recognition, object matching and recognition, machine learning, texture analysis and Fourier Transforms are directly relevant for this research.

2.1 Edge detection

Edge detection is one of the foundations of object recognition. This topic has been extensively researched [8][12]. Edges are defined as an abrupt change in the same colour channel, which are used as the object boundaries in an image [8][12].

2.1.1 Prewitt and Sobel Operators

Most edge detection algorithms find edges by applying a convolution filter. In 1975 Fram and Deutsch published an article, which contained a number of masks and

discussed their performance [13]. The most common masks are Prewitt and Sobel. Figures 2.1 and 2.2 show these 3×3 masks used for line and edge detection [8][14].

| | | | | | |
|----|----|----|----|---|---|
| -1 | -1 | -1 | -1 | 0 | 1 |
| 0 | 0 | 0 | -1 | 0 | 1 |
| 1 | 1 | 1 | -1 | 0 | 1 |

Figure 2.1 Prewitt operators

| | | | | | |
|----|----|----|----|---|---|
| -1 | -2 | -1 | -1 | 0 | 1 |
| 0 | 0 | 0 | -2 | 0 | 2 |
| 1 | 2 | 1 | -1 | 0 | 1 |

Figure 2.2 Sobel operators

These masks are applied at each pixel in an image.

2.1.2 Hough Transform

In 1962, Hough proposed a quite different way to find lines in an image, which was known as the Hough Transform [8][15]. It considers a point (x, y) and a line $y = ax + b$. This formula can also be written as $b = -xa + y$. This converts the xy plane to the ab plane. The problem of using $y = ax + b$ to describe a line is that neither slope nor y -intercept is bounded. This can be avoided by using formula

$$x \cos \theta + y \sin \theta = \rho \quad (1)$$

The relation between x, y and θ, ρ is show in Figure 2.3

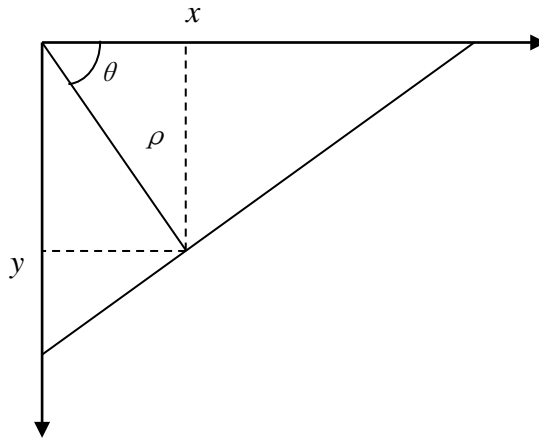


Figure 2.3 (ρ, θ) parameterization of line in the xy plane

For each edge point (x, y) obtained by the Prewitt or Sobel mask, $x \cos \theta + y \sin \theta = \rho$ that passes through the point is used to increment a counter. The points that have the high amount of trigonometric function curves crossed determine the probable lines in the image. [16] [17]

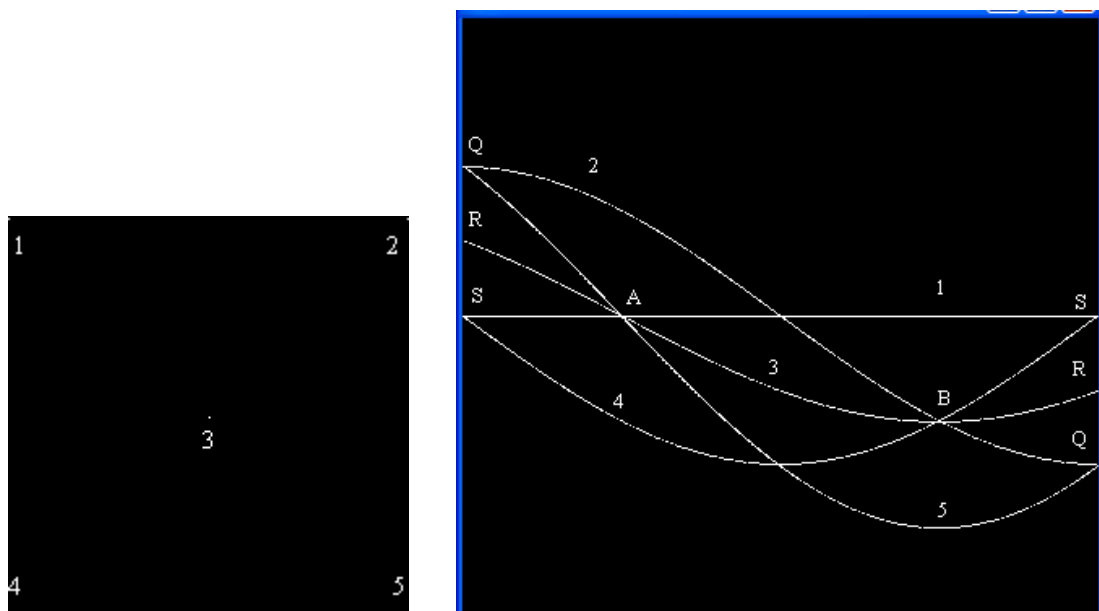


Figure 2.4 an example of using Hough transform to detect lines. Left image is xy plane. Right image is parameter space.

In Figure 2.4, points 1, 2, 3, 4 and 5 refer to points in the left image, the right image

shows that the most trigonometric function curves cross points A and B in $\rho - \theta$ plane. Point A, in the bottom image, determines points 1, 3 and 5 can be a line in the top image. Point B, in the bottom image, determines points 2, 3 and 4 can be a line in the top image.

2.1.3 Canny Edge Detection

Canny edge detection algorithm builds on the gradient vector that can find edges excellently in practice. The most significant contribution of Canny edge detection is the detector of edge pixels in an image, using non-maximum suppression and a double threshold [18][19][20]. It contains four steps:

Reduce noise

Noise of an image could result from various effects and is inevitable. In order to carry out successful edge detection, the noise in an image should be reduced. Canny proposed to use a Gaussian smoothing filter

Finding the intensity gradient for each pixel

This task can be done by Sobel operators. The length of each gradient vector is calculated and its direction would be classified into 4 directions horizontal, vertical, forward diagonal and backward diagonal.

Non – maximum suppression

In this step, a pixel is discarded if its gradient vector length is smaller than the value at either of the adjacent pixels in both direction of the gradient vector. This results in thin lines for the edges.

Double threshold

Two thresholds are built in this step. The ratio of upper threshold and lower threshold is chosen between 2:1 and 3:1. Any pixel's values above the upper threshold are

accepted as an edge pixel. Any pixels' below the lower threshold are rejected. For those pixel values, which are between upper threshold and lower threshold are accepted if and only if one of its neighbors is above the threshold.

Comparing with the basic edge detection such as gradient-based edge detection, the cost of Canny edge detection is higher and more complex although it is often found to work well in practice

2.2 Basic Geometrical Shape Recognition

2.2.1 Hough Transform for Circles Recognition

Similarly to before, the Hough transform can also be applied with the circles [21][22][23]. A circle equation $(x - a)^2 + (y - b)^2 = r^2$ could be written:

$$x = a + r \cos \theta, y = b + r \sin \theta \quad (2)$$

The difference between line recognition and circle recognition by Hough transform is that circle recognition works in the three dimensional space represented by (a, b, r) .

2.2.2 Proportion Based Shape Recognition

The proportion between edge length and enclosed area size can also be used for basic geometrical shape recognition. This can be measured by:

$$C = \frac{P^2}{A} \quad (3)$$

where P is the edge length and A is the area size (the size can be counted by number of pixels). C is greater than or equal to 4π . If C is equal to 4π , the shape must be a circle. If C is 16, the shape might be a square. If C is $6\sqrt{3}$, the shape might be equilateral triangle [24].

2.3 Contour Based Object Matching and Recognition

This approach considers the contour of an object, and matches with a known object in order to achieve object recognition.

2.3.1 Shape Context Descriptors

Shape context descriptors were introduced around the end of the last century. The performance was 99.4% accuracy on handwritten digits [25][26]. The basic idea is get some points on the edge of the object contours. Then all the points will be put into the log-polar histogram bin. The bin contains information on the direction of all the points on the edge relative to the selected point and the distance between all the points and the selected point. This gives a 3D histogram. The x-axis of the histogram shows the direction, whereas the y-axis shows distance and the intensity shows the number of points in a certain direction and distance to the selected point.

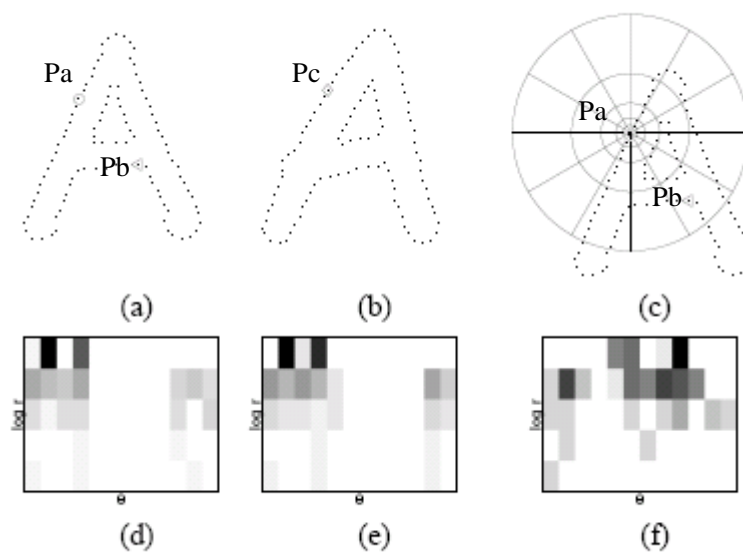


Figure 2.5 an example of Shape context. (d) is a log-polar histogram for point Pa. (e) is a log-polar histogram for point Pb. (f) is a log-polar histogram for point Pc.

[27]

However considering the algorithm may take n points, the total cost of computation

time will be n^2 which makes the cost of matching progress very high [27]. Mori improve the approach by two stages “Fast Pruning” and “Detailed Matching” in 2001 [28]. Fast Pruning selects likely candidate shapes from a potentially large collection of stored shapes. Detailed Matching cost more on matching the details to find the matching shapes. It has been further developed for 3D object [29][30]

2.3.2 Hu Moments

Moments can be used for contour based object recognition and can also be applied to image recognition [31][32][33]. The formula of moments for a pixel with width w and height h is given by:

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) dx dy \quad (4)$$

where m_{pq} is the p - q moment, the sum of p and q gives the order of the moments and $f(x, y)$ is the intensity at pixel x, y .

A uniqueness theorem of moments was given by Papoulis in 1965 [34]. The uniqueness theorem states that “if $f(x,y)$ is piecewise continuous and has nonzero values only in a finite part of the xy plane, moments of all orders exist, and the moment sequence (m_{pq}) is uniquely determined by $f(x,y)$. Conversely, (m_{pq}) uniquely determines $f(x,y)$.”

Moments represent some features of a shape or an image after centralization.

$$(i_c, j_c) = (M_{10} / M_{00}, M_{01} / M_{00}) \quad (5)$$

M_{00} is the sum of density of an image. And (i_c, j_c) represents the coordinates of centroid of an image. In other words, it represents the centroid of intensity of an image. The centralized moment is given by:

$$m_{pq} = \sum \sum (i - i_c)^p (j - j_c)^q f(i, j) \quad (6)$$

It represents the distribution of an image's intensity relative to the centroid. If $m_{20} > m_{02}$, it means the image or the object is more distributed on horizontal axis; m_{30} and m_{03} are showing the feature of horizontal symmetry and vertical symmetry.

In 1962, Hu proposed using moments for visual pattern recognition [35][36]. The great advantage of using moments is that they are invariant to image scaling, rotation, translation, even mirroring the image. [37][38].

The theory of Hu moments is that this approach is based on the centroid. The centralized moment is calculated. This gives moment invariance under translation. The normalized central moments give scaling invariance. Finally, seven invariant moments are calculated. During the final calculations for seven invariant moments, the rotation effects are taken into account [38]. Here is an example of how the rotation effect on moments μ_{20} and μ_{02} are taken off from second-order moments after rotation by α degree.

$$\mu'_{20} = \cos^2 \alpha \times \mu_{20} + \sin^2 \alpha \times \mu_{02} - \sin 2\alpha \times \mu_{11} \quad (7)$$

$$\mu'_{02} = \sin 2\alpha \times \mu_{20} + \cos 2\alpha \times \mu_{02} - \sin 2\alpha \times \mu_{11} \quad (8)$$

$$\phi'1 = \mu'_{20} + \mu'_{02} = (\cos 2\alpha + \sin 2\alpha)(\mu_{20} + \mu_{02}) \quad (9)$$

2.3.3 Colour Based Object Recognition

Colours have often been described in different colour spaces. RGB(red, green, blue colour space) and HSI(Hue, Saturation, Intensity colour space) are the most common colour spaces[39] in image processing. Other colour spaces include CMY, CMYK and Lab for different purpose. [40]

The simplest approach to colour based object recognition is by using a colour range. The whole image is searched and the pixels which fall into a given colour range are found. These pixels are considered to be the pixels of the target object. Ralph Brunner, Frank Doepke and Bunny Laden developed this way via a GPU-based program [41]. They assumed a tennis ball is purple. Then they build a mask to run over the image. The mask exposed the pixels which are purple. The centroid of the mask is computed for composing with other object.

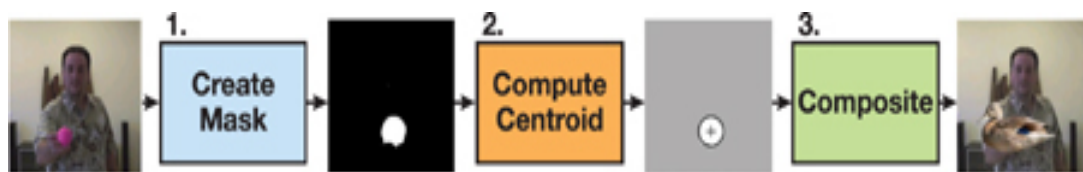


Figure 2.6 simple colour based object recognition[41]

Histogram back projection is another colour feature detection technique based on image histogram for object recognition. It considers the colour or intensity distribution of pixels on histogram [42]. There are two back projection approaches, pixel-based and patch based. Patch based back projection improves pixel-based by creating histograms in the neighbourhood for each pixel for matching.

2.3.4 Machine Learning Approaches

Minimum Distance Classifier is a popular machine learning approach. It belongs to pattern classification [41][42]. In this approach, the training data have to be analyzed and its features are found. A boundary line is located between two groups of object by the mean of feature vectors for each of the two groups. The equation of the boundary is given by

$$d_i(x) = x^T m_i - 0.5 m_i^T m_i \quad (10)$$

$$d_{ij}(x) = d_i(x) - d_j(x) \quad (11)$$

where $d_{ij}(x)$ is the smallest distance. m_i is the mean of feature vector i [43].

Fisher was the first person to apply minimum distance classifier in his paper. He recorded width and length data of three kinds of Iris (Iris Setosa, Iris Virginica and Iris Versicolor) [44].

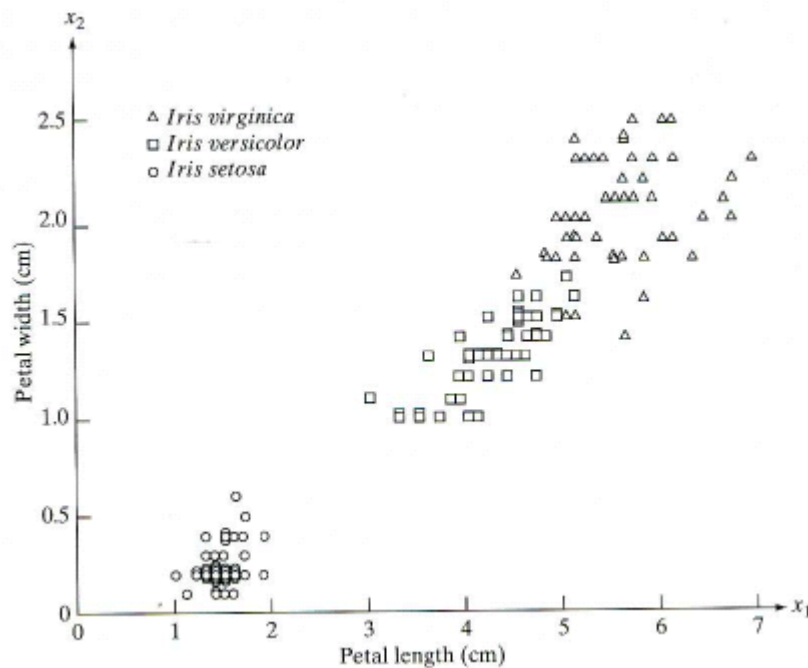


Figure 2.7 Three types of iris flowers described by two measurements. [44]

Using Figure 2.7 as an example, for Iris Versicolor and Setosa, the average vectors are $m_1 = (4.3, 1.3)^T$ and $m_2 = (1.5, 0.3)^T$.

$$\begin{aligned} d_1 &= x^T m_1 - 1/2 m_1^T m_1 \\ &= 4.3x_1 + 1.3x_2 - 10.1 \end{aligned}$$

$$\begin{aligned} d_2 &= x^T m_2 - 1/2 m_2^T m_2 \\ &= 1.5x_1 + 0.3x_2 - 1.17 \end{aligned}$$

$$\begin{aligned} d_{12} &= d_1(x) - d_2(x) \\ &= 2.8x_1 + 1.0x_2 - 8.9 \end{aligned}$$

The result is show in Figure 2.8.

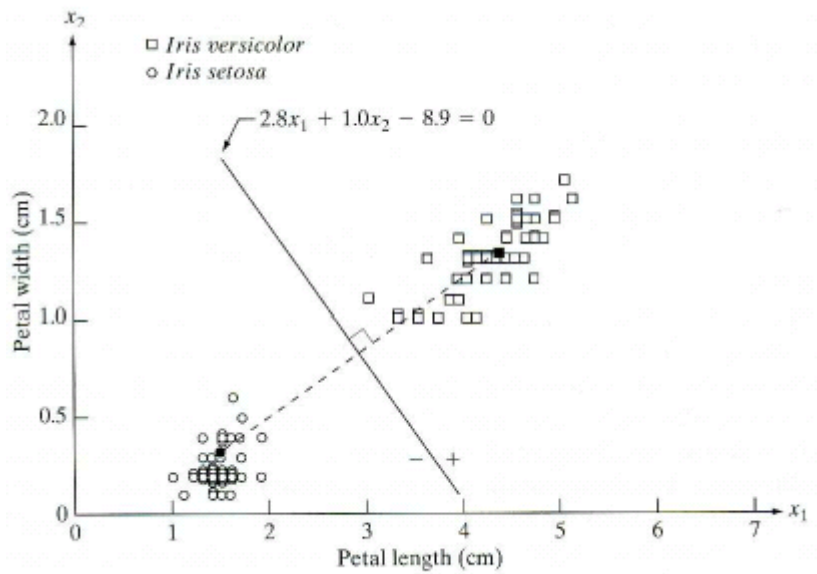


Figure 2.8 Minimum Distance Classifier for Iris Versicolor and Iris Setosa [44]

Support Vector Machine (SVM) is another machine learning approach. SVM was introduced by Vladimir [45]. It is good when there is limited sampled data. It often provides linear classification in a higher-dimensional space even if the original space data may be nonlinear [46]. SVM can also be applied for non-linear classification by using the kernel tick technique [47]. Recently, the most common approaches of One-against-One and One-against-Rest build more than one SVM.

2.4 Texture Analysis

A texture is another important feature of objects and can be considered as regions where patterns repeat. Textures can be categorized into artifact textures and natural textures. Artifact textures are often analyzed by using structure analysis and natural texture are often analyzed by using statistical techniques [24].

2.4.1 Gray-Level Co-occurrence Matrix (GLCM)

GLCM is a well known statistical approach for texture image analysis [24] [48]. The second order statistics features of an image are often suggested to get from GLCM by Haralick [49]. It measures which two neighboring pixels have the same patterns. This kind of relation represents the feature of the texture. It is one of statistical approaches and can be used for texture classification and texture feature extraction [50][51][52][53].

GLCM is defined as the probability of two gray-levels. $P_{\delta}(i, j)$ ($i, j = 0, 1, 2, \dots, L-1$) where i and j are the gray-levels, L is the number of grey-levels in an image, $\delta = (\Delta x, \Delta y)$ is the position relationship between two pixels, Δx and Δy are the distances between two pixels on the x and y axes. The position relationship also includes direction between each pixel pair. Often, the directions are chosen to be 0, 45, 90 and 135 degrees.

Here are two examples [24] for images that have $L = 4$ grey levels, where 0 degree is taken to the right and angles measured anticlockwise

| | |
|-----------------|-----------------|
| 0 0 1 1 2 2 3 3 | 0 1 2 3 0 1 2 3 |
| 0 0 1 1 2 2 3 3 | 1 2 3 0 1 2 3 0 |
| 0 0 1 1 2 2 3 3 | 2 3 0 1 2 3 0 1 |
| 0 0 1 1 2 2 3 3 | 3 0 1 2 3 0 1 2 |
| 0 0 1 1 2 2 3 3 | 0 1 2 3 0 1 2 3 |
| 0 0 1 1 2 2 3 3 | 1 2 3 0 1 2 3 0 |
| 0 0 1 1 2 2 3 3 | 2 3 0 1 2 3 0 1 |
| 0 0 1 1 2 2 3 3 | 3 0 1 2 3 0 1 2 |

Image A

Image B

$$p_{A\delta}(0^\circ) = \begin{bmatrix} 16 & 8 & 0 & 0 \\ 8 & 16 & 8 & 0 \\ 0 & 8 & 16 & 8 \\ 0 & 0 & 8 & 16 \end{bmatrix} \quad p_{A\delta}(90^\circ) = \begin{bmatrix} 28 & 0 & 0 & 0 \\ 0 & 28 & 0 & 0 \\ 0 & 0 & 28 & 0 \\ 0 & 0 & 0 & 28 \end{bmatrix}$$

$$p_{B\delta}(45^\circ) = \begin{bmatrix} 24 & 0 & 0 & 0 \\ 0 & 24 & 0 & 0 \\ 0 & 0 & 24 & 0 \\ 0 & 0 & 0 & 26 \end{bmatrix} \quad p_{B\delta}(135^\circ) = \begin{bmatrix} 0 & 0 & 25 & 0 \\ 0 & 0 & 0 & 24 \\ 25 & 0 & 0 & 0 \\ 0 & 24 & 0 & 0 \end{bmatrix}$$

The normalized matrices $p'_{A\delta}$ and $p'_{B\delta}$ are given by $P_\delta(i, j) / S$, where S is the sum of all the elements' values in the $P_\delta(i, j)$

$$p'_{A\delta}(0^\circ) = \begin{bmatrix} 1/7 & 1/14 & 0 & 0 \\ 1/14 & 1/7 & 1/14 & 0 \\ 0 & 1/14 & 1/7 & 1/14 \\ 0 & 0 & 1/14 & 1/7 \end{bmatrix} \quad p'_{A\delta}(90^\circ) = \begin{bmatrix} 1/4 & 0 & 0 & 0 \\ 0 & 1/4 & 0 & 0 \\ 0 & 0 & 1/4 & 0 \\ 0 & 0 & 0 & 1/4 \end{bmatrix}$$

$$p'_{B\delta}(45^\circ) = \begin{bmatrix} 12/49 & 0 & 0 & 0 \\ 0 & 12/49 & 0 & 0 \\ 0 & 0 & 12/49 & 0 \\ 0 & 0 & 0 & 13/49 \end{bmatrix} \quad p'_{B\delta}(135^\circ) = \begin{bmatrix} 0 & 0 & 25/98 & 0 \\ 0 & 0 & 0 & 12/49 \\ 25/98 & 0 & 0 & 0 \\ 0 & 12/49 & 0 & 0 \end{bmatrix}$$

After normalizing the matrices, if the non-zero values are all allocated on the diagonal elements in a certain angle, then the original image is symmetric about that direction. The distribution of non-zero elements also gives information of how rapidly the image change intensity. For example matrix $p'_{B\delta}(135^\circ)$'s nonzero numbers are distributed farther away to the diagonal line than matrix $p'_{A\delta}(0^\circ)$, which means image B on 135 degree is finer than image A on 0 degree.

Coarseness measures the changes of a texture from pixel to pixel. GLCM measures the coarseness of the texture by checking the distribution of the nonzero elements from the diagonal elements. It also can be measured by its sum of the energy which adds the squares of all the elements together.

Contrast measures the difference in intensity or colour between different parts of a texture. GLCM provides the formula of calculating the contrast.[24][54]

$$Contrast = \sum_{n=0}^{L-1} n^2 \left\{ \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} P(i,j) \right\} \quad (12)$$

where $n = |i - j|$

A disadvantage of GLCM is it may produce a huge matrix (the size is $L \times L$). Therefore before applying GLCM, the image's gray-level is typically reduced from 256 to 16. However, it may still need huge amount of computation. Mokji proposed GLCM computation based on Haar wavelets to reduce the computation burden [55].

2.4.2 Local Binary Patterns (LBP)

Local Binary Patterns (LBP) was proposed in 1994[56][57]. It is a very simple but useful technique of texture measurement[58]. Here are the basic steps:

- Divide an image into cells, for example 16*16 pixels
- Compare each pixel with its eight neighbors either clockwise or anti-clockwise
- If the pixel is bigger than all its neighbors record as 1 else record as 0
- Produce a histogram over the cell of 1s and 0s
- Normalize the histogram
- Concatenate the histogram for each cell. And this gives the feature vector of the window.

This approach is often used for finger print detection [59], object classes recognition[60], texture classification [61], face recognition [62]

2.4.3 Tamura's Texture Features

Tamura, Mori and Yamawaki selected six features to describe a texture in 1977. These feature descriptors are based on human perception [63][64]. Content-based image retrieval systems (CBIR systems) are often used for extracting features from the entire image or from regions but specific regions are more interested than the entire image [65]. CBIR often uses Tamura's texture features [66] and it is also considered to be a very promising approach in biomedical image data [67t]. The Tamura's features are named Coarseness, Contrast, Directionality, Linelikeness, Regularity and Roughness [68] [69]. Tamura texture features performs better than RGB Colour Histogram [70] and are good on the high-level perceptual texture attributes but does not well on finer texture discrimination [71].

2.4.3.1 Coarseness

Coarseness is about the distance of variations of an image's grey levels. It is based on each pixel and find the averages of six different sizes (size = $2^k \times 2^k$, $k = 0, 1, \dots, 5$) nonoverlapping windows in both horizontal and vertical directions. The largest difference between the windows is set as the best size then averaging the best size of each pixel over the entire image.

Hanish Aggarwal and Padam Kumar suggested combining coarseness information with histogram to improve Tamura coarseness [72].

2.4.3.2 Contrast

Contrast measures variance of grey levels q and its extension, which is based on black or white[73]. It is given by following formula

$$F = \sigma / \alpha_4^n \quad (13)$$

$$\alpha_4 = \mu_4 / \sigma_4; \quad \sigma^2 = \sum_{p=0}^{\max} (p - m)^2 \quad (14)$$

where q_0 is the first pixel in the image, q_{\max} is the last pixel in the image.

$$\mu_4 = \sum_{p=0}^{\max} (p - m)^4 \quad (15)$$

Where m is the mean grey level.

2.4.3.3 Directionality Degree

Tamura Directionality Degree is the total degree of Directionality. It is firstly using Sobel edge detector windows $\Delta_x(x, y)$ and $\Delta_y(x, y)$ to find all the edge pixels. The edge strength $e(x, y)$ and the directional angle $a(x, y)$ is computed by following formula.

$$e(x, y) = 0.5(|\Delta_x(x, y)| + |\Delta_y(x, y)|) \quad (16)$$

$$a(x, y) = \tan^{-1}(\Delta_y(x, y) / \Delta_x(x, y)) + \pi/2 \quad (17)$$

Then a histogram of the percentage of the edge pixels with its angle is built but for each edge pixel, the edge pixel will be counted if its edge strength is greater than a threshold.

$$F_{\text{dir}} = \sum_p^{n_p} \sum_{\varphi \in \omega_p} (\varphi - \varphi_p)^2 H_D(\varphi) \quad (18)$$

Where n_p is the number of peaks, ω_p is the range of the peak, φ_p is the peak value of p th peak. Some research shows that Tamura Directionality does not differentiate between orientations or patterns and the directionality score does not reflect human perception [74][75][76].

2.4.3.4 Linelikeness

Linelikeness is basically built on the Directionality by averaging coincidence of the edge directions. It applies a given threshold to select the edge strength, which is greater than the threshold. Then coincidence is calculated by the cosine of the difference of the angles. If the edges are the same direction, it adds one to a counter. If the edges are in the perpendicular direction, it subtracts one.

2.4.3.5 Regularity

Regularity is measured by the following formula

$$F_{\text{reg}} = 1 - r(s_{\text{crs}} + s_{\text{con}} + s_{\text{dir}} + s_{\text{lin}}) \quad (19)$$

Where r is a normalising factor and all the s terms are the standard deviation of all the texture features (coarseness, contrast, Directionality and Linelikeness).

2.4.3.6 Roughness

Roughness is measured by sum of coarseness and contrast.

Linelikeness, regularity and roughness are correlated with coarseness, contrast and Directionality. Furthermore coarseness, contrast and Directionality are correlated with human perception. Therefore, these three features are often used and applied in CBIR system [77].

2.5 Fourier Transform

Fourier series for periodic functions were introduced by the French mathematician Fourier. The formulation for any functions is called the Fourier transform. However it did not find broad application until the Fast Fourier Transform (FFT) was introduced.

The most common algorithm is called the Cooley-Tukey algorithm. There are also some other algorithms, such as Prime-factor FFT algorithm, Bluestein's FFT and Bruun's FFT algorithm[78][79].

The Asymptotic Complexity of Fast Fourier Transform is:

$$O = N(M\log M) + M(N\log N) = MN\log MN \quad (20)$$

2.5.1 Fourier Spectrum and Fourier Phase Angle

The Fourier Transform is an excellent tool for image processing. It turns spatial domain data into frequency domain data with both real and imaginary parts [80]. The Fourier spectrum and phase angle both hold information for image processing. The Fourier spectrum consists of the magnitudes of the complex numbers in the complex plane, and contains intensity information of the original image. Fourier phase angle consists of the displacements of the various sinusoids, and contains positional information of discernable objects in the original image [81][82][83][84].

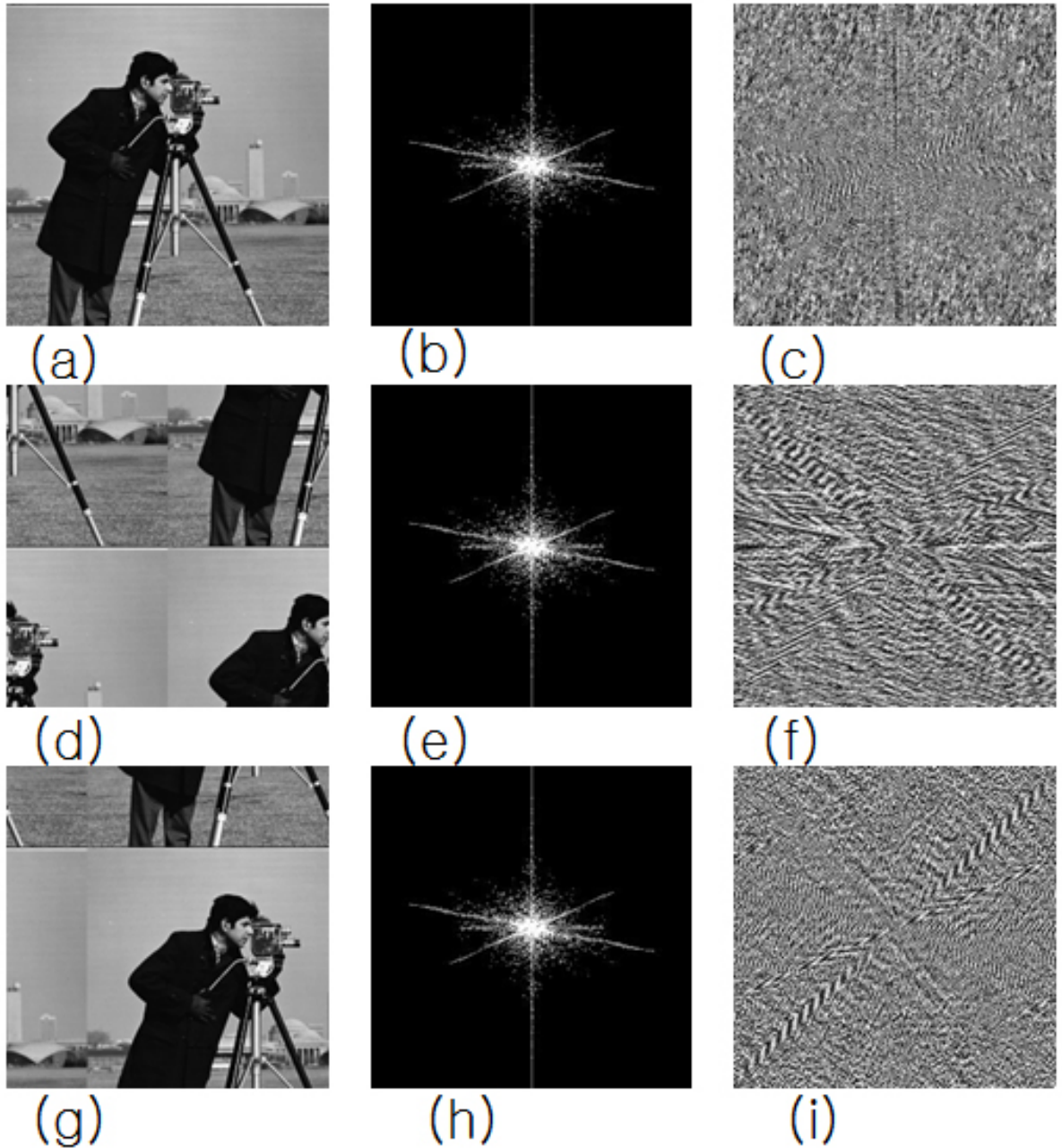


Figure 2.9 (a) Original image. (b) The magnitude of the Fourier spectrum from (a) represented as an image. (c) The phase of the Fourier spectrum from (a) represented by an image. (d) and (g) are original image shifted by 128 rows and 128 columns, and 64 rows and 64 columns, respectively. (e), (h) The magnitude spectrum. (f), (i) Phase spectrum. These images illustrate that when an image is translated, the phase changes, even though magnitude remains the same. (Images are from [http://lab.must.or.kr/\(S\(2iatfs55hncsa455z1rxqt55\)\)/Print.aspx?Page=Fourier-Transform-Properties](http://lab.must.or.kr/(S(2iatfs55hncsa455z1rxqt55))/Print.aspx?Page=Fourier-Transform-Properties))

As phase angle dominates the position information of an image, it determines an image's edges. Figure 2.10 shows the result of merging two images: the phase angle and the spectrum.

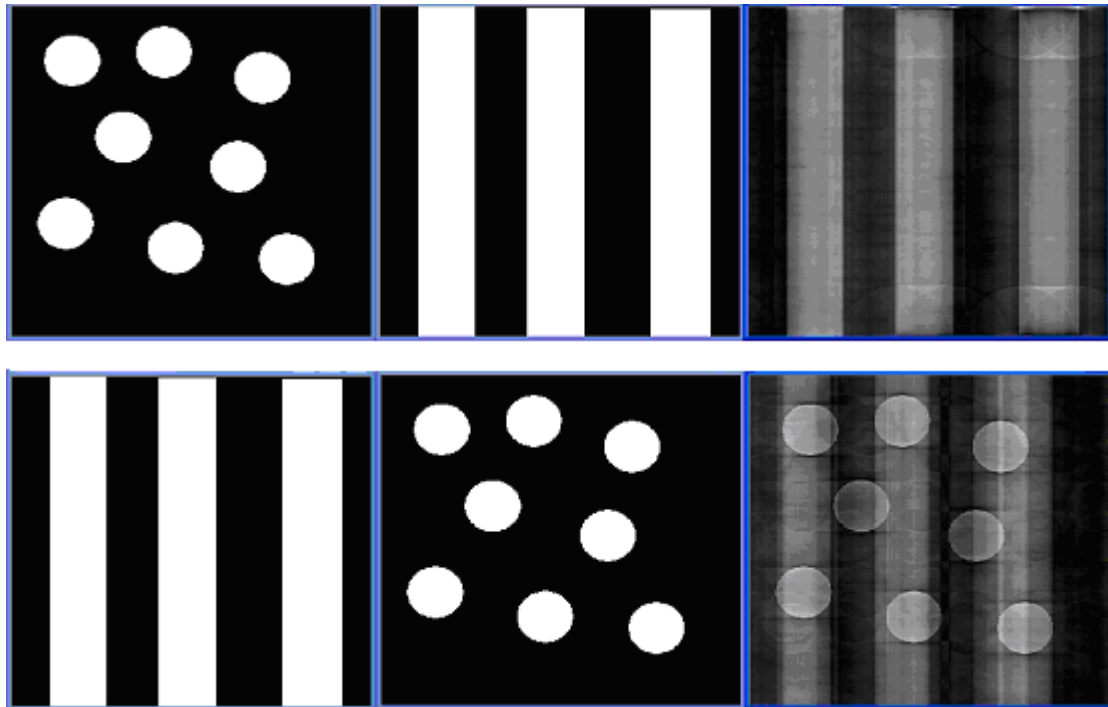


Figure 2.10 shows combination of the different images' phase and spectrum. The first column is the images, which contribute the spectrums to the reversed FFT of the last column. The second column is the images, which contribute the phase to the reversed FFT of the last column. The third column is the combination of these phases and spectrum. The phases stores the position and the spectrum stores the energy. Therefore the top image from the third column keeps the bars' position, but lost the circles'. The bottom image from the third column keeps the position information of the circles, but the bars' positions are lost

The merged image keeps the objects from the image which provides the phase angle. However the intensity depends on the image, which provides the spectrum [85]. Comparing with the spectrum, phase angle is not often used in many applications [86].

2.5.2 Aliasing and Leakage

The Discrete Fourier Transform (DFT) is used in digital image processing. It has two issues: aliasing and leakage. If the initial samples are not sufficiently closely spaced to represent the high-frequency components present in the underlying function, then the DFT values will be corrupted by aliasing. As before, the solution is either to increase the sampling rate (if possible) or to pre-filter the signal in order to minimise its high frequency spectral content. [87]

The continuous Fourier transform of a periodic function requires integration to be performed over the interval $-\infty$ to $+\infty$, or over an integer number of cycles of a periodic function. However, DFT leakage appears as the phenomenon in which DFT is calculated over a partial integral of the input signal or a finite signal, which can be considered as the product of an infinite signal with a square window function. [88]. A typical window function is a non-negative bell shaped curve [89]. The polyphase filter bank (PFB) technique is a mechanism for helping to alleviate the above problem.

Mathematically, the solution to DFT leakage involves suppressing the side-lobes of the sinc function by changing the single-bin frequency response of the DFT to approximate a rectangular function. The PFB approach multiplies the pixels in an image point-by-point with a sinc function, which has the same size as the image. A sinc function is chosen because the Fourier Transform of a sinc function is a rectangular function. Then the product of image and sinc function can be split into a number of blocks with the same size. All the blocks are summed point-by-point to pass to the DFT [88].

Window functions are also used in spectral analysis and they were designed for finite signal filters in order to reduce the problem of leakage. The most common window functions are Blackman window function, Gaussian window function, Hamming window function and Kaiser window function [90].

2.5.3 Fourier Spectrum for Variance Images

2.5.3.1 Direct Current of Spectrum Image

After Fourier transform, a pure white image or a constant gray image appears as a single pixel at the center (zero frequency) and that pixel does not represent a sine wave. It is only the overall intensity of the image and is called the Direct Current (DC). The value of that pixel is the intensity of the original gray constant image or the colour of the constant image if the image has colour.

However if the image is not a constant image, the center pixel of the spectrum image or the DC value represents the average colour or intensity of the image. Conversely, if the value of the center pixel of a spectrum image is changed, the image's colour will be changed after reverse FFT [91].

2.5.3.2 Spectrum of Different Pattern image

A single sine wave pattern images (the negative value to be 0 see Figure 2.11 and Figure 2.12) are transformed three pixels in the spectrum image. The middle pixel will be at the center of the spectrum image, and the distance in pixels between the center pixel and the other two pixels represents the number of sine cycles that the original image has.



figure 2.11 a sine wave image with four sine cycles. The spectrum of this sine wave image has three pixels the distance between the center pixel and the other pixels is four pixels

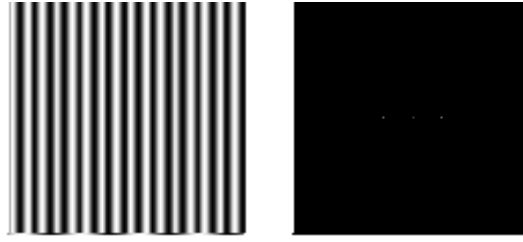


figure 2.12 a sine wave image with sixteen sine cycles. The spectrum of this sine wave image has three pixels the distance between the center pixel and the other pixels is sixteen pixels

The spectrum image of a rectangular pattern will be repeated grids. The size and the number of the grids in x and y direction depend on the size of the rectangle and the original image size. The Pattern Size in spectrum image in x and y direction is N/a and M/b , where a and b are the width and high of the rectangle and the original image size is $N \times M$. [92]

Rotation of a rectangle pattern rotates the spectrum pattern, but the spectrum pattern is rotated in the reverse direction.

Theoretically, a circular pattern image is transformed to be circles, but a digital image is made by a matrix of square pixels. Therefore in practice, a circular pattern is not as round as expected. A large circular's spectrum image contains fewer artifacts than a small circular pattern image

If an image has a diameter d circle pattern, the Pattern Size in spectrum image will be $1.22 \times N/d$. [92][93][94][95].

The transform of a Gaussian circular pattern is another Gaussian circular pattern. The relationship between these two Gaussian circular pattern is:

$$\text{Transformed Sigma} = N / (2 \times \text{Original Sigma}) \quad (21)$$

Where N is the size of the image and σ is the variance in the two dimensions Gaussian function [92].

2.5.4 Basic Applications of Fourier Transform

The most popular applications of Fourier Transform in image processing include adjusting the contrast of an image, blurring or sharpening an image, noise removal (perfectly for removal of periodic noise [96]), edge detection [97][98] and embedded water marks in spectrum of the images [99].

2.5.4.1 Fourier Spectrum and Contrast

Increasing the spectrum after applying Fourier transform can increase an image's contrast. The intensity of the spectrum relates to the contrast of the original image. Increasing the spectrum energy (intensity) is increasing the changes of intensity from the original image. If the 0 frequency (DC) is kept as it was, the overall intensity of the image will not be changed [92].

2.5.4.2 Fourier Spectrum and Edge Detection

It has been widely used that applying Fourier Transforms in image processing as frequency domain filters with a centralized Fourier spectrum, the centre of the image in Fourier plane is the main intensity and higher frequency components of the image are on the edges. This provides the ability to perform image edge detection, smoothing or sharpening [100]. Image smoothing often applies lowpass filters; edge detection and sharpening often apply highpass filters [101][102].

Fourier Transform has been widely used for edge detection [103][104][105][106] Edge detection utilises the high frequency channels of the spectrum to keep detailed information, which includes edge information, after getting the high frequency components, then applying reverse transform. A high pass filter is used for this application. A step circular high pass filter is the most simple high pass filter.

However as this filter has a sharp edge, it can produce “ringing” artifacts on the reversed image. A Gaussian high pass filter has smooth edge and its Fourier transfer is a Gaussian feature, therefore the edge can be better detected than a step circular high pass filter. Furthermore, the edge detection can be used for the segmentation algorithm [107].

2.5.4.3 Fourier Spectrum and Image Smoothing and Sharpening

Image smoothing can be achieved by using a low pass filter. The filter passes the low frequency components, which includes zero frequency so the intensity of an image will be kept unchanged. The common low pass filters are Ideal Low Pass Filters (ILPF), Butterworth Low Pass Filters (BLPF), Gaussian Low Pass Filters (GLPF), Exponential Low Pass Filter (ELPF) and Trapezium Low Pass Filter (TLPF) [92][108].

The most common application of using low pass filter is reducing the noise [109]. The filter passes the low frequency, which keeps the main features of the image but the high frequency channels, which contain noise, will be blocked. After reverse FFT, the noise in the image is usually reduced. Another application is increasing the quality of documents for machine recognition. Because of the age of the documents, some old documents have noise and broken letter problems (where a letter is corroded by noise and some parts of a letter are broken), which makes machine recognition task harder. The low pass filter helps reconnect the broken point of the letters.

Besides using a high pass filter to detect edges in an image it can also be used to sharpen an image. This approach is called High-boost Filtering. The simplest way of using high-boost filtering is applying the high pass filter in the frequency domain then doing reversed FFT then blending with the original image. Here is a formula for applying high-boost filtering for sharpening.

$$f_{hb}(x, y) = (A - 1)f(x, y) + f_{hp}(x, y) \quad (22)$$

where $f_{hb}(x, y)$ is an output image $A \geq 1$, $f(x, y)$ is an original image, and $f_{hp}(x, y)$ is the high pass filtered version of the original image [91].

2.5.4.4 Fourier Spectrum and Notch Filters

Fourier Transform also can be used as notch filters. Notch filters are the most useful of the selective filters. In theory, they reject certain frequencies, usually chosen to remove noise. After removing the undesired frequencies, the noise on the image will be reduced [110][111]. This technique has been used as a function within image processing applications such as Photoshop.

Usually a notch filter is created manually, and a circular Gaussian function is used to avoid “ringing” artifacts. Figure 2.13 illustrate a comparison of using two different shape notch filters.

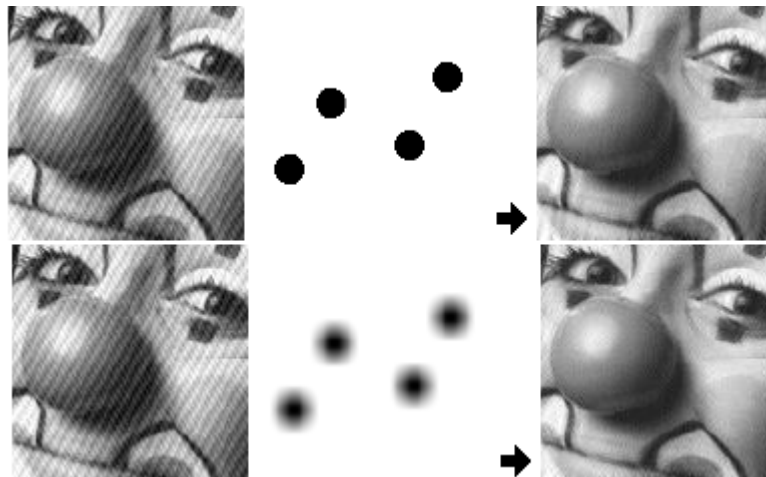


Figure 2.13 An example comparing circular and Gaussian notch filters

2.5.5 Advanced Applications of Fourier Spectrum (Blur and Deblur)

In the spatial domain, blurring is the result of convolutions of an image and a filter. And deblurring is the result of deconvolution (a reverse of convolution, it takes an convoluted image to retrieve the original image) of an image and a filter. However deblurring an image in the spatial domain is not as easy as blurring an image. Fortunately, convolution or deconvolution in the spatial domain is the same as complex multiplication and division in the frequency domain [92][112] and Fourier Transform helps a lot of deblurring an image [113][114].

Ideally, an image and a filter are both transferred to the frequency domain. Then they are multiplied together element by element.

If the filter is a horizontal line, then a motion blur can be created. The image is blurred horizontally. If the filter is a circle, then the image is blurred evenly to all directions.



Figure 2.14 horizontal motion blur



Figure 2.15 even motion blur

Likewise, we can apply deblurring by complex number division.



Figure 2.16 refocus of horizontal motion blur

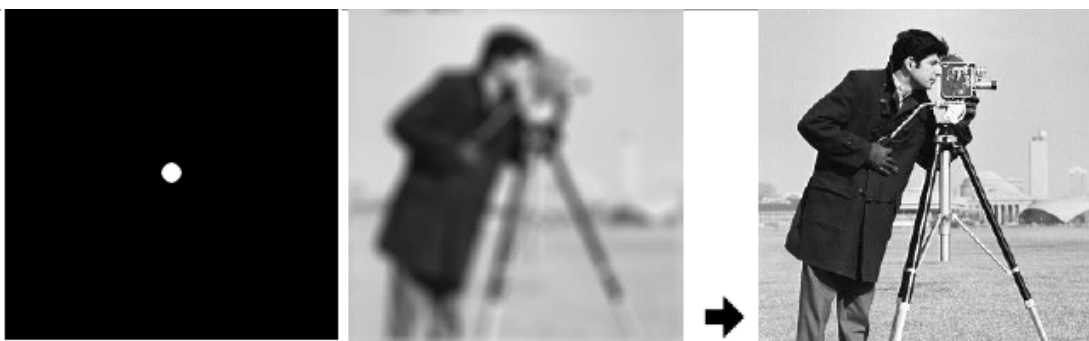


Figure 2.17 refocus of even motion blur

However in practice, noise occurs when a photo is taken. The noise can be amplified by this approach. This is because the denominator is very small and it almost equals to zero (see Figure 2.18). Norbert Wiener suggested a solution, which is called Wiener Filter [115].



Figure 2.18 realistic refocus of horizontal motion blur cause more noise

$$\frac{P}{F} = \frac{F^*P}{F^*F} \rightarrow \frac{F^*P}{|F|^2 + n} = \frac{(A - iB)(C + iD)}{(A^2 + B^2) + n} = \frac{(AC + BD)}{(A^2 + B^2) + n} + i \frac{(AD - BC)}{(A^2 + B^2) + n} \quad (23)$$

Equation 23 shows improved complex number division. Where P and F are two complex numbers. A little value n is added, where n is the ratio of the noise. It can be calculated by squaring the overall image variance (squared standard deviation) dividing the noise variance. Where the noise variance can be calculated by a segmented the image with a constant gray level. The segmented the image is cut a part of an image which has a constant gray level.

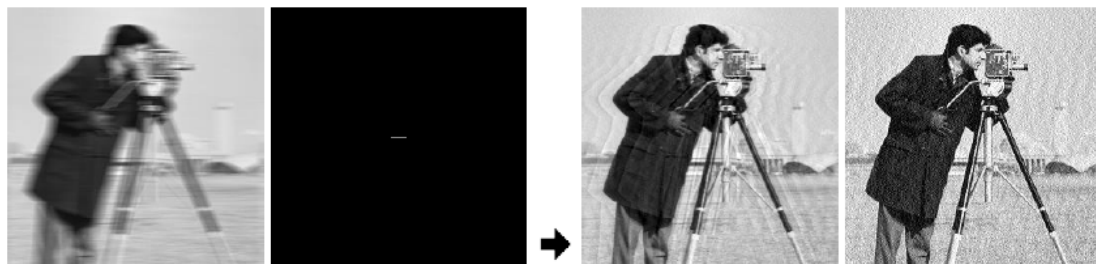


Figure 2.19 balancing the noise and “ghosting” artifacts

In the example above (Figure 2.19), the left result is used $n = 0.005$, which is bigger than the ratio of the noise. The right result is used $n = 0.0005$, which is the ratio of the noise. The left result has “ghosting” artifacts but right image reduced this artifact however it contains more noise. Comparing the result which is without n , the $n = 0.0005$ keeps the balance between the artifacts and the noise level.

2.5.6 Fourier Spectrum descriptors

The Fourier spectrum has three features that are useful for texture description [91]

The prominent peaks in the Fourier spectrum give the principal directions of texture patterns.

The location of the peaks in the frequency plane gives the fundamental spatial period of the patterns.

Eliminating any periodic components via filtering leaves nonperiodic image elements, which can then be described by statistical techniques.

The Fourier spectrum can be describe by polar coordinates $S(r, \theta)$. It can be disassembled into two function $S_\theta(r)$ and $S_r(\theta)$. $S_\theta(r)$ represents the amplitude of the frequency component on the line that starts from the origin with θ degree. $S_r(\theta)$ represent the feature on the circle with r radius.

$$S(r) = \sum_{\theta=0}^{\pi} S_\theta(r) \quad (24)$$

$$S(\theta) = \sum_{r=1}^{R_0} S_r(\theta) \quad (25)$$

The functions above provide spectral-energy description of textures. The main propose of it in this thesis is for finding the location of the highest value, the mean and variance of both the amplitude and axial variations.

Chapter 3

Image Preparation

This research has been designed to have three phases: Image Preparation, Conversion to Frequency Domain and Frequency Domain Pattern Description and Recognition. In order to have a clear texture pattern, several steps need to be applied before converting to frequency domain. This is called Image Preparation or Pre-processing, which mainly includes developing techniques for reducing the effect of shadows and for segmentation.

3.1 Shadow Removal

A texture will be affected by a shadow if a part of the texture is within the shadow when it is caught by a camera. It may lead to mismatching for the final result.

Most shadow removal processes remove the shadow and replace it by the correct colour [116]. The shadow removal process applied here does not totally remove the shadow and replace it by the correct colour. Instead it reduces the shadow effect to within a tolerance level so that the texture pattern recognition algorithm can work well.

Shadow is caused by a lighting source being blocked by objects. The lighting model is written as below.

$$B(x) = \sum_{k \in \text{all sources}} B_k(x) \quad (26)$$

Considering in RGB space, an object can appear different colours when it is under different coloured lighting sources. Therefore an image needs to set the white balance first, which corrects the colour for an object and convert the overall lighting sources to be white. This can be done by a camera white balance setting feature (see Figure

3.1).



Figure 3.1 the top image was taken under the white lighting sources. The bottom two images were taken under the lighting sources with 3000k correlated colour temperature. The bottom left image was applied white balance but the bottom right image was not applied white balance.

After the white balance has been set, the lighting sources colour will be corrected to be white. The Spectral Response Function (SRF) [117] is used to determine what fractions of lighting from a source are absorbed, transmitted or reflected.

The reflection colour of an object surface can be determined by the product of the lighting source colour with the SRF. SRF is a constant attribute of an object surface. A shadow only affects the amount of lighting incident on the object surface. In other words, the difference between shadowed surface and unshadowed surface is the difference of energy of the reflected light. Therefore, if shadowed surface reflection

light energy and unshadowed surface reflection light energy are manipulated to be the same energy, the shadow effect can be reduced.

The intensity of a pixel is the projection of the pixel's colour vector onto the white colour vector, which is vector [255,255,255] in RGB space.

If a colour vector is [r, g, b], the intensity formula is (r+g+b)/3. To justify this suppose this colour vector's projection which is on to white colour vector [255, 255, 255] is [n, n, n]. Then point (n, n, n) is the point on the vector [255, 255, 255] with shortest distance to the point (r, g, b). Point (n, n, n) is also shortest for distance square.

$$\text{Distance}^2 = (r - n)^2 + (g - n)^2 + (b - n)^2$$

$$(\text{Distance}^2)' = -2(r - n) - 2(g - n) - 2(b - n)$$

$$\text{At the minimum } (\text{Distance}^2)' = 0$$

$$-2r + 2n - 2g + 2n - 2b + 2n = 0$$

$$n = (r + g + b) / 3$$

A pixel's intensity $n = (r + g + b) / 3$ is not the length of the pixel as a vector in Euclidean space. The direction of the pixel vector is the hue of the pixel. Normalizing pixels will not change the hue value but converts all pixels with the same hue to be the same intensity. Therefore if a texture appears in different colour (or hue), the shadow effects can be reduced but the texture will be kept (see figure 3.2).

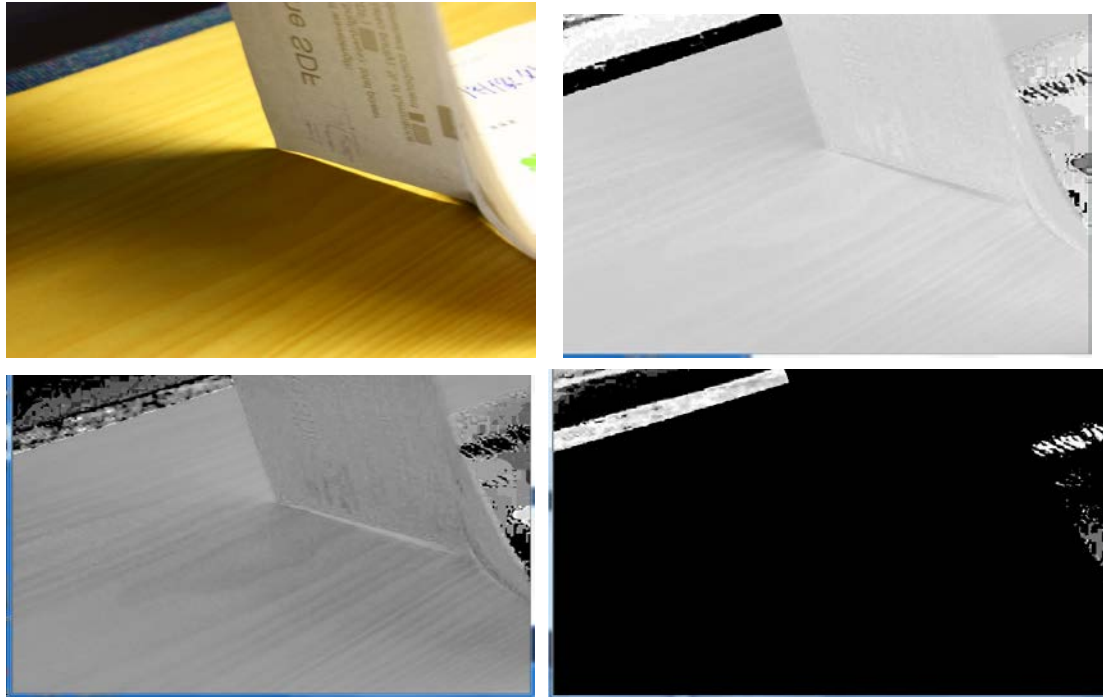


Figure 3.2 Top left shows an original image with shadow. Top right shows the R channel of the L2 normalized pixel vectors image. Bottom left shows the G channel of the L2 normalized pixel vectors image. Bottom right shows the B channel of the L2 normalized pixel vectors image.

3.2 Segmentation

Segmentation is an important part of texture recognition [118][119]. This determines the position of testing texture and distinguishes background and foreground objects. If an image contains too many background objects, the foreground object texture will be too small to be the main texture signal in the frequency domain. However most common approaches for segmentation use spatial domain transform based segmentation or colour based segmentation. Transform based approach such as background subtraction segments all the moving objects. It even segments the background moving objects. The colour based segmentation is often only used for uniformly coloured objects.

3.2.1 Issues of Spatial Domain Segmentation

The most common approach for spatial domain segmentation is called background subtraction. This can be done either by using a specified colour or by the difference of two images [41]. These two approaches are simple but they have some issues.

A disadvantage of using a given colour is that the target object must have a uniform colour. If the target object contains more than one colour or a texture pattern, this approach can fail.

A disadvantage of using the difference of two images is that it only works for moving object with a stationary background. If any part of the background is also moving, it gets included too..

3.2.2 Frequency Domain Segmentation

In order to avoid the previous issues, frequency domain segmentation is investigated. A background image is converted to frequency domain by using the Fourier Transform. All repeated patterns appear in the frequency domain, which can be viewed in spectrum image, and the position information is kept in the phase angle

image. If an object with certain texture features appears in an image, it will replace a part of background frequency pattern by an object texture's frequency patterns which also creates a new texture pattern in the frequency domain. Comparing two Fourier spectrum images, the differences are made by foreground object replacing the background objects. This gives two approaches to segmentation (filter based approach, which cuts off the texture frequency and replacing based approach which replace the texture frequency by the background frequency then segment the image).

Filter based approach is similar as using Notch filters (see details of Notch filters in Section 2.5.4.4), but Notch filters are intended for reducing noise from an entire image. However this approach treats the foreground object texture as noise and tries to find the foreground texture pattern by removing the foreground texture's spectrum pattern.

A spectrum pattern of a foreground object's texture can be used as a filter to remove the background frequency and only the foreground object's frequency may remain. Due to the linearity of Fourier transform, an image's spectrum contains both features of the image's foreground spectrum features and its background spectrum features. The filter approach can also remove a part of background spectrum when it is removing the foreground spectrum. Also this approach sets foreground object's spectrum as zero, then the position information of the foreground object might be lost.

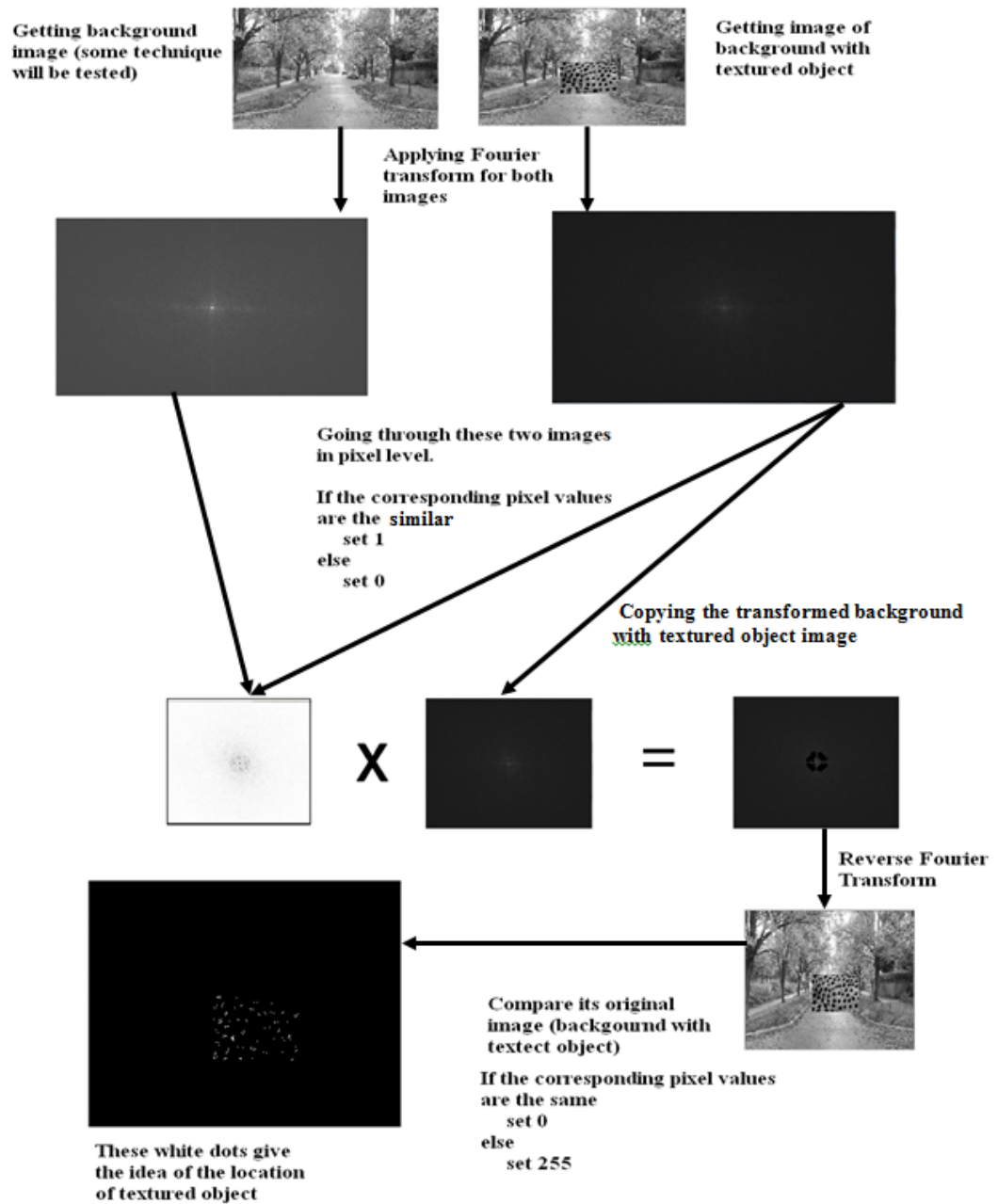


Figure 3.3 A schematic representation of how frequency domain segmentation can be achieved.

The replace spectrum approach is similar to the filter based approach. It requires the background image and the segmenting image to both be Fourier transformed to get spectrum image and phase image. Then it combines the background image's spectrum

with the segmenting image's phase image.

The phase angle image stores the position information and the spectrum stores the energy information. Therefore the replace spectrum approach actually changes the energy of the texture pattern, but the energy of the background texture pattern is not affected. Then applying the reverse Fourier transform of combined image. Using reverse transformed image to subtract the testing, the position of the texture is explored.



Figure 3.4 a test example of filter approach segmentation. The first image is the background image, the second image is the background image with foreground texture. The third image is the difference of the foreground image and the result image of filter approach segmentation



Figure 3.5 a test example of spectrum replace approach segmentation. The first image is the background image, the second image is the background image with foreground texture. The third image is the difference of the foreground image and the result image of spectrum replace approach segmentation

Comparing two approaches above, it is clearly seen that the filter approach

segmentation gets the edges of texture pattern, which demonstrates that spectrum indicates the changes of energy (in the difference image, the difference appears where the cheetah's texture patterns are). However the filter approach also detects some background texture pattern. This is because when two texture patterns have the Fourier transform applied, the result is applying Fourier transform on each of them then summing the individual result together. When applying the filter, some parts of background spectrum will be lost with the foreground texture pattern.

After the testing results in figure 3.4 and 3.5, the spectrum filter approach segmentation can be applied better in practice as it has less effect by the moving background objects and allows foreground objects to have texture patterns. Furthermore, the subtraction of foreground and background spectrums is giving a filter for the foreground objects. For each foreground object texture pattern, each filter can be found. The filter can be applied on other images to find this particular object with the same texture pattern.

Chapter 4

Fourier Transform and Spectrum Pattern Enhancement

After the image has been prepared by segmentation and reducing the effects of shadows, the texture is further reprocessed to reduce leakage when using the Fourier transform. This chapter discusses some window functions, which help to reduce the issue of leakage and also discusses approaches to enhance the spectrum, to improve the spectrum pattern for building texture feature descriptors.

4.1 Windows Functions Selection

Different types of window functions are compared in this section. Some window functions for reducing the issue of leakage are investigated.

4.1.1 Discrete Fourier Transform Issues and Solutions

The Fourier transform converts an infinite periodic signal to the frequency domain. However the Fourier transform suffer from spectral leakage when it transforms an image (see Figure 3.6). The continuous Fourier transform of a periodic function requires integration to be performed over the interval $-\infty$ to $+\infty$, or over an integer number of cycles of a periodic function but Discrete Fourier Transform (DFT) leakage appears as the phenomenon in which DFT is calculated over a finite width sample of the input signal or a finite signal, which can be considered as the product of an infinite signal with a rectangular window function.

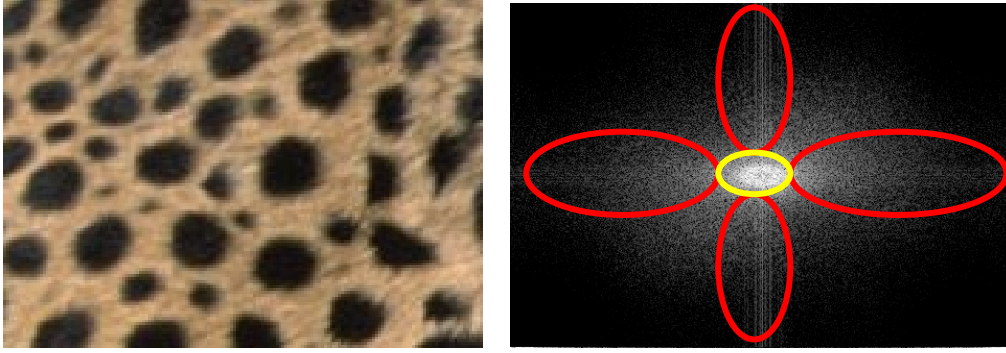


Figure 4.1 original input texture image with its Fourier Spectrum. The red ellipses show the side-lobes of leakage, which is caused by horizontal clipping and vertical clipping of the image. The yellow ellipse shows the most significant parts of the spectrum of the texture pattern.

Mathematically, the solution to DFT leakage involves suppressing the side-lobes of the window function by changing the single-bin frequency response of the DFT to approximate a rectangular function. In order to apply window functions in image processing, two-dimensional window functions must be formed from one-dimensional window functions. The separable form (outer product) and the radial form (isotropy or anisotropy) are the most common way of building two-dimensional window function.

The candidate functions for this research are Higher-order generalized cosine window functions such as Blackman window function [120]; Generalized Hamming window function[121][122], Adjustable window functions such as Kaiser window function and finite impulse response (FIR) filter designed Sinc function (domain is $[0, \pi]$) [123].

Two tests are run at this stage. The first test (candidate test) finds which window functions are good with both the reduction of side-lobes and preserving the original spectrum. The second test (forming two-dimensional window test) compares the separable form, radial form with fixed radius and radial form with unfixed radius (see more detail in Section 3.2.1.2) for forming two-dimensional window functions to investigate the different effects on the Fourier spectrum.

4.1.2 Candidate Test

The candidate test compares candidate window functions applied to a texture and original texture spectrum with leakage. Three types of texture are involved in the comparison. They are zebra texture (liner pattern), cheetah texture (circularity pattern evenly distributed) and dalmatian texture (circularity pattern with randomly distributed). The best window function should reduce as much leakage as possible, but should keep as much texture spectrum as possible.



Figure 4.2 testing images for comparing window functions

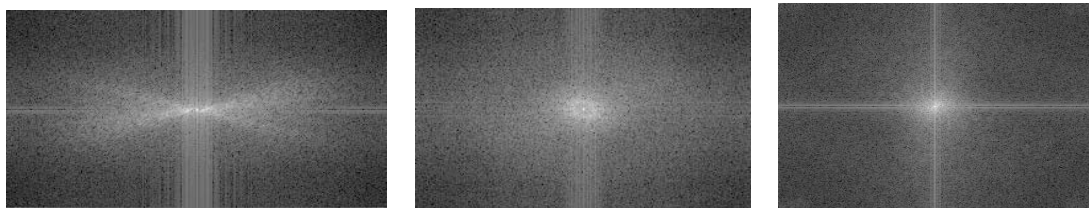


Figure 4.3 Original Fourier Spectrum

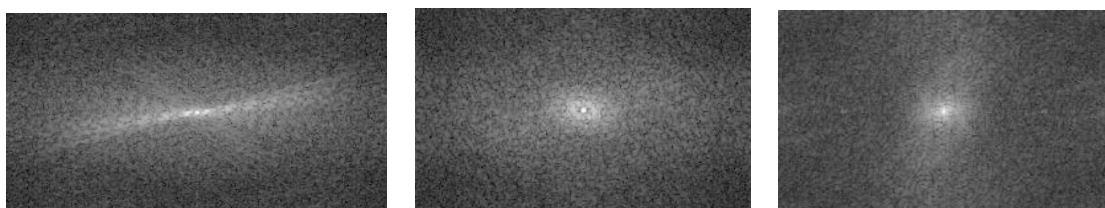


Figure 4.4 Applied Blackman Function Fourier Spectrum



Figure 4.5 Applied Hamming Function Fourier Spectrum

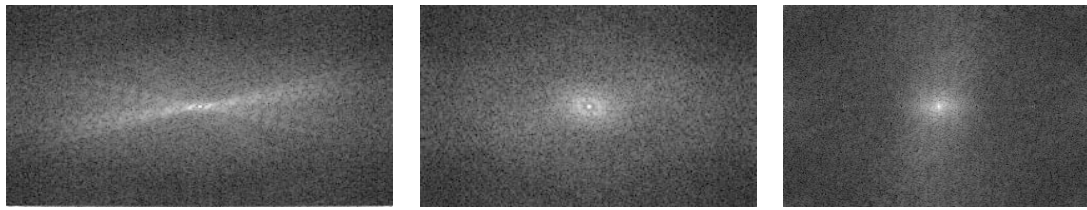


Figure 4.6 Applied Kaiser Function Fourier Spectrum

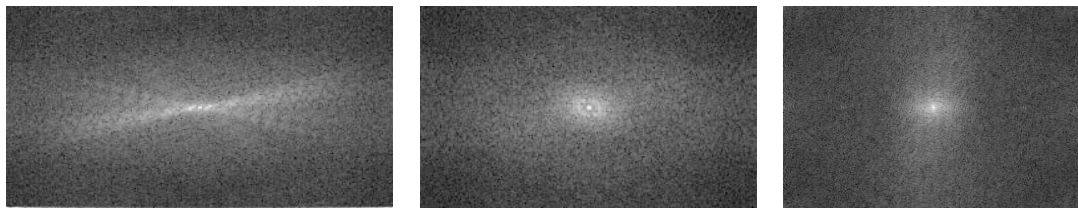


Figure 4.7 Applied Sinc Function Fourier Spectrum

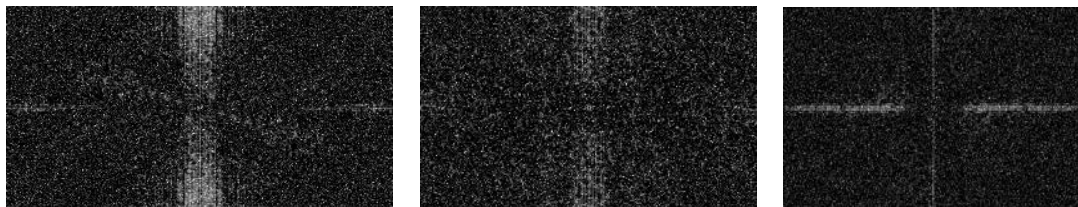


Figure 4.8 Blackman Fourier Spectrum verses Original Fourier Spectrum

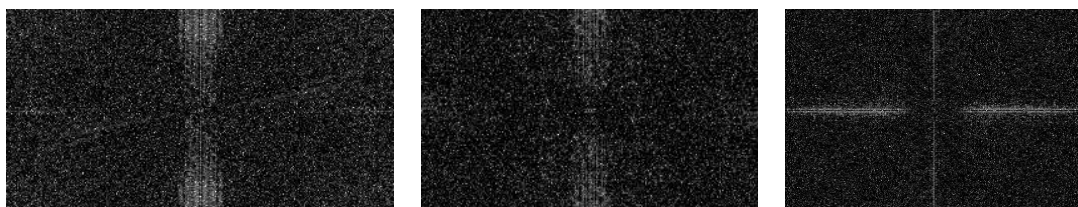


Figure 4.9 Hamming Function Fourier Spectrum verses Original Fourier Spectrum

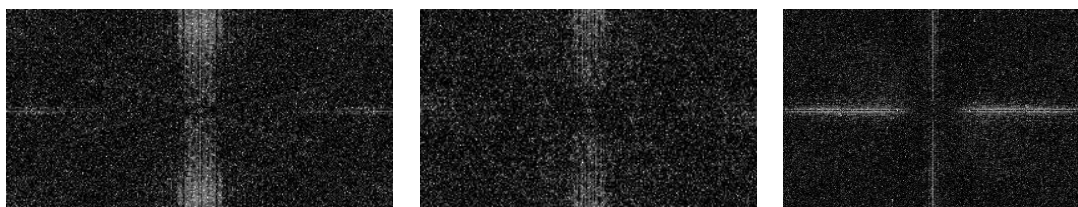


Figure 4.10 Kaiser Function Fourier Spectrum verses Original Fourier Spectrum

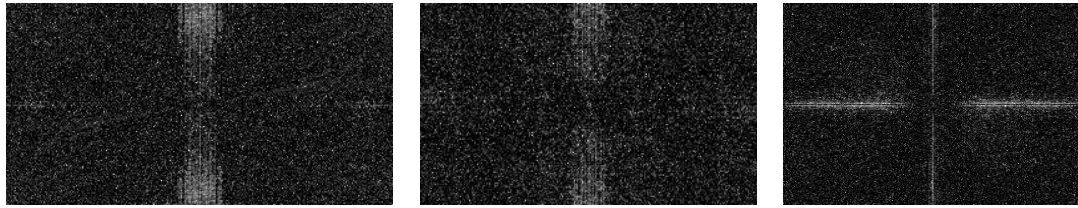


Figure 4.11 Sinc Function Fourier Spectrum versus Original Fourier Spectrum

From the test results (from Figure 4.3 to Figure 4.7) and comparing with the original Spectrum, Blackman window function changes the texture pattern spectrum. There are some original signal has been lost by the Blackman window function. In Figure 4.12, the yellow box indicates loss from the original signals.

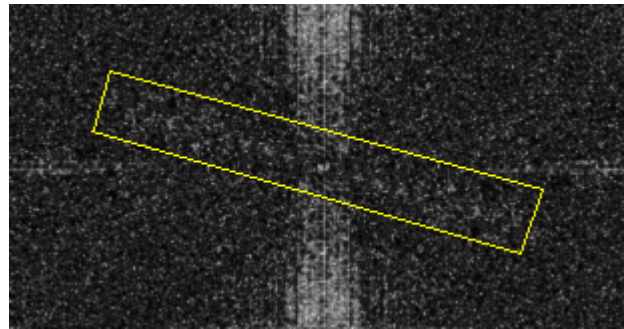
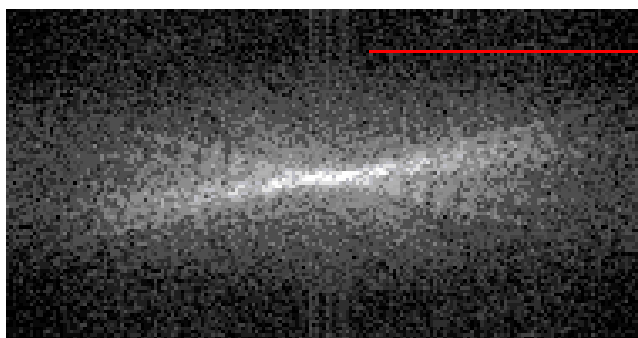


Figure 4.12 the spectrum difference between zebra texture applying Blackman window function with spectrum not applying Blackman window function.



→ side-lobes

Figure 4.13 the spectrum of Hamming window function applied to zebra texture.

Hamming window function does not reduce the side-lobes as much as other functions do (comparing the Hamming with Kaiser window function, Hamming window

function reduces 8.41% less than Kaiser window function). Sinc function and Kaiser's performance are quite similar, with 1.44% difference. (All the data in this section are calculated by taking the selected samples from the Figure 4.8 to Figure 4.11. The selection algorithm is evenly vertically cutting a image into three parts and the second part is the are taken as the sample).

4.1.3 Forming Two-Dimension Test

The separable form and radial form are the most common ways of forming two-dimensional window functions. The separable form applies a product of a one dimension window function along all the rows with one down all the columns, so that each corresponding pixel uses a two-dimensional rectangular window function. The radial form is set the image center as window function center, then rotates the window function to form a two-dimensional window function.

As an image is a rectangle with different width and height, the radial form can be categorized into fixed radius and unfixed radius. The fixed radius takes the smaller value of image's width and height as radius, so loses information contained within the image that falls outside the circle. The unfixed radius takes the length of a line segment that passes through the center point of the image and the current pixel position. The window functions are all decreasing the image intensity smoothly towards to the edge. This causes loss of the original pattern information, especially texture patterns near the edges. Forming two-dimensional window function should retain as much original texture information as possible. The similarity between the original texture pattern image and applied window functions texture pattern image is the main criterion for comparing each of those two-dimension forming techniques

The comparing test is designed to find the difference of intensity in percentage between a white rectangle image and the image that applied each forming technique.

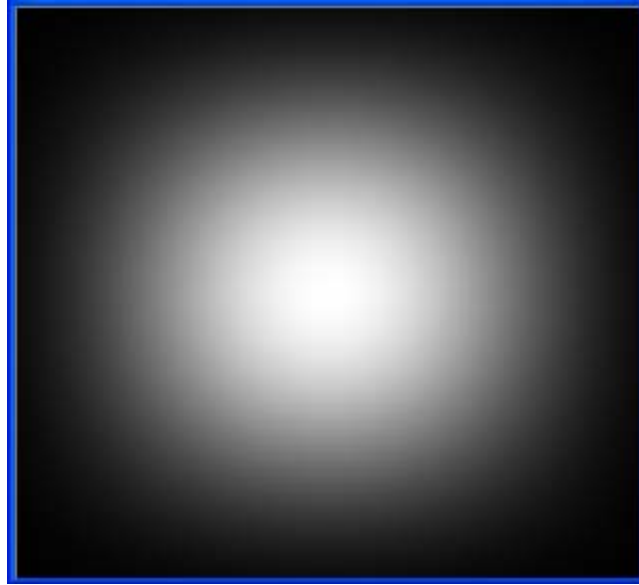


Figure 4.14 this is a white image applied Separable two-dimension form of Kaiser window function

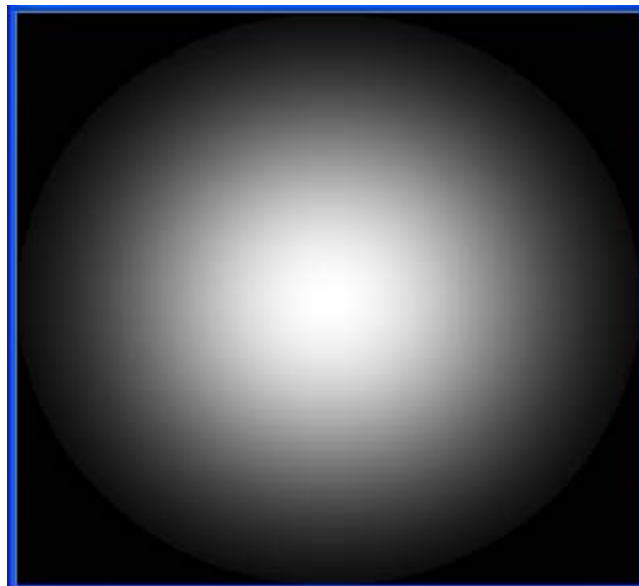


Figure 4.15 this is a white image applied fixed radius two-dimension form of Kaiser window function

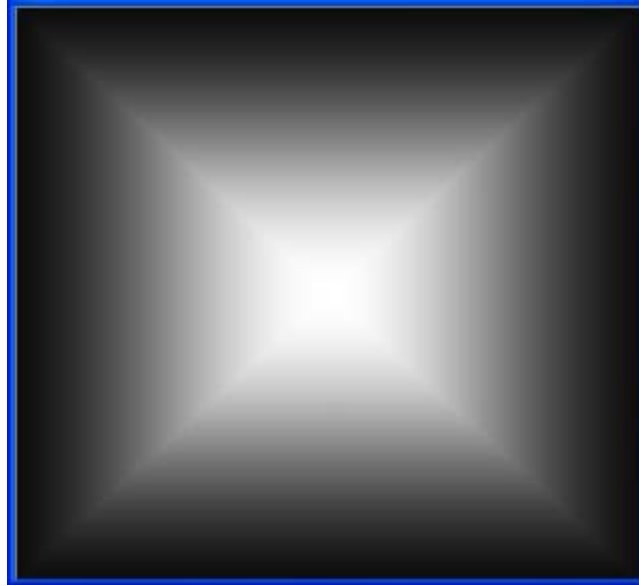


Figure 4.16 this is a white image applied unfixed radius two-dimension form of Kaiser window function

Figure 4.14, Figure 4.15 and Figure 4.16 show these three ways of forming a two-dimensional Kaiser window function. After combining the testing image (see Figure 4.17) with the two-dimension Kaiser window function, it is clearly to be seen that Radial Form (fixed radius) loses the pattern on the corner (see Figure 4.19). Separable Form weakly keeps the pattern on the corner. Radial Form (unfixed radius) keeps more detail over the entire image.

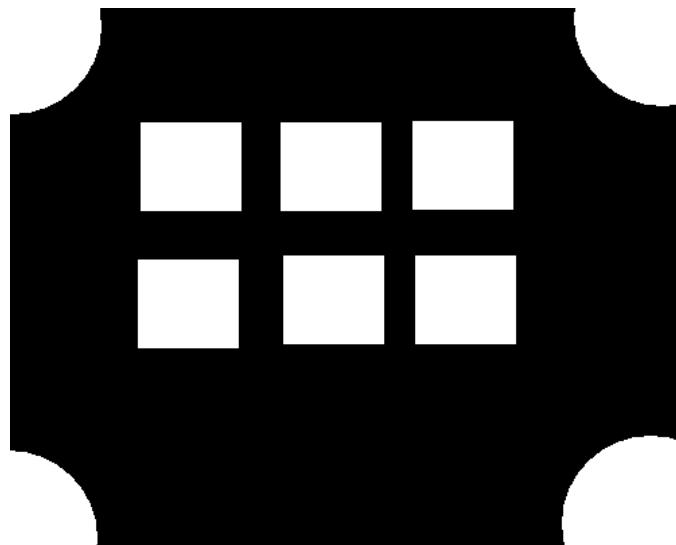


Figure 4.17 a testing image for comparing three two-dimension forming techniques

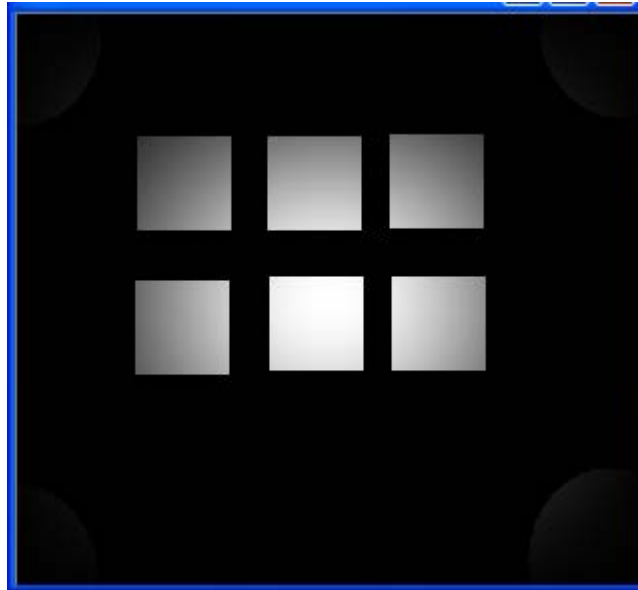


Figure 4.18 shows Separable Form applied to a two-dimensional window function then applied to the testing image.

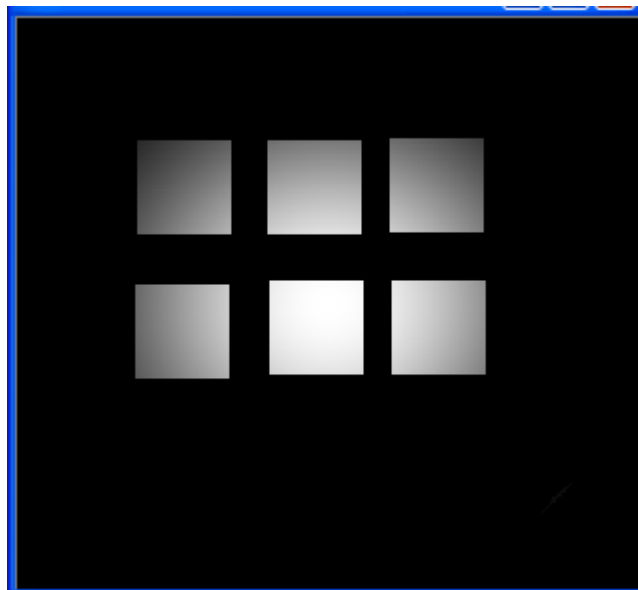


Figure 4.19 shows Radial Form (fixed radius) applied to a two-dimension window function then applied to the testing image

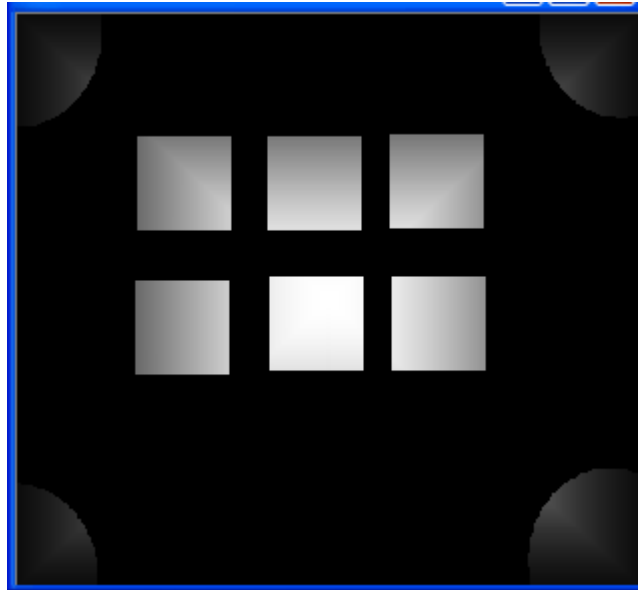


Figure 4.20 shows Radial Form (unfixed radius) applied to a two-dimension window function then applied to the testing image

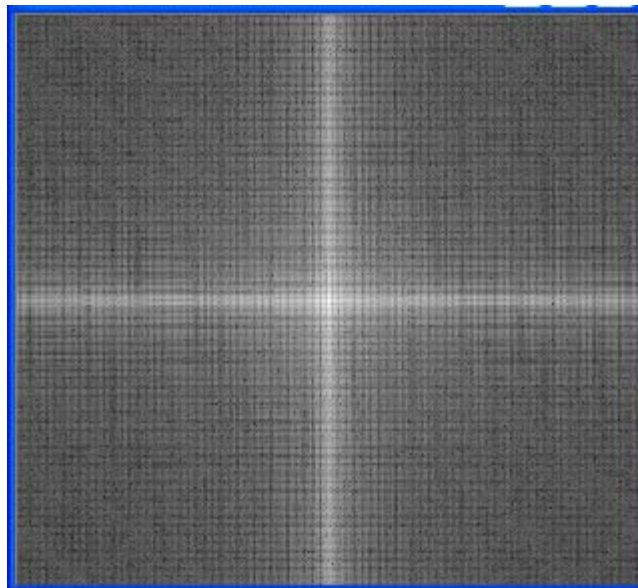


Figure 4.21 the spectrum image of testing image combined with Separable Form 2D Kaiser window function

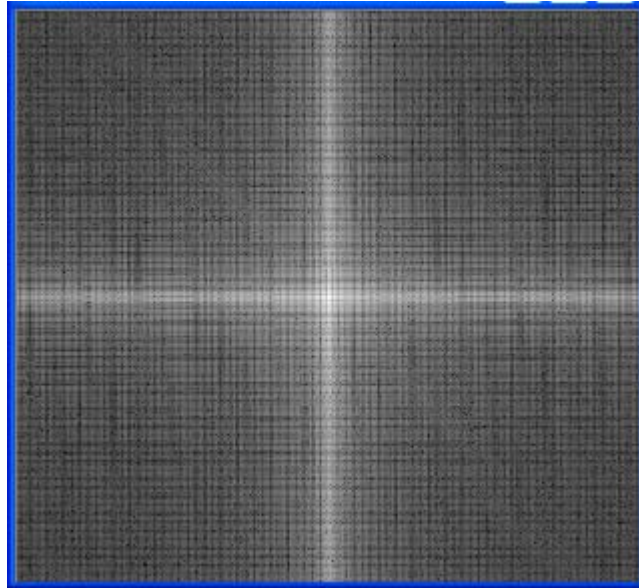


Figure 4.22 the spectrum image of testing image combined with Radial Form (fixed radius) 2D Kaiser window function

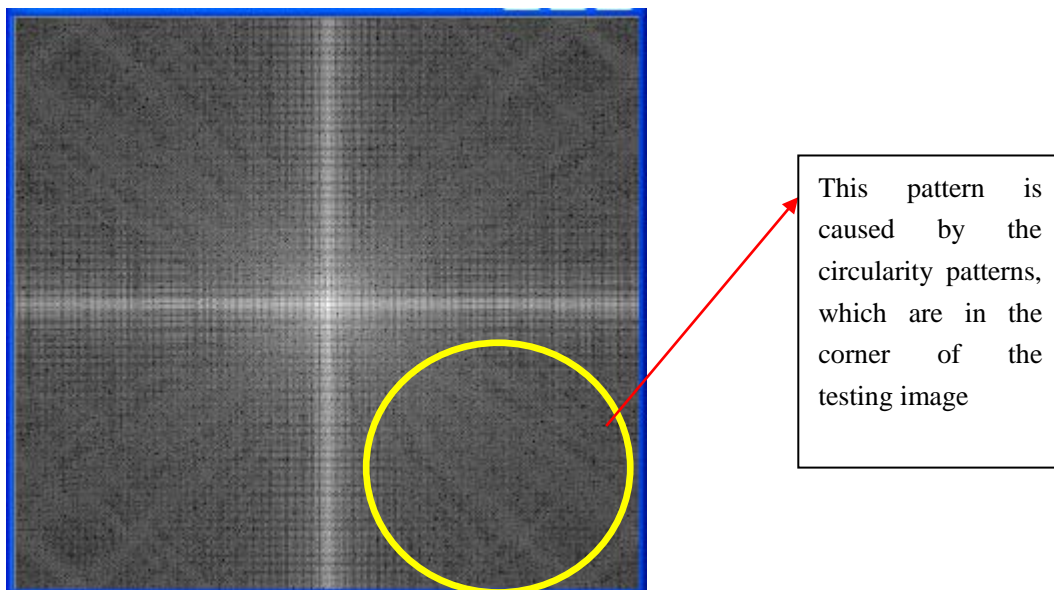


Figure 4.23 the spectrum image of testing image combined with Radial Form (unfixed radius) 2D Kaiser window function

Figure 4.21, Figure 4.22 and Figure 4.23 show spectrum results of these three forming techniques applied and passed through Fourier transform. Radial Form (unfixed radius) performs the best of these three techniques. A circularity pattern is clearly

shown on the spectrum image, but is not shown on other two-dimension forming approaches.

| Way of Forming 2D Windows | Average Intensity | Original Texture Information Retained (%) |
|------------------------------|-------------------|---|
| Separable Form | 79.68 | 31.25% |
| Radial Form (fixed radius) | 75.48 | 29.60% |
| Radial Form (unfixed radius) | 96.07 | 37.67% |

Table 4.1 a comparison of Separable Form, Radial Form (fixed radius) and Radial Form (unfixed radius). These three forms are applied on a white image and Average Intensity and Original Texture Information Retained in percentage are chosen on to compare these three forms

As the result shows in table 4.1, Radial Form (unfixed radius) retains more original texture information. Therefore it will be applied to all the Fourier transforms in this thesis.

4.2 Cleaning Spectrum Pattern

After applying Fourier transform, the spectrum values can have high dynamic range (the range could be from 0 to 10^6). When a spectrum values are represented on a image, the fainter channels part of the spectrum will not be as apparent as stronger channels. The most common way of solving this problem is to apply log function. [91]

However this increases the background details' spectrum values significantly and these background details cause much “noise” in the spectrum. It can be hard to find the spectrum pattern as the noise has almost the same intensity as the pattern has (see Figure 4.24). This will make it difficult for shape-based object recognition approaches to work out texture features such as Hu moments.

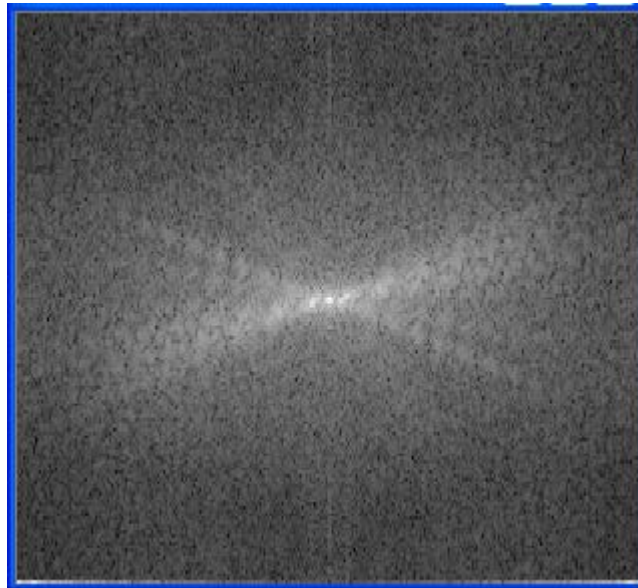


Figure 4.24 a spectrum image applied $\log_{10}(x+1)$ function. The high frequency channels contains noise.

4.2.1 Power-Law Transformations and Cosine Function

The log function shows too much details of background high frequency and the

spectrum image has lower contrast. The Power-law functions can be used which can increase the pattern intensity with higher contrast.

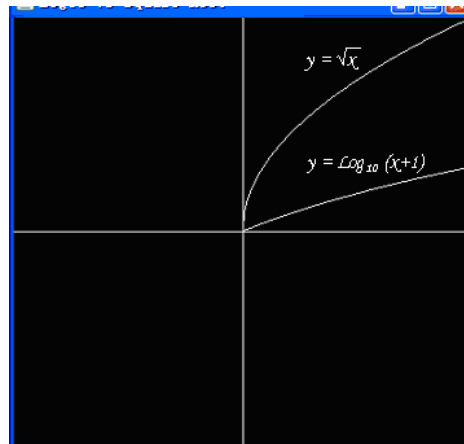


Figure 4.25 the $\log_{10}(x+1)$ function and square root function with domain from 0 to 1



Figure 4.26 the spectrum image with square root function replacing log function

Comparing Figure 4.24 and Figure 4.26, square root function amplifies the noise less significantly. The problem is that the intensity of the spectrum pattern is not high enough. If it is simply applied intensity increasing techniques, the noise will come back. The image needs to increase the contrast rather than the intensity.

The cosine function can be used to increase the contrast of the image

$$\text{Output Intensity} = (\cos(i / a \times \pi + \pi) + 1) / 2 \times 255 \quad (27)$$

where i is the intensity value of an image, a is a half period which controls which range of intensities are going to increase and which range of intensities are going to decrease. a is the boundary of the intensity for increasing or decreasing. If the pixel's intensity is higher than a the intensity will be increased; if the pixel's intensity is lower than a the intensity will be decreased.

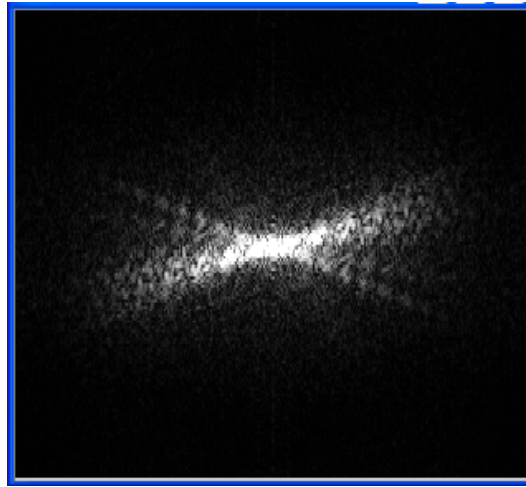


Figure 4.27 a cosine function applied after using square root function. Within this example, $a = 32$.

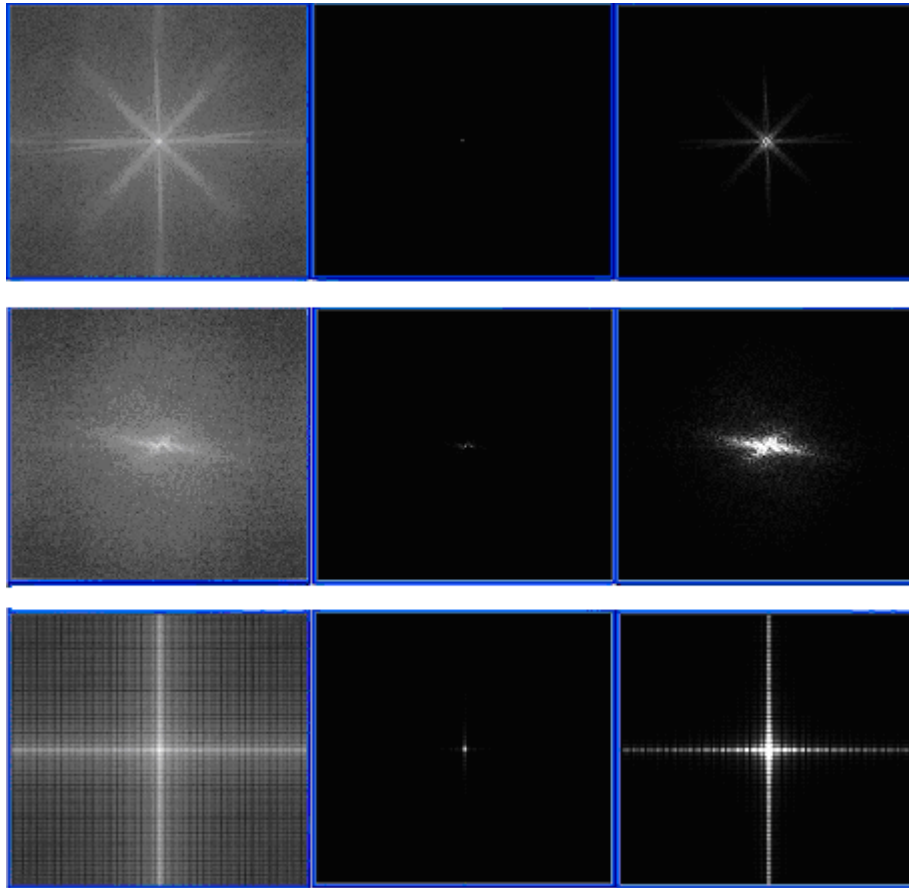


Figure 4.28 the left side images are the spectra which have had the log function applied. The middle images are the spectrums, which applied power root function and cosine function with a is 255. The right side images are the spectrums, which applied power root function and cosine function with $a = 32$.

It's clear to see that Power-Law Transformations plus Cosine Function with $a = 32$ captures the main characteristics of the frequency pattern.

Chapter 5

Texture Spectrum Feature Descriptors

This chapter investigates different ways to extract the frequency-domain based approach of texture features descriptors, and introduces two more new texture feature descriptors.

5.1 Hu Moments

Hu moments are often used to describe shape-based images [124][125]. The images can be translated, rotated, scaled and flipped without changing the Hu moments. After a texture image has been transformed to a spectrum, the spectrum pattern keeps the information of the texture pattern shape and the pattern can be described by Hu moments. The similarity of Hu moments can be calculated by formula (28)[24]

$$S = \frac{\sum_{i=1}^7 (m1i \times m2i)}{\sqrt{\sum_{i=1}^7 m1i^2 \times \sum_{i=1}^7 m2i^2}} \quad (28)$$

Where $m1i$ is the i th Hu moment of one image and $m2i$ (i is from 1 to 7) is the i th Hu moment of the other image.

However, as mentioned in the Chapter 2, Hu moments cannot describe a center symmetry pattern well. The second order moments are all zeros. Unfortunately, one of the features of spectrum pattern is center symmetry. This can reduce the accuracy for the Hu moments descriptor. In order to make the Hu moments contain more useful information, the spectrum can be cut and only half of the spectrum will be used.

Even when the image has been cut the half off to avoid symmetry problem, the Hu moments still do not work very well as described below.

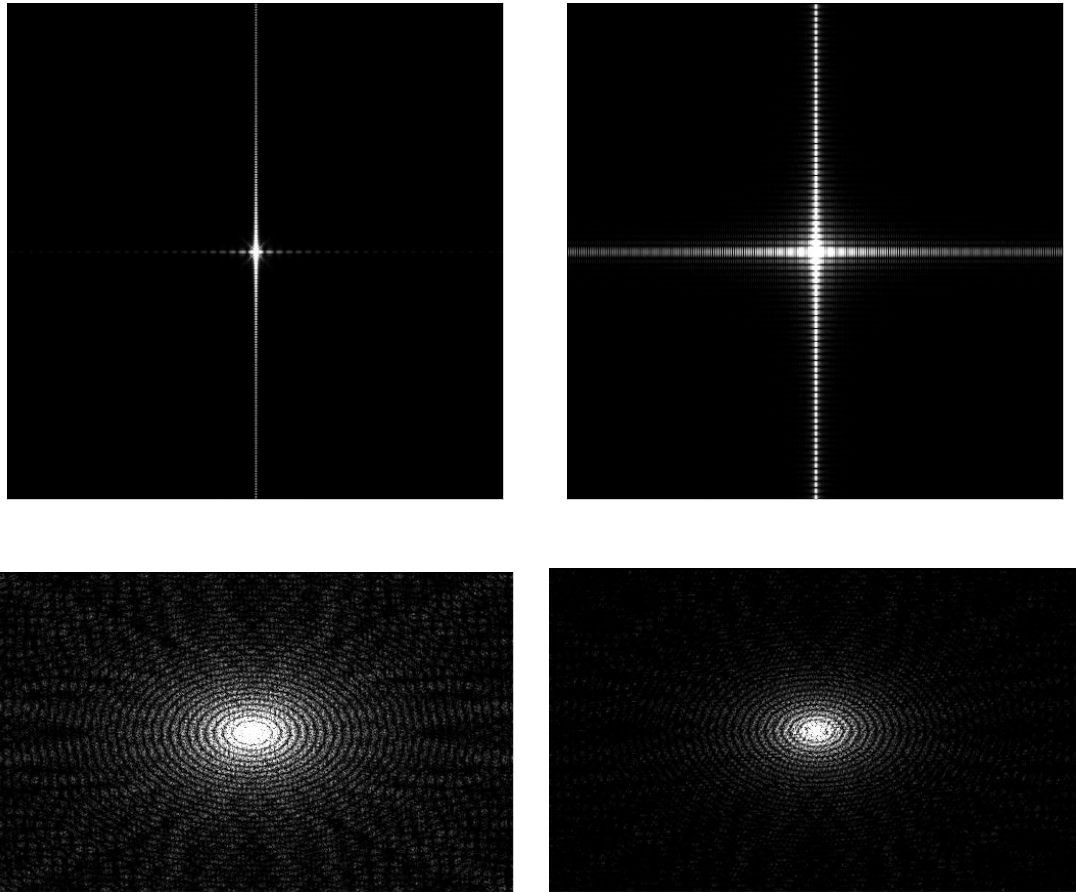


Figure 5.1 the spectrum testing images (top left is a big rectangle pattern's spectrum, top right is a small rectangle pattern's spectrum, bottom left is a small circular pattern's spectrum, bottom right is a big circular pattern's spectrum) for Hu moments

| | Big rectangle pattern's spectrum | Small rectangle pattern's spectrum | Small circular pattern's spectrum | Big circular pattern's spectrum |
|---|--|---|---|---------------------------------------|
| Big rectangle pattern's spectrum | 0 | 2.75% | 11.41% | 5.19% |
| Small rectangle pattern's spectrum | 2.75% | 0 | 10.33% | 5.03% |
| Small circular pattern's spectrum | 11.41% | 10.33% | 0 | 7.97% |
| Big circular pattern's spectrum | 5.19% | 5.03% | 7.97% | 0 |

Table 5.1 the difference for the spectrum testing images in Figure 5.1. The percentages are calculated by average difference of each pixel over 255 (getting average of the absolute values of the differences between each corresponding pixels).

| | Big rectangle pattern's spectrum | Small rectangle pattern's spectrum | Small circular pattern's spectrum | Big circular pattern's spectrum |
|-------------|--|--|---|---------------------------------------|
| Φ_1 | 0.143138 | 0.0255142 | 0.00885293 | 0.039267 |
| Φ_2 | 0.0072561 | 0.00008.49876 | 0.0000478299 | 0.00127028 |
| Φ_3 | 3.80845 | 1.64821 | 0.734671 | 30.0512 |
| Φ_4 | 25.80845 | 5.83153 | 0.604139 | 39.503 |
| Φ_5 | 133.744 | 6.10695 | 0.384114 | 1361.04 |
| Φ_6 | 2.01868 | 0.0525726 | 0.00339052 | 1.40178 |
| Φ_{7s} | 218.082 | 17.0165 | 0.120219 | 7.09784 |

Table 5.2 the Hu moment invariance values for each testing images in Figure 5.1

| | Big rectangle pattern's spectrum | Small rectangle pattern's spectrum | Small circular pattern's spectrum | Big circular pattern's spectrum |
|---|--|---|---|---------------------------------------|
| Big rectangle pattern's spectrum | 1 | 0.9554 | 0.3614 | 0.5274 |
| Small rectangle pattern's spectrum | 0.9554 | 1 | 0.4634 | 0.3355 |
| Small circular pattern's spectrum | 0.3614 | 0.4634 | 1 | 0.4049 |
| Big circular pattern's spectrum | 0.5274 | 0.3355 | 0.4049 | 1 |

Table 5.3 the similarity of Hu moment for the images in Figure 5.1. The numbers are calculated using Formula (28).

Table 5.1 shows the difference for the spectrum testing images. It's clear that the Small circular pattern's spectrum image has the most difference compared to the other images. And it has strongest energy at the high frequency channels as show in Figure 5.1. All the other images have lower energy at the high frequency channels. If the Hu moments are calculated for the spectrum images, the high frequency channels will have a large influence on the moments. Unfortunately, the purpose of applying Hu moments is trying to compare the signals at the low frequency channels only. This can be failed when the low frequency channels and the high frequency channels have similar strength energy.

As a conclusion it appears that Hu moments do not provide a good description for the above spectrum patterns although Hu moments invariance have been widely used for recognizing all kind of objects and patterns. However, symmetric object have some zero moments and so possibly lose the power of discrimination. If the object is centrosymmetric, all odd-order moments are identically equal to zero [126]. Furthermore all the spectrum images have some common features that cause similar Hu moments, so less able to distinguish between images:

- Firstly, the spectrum patterns are linear or circular, they all have higher energy at the lower frequency part of the spectrum.
- Secondly, when the size of the texture pattern is changing, it changes the wavelength on the spectrum (see Section 2.5.3.2).
- Thirdly, the circular pattern cannot represent as a perfect circle, because of the square shaped pixels [92]. The circular pattern cannot be perfectly transformed to a circle and it does has some linear effects on the spectrum.

All of above reasons causes Hu movement to not perform well for describing spectrum features..

5.2 Tamura's Texture Features And Spectrum Texture Features

Tamura gave six features to describe a texture. They are coarseness, contrast, Directionality, Linelikeness, regularity and roughness. One limitation of Tamura's Texture Features is the Directionality. Tamura texture Directionality algorithm works well on a natural texture image but not on a computer-produced pixelated texture image (see more details in Section 5.2.7). The Linelikeness, the regularity and the roughness are related with the first three features and are not used for CBIR and they are not very effective for texture recognition [68].

These features can also be described in frequency domain by the spectrum. Some of them such as Linelikeness and Directionality are more accurate in frequency domain rather than in the spatial domain because Frequency domain approach focuses on the entire texture image rather than on pixels within local neighbourhoods.

5.2.1 Coarseness Descriptor on Texture Pattern Size

Coarseness has a variety of meanings. It can be the opposite measurement of smoothness which is the variations of grey levels [127][128]. It also can be a description of Texture Pattern Size which is how frequent the grey levels change [24].

The size of a texture pattern and smoothness both affect the distribution of energy in the spectrum image. The definition of coarseness used in this research is the Pattern Size and the smoothness is considered as the low frequency part of a spectrum.

The contrast affects the energy distribution in the spectrum image. The directionality affects the direction of the energy that is distributed in the spectrum image. The research discusses the contrast, directionality and image size variance approach of coarseness on Texture Pattern Size.

As mentioned in Section 2.5.3.2, the relation between Texture Pattern Size and the distance between each peak on the spectrum (one periodic) is N/a , where N is the image size (the number of pixels) and a is the Texture Pattern Size in pixels. a/N measures number of periodics for each pixel (An extreme case will be one periodic per pixel then $a/N = 1$, which means the texture image is a constant intensity image). When a (the Texture Pattern Size) gets bigger, more details of a periodic have to be represented by one pixel. When a is almost closes to N , one pixel contains almost one periodic and the pixel value is the average intensity of the periodic, which makes spectrum pattern smoother. This gives a possible way of describing the coarseness of the texture in frequency domain. The smoothness of the spectrum pattern changes of pixels can be used for describing the Texture Pattern Size.

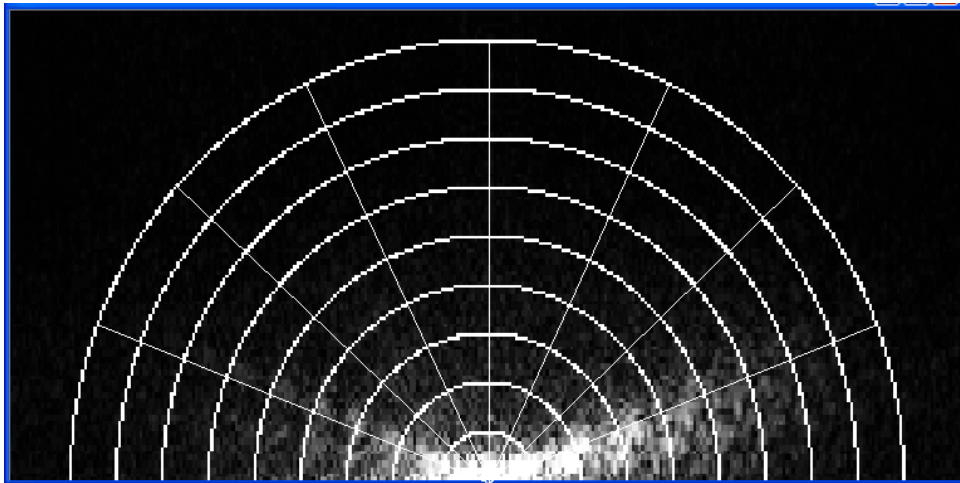


Figure 5.2 a spectrum image with the bins.

The spectrum bins approach takes the top half of the spectrum image and subdivides the spectrum into bins (see Figure 5.2). These bins are designed based on equations (24) and (25). Each bins measures different angle range and distance range. The bins' values are the average intensity of each pixel in the bins This design allows the bins to be compared to each other even they may contain different numbers of pixels.

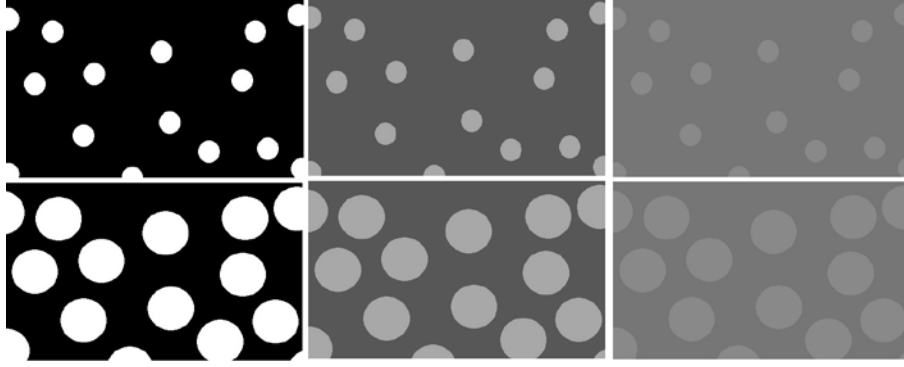


Figure 5.3 The testing images of comparing the smoothness for different contrast and Texture Pattern Size the top three images contain a small pattern, the bottom three images contain a larger pattern. The contrast is getting lower from the left to the right.

The spectrum energy can be affected by coarseness, contrast and intensity. Intensity only affects the zero frequency. If the zero frequency's energy is not considered by the algorithm, then the intensity effects can be removed.

In order to be directionally invariant, the algorithm groups all the bins with the same direction then sums each group bin values. The group with the biggest sum value is chosen as the sample to calculate the coarseness descriptor.

The formula of changes of brightness [129] is:

$$g(x, y) = f(x, y) + r \quad (29)$$

where $f(x, y)$ is the original intensity of the pixel and r is the additive factor of changes of contrast.

For contrast the formula [129] is:

$$g(x, y) = (f(x, y) - 128)r + 128 = f(x, y)r - 128r + 128 \quad (30)$$

The formula is divided into two parts $f(x, y)r$ and $-128r + 128$. Only $f(x, y)r$ has an effect on the contrast, but not $-128r + 128$, and the contrast effect is a liner relationship between $f(x, y)$ and $g(x, y)$. Base on the formula, the changes of contrast also causes spectrum energy times r .

For reducing the contrast effect on the spectrum energy, the algorithm of Texture Pattern Size, which is designed by this thesis, starts from the origin, and takes each bin divided by its neighbouring bin, which is father away from the spectrum's origin. Then the ratios are used for coarseness calculation, and effect of r can be reduced (See Figure, 5.4)

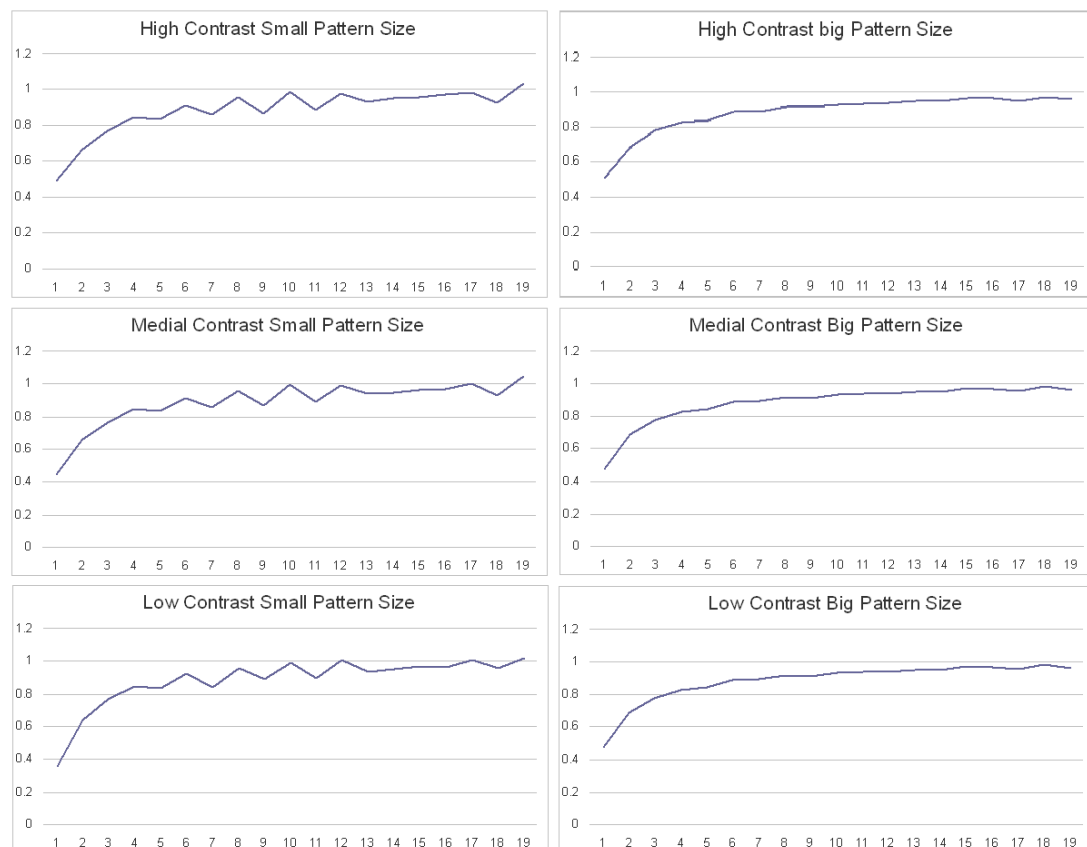


Figure 5.4 the ratios of the images from Figure 5.3. The y-axis is the ratios of the neighbor bins. The x-axis is the bins.

Each ratio is averaged by its two neighbours, then the difference between the ratios found and its corresponding average. In the end, the average of the differences can be

used for describing the smoothness of the spectrum pattern and the size of the pattern.

| Descriptions of testing image | Average of the difference between smoothed and original ratio | Descriptions of testing image | verage of the difference between smoothed and original ratio |
|------------------------------------|---|----------------------------------|--|
| High Contrast Small Pattern Size | 0.036 | High Contrast Big Pattern Size | 0.010 |
| Medial Contrast Small Pattern Size | 0.039 | Medial Contrast Big Pattern Size | 0.010 |
| Low Contrast Small Pattern Size | 0.040411 | Low Contrast Big Pattern Size | 0.015 |

Table 5.4 the result of smoothness testing for contrast tolerance.

Table 5.4 shows the results of smoothness testing and these results can be used as the texture coarseness descriptor. The algorithm compresses the effect of the texture contrast. However when contrast changes a lot, the result of the algorithm will be affected by the contrast.

This algorithm is also invariant to image size. The distance between each peak is N/a . If the image's scaling factor is s , the distance between each peak is $s N/a$.

$$s N / (s N/a) = a \quad (31)$$

Equation (31) gives the number of periods in the image. The number of periods depends on the size of texture pattern. Therefore changes of the image size do not affect the result.

5.2.2 Calculation of Texture Pattern Size

Another approach of finding the Pattern Size is by measuring the size of the spectrum pattern. The spectrum pattern is dependent on the size of the texture pattern. Ideally the spectrum Pattern Size can be used as a description of texture coarseness.

It has been mentioned in Chapter Two that the Pattern Size in spectrum image in x and y direction are N/a and M/b , where a and b are the width and height of the rectangle texture pattern which is in a $M \times N$ pixels image. If a texture pattern is a circular pattern with the diameter d , the Pattern Size in spectrum image will be $1.22 \times N/d$.

If the trough of each periodic can be found in the spectrum, then the size of the spectrum pattern can be calculated and the coarseness of the texture can be described.

| | Small Pattern Size | Big Contrast |
|-----------------|--------------------|--------------|
| High Contrast | 113 | 106 |
| Medial Contrast | 113 | 106 |
| Low Contrast | 122 | 106 |

Table 5.5 shows the position of the first trough in pixel for testing images in Figure 5.3. all the spectrum images have been blurred

Unfortunately, the trough can be misled by the texture of the spectrum pattern itself (see Figure 5.5). Blurring the image can be the way to reduce this problem but it can cause finding the trough harder when the contrast is too low.

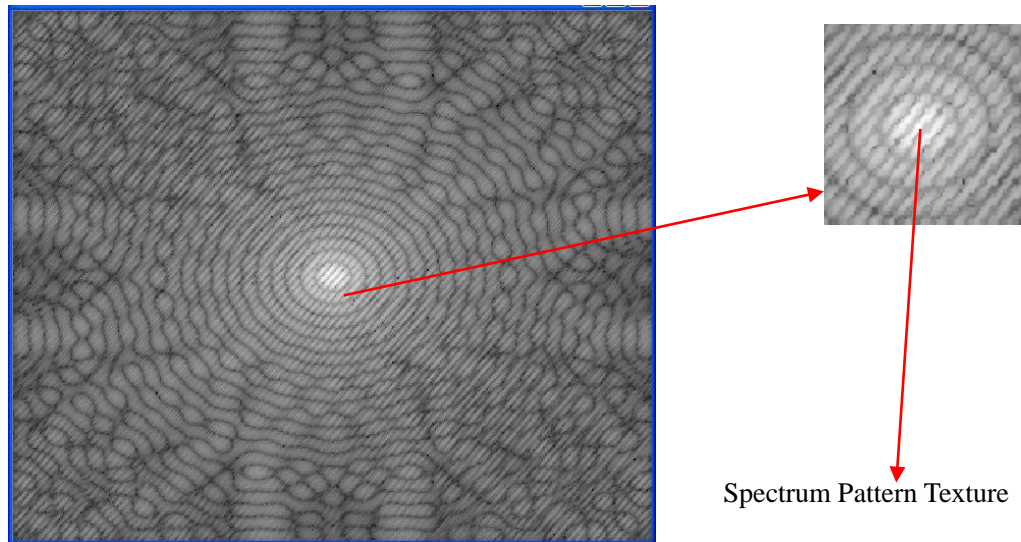


Figure 5.5 the texture of the spectrum pattern

Not all the spectrum patterns have texture, but if the image in spatial domain has more than one repeated pattern, then it possibly has spectrum pattern texture. These textures can mislead finding the position of trough. (see more useful application of spectrum pattern texture details in section 5.3)

5.2.3 Contrast

Contrast is represented as energy (or intensity) in the frequency domain. Therefore the total energy can be used to describe the contrast of a texture. However it can be affected by the Texture Pattern Size. Only when the Pattern Sizes are the same, then the contrast can be described by the total energy of the spectrum image.

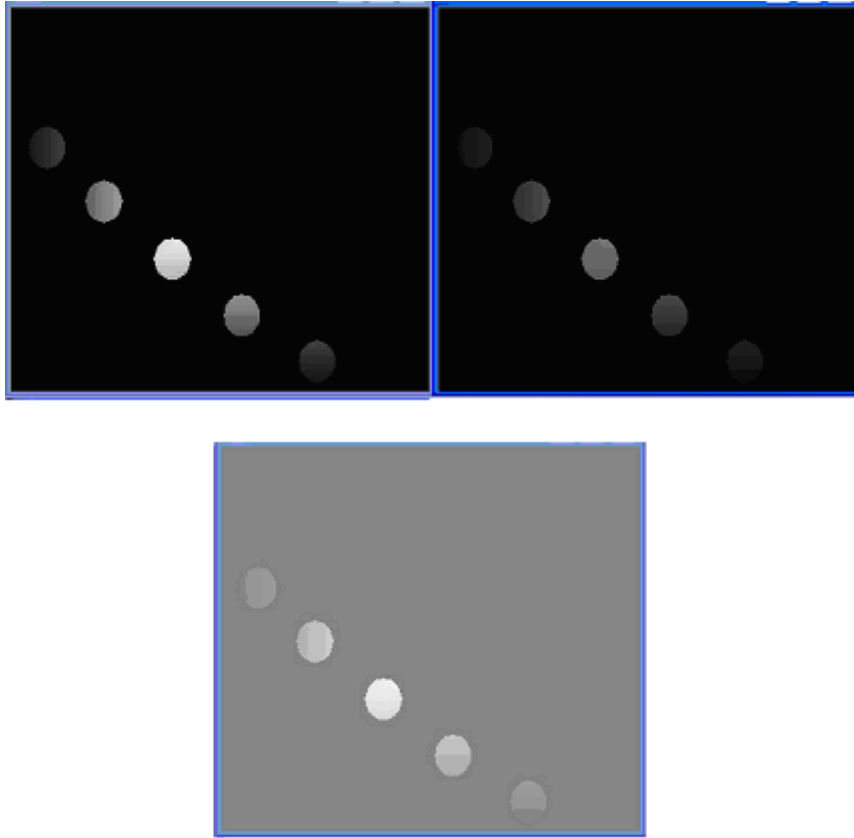


Figure 5.6 the top left image is the original image that is combined with Kaiser window function. The top right image is the reversed FFT image with halved energy in frequency domain. The bottom image is the reversed FFT image with halved energy in frequency domain and double the intensity in spatial domain.

However, contrast has an issue to be one of the texture features. The contrast of a texture image can be significantly affected by lighting. The spectral Response Function determines that the surface gray level is depended on the surface colour reflection rate and the lighting intensity. Contrast measures the difference of gray levels (intensity level) in an image. Therefore when the lighting intensity changes, the range of the maximal or minimal intensity value of each pixel will be changed. This causes the contrast of a texture image to also be changed. In one word, if intensity is getting higher or lower, contrast will be lower.

Suppose the lighting is getting dark and each pixel's intensity is ld (l is the original intensity, d is the lighting factor). Here are the formulas of Tamura contrast.

From formula (14)

$$(\sigma^d)^2 = (q_0 - m)^2 + (q_1 - m)^2 + \dots + (q_{\max} - m)^2 d^2$$

where $(\sigma^d)^2$ is the σ^2 with the lighting factor d

From formula (15)

$$\mu_4^d = ((q_0 - m)^4 + (q_1 - m)^4 + \dots + (q_{\max} - m)^4) d^4$$

$$\alpha_4^d = \mu_4 / \sigma_4 = \alpha_4$$

where μ_4^d is the μ_4 with the lighting factor d

From formula (13)

$$F = \sigma / (\alpha_4)^n ; F^d = \sigma / (\alpha_4)^n d$$

where F^d is the F with the lighting factor d

The other formula for contrast is Root mean square (RMS) contrast formula it is

$$Contrast = \sqrt{\frac{1}{MN} \sum_{i=0}^{N-1} \sum_{j=0}^{M-1} (I_{ij} - I_{mean})^2} \quad (32)$$

Obviously, formula (32) is also affected by the intensity factor d . Therefore the most common algorithm of contrast descriptions have difficulty to avoid effects from the lighting source.

5.2.4 Linelikeness

As mentioned previously, the coarseness and contrast are easily affected by the environment. Therefore coarseness and contrast are not good descriptors of texture features.

Textures can be described by Linelikeness. Texture patterns that are made up by lines

or curves will be represented as lines or curves respectively in the spectrum image. Linelikeness measures how the pattern is likely to be a line and whether it can be found by a spectrum bins based approach (see 5.2.1 Figure 5.2) and the spectrum bins based approach is translation independence.

If the texture pattern is likely to be lines, then in the spectrum image there will be some strong energy (some pixels with high intensity in the spectrum image) along a certain direction and this direction will be strong compared to the other directions. This can be measured by modified RMS contrast formula (see formula 33) where I_i is the directional bins' values and M is the total number of directions:

$$Linelikeness = \sqrt{\frac{1}{M} \sum_{i=0}^{M-1} (I_i - I_{mean})^2} \quad (33)$$

All bins' values with the same angle range are summed up. Then each sum divides the maximum of these sums, therefore the bins value range will be between 0 to 1 and different size or different contrast of the texture patterns can be compared.

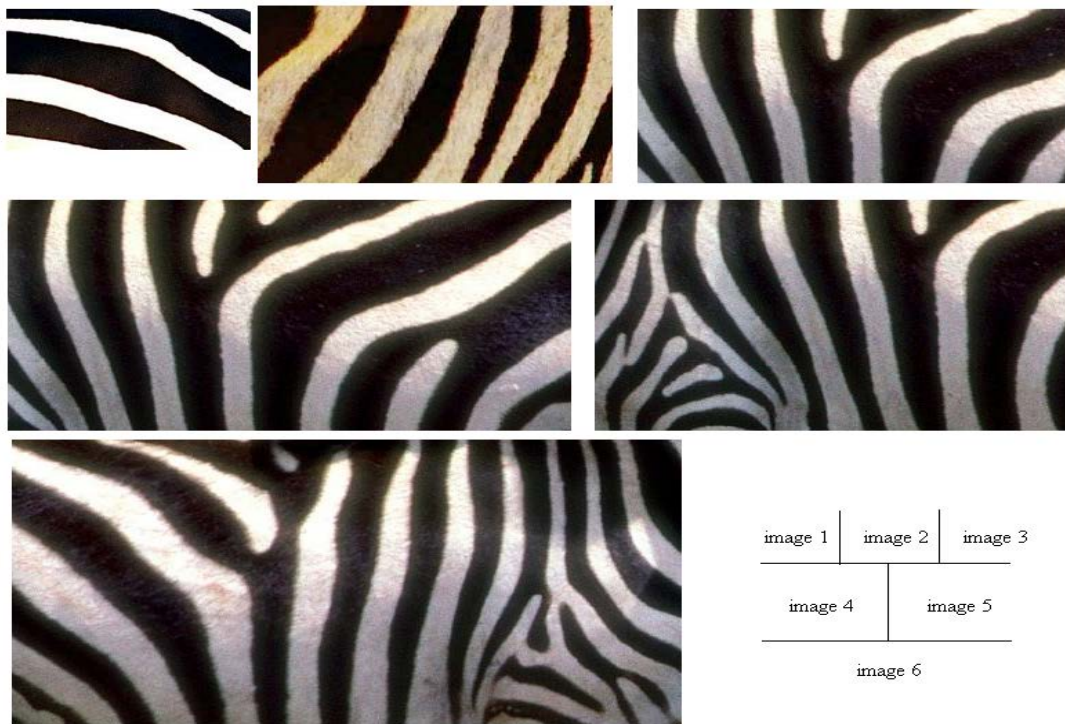


Figure 5.7 zebra Linelikeness testing texture images

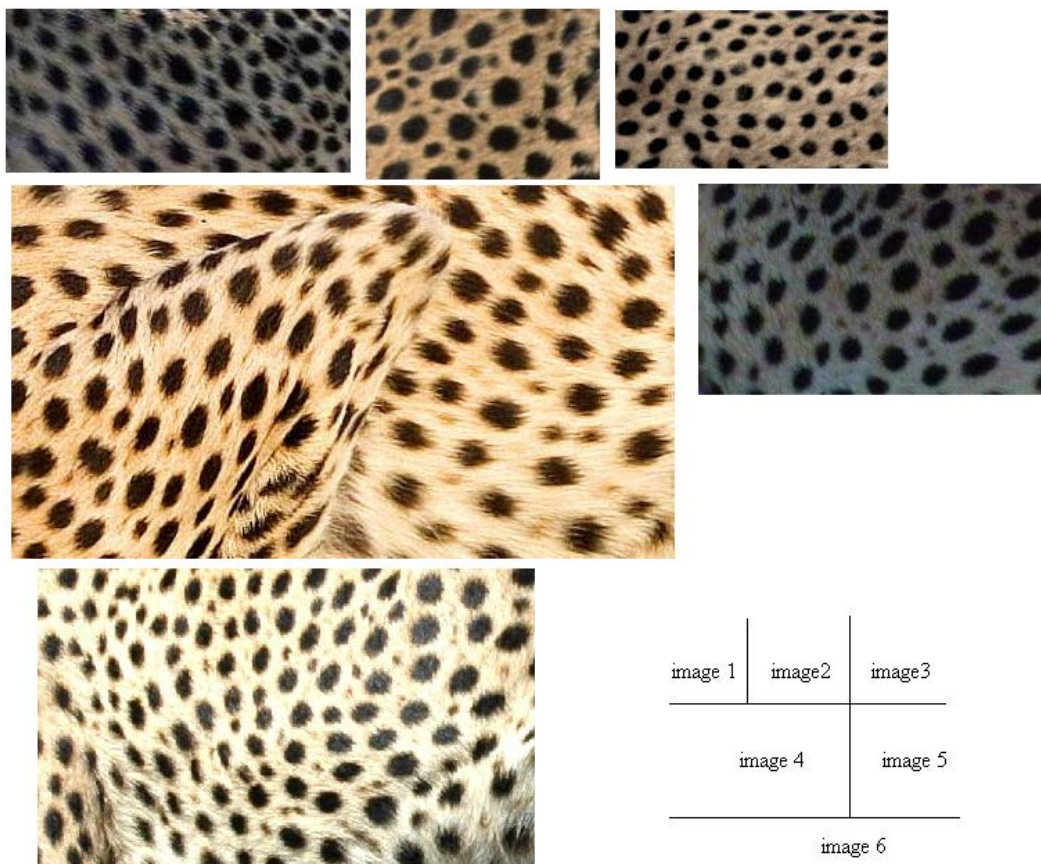


Figure 5.8 cheetah Linelikeness testing texture images

| Image Name | Linelikeness | Image Name | Linelikeness |
|------------|--------------|------------|--------------|
| Zebra 1 | 0.3313 | Cheetah 1 | 0.2090 |
| Zebra 2 | 0.3276 | Cheetah 2 | 0.0812 |
| Zebra 3 | 0.3216 | Cheetah 3 | 0.1040 |
| Zebra 4 | 0.3231 | Cheetah 4 | 0.2344 |
| Zebra 5 | 0.3009 | Cheetah 5 | 0.2320 |
| Zebra 6 | 0.3283 | Cheetah 6 | 0.2029 |

Table 5.6. Linelikeness testing results of Figure 5.7 and Figure 5.8

From table 5.6, it is clearly to be seen that there are basically two texture pattern groups. They are above 0.30 and around 0.20 groups (Cheetah 2 and Cheetah 3 have less hair effect and the Linelikeness values are lower), and they are belong to zebra and cheetah. Respectively even convex polygon, which has straight edge, can be detected as low Linelikeness. There are two testing examples (See Figure from 5.9 to Figure 5.12). A line based texture pattern (Figure 3.44) has high Linelikeness value which is 0.38684

However one thing should be pointed out. If the spectrum is cut more slices in a particular direction, line likely patterns' Linelikeness value changes more (All the data of table 5.6 is based on 5 directions).

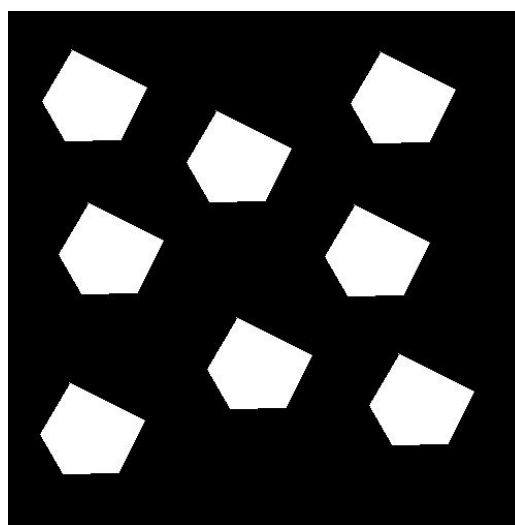


Figure 5.9 a pentagon shape pattern texture with Linelikeness is 0.1936

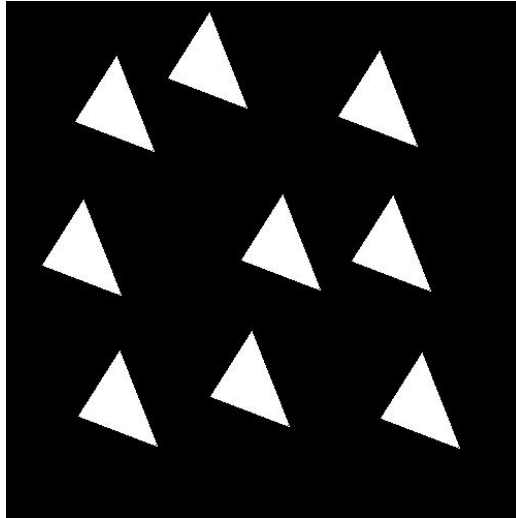


Figure 5.10 a triangle shape pattern texture with Linelikeness is 0.1888

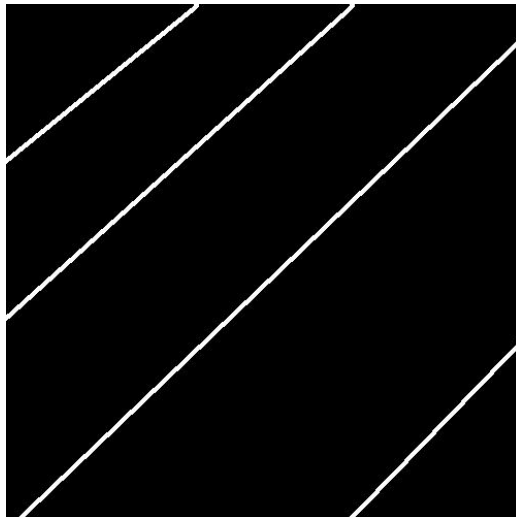


Figure 5.11 Linelikeness value of a line-based pattern is 0.3868.

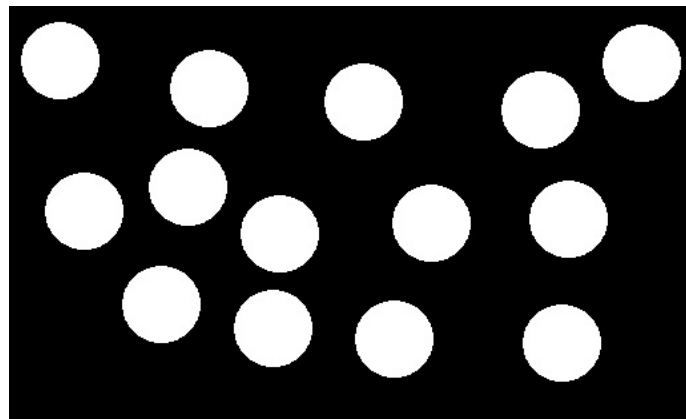


Figure 5.12 Linelikeness value of a line-based pattern is 0.1492.

5.2.5 Texture Pattern Edge Directional Unification Value

The texture pattern edge Directional Unification measures whether and how the line of the edges are directionally unique. If the lines on the edges are in the same direction angle, then on the spectrum image, all the strong signals will lie along one direction, which is perpendicular to the direction in spatial domain. Therefore that direction will stand out from all the other directions on the spectrum image. Basically after passing the spectrum image to the spectrum bins, and calculating the sum of values for each direction, one minus the quotient of the average of the sums over the maximal of the sum gives the Directional Unification value (see formula 34).

$$1 - \text{AVERAGE}(s_1, s_2, s_3 \dots s_n) / \text{MAX}((s_1, s_2, s_3 \dots s_n)) \quad (34)$$

where s_n is the sum of the spectrum intensity for one direction. This is similar idea to texture Linelikeness, but Linelikeness measures how widely the energy range of directions is. The Directional Unification measures how strong the outstanding direction is compared with overall energy in all directions.

| Image Name | Directional Unification Value | Image Name | Directional Unification Value |
|------------|-------------------------------|------------|-------------------------------|
| Zebra 1 | 0.6347 | Cheetah 1 | 0.3494 |
| Zebra 2 | 0.7143 | Cheetah 2 | 0.2180 |
| Zebra 3 | 0.6727 | Cheetah 3 | 0.2567 |
| Zebra 4 | 0.5824 | Cheetah 4 | 0.5037 |
| Zebra 5 | 0.5974 | Cheetah 5 | 0.3791 |
| Zebra 6 | 0.6808 | Cheetah 6 | 0.3210 |

Table 5.7 Texture Pattern Edge Directional Unification testing results of Figure 5.7 and Figure 5.8

Table 5.7 shows that zebra texture's edge directions are more unique than cheetahs'

(the zebra's average of Directional Unification value is 0.6471 and cheetah's average of Directional Unification value is 0.3380). Image Cheetah 4 is a little bit different with all the others. This is because the cheetah's shoulder is in the image, which make some strong linear edges and these linear edges are almost in the same direction.

If a circular texture is considered as a multiple line segments in variant directions, then the circular texture will has lowest value of Directional Unification.

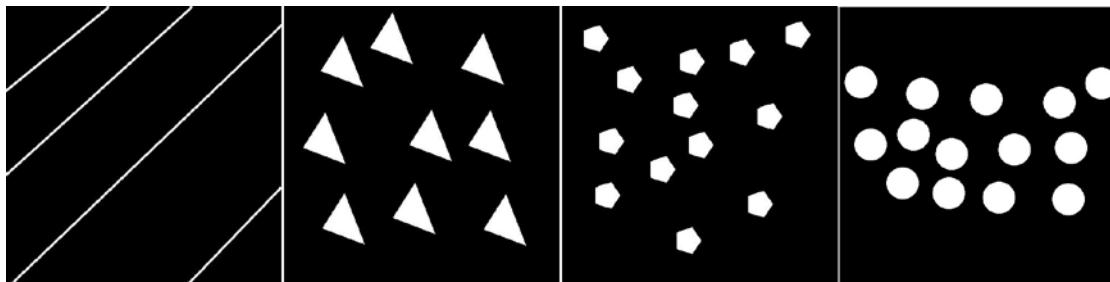


Figure 5.13 a set of testing texture images for texture pattern edge Directional Unification test

| Linear texture | Triangle texture | Pentagon texture | Circular texture |
|----------------|------------------|------------------|------------------|
| 0.9078 | 0.6179 | 0.3673 | 0.2138 |

Table 5.8 Comparing Texture Pattern Edge Directional Unification for different shapes of texture pattern (testing images are from Figure 5.13)

Table 5.8 clearly shows that linear texture pattern with the same directional angle has highest value of Directional Unification and the circular texture pattern has the lowest value of Directional Unification.

5.2.6 Directionality

Some texture patterns such as triangles or polygons, which have strong straight edges in certain directions, can be described by the directionality degree of these edges. If a texture pattern's edge is in θ radians, then the edge will be represented in the spectrum at an angle $\theta \pm \pi/2$. Therefore if the degree of the angle in spectrum can be found, then the texture pattern's angle is also found.

The spectrum bins approach, which has been discussed in previous sections, is suitable for the job of finding edge directionality degree. The bins are allocated by the distance ranges and angle ranges. Comparing overall bins in different angle range, if there are strong energies in a certain angle range, then the original texture pattern's edge directionality degree must be that angle range's degree plus or minus $\pi/2$.

After all the energies has been allocated into the bins by the angle range and the distance ranges, the sum of the energies which are in the same angle range can be calculated, and the angle range scanned in the order of the angles. The peaks of the angle range is the directionality degree of the texture pattern's edge angle on the spectrum image.

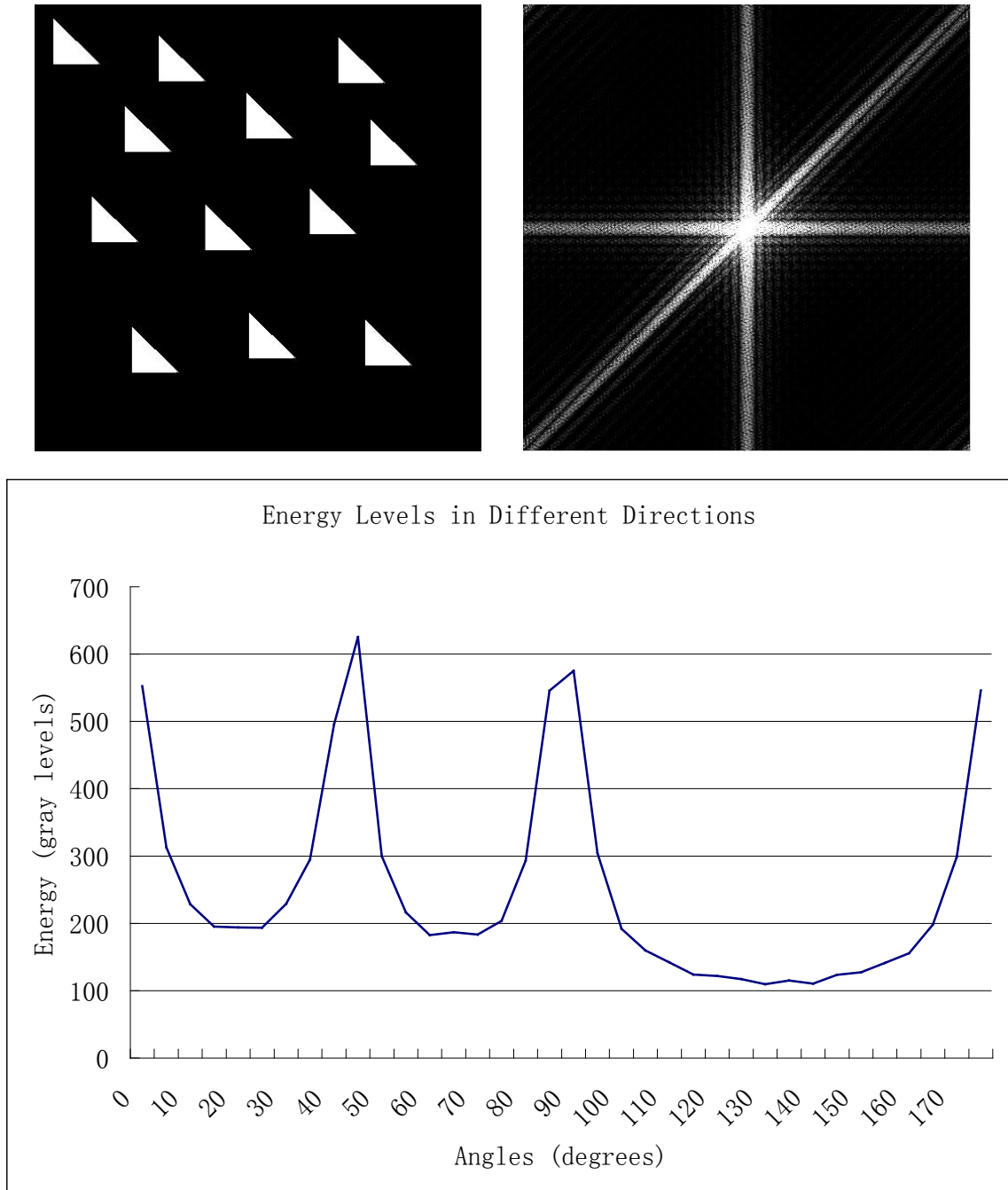


Figure 5.14 edges' directionalities of a right-angled triangle with 45 degree texture pattern. Top left image is the original triangle texture pattern. Top right image is the spectrum of the triangle texture pattern. Bottom image is the Energy Values of different angles range (bin range is 5 degrees).

Figure 5.14 shows a testing example of spectrum bins approach for finding the directionalities for each edge of the triangle. There are four peaks for the image. They

are 0 to 5 degree, 45 to 50 degree, 90 to 95 degree and 175 to 179 degree. 0 to 5 degree and 175 to 180 degree can be treated as one angle because mathematically 0 degree line and 180 degree line are in the same direction and the spectrum is symmetric about its centre. Now, it is clearly seen that the texture pattern has three lines and they are in 180 ($180 = 90 + 90$) degree (or 0 degree), 135 ($135 = 45 + 90$) degree, and 90 ($90 = 0 + 90$) degree.

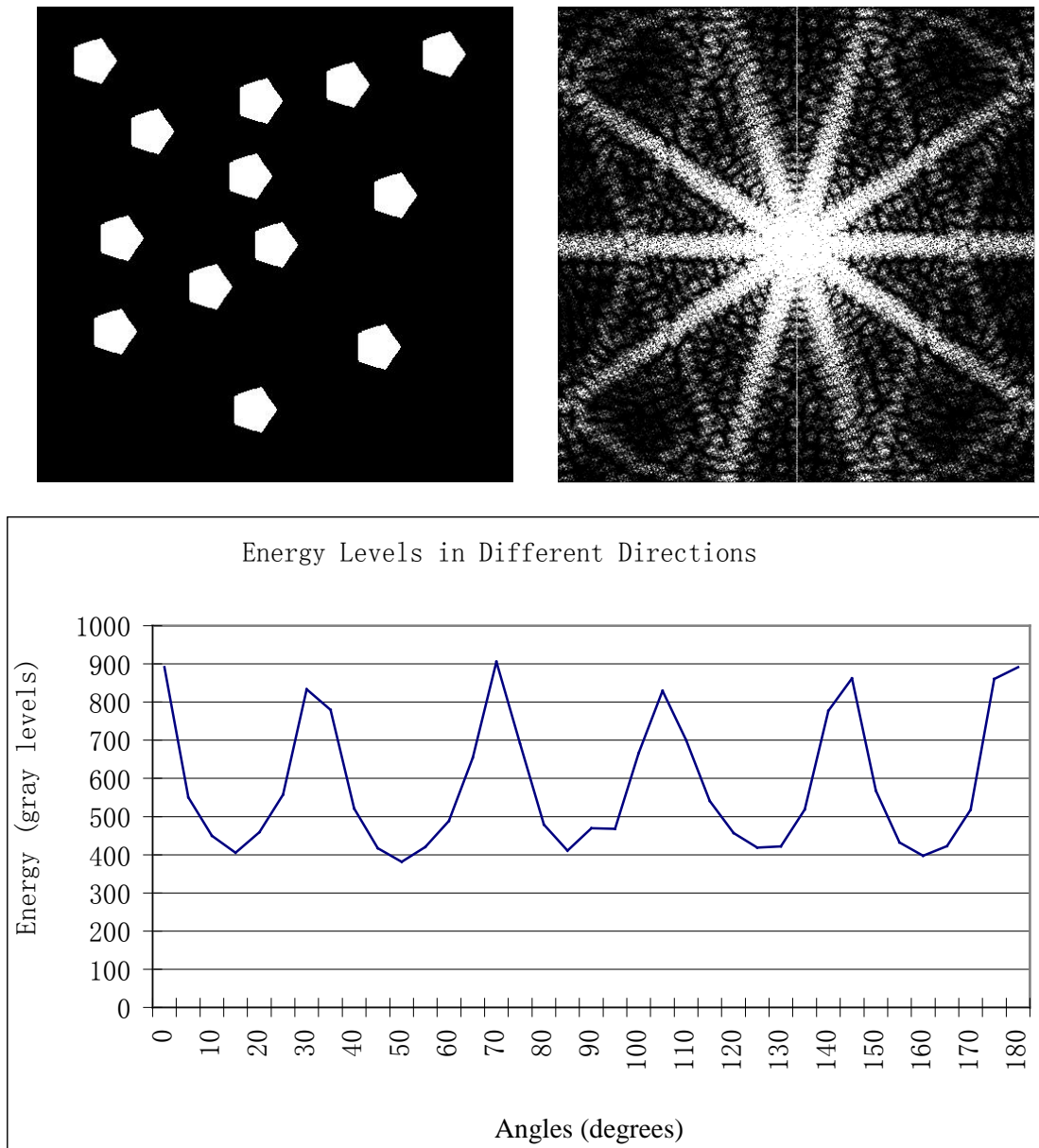


Figure 5.15 edge directionalities of a pentagon texture pattern. Top left image is the original pentagon texture pattern. Top right image is the spectrum of the pentagon texture pattern. Bottom image is the Energy Values of different angles range (bin range is 5 degree).

Figure 5.15 shows that the spectrum bins approach catches the edge angles in degrees of a pentagon.

5.2.7 Spectrum Bins Approach Versus Tamura Directionality Degree

An image's pixels can be categorized into three groups, edge pixels (which is on the edges of a shape), no-edge pixels and edge neighbor pixels. The edge neighbor pixels' intensity (black and white image) or colour (coloured image) is affected by the edge pixels. This is because when a photo is taken, it is hardly to have an edge exactly on pixels and it often blurs the pixels and its neighbors. Tamura Directionality algorithm relies on the neighbor pixels' intensity to calculate the edge directionality and it can cause incorrect result when those edge neighbor pixels are surrounded by multiple edges. Also it can be affected by the pixelation of a computer generated line.

Tamura Directionality Degree algorithm works well on applying to a camera captured texture pattern, but it does not work for computer produced texture pattern.

The difference between camera captured texture pattern and the computer produced texture pattern is the aliasing. Computer only can produce line angle in $n\pi/4$ ($n \in \mathbf{Z}$), therefore all the other degree of angles must be approximated by $n\pi/4$ line segments. Tamura Directionality Degree algorithm applies Sobel edge detector. This causes Tamura Directionality Degree algorithm to only be able to recognize $n\pi/4$ even for a curve.

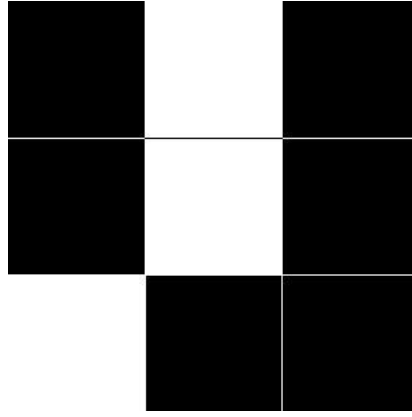


Figure 5.16 a part of a line with directionality degree is between 45 and 90 degree

Taking Figure 5.16 as an example, the Sobel edge detector is 3×3 window. $\Delta_y(x, y) = 0$ and $\Delta_x(x, y) = 1$, the arctan value of this part of line will be $\tan^{-1} (0/1) = \pi/2$.

The problem of Tamura Directionality Degree algorithm is when lines are not anti-aliased the Sobel edge detector only focuses on each 3×3 window (in Figure 5.16) so lines get classified as having angles that are only multiplier of $\pi/4$ if an entire image is considered. Spectrum bins approach takes advantage of considering entire image and based on bin counting to find out the directionality degrees of a texture pattern's edge. Furthermore, Tamura's approach is easily affected by noise as it only focuses on 3×3 windows, but spectrum bins approach does not, as noise often only appears in the high frequency channels. These high frequency channels usually have low intensity in the spectrum and the square root function suppresses the intensity even lower.

5.3 Texture Pattern Formation

Some textures have a pattern that repeats in some formation other than both horizontally and vertically. For these textures, which the patterns' formation is in a certain order, this order should be considered as a feature of the texture.

However the pixel-based approaches have huge difficulties to calculate the formation features. Frequency-based approaches take advantage of viewing the texture image in frequency domain and more easily get the formation.

5.3.1 Spectrum Pattern Texture

Usually a circular pattern appears a circular pattern in the spectrum image, and a rectangle pattern appears a rectangle in the spectrum image. However in Section 5.2.2, Figure 5.5 shows the spectrum image does not only contain a circular pattern but also contains parallel diagonal lines. These parallel diagonal lines are the barrier for finding the Texture Pattern Size but they show the formation of original texture pattern.

Figure 5.17 shows an original texture image, the texture is a group of circular patterns and they are all laid on line which is in 45 degree. Figure 5.18 shows the spectrum pattern texture appears as lines in 135 degree, which is perpendicular to the original texture's formation line in Figure 5.17

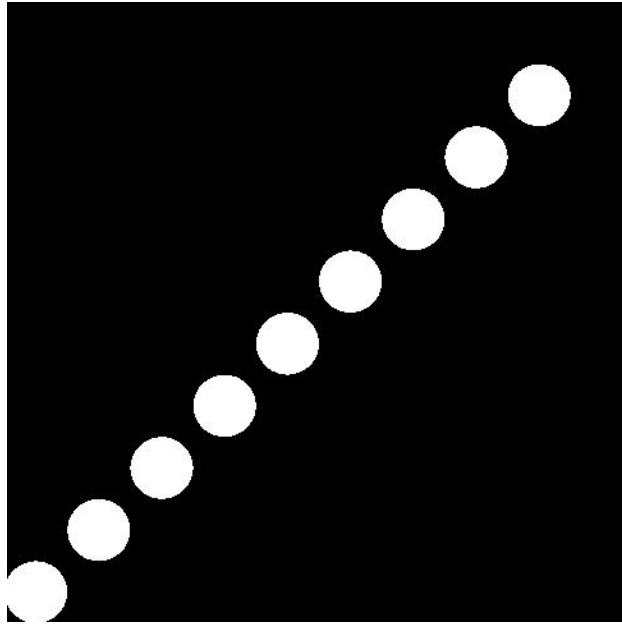


Figure 5.17 circular texture pattern formed in 45 degree line.

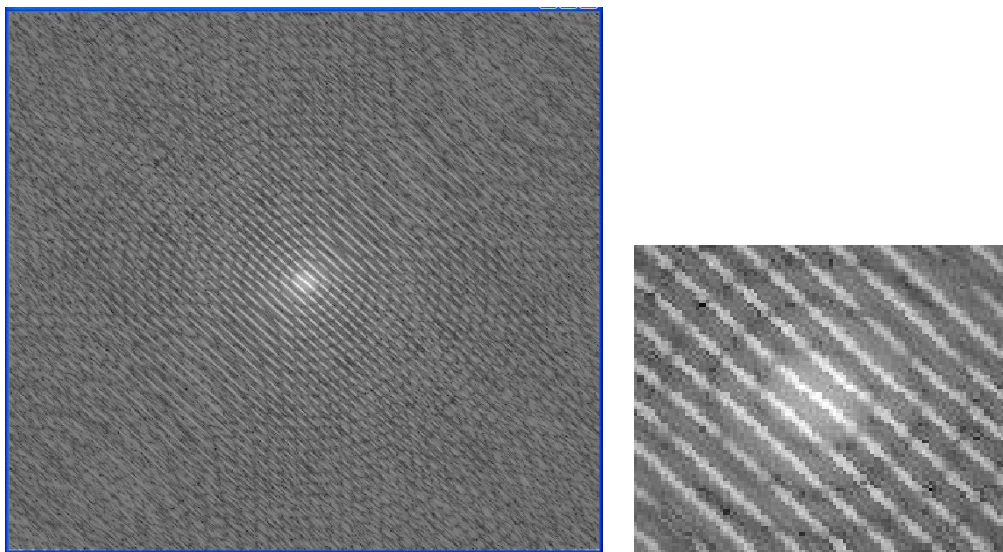


Figure 5.18 spectrum pattern texture of Figure 5.17

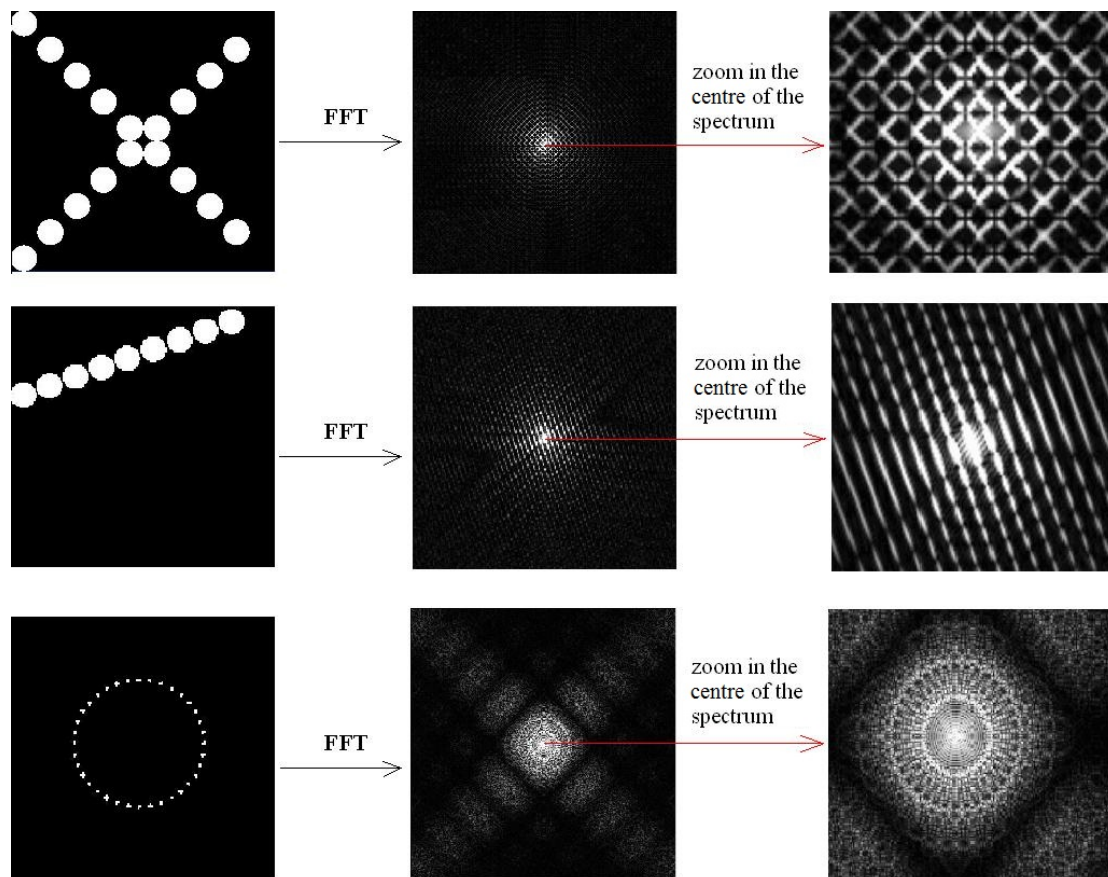


Figure 5.19 examples of spectrum pattern textures. The first column shows the original texture image, the second column shows the spectrum images which is the Fourier Transform of the first column. The last column (the spectrum texture) shows the zoom in the image of the second column.

Figure 5.20 shows the relationship between the formation of the texture and the spectrum pattern texture. The first example is the circular patterns format in two diagonal lines and cross perpendicularly. This formation is captured in the spectrum image and represented as the spectrum pattern texture. The spectrum pattern texture appears as lines and these lines cross perpendicularly.

The second example is the circular patterns format in 20 degree. The spectrum pattern texture shows parallel lines and these lines are in 110 degree, which is perpendicular to the original Texture Pattern Formation.

The last example shows the spectrum pattern texture represents the formation of the

original texture patterns which is not only true for the linear formation, but also can be applied for circular formation. The rhombic texture patterns format in circle in the last example. The spectrum pattern is rhombic in the centre and the pattern has circular texture on it.

5.3.2 Finding Formation Information by Spectrum Pattern Texture

If the spectrum pattern texture is considered as the noise of the spectrum, then it becomes the similar case of applying notch filters. The Fourier transform applies notch filters on the image to get spectrum. Some high intensity spots are filtrated out then the reverse transform applied to remove noise from the original image. This means that those spots are the frequency of the noise. Similar idea can be applied to get those spots. The difference is the notch filters apply Fourier transform to remove the noise but the purpose of this research to apply the Fourier transform on the spectrum image to find the frequency feature of these “noise”

In order to describe the spectrum pattern texture, the second Fourier transform is applied on the spectrum image (the first Fourier transform produces a spectrum image, the second Fourier transform on the spectrum which produces the spectrum image of the spectrum image). The second Fourier transform is similar to reverse Fourier transform but without the position information (as phase angles are not involved).

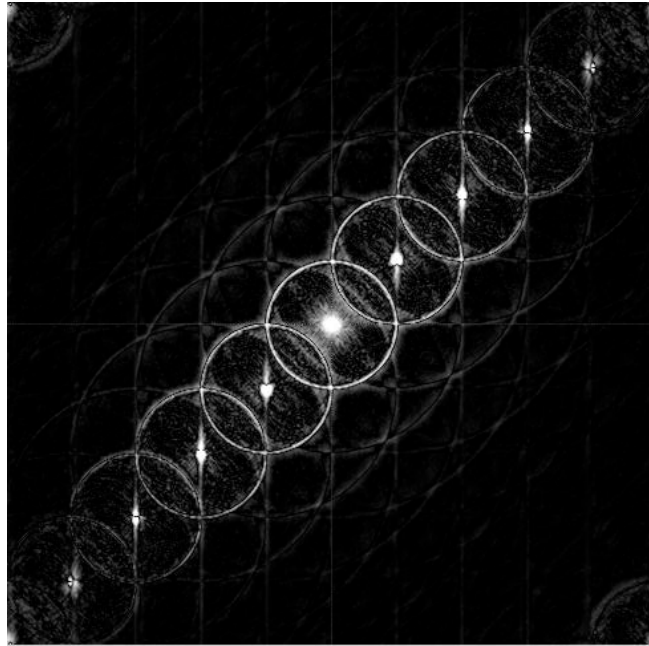


Figure 5.20 spectrum of Figure 3.51

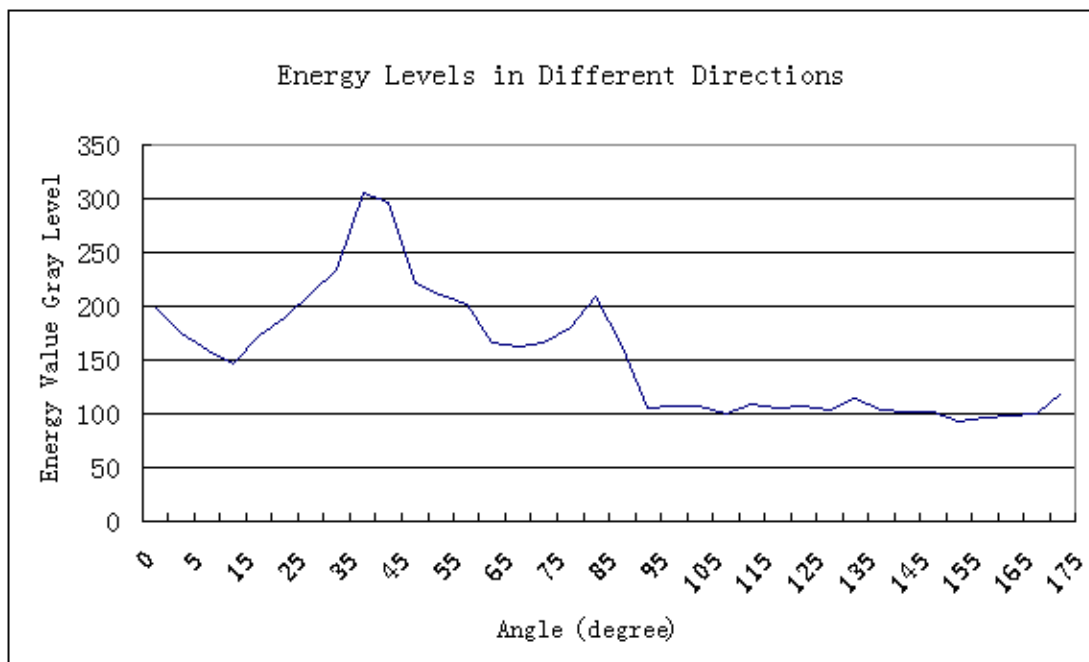


Figure 5.21 the energy values in Figure 3.53 versus angle.

Figure 5.20 to Figure 5.21 show the spectrum image of second Fourier transform. After passing to the spectrum bins, the result can be seen in Figure 5.21. The highest peak is at 45 which determines the direction of the formation of original texture patterns (once Fourier transform rotate in 90 degree second time Fourier transform

rotate in 90 degree again which meets original direction in degree as spectrum image is center symmetric)

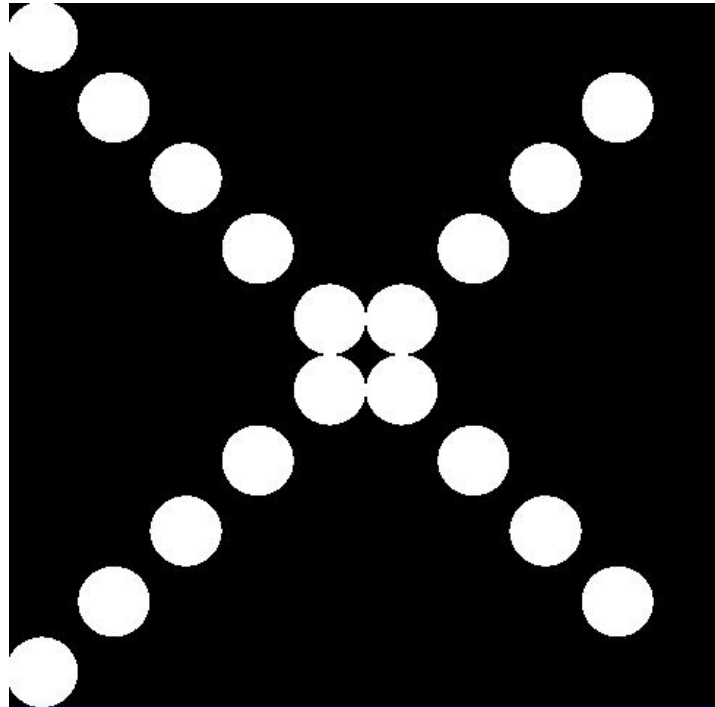


Figure 5.22 texture of two lined formation of circular pattern.

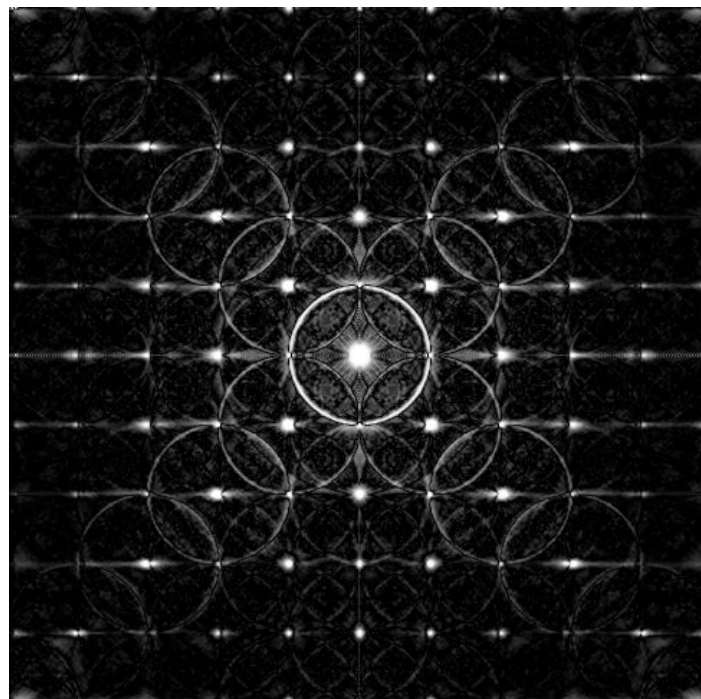


Figure 5.23 the second spectrum of Figure 3.55

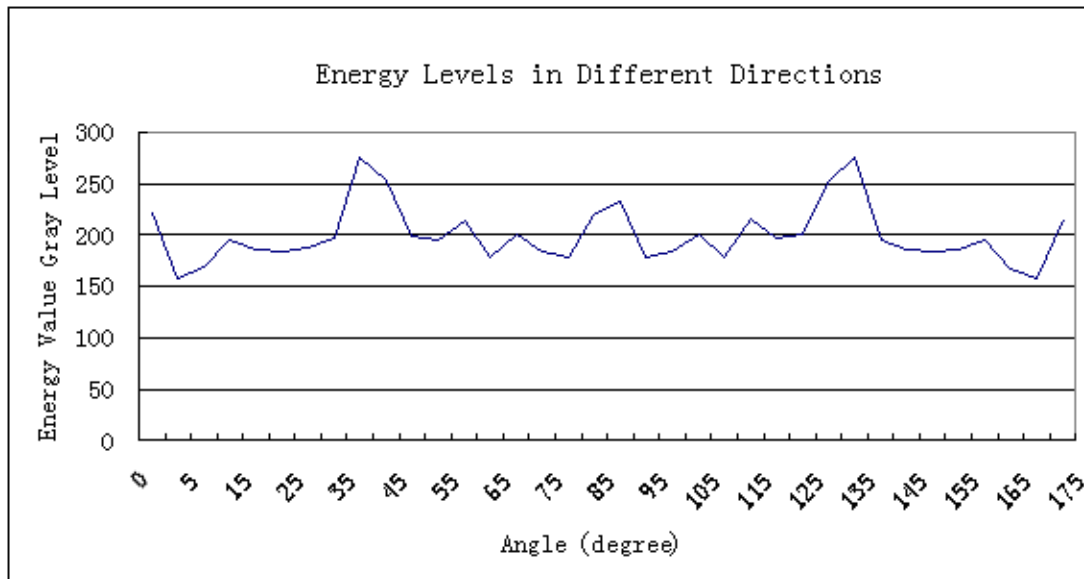


Figure 5.24 the chart of energy values in Figure 5.23 changes versus degree of angles.

Figure 5.22 to Figure 5.24 show a way of finding the directions of two linear formations of circular patterns. There are two high peaks with very similar a gray level with difference is 0.33. These two peaks show that the original texture patterns form two lines and the directions are at 45 degrees and 135 degrees.

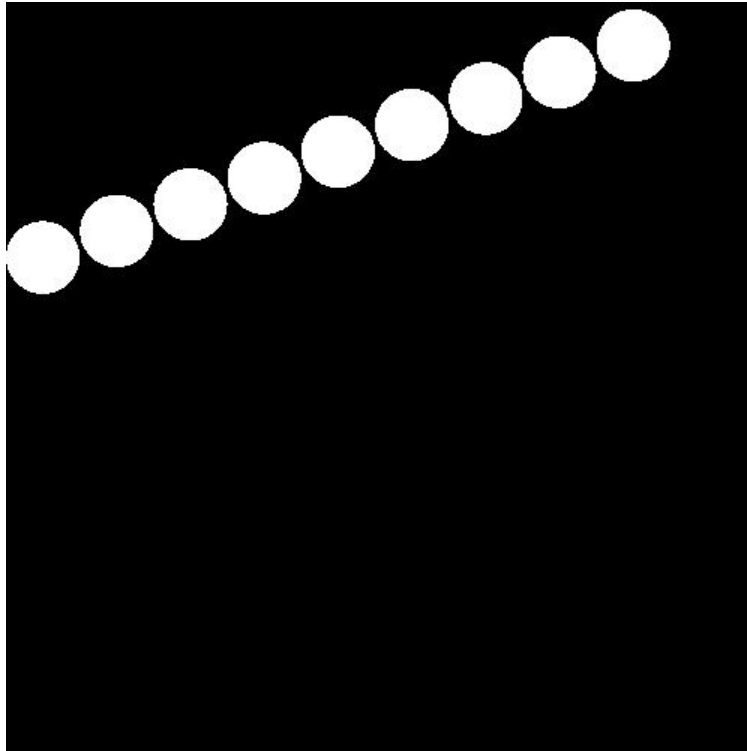


Figure 5.25 a circular texture pattern formed in 20 degree line.

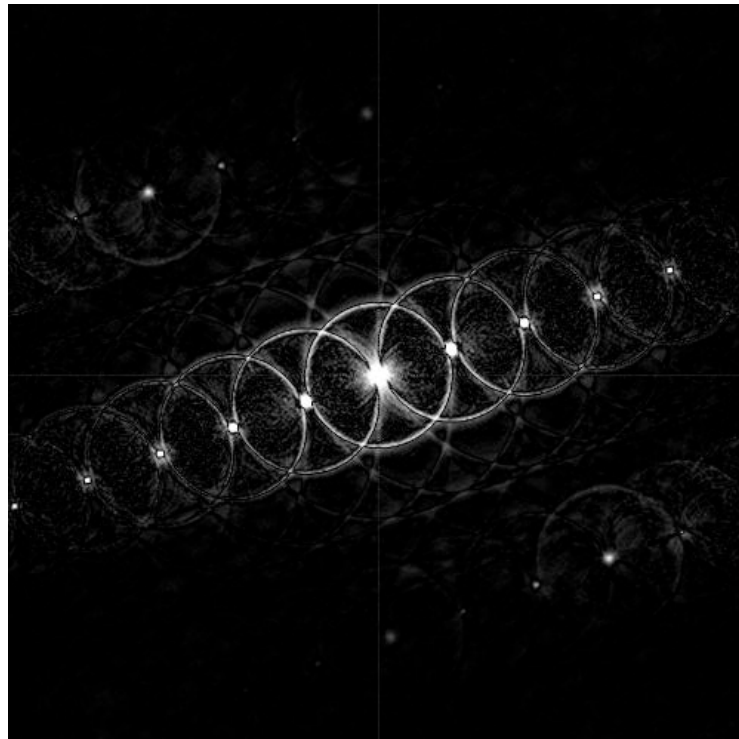


Figure 5.26 the second spectrum of Figure 3.58

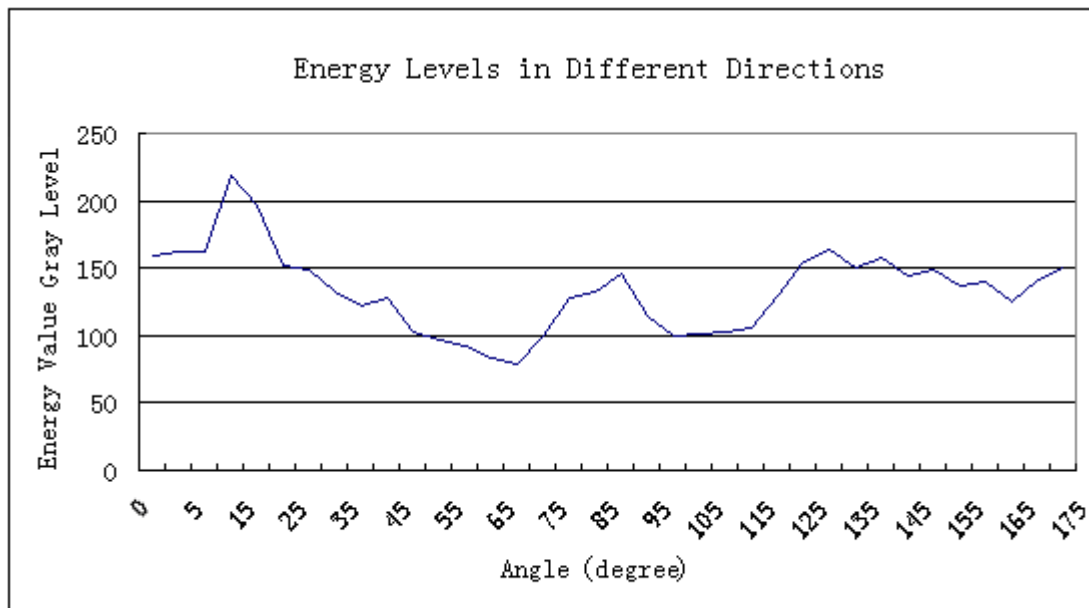


Figure 5.27 the energy values in Figure 3.59 versus angle.

Figure 5.25 to Figure 5.27 show that even when the circular texture patterns is shifted away from the center of the image, the spectrum bins approach still finds the patterns format is a line at 20 degrees, where the peak of the energy value is at 20 in Figure 3.60 and there is no other peak close to the highest peak.

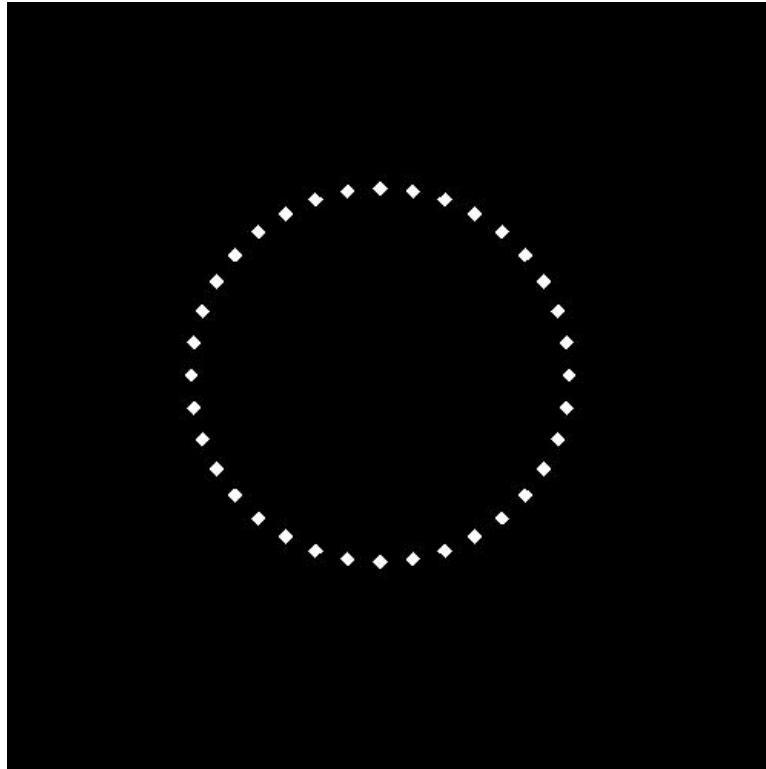


Figure 5.28 square texture patterns form in circularity

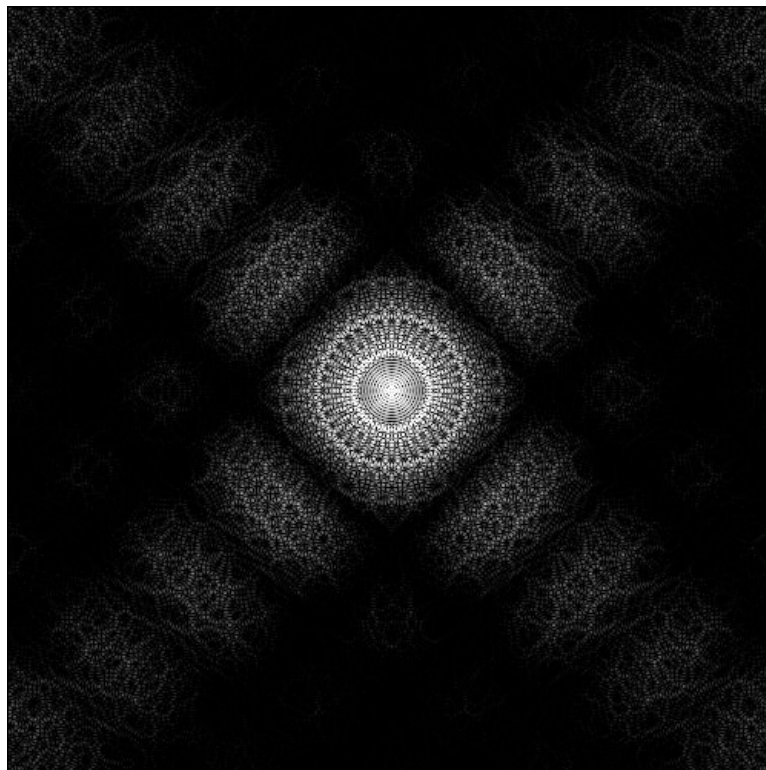


Figure 5.29 spectrum of Figure 3.61

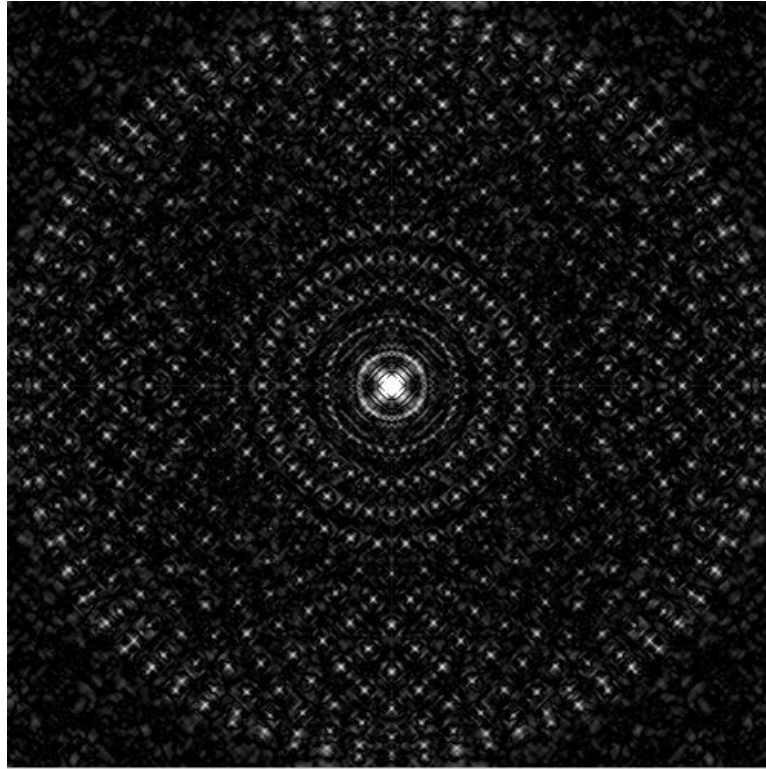


Figure 5.30 spectrum of Figure 3.62

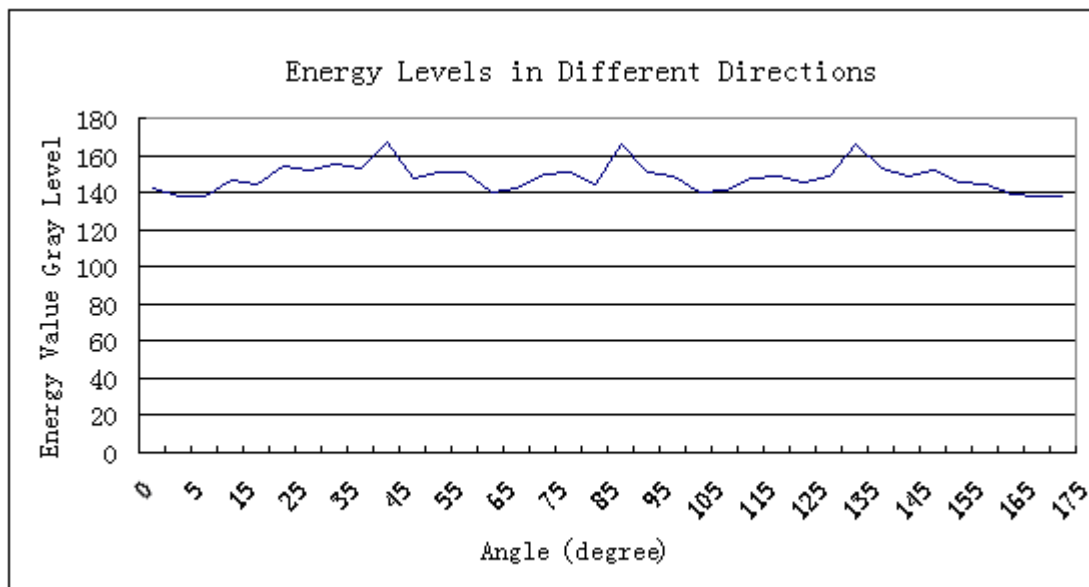


Figure 5.31 energy values in Figure 3.63 versus angle

Figure 5.29 shows the spectrum image of Figure 5.28. It is clear to see that the spectrum pattern texture appears as circularities, which determines the formation of the texture. Passing the spectrum to the Fourier transform again, the second spectrum (Figure 5.30) shows a circular symmetry. Then spectrum bins approach takes place and produces the graph of the changes of energy value versus angle. It shows that energy changes very little over angles from 0 to 180 degrees. Comparing with Figure 5.21 and 5.27 for Linelikeness, the standard deviation of intensities for Figure 5.21 is 0.1858, for Figure 5.27 is 0.1388, whereas for Figure 5.31 the standard deviation is only 0.0451.

Chapter 6

Testing and Results

Chapter five discusses some algorithms of frequency domain texture feature descriptors. Some of these algorithms' limitations have been explored and discussed in Chapter five. These algorithms are Hu moment for spectrum images and Texture Pattern Size. Others need further tests to explore the limitations. This chapter tests and try to find out the limitation of the algorithms for Linelikeness, Directional Unification, Directionality and Texture Formation texture feature descriptors.

6.1 Linelikeness Test

There are three sets of testing texture images used in this thesis for Linelikeness and Directional Unification testing (Images are taken from the standard image database at University of Southern California

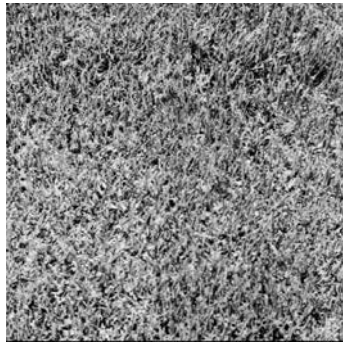
<http://sipi.usc.edu/database/database.php?volume=textures>

Within each set, the images can be visually partitioned into two groups: line pattern based texture images, and non-line pattern based texture images. The non-line based group can be further divided to the regular shape pattern (such as rectangles) subgroup and the irregular shape pattern subgroup. The Linelikeness algorithm will calculate a Linelikeness value for each image and these values are used for classifying the images into those groups.

The second set of images is histogram-equalized version of the first set. This testing set is used to determine how histogram equalization affects the Linelikeness algorithm. As the contrast changes, some edges of a texture may change. These could cause the texture's Linelikeness value to change. However it is not expected to change a lot [130].

The third set uses the same objects as the first set without histogram equalization but enlarged parts from the first set. This set of tests is used for testing how the Linelikeness is affected by image scaling.

The following images are testing textures from the first set



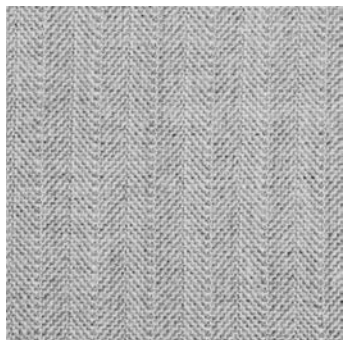
1.01



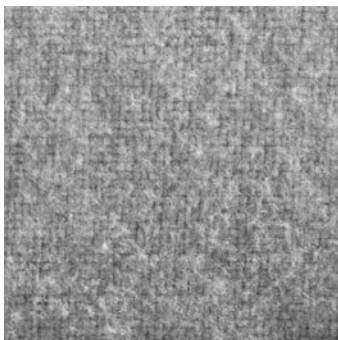
1.02



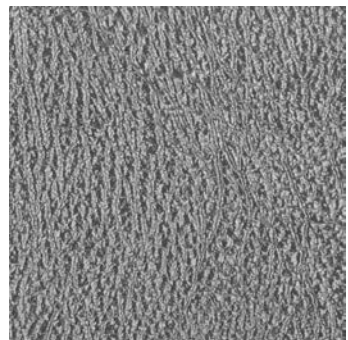
1.03



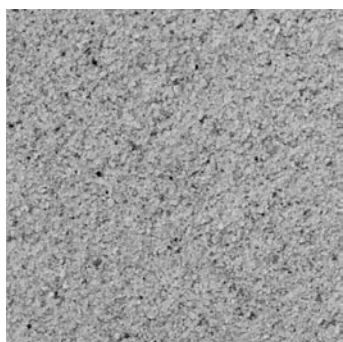
1.04



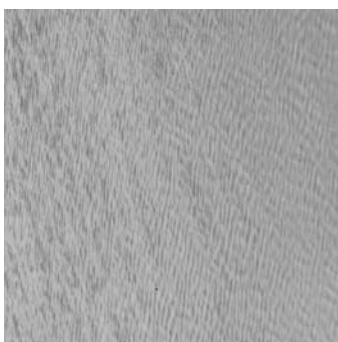
1.05



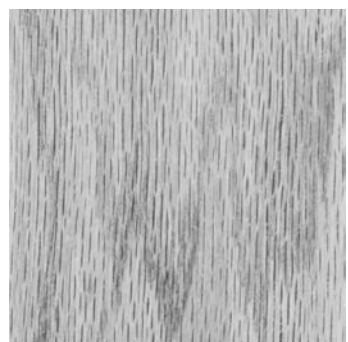
1.06



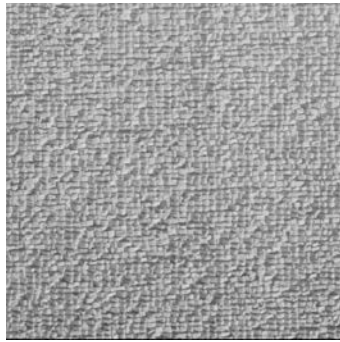
1.07



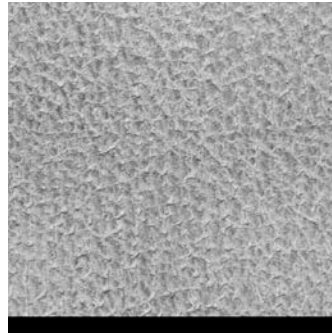
1.08



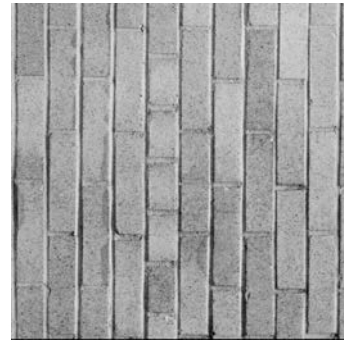
1.09



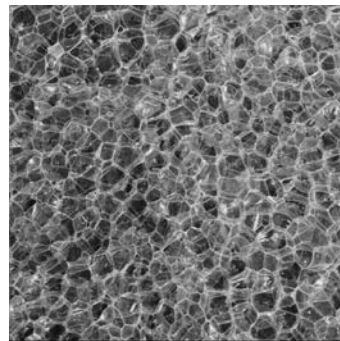
1.10



1.11



1.12



1.13

Figure 6.1 The first set of testing image

| Image name | Linelikeness Value for First Set | Linelikeness Value for Second Set | Linelikeness Value for Third Set |
|------------|-------------------------------------|--------------------------------------|-------------------------------------|
| 1.01 | 0.086 | 0.083 | 0.072 |
| 1.02 | 0.117 | 0.103 | 0.092 |
| 1.03 | 0.224 | 0.225 | 0.225 |
| 1.04 | 0.114 | 0.097 | 0.092 |
| 1.05 | 0.096 | 0.102 | 0.102 |
| 1.06 | 0.195 | 0.186 | 0.166 |
| 1.07 | 0.073 | 0.073 | 0.065 |
| 1.08 | 0.220 | 0.215 | 0.192 |
| 1.09 | 0.204 | 0.205 | 0.183 |
| 1.10 | 0.139 | 0.135 | 0.134 |

| | | | |
|------|-------|-------|-------|
| 1.11 | 0.090 | 0.068 | 0.087 |
| 1.12 | 0.142 | 0.117 | 0.129 |
| 1.13 | 0.060 | 0.050 | 0.048 |

Table 6.1 Linelikeness result for set one, two and three

From Table 6.1 is clear to see that the change of contrast (histogram equalization) only makes little change in the images' Linelikeness value to be changed. Image 1.12's Linelikeness value changes the most with reduced about 17.76%. This is because the histogram equalization enhances each individual brick's texture pattern which almost changes the original wall's pattern. However overall the set texture images, the Linelikeness value changes 8.27% on average.

This does not affect the images to be grouped and the groups match the visual result. From the data which is listed on the table, image 1.03, 1.06, 1.08 and 1.09 are line based textures. The difference between the image which has lowest Linelikeness value in the line based texture group and the image which has highest Linelikeness value in the non-line based group drops 27.22% in set 1 and drops 37.30% in set 2.

Comparing the first set and second set, 1.03, 1.06, 1.08, 1.09 are still the top 4 textures with stronger Linelikeness value. The difference between line based texture group and non-line based texture group drops 19.31%. In the second test group, the algorithm takes consideration of which parts are chosen. Parts of image 1.06 loses a lot of Linelikeness texture pattern when zoomed in and the irregular gaps are increased by zooming in. This reduces its Linelikeness value. This is also the probable reason why most textures' Linelikeness value reduce and the difference between first set and third set is about 11.59% on average.

The algorithm is a little affected by histogram equalization and also affected by zoom in the image, but it still picks up the textures which are visually strong Linelikeness.

These effects are because the texture details are becoming clearer and these details are considered as a part of texture such as testing image 1.12 that brick's texture becomes to be a part of wall's texture.

There are more Linelikeness test images in Figure 6.2 and all ordered by the Linelikenesses values.



Figure 6.2 extra testing images for Linelikeness.

There are some very interesting points base on these results.

Firstly, the second image and the sixth image are both wood texture image. The difference between these two images is about 10% reduced from second image to sixth. This because of the sixth image is blurrier than the second image. Blur causes image edge to be weak and the linearity is weak.

Secondly, eleventh, twelfth, thirteenth image are all camouflage textures. The eleventh image's texture patterns are made of squares, but the other two are made of curves. Therefore the eleventh image's Linelikeness is 7.67% higher than the other two.

Thirdly, in the last four images, although the fibres make the leaf texture have stronger edges, these edges also build closed irregular polygons. In the last image, the edges which made up the patterns are all small curves which helps explain its low Linelikeness value.

6.2 Direction Unification Value Test

Direction Unification Value is used for describing how the direction of the lines are in the same direction. The testing images are selected from the Linelikeness testing sets with high Linelikeness values. The reason for using images with high Linelikeness value is because only the liner textures are worth getting Direction Unification Values. If an image's edges make up a closed polygon, definitely, the lines' directions are not unified. As in the previous testing the original image set, the histogram equalized image set and the zoomed image set are used.

These three sets of tests are used to find how much the Direction Unification Value changes when the contrast is altered or different photos of the same object are used.

| Image name | Direction Unification Value for First Set | Direction Unification Value for Second Set | Direction Unification Value for Third Set |
|------------|---|--|---|
| 1.03 | 0.526 | 0.517 | 0.713 |
| 1.08 | 0.563 | 0.552 | 0.563 |
| 1.09 | 0.804 | 0.733 | 0.817 |
| 1.06 | 0.384 | 0.352 | 0.327 |

Table 6.2 Direction Unification test for the first four most linelike textures

From the table, the average difference between set one and set two are 5.16%. The highest difference comes from image 1.09. When the contrast of testing image 1.09 is increased, more details of the wood become clear which changes the Direction Unification Value. In the original image 1.09, the details are weak, which have weak energy in the spectrum and the Direction Unification Value only considers the main texture pattern.

When histogram equalization is applied, the image's details become stronger and

become more noticeable in the main texture pattern. More energy in the spectrum is represented by the details and the Direction Unification Value takes account of these changes and these noticeable details were not considered by the Direction Unification Value in the first set. Therefore it ends up reducing the Direction Unification Value.

Changes of zoom can be considered as two different images sometimes, especially for image 1.03. For image in the first and second set, the value changes very little (about 1.73%). However when it is zoomed in, the Direction Unification Value has changed about 35.67%.



Figure 6.3 the original image of 1.03 Figure 6.4 the partially zoom in image of 1.03

Comparing the two images in Figure 6.3 and Figure 6.4, it is easily to be found that although they have similar Linelikeness (original image: 0.224, zoomed image: 0.225), but the directions of the grasses in the zoomed image are set to be more unified than the original image. The Direction Unification Value is very sensitive on direction varies. For all the other three images, the average of changes is 5.48%.

There are two more group of direction unification tests. They are cactus photos and bamboo sticks. They are all ordered visually Directional Unification. The test is expected to provide the same order as the same as human perception. All the images are from google texture image searching.



Figure 6.5 Direction Unification: 0.594



Figure 6.6 Direction Unification: 0.499



Figure 6.7 Direction Unification: 0.850



Figure 6.8 Direction Unification: 0.731



Figure 6.9 Direction Unification: 0.669



Figure 6.10 Direction Unification: 0.512

If an object's texture direction varies, the Direction Unification Value can catch the changes and returns different Direction Unification Values, even if they have similar Linelikeness value, such as in Figure 6.3 and 6.4, Figure 6.11 and 6.12.



Figure 6.11 Direction Unification: 0.747

Linelikeness value: 0.241



Figure 6.12 Direction Unification: 0.407

Linelikeness value: 0.249

The Direction Unification Value is not a duplicate of the Linelikeness value. Linelikeness give information on how strong the texture edges build up lines based patterns. Direction unification instead shows how similar these line directions are. This is illustrated by the human hair textures in Figure 6.11 and 6.12. Although their Linelikeness values are very close, the hair in the Figure 6.11 is almost horizontal but the hair in Figure 6.12 are both horizontal and vertical. These changes are not caught by Linelikeness but are by the Direction Unification Value.

6.3 Directionality Test

These tests compare the texture's Directionality with the Tamura's Directionality algorithm to find the better approach of Directionality descriptor. The Hough transform is used for checking the correctness of the two algorithms. The energy value of each angle is calculated by the bin approach. The direction angles are chosen to be any peaks that are above half the height of the highest peak.

The testing images are chosen from google texture image searching. Figure 6.13 is testing whether the algorithm can determine the main direction of the image when there are some other directions included in the image. Figure 6.14 is testing whether the algorithm can determine the main direction of the image when there are fake edges (made by the ends of the sticks lining up). Figure 6.15 is testing how the algorithm can determine the main directions, even some immethodical lines may interfere the algorithm. Figure 6.16 is testing whether the algorithm can determine the main direction of curves. Figure 6.17 is testing whether the algorithm can determine the main direction from a non-antialiased computer-generated image. As a computer image is a matrix of square pixels all lines are built up by short line segments that are multiples of 45 degrees. The algorithm is expected to determine the main direction but not that of the segments.

Each test includes provide the main direction angles in degrees and spectrum energy value versus different direction angles in degree is provided. Tamura and Hough transform results are also drawn for comparison.



Image of sticks actual angles: 6° , 12° , 18° , 36°

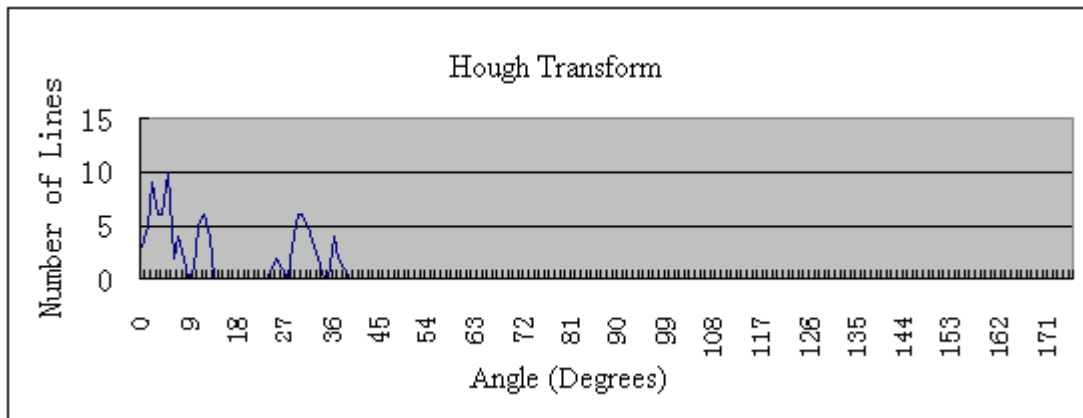
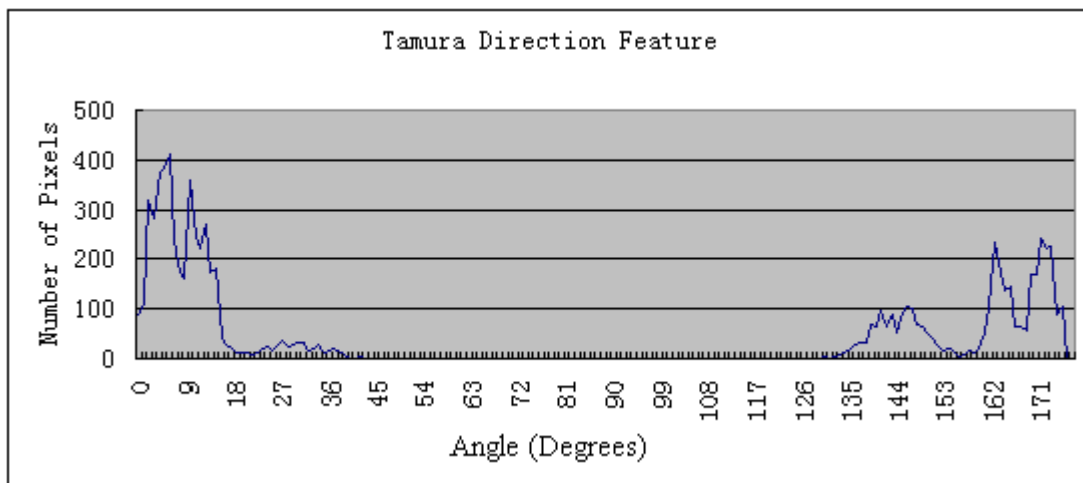
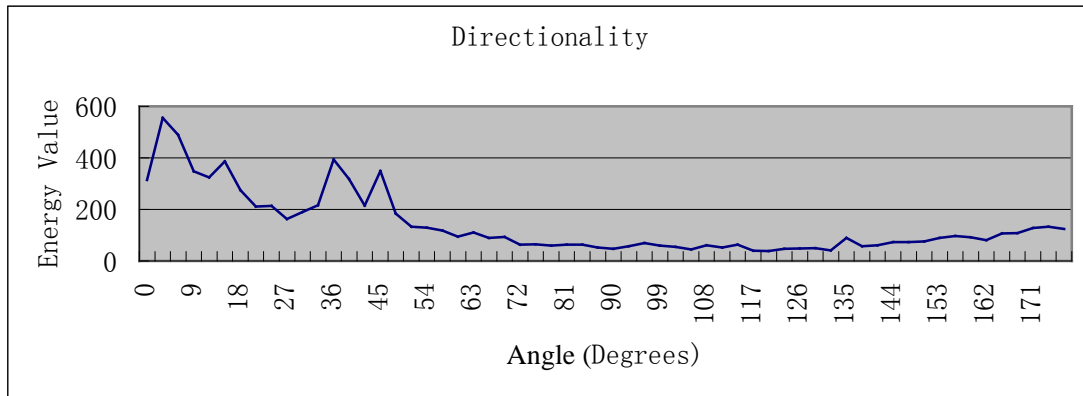


Figure 6.13 Directionality test for Figure 6.8



Image of sticks actual angles: 156° , 155°

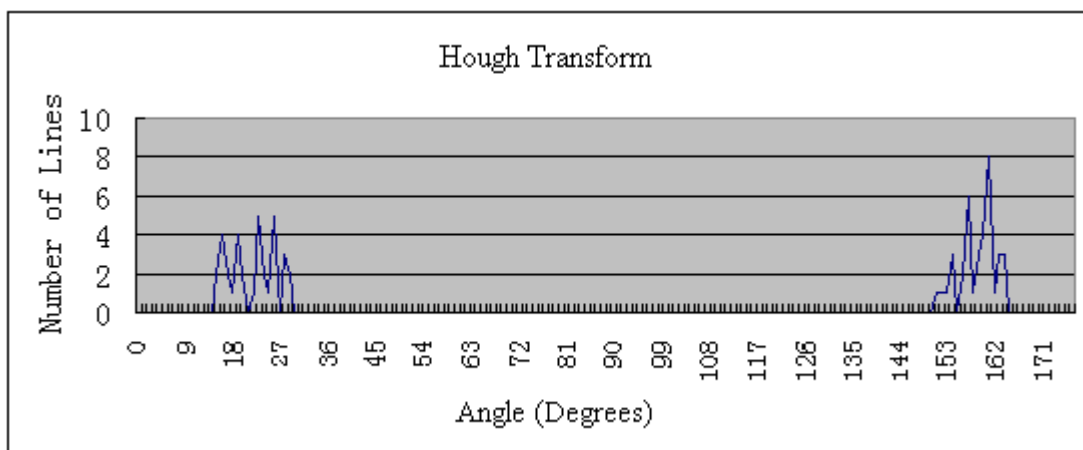
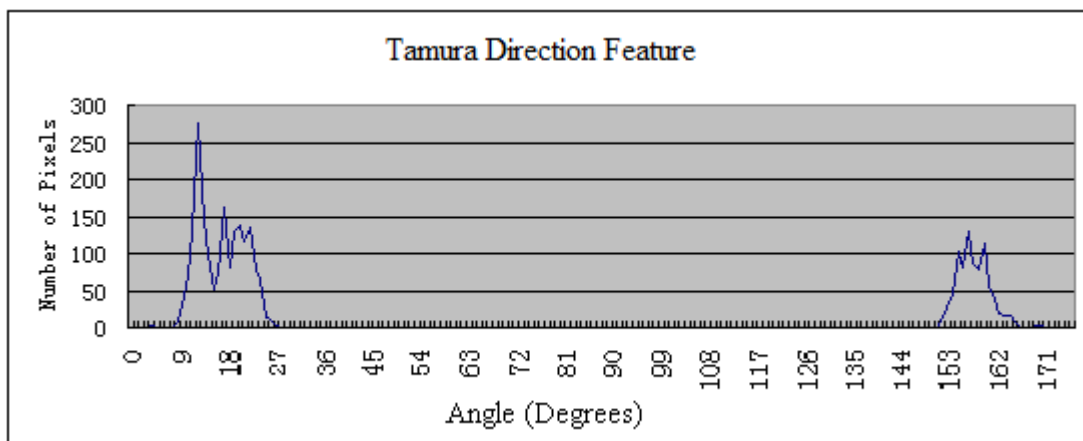
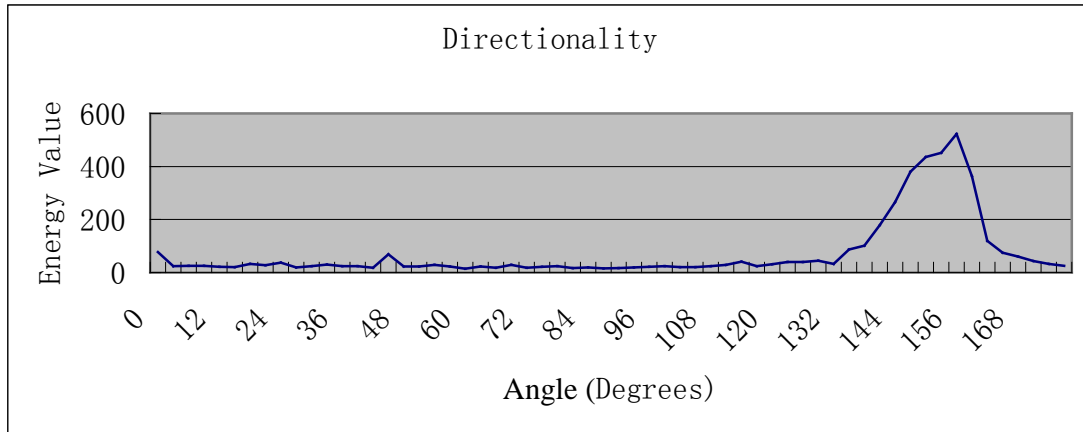


Figure 6.14 Directionality test for Figure 6.7



The angle in image: 78°

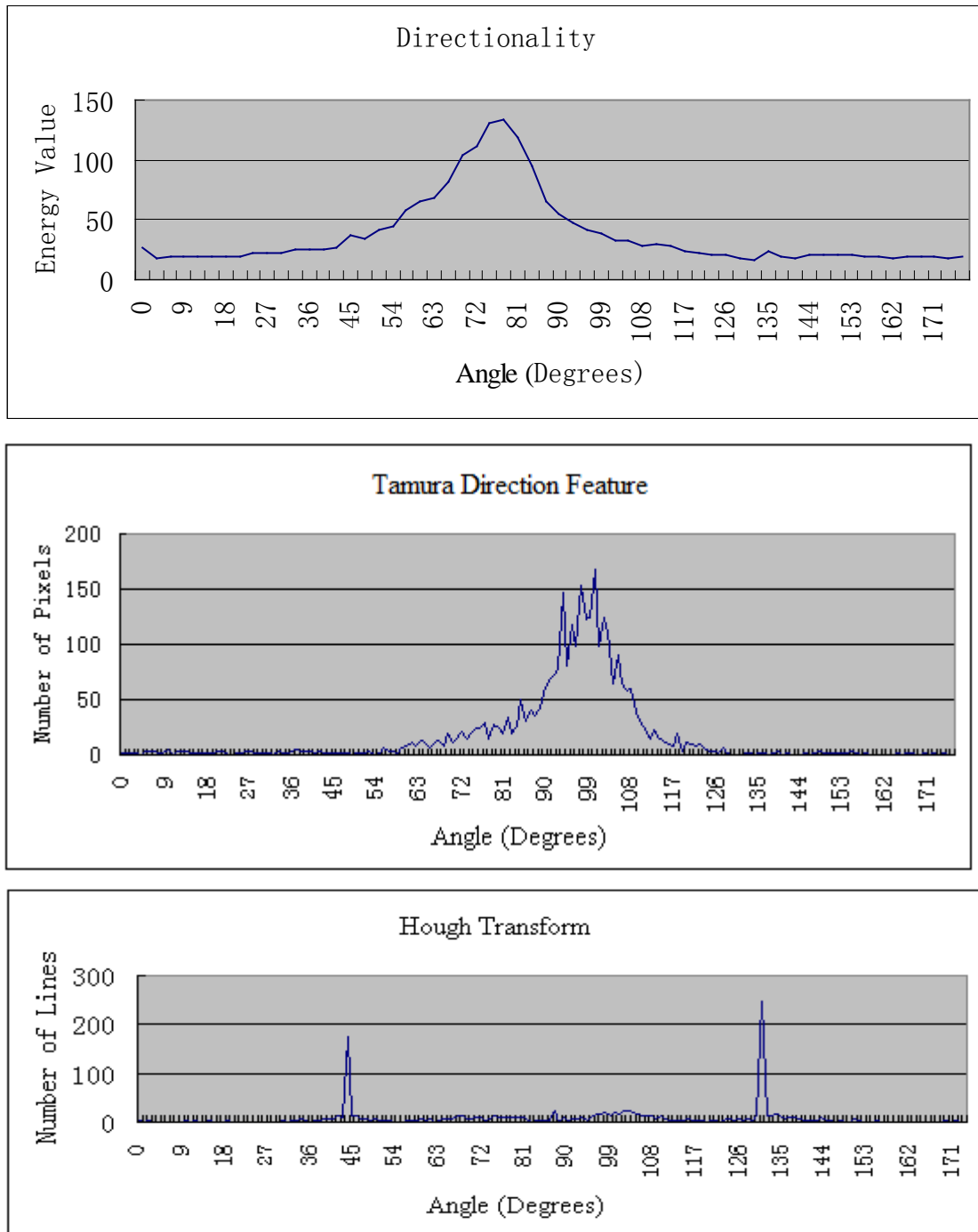


Figure 6.15 Directionality test for Figure 6.4



The angle in image: 168°, 174°, 162°

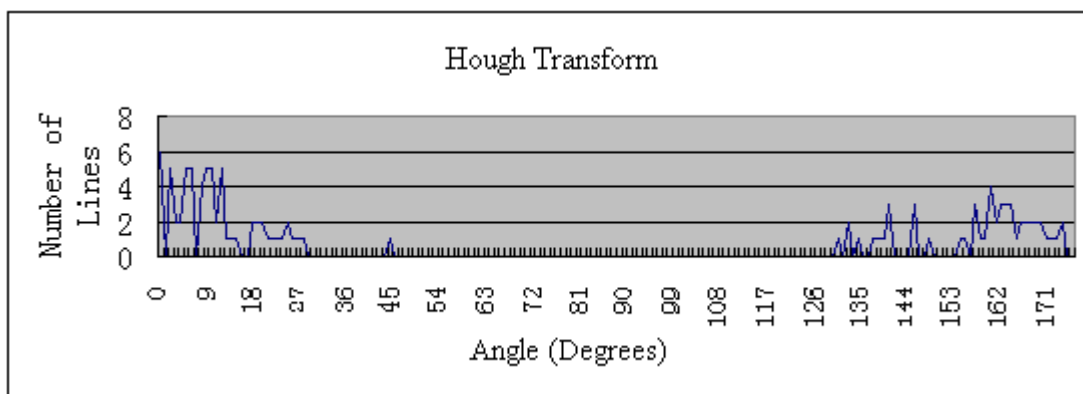
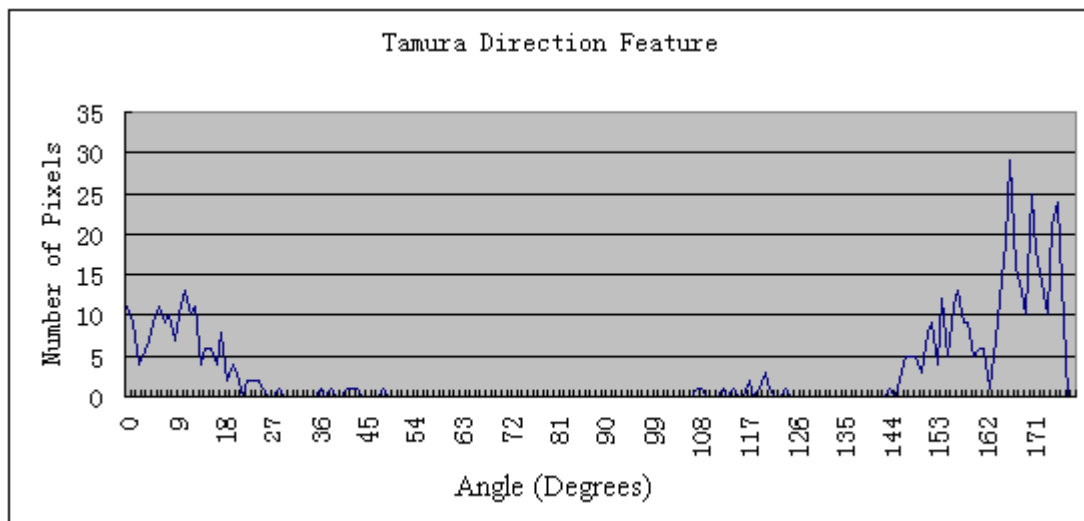
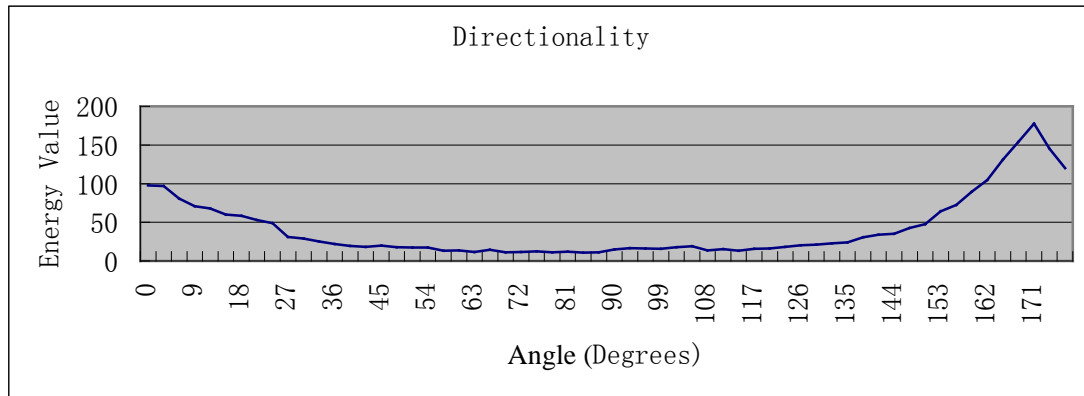


Figure 6.16 Directionality test for Figure 6.11

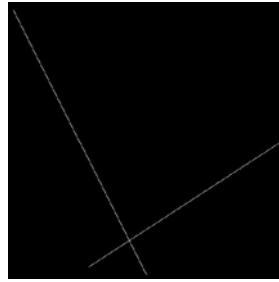


Figure 6.17 computer produced image. It has two lines which are at 30 degrees and 144 degrees

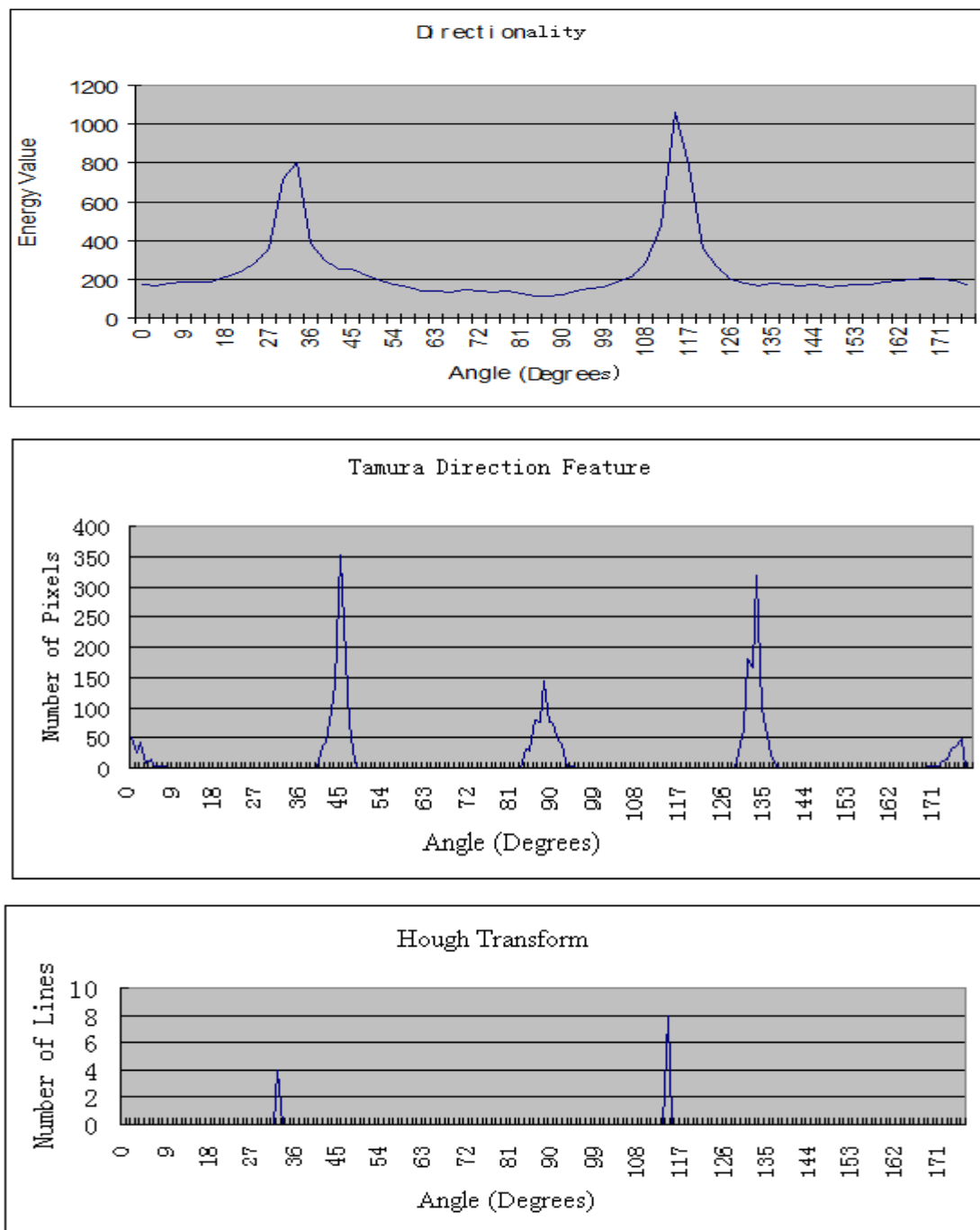


Figure 6.18 Directionality test for Figure 6.17

In the test for Figure 6.13, the Hough Transform and the spectrum-bins approach determine that the most line segments' directional angle is between 6 degree and 36 degree which is not disturbed by those five larruping sticks. However Tamura algorithm shows that the angle also includes between 140 degrees and 177 degrees.

In the test for Figure 6.14, the spectrum-bins approach claims the image mainly has line segments at about 155 degrees with a little energy also at about 45 degrees. This is made by the right side of all the sticks with 38 amount of energy in gray levels. Actually the blue stick (at the bottom) is 150 degree and the green stick (at the top) is 165 degree. Both in the Tamura's directional test result and the Hough Transform test result all show the angles of line directional at about 160 degrees, also show there are some lines' directional in between 15 and 25 degrees. The Tamura algorithm measures number of pixels, and the Hough Transform measures number of potential lines. Therefore the length of the line, which may be the main directionality is not taken account in the Hough Transform. The more short line segments lead the Hough Transform and Tamura algorithm results to give more consideration those minor directionalities.

In the test for Figure 6.15, the spectrum-bins approach claims the main direction of image is about 78 degrees, whereas Tamura Directionality claims that the directionality of the image is around 100 degrees. The directionality of the texture actually is less then 90 degrees, which is captured by the spectrum-bins approach. The Hough transform fails in this test because of failures of Canny edge detection. Figure 6.19 shows the line segments which are detected by Canny edge detection.

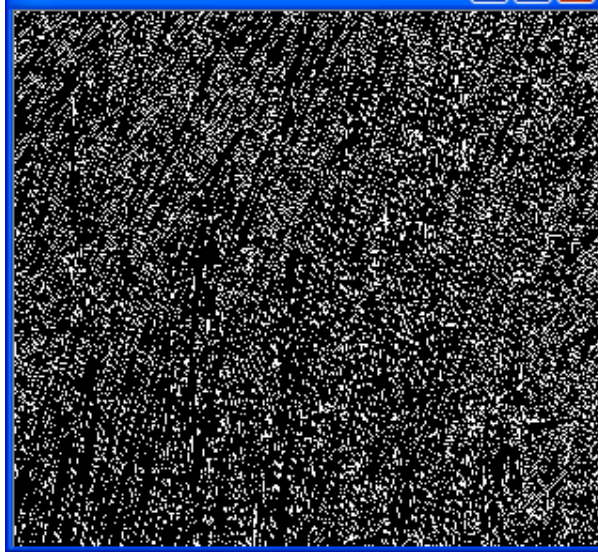


Figure 6.19 Most the lines edges from Figure 6.15 are caught as some single pixels via Canny edge detection. This causes the Hough transform to fail detect the lines.

In the test for Figure 6.16, all of these three algorithm claim that the texture directionality is from 0 degree to 28 degree and from 140 degree to 179 degree. These three algorithms work fine to find the direction for the curve edges.

In the test for Figure 6.17, both spectrum-bins approach and Hough Transform claim the texture directionality is about 30 degrees and 115 degrees, but Tamura Directionality claims that the directionality of the texture is at 0 degrees, 45 degrees, 90 degrees and 135 degrees. Without anti-aliasing, the edges' neighbor pixels do not contain any directionality information of the edges. This causes failure of the Tamura Directionality algorithm.

6.4 Texture Pattern Formation Test

These tests are testing the algorithm of finding Texture Pattern Formation information. The information includes the Linelikeness and the directions of the pattern repeating formation. The images are chosen from google texture image searching. The important feature of these texture images is that the texture patterns are repeated in certain directions in these tests.

The pattern is repeated in horizontal and vertical direction in Figure 6.20. This image is not easily to find the repeating direction visually, but the algorithm is expected to determine the directions in which texture patterns are repeated.

The image patterns in Figure 6.21 are repeated in multiples of 60 degrees. The algorithm is expected to perform well not only in horizontal and vertical but also in any degree.

The image texture patterns for Figure 6.22 are circles and repeated in the top of the image. The algorithm is expected to be transfer invariant. Comparing with Figure 6.21, the algorithm should also be texture pattern invariant (the algorithm should not take effect of the texture pattern are triangles or circles).

Figure 6.23 texture patterns are formatted naturally. This test is used to check whether the algorithm can also work for nature-designed images. Previously, the tests were testing regular computer-generated patterns, which give very strong directions in they are repeating. Comparing with the nature-designed image, the human-designed images are more regular.

In the test image for Figure 6.24, the pattern is repeated horizontally and vertically. It also can be consider that the patterns repeat in 45 degrees and 135 degrees. However to the camera was slanted. The algorithm's result may be affected by the angle of the

image's slant.

Figure 6.25 texture patterns are circular patterns, and weakly repeated in the direction of multiples of 45 degrees. This tests whether the algorithm can detect it is a weak Linelikeness formation.

In Figure 6.26, square texture patterns repeat in a circular formation. The algorithm should return the lowest formation Linelikeness value within these seven tests.

In for these seven tests, the bins' angles are set to in each three degree, which means the accuracy of the results are in three degrees.

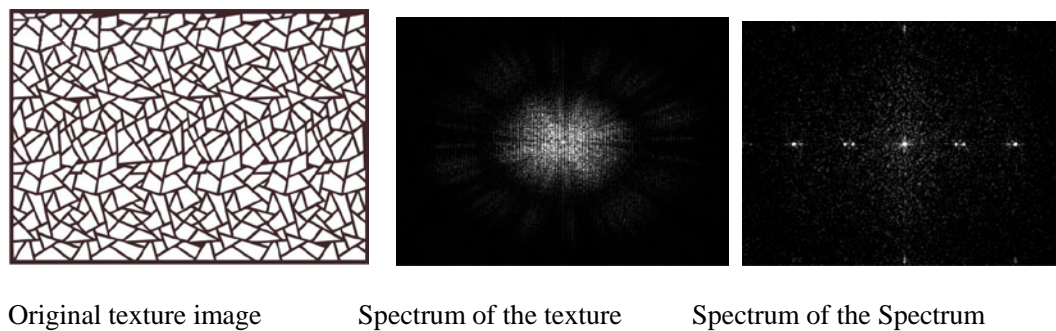


Figure 6.20 Texture patterns are repeated in 0, 90 degrees
Texture patterns repeating Linelikeness: 0.099

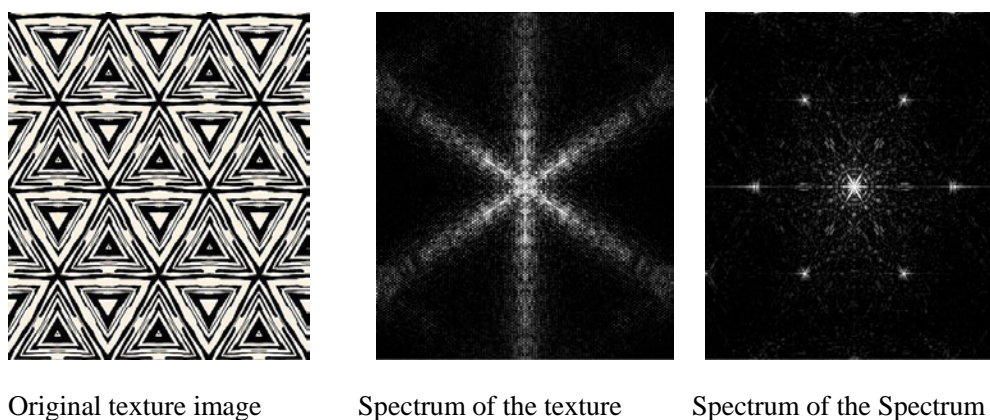
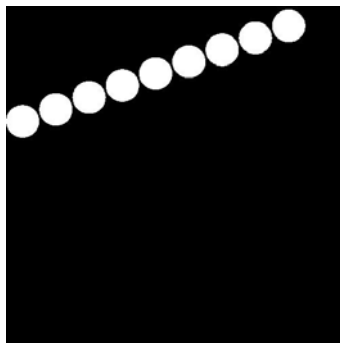
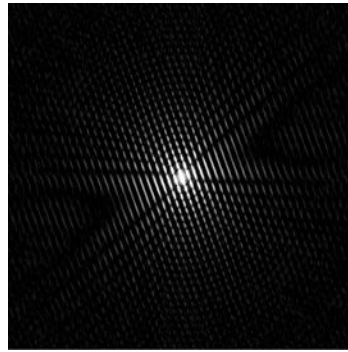


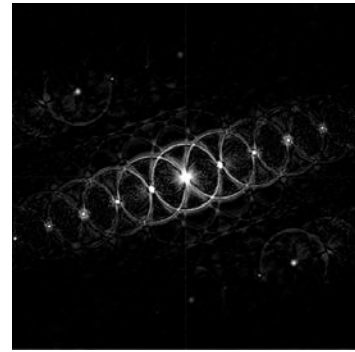
Figure 6.21 Texture patterns are repeated in 0, 60, 120 degrees
Texture patterns repeating Linelikeness: 0.138



Original texture image



Spectrum of the texture



Spectrum of the Spectrum

Figure 6.22 Texture patterns are repeated in 18 degrees

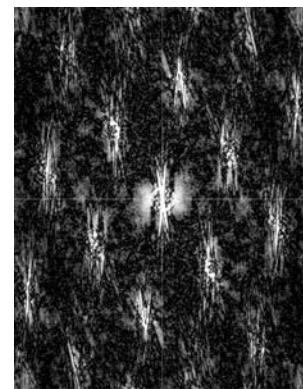
Texture patterns formation Linelikeness: 0.135



Original texture image



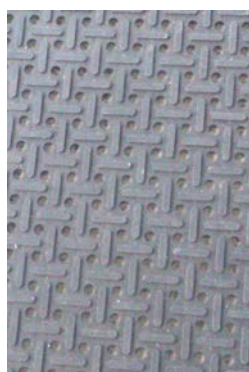
Spectrum of the texture



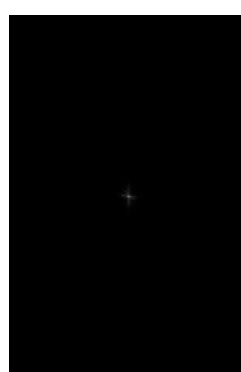
Spectrum of the Spectrum

Figure 6.23 Texture patterns are repeated in 40, 80, 120 degrees

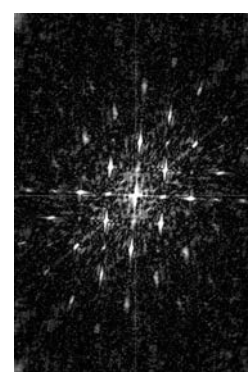
Texture patterns formation Linelikeness: 0.113



Original texture image



Spectrum of the texture



Spectrum of the Spectrum

Figure 6.24 Texture patterns are repeated in 84, 42, 174, 132 degrees

Texture patterns formation Linelikeness: 0.115

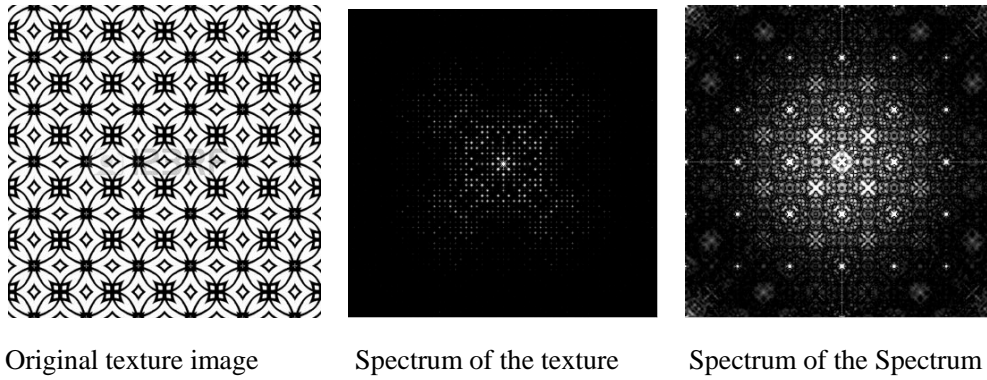


Figure 6.25 Texture patterns are repeated in 42, 132, 90, 0 degrees

Texture patterns formation Linelikeness: 0.097

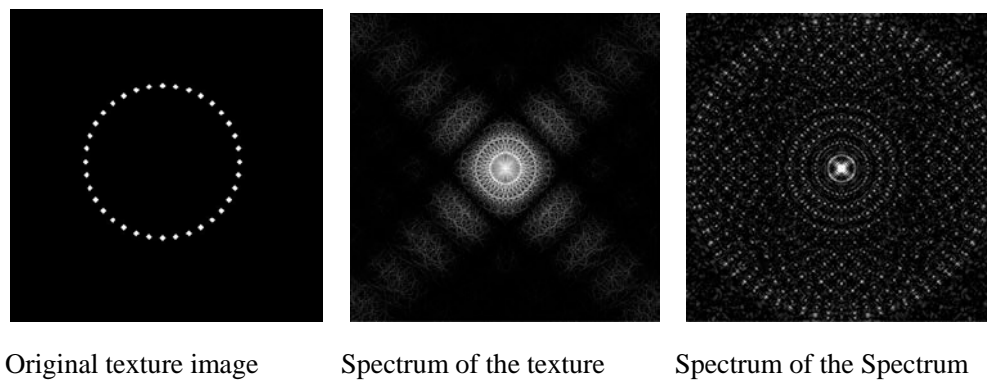


Figure 6.26 Texture patterns are repeated circularly

Texture patterns formation Linelikeness: 0.048

The spectrum images contain the information of the texture patterns' formation. This information can be retrieved by a second application of Fourier transform on the spectrum image. The second application of Fourier transform is similar to a reverse Fourier transform but without the position (phase) information. It brings the original texture pattern shifted to the center of the second spectrum without changing of the formation. This means Linelikeness and Directionality can be measured by the spectrum bins approach. From the results, the algorithm correctly works out the feature of the Texture Pattern Formation features.

Figure 6.20 is hardly to be found out the way of the texture pattern has been repeated visually. It becomes easier by the second application of Fourier transform. Furthermore the spectrum bins approach gives the directionality (0 degrees and 90

degrees) in which the pattern is repeated, which is not able to be obtained from the pixel-based approach.

The results for Figure 6.21 and 6.22 show that the algorithm is robust on texture pattern shapes. The texture pattern itself can be various shapes but the algorithm only focus on the pattern's formation and gives result that directionality in 0, 60 and 120 degrees for Figure 6.21 and 18 degrees for Figure 6.22.

The results for Figure 6.23 and 6.24 show that the algorithm work very well as expected on the natural-produced pattern image which gives directionality in 40, 80 and 120 degrees for Figure 6.23 and 84, 42, 174 and 132 degrees for Figure 6.24 but it needs to be aware that the algorithm is sensitive on rotating the cameras.

The result for Figure 6.25 shows that the algorithm works out that the texture is weak Linelikeness formation (Linelikeness is 0.097) to compare with previous tests. The Directionality can be found by linking any two texture patterns. Therefore the algorithm is reasonable giving low value for the Linelikeness. Even the Linelikeness value is low, but the Directionality still picks up the main repeat directions.

The result for Figure 6.26 shows the algorithm is able to pick up the circular formation as expected. The texture patterns formation Linelikeness is 0.048, which is much lower than all the others (it is about 59% lower). The first spectrum only determines the patterns are squares shape but the second spectrum clearly gives the formation of these squares.

Overall, the second Fourier transform contains the pattern formation information and the formation information can be quantitative by the Linelikeness and the Directionality algorithms.

Chapter 7

Conclusion

Object detection and recognition are important research areas within computer vision. This research focuses on developing feature descriptors to describe textures. It mainly focuses on designing algorithms for a texture's spectrum image to extract its texture feature descriptors and evaluates their applicability under several effects, including change in contrast or zooming into parts of an image. The descriptors include local texture features, which are Texture Pattern Size, Linelikeness, Directionality, and global texture features, which are Directionality Unification, and Texture Pattern Formation, which describe how a pattern repeats within a texture.

7.1 Key Achievements and Limitations

7.1.1 Texture Pattern Size

This research proposes two Texture Pattern Size algorithms (which are in Sections 5.2.1 and 5.2.2). Both of them are shown to be mostly invariant to contrast and direction. The first Texture Pattern Size descriptor algorithm (in 5.2.1) utilises the energy that distributes in the spectrum to find the Pattern Size. This algorithm reduces contrast effects compared to spatially-based descriptors. The second algorithm, which is discussed in Section 5.2.2, provides a more accurate result for the Texture Pattern Size (see Table 5.4 and Table 5.5). There are not significant effects on the contrast even when the contrast drops down to 20%.

Although the first algorithm reduces the effects of changing contrast it can still fail. When the contrast is decreased, the spectrum drops off more quickly at higher frequencies. This appears as though the spectrum is compressed which is equivalent to the image getting larger, so changing the value of texture Pattern Size.

The second algorithm works well even the contrast drops below 20% . However it is very sensitive to the Texture Pattern Formation. If the texture patterns are repeated in certain way (such as linearly), and the texture image has low contrast, the descriptor is quite affected by the Spectrum Pattern Textures. This research has discovered the importance of the spectrum pattern textures. These spectrum pattern textures contain information about how the texture units repeat. It can be greatly helpful for finding texture formation information (see the discussion in Sections 5.2.2 and 5.3.1).

The frequency domain approach has the ability to extract the Pattern Size of a texture, but this ability is limited by changes of contrast and the way texture units repeat. Comparing with the spatial domain approach, the frequency domain approach is complicated especially compared with CLCM. Due to the limitations, it is not as accurate as the spatial domain approaches.

7.1.2 Linelikeness

The research proposes a frequency-based Linelikeness descriptor algorithm, which is described in Section 5.2.4. This descriptor can be used to distinguish between texture patterns that are comprised of lines versus closed shapes (such as rectangles and polygons) by the descriptor's value.

The Tamura algorithm for Linelikeness is built on its directionality result. Section 5.2.7 has discussed one scenario where the Tamura algorithm fails the directionality test. This can lead a wrong result of Linelikeness. The Frequency domain approach directly takes advantage of spectrum properties that a linear texture in spatial domain must have a strong linear pattern in frequency domain. Differences between each directional bin provides the result of linelikness.

Histogram equalization is one of popular approaches for improving the contrast of an image. It makes details of an image clearer and these details may have strong

intensities in the spectrum. Because the Linelikeness descriptor algorithm is based on the intensities of the spectrum, it can be affected by some contrast improvement approaches such as Histogram equalization. The same issue occurs when the image is zoomed in. The details of an object surface become a dominating texture pattern which affects the result of Linelikeness descriptor.

One example of these effects is illustrated in Table 6.1 (Testing Image 1.12). In this image the wall has many lines along the mortar edges but each brick also has its own texture, which becomes more apparent when Histogram equalization is applied.

7.1.3 Direction Unification

In Section 5.2.5, the thesis introduces a new texture feature descriptor, which is called Directional Unification. It measures the unification of the texture pattern's directionality. This measurement is used with patterns that have a high Linelikeness texture pattern and for further categorizing these high Linelikeness texture patterns into directionally unified and directionally non-unified groups. Some textures may have very similar Linelikeness values, but the Directional Unification measure can still distinguish between them. This provides an opportunity for more sensitive texture pattern matching.

A spatially-based approach such as Hough transform and Tamura texture feature descriptor, can be used for Directional Unification descriptor by counting the number of directionalities but they all based on the directionality. However the spatially-based approach does not consider the number of lines for each direction. A spatially-based approach may have low Directional Unification value if one line has different directionality from others. However the number of lines can be represented as intensity on the spectrum. Therefore a frequency-based approach performs better for texture Directional Unification descriptor.

7.1.4 Directionality

This research proposes a frequency-based directionality descriptor and compares it with spatially-based Tamura directionality in Sections 5.2.6 and 5.2.7. The comparison shows the frequency-based directionality descriptor has advantages over spatially-based Tamura directionality descriptor. Tamura directionality has difficulty obtaining appropriate feature descriptors for computer generated textures, due to pixelation effects (see Section 5.2.7). Frequency-based directionality could be a replacement for Tamura directionality measure for such textures.

Hough transform can also find the directionality of a texture image but it is based on an edge detection algorithm such as Canny edge detection. The Canny edge detection algorithm needs to set an upper threshold and a lower threshold. Different ratios of these two thresholds may cause different directionalities to be detected. When the images are like Figure 6.15, Hough transform directionality can fail.

GLCM has ability of detecting texture directionality, but it only allows a very limited range of angles to be detected. GLCM is build on eight neighbourhood pixels therefore it only can detect directionality in four directionalities.

A potential limitation of a frequency-based approach is setting up the bins. The directionality of a texture can be on the boundary of the bins. The algorithm detects two consecutive directional bins have the same high intensity. In this case, the directionality is not going fail by averaging two consecutive directionality bins.

7.1.5 Texture Pattern Formation

The research introduces an algorithm for Texture Pattern Formation descriptors. These descriptors describe how the texture pattern units are repeated and the directionality of

the units repeat. These descriptors utilise the spectrum pattern texture together with Linelikeness and Directionality. These descriptors provide accurate measures for describing how the pattern has been repeated (see Section 5.3).

The Texture Pattern Formation descriptors strongly support that frequency-based approach is suitable for global texture features. A limitation of frequency-based texture pattern descriptors is that it require running the Fourier transform twice. This approach may not be suitable for some time-critical applications.

7.1.6 Hu Moments

In Section 5.1, the thesis discusses applying Hu moments for describing texture spectrum patterns, which was found to not be very successful. A spectrum image has center symmetry and Hu moments cannot describe a center symmetry pattern well. The second order moments are all zeros. Although the algorithm only uses half of the spectrum image, the high similarity of the spectrum across different images causes the algorithm to not work very well (see Table 5.1, 5.2, 5.3).

7.1.7 Image Pre-processing

In order to develop practical frequency-based texture feature descriptors the research also develops and proposes techniques for preprocessing images before the descriptors are extracted. Section 3.1 discusses a shadow-removing algorithm. Although the image will have its original colour scaled, it still maintains its texture pattern.

The thesis also discusses two frequency-based segmentation algorithms in Section

3.2.2. These two segmentation algorithms are background translation invariant and allow multiple foreground colours to be present, which cannot be achieved by colour based or motion based segmentation.

Often the Fourier transform suffers from leakage and the spectrum pattern is corrupted by noise in the spectrum image. The thesis compares different window functions for reducing the issue of leakage in Section 4.1, using Blackman, Hamming, Kaiser window functions and Sinc window functions. The research also introduces a new approach to improve the appearance of a spectrum image by applying square root function and cosine function based on the approach, which is proposed in [91].

Compare the cosine function with histogram equalization, the cosine function is faster than histogram equalization. Histogram equalization needs to scan an image at least twice. Although histogram equalization is fully “automatic” and does not need further parameter specifications, it also enhances the intensity of the noise in the spectrum image.

7.1.8 Overview of Frequency-Based Approach

The frequency-based approach has been shown to have translation invariance and be less affected by image intensity and image blurring. To compare with the spatially-based approach, the frequency-based approach is better for global texture features. It provides better accuracy for Linelikeness, Directional Unification and Directionality. Furthermore, it also provides Texture Pattern Formation descriptors, which are new feature descriptors introduced in this thesis. The limitations are also obvious, a pure frequency-based approach does a poor job extracting some local texture features such as Texture Pattern Size.

Overall, the spatially-based approach is like watching an image by locally through a sliding window. The magnifier is the sliding window. It is hard to consider the relations between the pixels which are inside the sliding window and outside the sliding window. However the frequency-based approach is like watching an entire image at one time, which makes it harder to extract fine details from the image.

7.2 Further Research

The Frequency-based approach provides a better way to extract global texture features, but it does not strongly support extracting local texture features. However the second Fourier transform potentially contains some information about local texture features. This could be explored more.

In fact, the second Fourier transform is back to spatial domain. However it loses the position information, as the phase angles are not involved in the second Fourier transform. This reorganizes the texture units' position to be center symmetric. It is possible to get more texture units information by following the direction, which is provided by texture formation's directionality. Further research of the texture feature descriptors could includes the texture unit size and density.

Another direction of further research could be improving texture classification and machine learning by using these texture descriptors. It also has many potential practical applications, such as object identification, automated navigation, visually impaired assistance and surveillance.

References

- 1 Efford, N. 2000 “*Digital Image Processing: A Practical Introduction Using Java.*” Addison Wesley.
- 2 Haykin, S. 1996 “*Adaptive Filter Theory*” Prentice Hall, Upper Saddle River, N.J.
- 3 Umbaugh, S.E 2005. “*Computer Imaging: Digital Image Analysis and Processing*” CRD Press, Boca Raton, FL.
- 4 Rheoridis, S. and Konstantinos, K. 2006 “*Pattern Recognition*” 3rd edition, Academic Press, New York.
- 5 Huang, K.Y 2002 “*Syntactic Pattern Recognition for Seismic Oil Exploration*” World Scientific, Hackensack, NJ.
- 6 Finlayson, G. D., Hordley, S. D. and Drew, M. S. 2002 “*Removing Shadows from Images*” In European conference on computer vision (pp. 823-836). Springer, Berling, Heidelberg. School of Information Systems, University of East Anglia Norwich England.
- 7 Peebles, P.Z. 1993. “*Probability, Random Variables, and Random Signal Principles*”, 3rd edition McGraw-Hill New York.
- 8 Rafael C. Gonzalez, Richard E. Woods 2008 “*Digital Image Processing*” 3rd Edition Pearson Prentice Hall. Chapter 10, 11
- 9 Brutzer, S., Höferlin, B., & Heidemann, G. 2011. *Evaluation of background subtraction techniques for video surveillance*. In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on (pp. 1937-1944). IEEE.
- 10 Sobral, A., & Vacavant, A. 2014. *A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos*. Computer Vision and Image Understanding, 122, 4-21.
- 11 Whitney, D., & Levi, D. M. 2011. *Visual crowding: A fundamental limit on conscious perception and object recognition*. Trends in cognitive sciences, 15(4), 160-168.
- 12 Xin, W., 2007 “*Laplacian Operator-Based Edge Detectors.*” IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(5), 886-890.

- 13 Fram, J. R., and Deutsh, E. S. 1975 “*On the Quantitative Evaluation of Edge Detection Schemes and Their Comparison with Human Performance.*” IEEE Trans, Computer, vol. C-24,100(6), 616-628
- 14 Prewitt, J. M. S. 1970 “*Object Enhancement and Extraction.*” In picture processing and Psychopictorics, Academic Press, New York.
- 15 Houth, P. V. C. 1962 “*Methods and Means for Recognizing Complex Patterns.*” U. S. Patent 3 069 654.
- 16 Leandro, A. F. F and Manuel, M. O 2008 *Real-time line detection through an improved Hough transform voting scheme* ResearchGate in Pattern Recognition 41:299-314.
- 17 Masci, F. 2009 *Line Detection by Hough transformation* http://web.ipac.caltech.edu/staff/fmasci/home/astro_refs/HoughTrans_lines_09.pdf .
- 18 Canny, J. 1986 “*A Computational Approach for Edge Detection*” IEEE Trans. Pattern Anal. Machine Intell., Vol. 8, no. 6, 679-698
- 19 Syed Irfanullah, A 2004 “*Canny edge detection report*” Massey university, Image processing.
- 20 Zhou, P., Ye, W., & Wang, Q. (2011). “*An Improved Canny Algorithm for Edge Detection.*” Journal of Computational Information Systems.
- 21 Ballard, D. H.1981 “*Generalizing the Hough Transform to Detect Arbitrary Shapes*” Patt. Recognition, 13(12), 111-122.
- 22 Weijer, J. van de, Gevers, T and Smeulders, 2006 “*A Robust Photometric Invariant Features from the Color Tensor*” IEEE Trans, Image Process, Vol. 15(1), 118-127.
- 23 Rastislav L and Konstantions N. P. “*Color Image processing Methods and Applications*” CRC Press Taylor & Francis Group, chapter 9.
- 24 Fan yang, 2010 “*Digital Image Processing and Analysis*” second edition, Beijing University of Aeronautics and Astronautics, chapter 8.
- 25 Amit, Y. and Kong, A 1996 “*Graphical Templates for Model Registration*”. IEEE Transactions on pattern analysis and machine intelligence, 18(3), 225-236.
- 26 Belongie, S. Malik, J. and Puzicha, J. 2000 “*Shape Context: A New Descriptor for Shape Matching and Object Recognition.*” In Advances in neural information

processing systems, 831-837

27 Mori, G. and Malik, J 2004 “*Recognizing Objects in Adversarial Clutter: Breaking a Visual CAPTCHA*” Computer Science Division University of California, Berkeley, CA 94720.

28 Mori, G. and Malik, J 2001 “*Learning to Find Brightness and Texture Boundaries in Natural Images.*” In NIPS.

29 Kokkinos, I., Bronstein, M. M., Litman, R., & Bronstein, A. M. 2012. *Intrinsic shape context descriptors for deformable shapes*. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on (pp. 159-166). IEEE.

30 Polewski, P., Yao, W., Heurich, M., Krzystek, P., & Stilla, U. 2015. *Free Shape Context descriptors optimized with genetic algorithm for the detection of dead tree trunks in ALS point clouds*. In LS2015—ISPRS Workshop Laser Scanning 2015. 41-48

31 Sabhara, R. K, Lee, C. P. and Lim, K. M. 2013 *Comparative Study of Hu Moments and Zernike Moments in Object Recognition* Smart Computer Review, Vol.3, no. 3.166-173

32 Flusser, J. Suk, T. Zitova, B 2009 *Moments and Moment Invariants in pattern Recognition*. Wiley Online Library from <http://onlinelibrary.wiley.com/book/10.1002/9780470684757> .

33 Huang, Z. and Leng, J 2010 *Analysis of Hu 's Moment Invariants on Image Scaling and Rotation* Computer Engineering and Technology 2nd International Conference, on Vol7, V7-476. IEEE. INSPECT Accession Number 11522050, DOI 10.1109/ICCET.2010.5485542.

34 Papoulis, A. 1965 “*Probability, Random Variables, and Stochastic Processes.*” Third edition, McGraw-Hill, New York.

35 Hu, M. K. 1962 “*Visual Pattern Recognition by Moment Invariants.*” IRE Trans. Info. Theory, vol. IT-8.

36 Bell, E.T. 1965 “*Men of Mathematics*”, Simon & Schuster, New York.

37 Huang, Z. and Leng, J 2010 “*Analysis of Hu’s Moment invariants on Image Scaling and Rotation.*” 2nd International Conference on Computer Engineering and Technology (ICCET) (Vol. 7, V7-476.

38 Flusser, J., Suk, T. and Zitova, B. 2009 “*Moments and Moment Invariants in*

Pattern Recognition.” John Wiley & Sons, Ltd. ISBN: 978-470-69987-4.

39 Giessen, M. and Schmidhuber, J 2005 “*Fast Color-Based Object Recognition Independent of Position and Orientation*” Springer-Verlag Berlin Heidelberg ICANN 2005, LNCS 3696.

40 Moghimi, M 2011 “*Using Color for Object Recognition*” Computer Science and engineering Department University of California, San Diego.

41 Duda, R.O., Hart, P. E., and Stork, D. G. 2001 “*Pattern Classification*” John Wiley & Sons, New York.

42 Tou, J. T., and Gonzalez, R. C. 1974 “*Pattern Recognition Principles*” Addison-Wesley, Reading, Mass.

43 Rafael C. Gonzalez, Richard E. Woods 2008 “*Digital Image Processing*” 3rd Edition Pearson Prentice Hall chapter 12.

44 Fisher, R. A 1936 “*The Use of Multiple Measurements in Taxonomic Problems*” Annals of human genetics, Vol. 7, Part 2. 179-188.

45 Cortes. C. and Vapnik. V. 1995 “*Support-Vector Networks*”, Machine Learning, 20(3), 273-297.

46 Bradski, G. and Kaebler, A 2009 “*Learning OpenCV Computer Vision With the OpenCV Library*” O’Reilly chapter 13.

47 Mohri. M, and Rostamizadeh A and Talwalkar A 2012 “*Foundations of machine Learning*” The MIT press ISBN 978262018258.

48 Haralick, R. M. Shanmugam, K. and Dinstein, I 1973 “*Textural Features for Image Classification*” IEEE Trans, on System, Man and Cybematics, Vol. SMC-3, No.6. 610-621.

49 Howarth, P., & Rüger, S. 2004. “*Evaluation of texture features for content-based image retrieval.*” In International conference on image and video retrieval (pp. 326-334). Springer Berlin Heidelberg.

50 Albregtsen, F. 2005 “*Statistical Texture Measures Computed from Gray Level Cooccurrence Matrices*” Image Processing Laboratory Department of Informatics, University of Oslo.

51 Gadkari, D. 2004 “*Image Quality Analysis Using GLCM*” the College of Arts and Sciences, University of Central Florida, Orlando, Florida.

- 52 Mohanaiah, P. Sathyanarayana, P. and GuruKumar, L. 2013 "*Image Texture Feature Extraction Using GLCM Approach*" International Journal of Scientific and Research Publications, Volume 3, Issue 5, ISSN 2250-3153. 1
- 53 Pathak, B. and Barooah, D 2013 "*Texture Analysis Based On the Gray-Level Co-occurrence matrix considering possible orientations*" International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering Vol.2, Issue 9 ISSN 2320-3765. 4206-4212.
- 54 Howarth, P. and Ruger, S 2004 "*Evaluation of Texture Features for Content-Based Image Retrieval*" Springer-Verlag Berlin Heidelberg LNCS 3115.
- 55 Mokji, M. M. 2007 "*Gray Level Co-Occurrence Matrix Computation Based On Haar Wavelet*" IEEE Digital Library 273-279. (Also see Computer Graphics, Imaging and Visualisation, 2007. CGIV '07).
- 56 Ojala, T. Pitikainen, M. and Harwood 1994 "*Performance Evaluation of Texture Measures with Classification Based on Kullback Discrimination of Distributions*", Proceedings of the 12th IAPR International Conference on Pattern Recognition, vol.1. 582-585
- 57 Ojala, T. Pitikainen, M. and Harwood 1996 "*A comparative Study of Texture Measures with Classification Based on Feature Distributions*", Pattern Recognition, vol.29. 51-59
- 58 Shapiro, L. and Stockman, G. 2000 "*Computer Vision*" Computer Science and Engineering, University of Washington.
- 59 Chen, C. H. and Patrick, W 2005 "*Handbook of Pattern Recognition and Computer Vision*" River Edge, NJ: World Scientific, c2005. ISBN 9812561056.
- 60 Zhou, C., Bichot, C.E. and Chen, L 2010 "*Multi-scale color local binary patterns for visual object classes recognition*" In Pattern Recognition(ICPR), 2010 2nd International Conference on Pattern Recognition. 3065-3068
- 61 Zhang, L., Zhang, L., Guo, Z. and Zhang, D. 2010 "*Monogenic-LBP: A new approach for rotation invariant texture classification.*" In Image Processing (ICIP) International Conference 2677-2680
- 62 Zhang, B., Gao, Y., Zhao, S. and Liu, J. 2010 "*Local derivative pattern versus local binary pattern: Face recognition with high-order local pattern descriptor.*" IEEE Transaction on Image Processing. 19(2) 533-544

- 63 Bagri, N and Johari, P. K. 2015 “*A Comparative Study on Feature Extraction Using Texture and Shape for Content Based Image Retrieval*” International Journal of Advanced Science and Technology Vol.80. 41-52
- 64 Chang, W. Y., Huang, A., Yang, C. Y., Lee, C. H., Chen, Y. C., Wu, T. Y., & Chen, G. S. 2013. “*Computer-aided diagnosis of skin lesions using conventional digital photography: a reliability and feasibility study.*” PloS one,8(11), e76212.
- 65 Ax Liu, Y., Zhang, D., Lu, G., & Ma, W. Y. 2006. “*Study on texture feature extraction in region-based image retrieval system.*” In 2006 12th International Multi-Media Modelling Conference (pp. 8-pp). IEEE.
- 66 Qi, Y. L. 2009. “*A relevance feedback retrieval method based on Tamura texture.*” In Knowledge Acquisition and Modeling, 2009. KAM'09. Second International Symposium on (Vol. 3, pp. 174-177). IEEE.
- 67 Majtner, T., & Svoboda, D. 2012. “*Extension of tamura texture features for 3d fluorescence microscopy.*” In 2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization & Transmission (pp. 301-307). IEEE.
- 68 Castelli, V. and Bergman, L. D. (Eds.). 2002 “*Image Databases: Search and Retrieval of Digital Imagery*”. Wiley: New York.
- 69 Martin J. T, Jonathan M. B, Patrick R. A 1998 “*Fractal Geometry in Digital Imaging*” ACD Press chapter 2.2.
- 70 Haridas, K., & Thanamani, A. S. 2014. “*Well-organized content based image retrieval system in RGB Color Histogram, Tamura Texture and Gabor Feature.*” International Journal of Advanced Research in Computer and Communication Engineering, 3(10), 1-7.
- 71 Manjunath, B. S., & Ma, W. Y. 2002. “*Texture Features for Image Retrieval.*” Image Databases, John Wiley & Sons, Inc 313.
- 72 Aggarwal, H., & Kumar, P. 2014. *Indian currency note denomination recognition in color images*. International Journal on Advanced Computer Engineering and Communication Technology, 1(1), 12-18.
- 73 Majtner, T. and Svoboda, D 2012 “*Extension of Tamura Texture Features for 3D Fluorescence Microscopy*” IEEE, 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on. 301-307
- 74 Zhang. X., Lok. Y, and Lyu, M. R. 2005 “*A New Feature of Uniformity of Image*

Texture Directions Coinciding with the Human Eyes Perception” Springer-Verlag Berlin Heidelberg, LNAI 3614. 727-730

75 Islam, M. M., Zhang, D. and Lu, G 2008 “*A Geometric Method to Compute Directionality Features for Texture images*” IEEE international Conference DOI:10.1109, ICME 2008.4607736. 1521-1524

76 Bagri, N., and Johari, P. K. 2015. “*A Comparative Study on Feature Extraction using Texture and Shape for Content Based Image Retrieval.*” International Journal of Advanced Science and Technology, 80, 41-52.

77 Mohammadi, S. M., Helfroush, M. S. and Kazemi, K. 2012 “*Novel Shape-Texture Feature Extraction for Medical X-Ray Image Classification*” International Journal of Innovative Computing, Information and Control Volume 8, Number 1(B), ISSN 1349-4198.

78 Georg B, 1978 “*z-Transform DFT filters and FFTs,*” IEEE Trans, on Acoustics, Speech and Signal Processing. 26, no. 1(1978) 56-63

79 Temperton, C, 1992 “*A Generalized Prime Factor FFT Algorithm For Any $N = 2^p 3^q 5^r$* ” Society for Industrial and Applied Mathematics. SIAM Journal on Scientific and Statistical Computing, 13(3), 676-686

80 Kulkarni, S. R. 2002 “*Frequency Domain and Fourier Transforms,*” Information Sciences and Systems, Princeton University.

81 Bracewell, R. N. 1995 “*Two-Dimensional Imaging*” Prentice Hall, Upper Saddle River.

82 Bracewell, R. N. 2000 “*The Fourier Transform and its Applications*”, 3rd edition McGraw Hill, New York.

83 Smith, J. O. 2003 “*Mathematics of the Discrete Fourier Transform*” W3K Publishing, CCRMA, Stanford, CA.

84 Das, A. 2012 “*Signal Conditioning*” Signals and Communication Technology, Springer-Verlag Berlin Heidelberg DOI: 10.1007/978-3-642-28818-0_2.

85 Edwin J. S. and Raymond A. A. 2004 “*The Importance of the Phase Transfer Function to Visual Function and Visual Quality Metrics*” Journal of Refractive Surgery vol 20 Issue 5: S504-S507.

86 Owens, R. 2004 “*Fourier Transform Theory*” Computer Vision, Department of Computer Science, The University of Edinburgh School of Informatics.

87 Roberts, S. 2000 “*The Discrete Fourier Transform*” Information engineering, Department of Engineering Science, University of Oxford. from <http://www.robots.ox.ac.uk/~sjrob/Teaching/SP/> .

88 2012 “*The Polyphase Filter Bank Technique*” from https://casper.berkeley.edu/wiki/The_Polyphase_Filter_Bank_Technique .

89 Roads, C 2002 “*Microsound 1st edition.*” MIT Press ISBN 0-262-18215-7..

90 Weisstein, E. W. 2003. “*CRC Concise Encyclopedia of Mathematics*”. CRC Press. ISBN 1-58488-347-2.

91 Rafael C. Gonzalez, Richard E. Woods 2008 “*Digital Image Processing*” 3rd Edition Pearson Prentice Hall chapter 4.

92 Weinhaus, F 2011 “*ImageMagick v6 Examples – Fourier Transforms*”. from <http://www.imagemagick.org/Usage/fourier/> .

93 Shepherd, M. 2012 *Correct Sampling of Diffraction Limited Image* California Institute of Technology from http://wiki.astro.cornell.edu/twiki/pub/CCAT/CCAT_Memos/DiffractionLimitedSampling111212.pdf .

94 Roger, D. B. and Kip, S. T. 2012 *Applications of Classical Physics* Division of Physics, Caltech Mathematics and Astronomy chapter 8: Diffraction From <http://www.pmaweb.caltech.edu/Courses/ph136/yr2012/1208.1.K.pdf> .

95 Blahut, R. E. 2004. “*Theory of remote image formation.*” Cambridge University Press.

96 Abdel-Karim, A. and Abu-Ein, H 2014 “*A Novel Methodology for Digital Removal of Periodic Noise Using 2D Fast Fourier Transforms*” Contemporary Engineering Sciences, Vol.7, No.3.

97 Bracewell, R. N 2000 “*The Fourier Transform and Its Applications*” McGraw-Hill Science Engineering ISBN 10: UCSD: 31822031267685.

98 Karam, M., Khazaal, H. F., Aglan, Heshmat and Cole, C 2014 “*Noise Removal in Speech Processing Using Spectral Subtraction*” Journal of Signal and Information Processing, Vol. 5, No. 2. 32

99 Poljicak, A., Mandic, L., & Agic, D. 2011. “*Discrete Fourier transform-based*

watermarking method with an optimal implementation radius.” Journal of Electronic Imaging, 20(3), 033008-033008.

100 Panyuskin, S. V. 2006 “*Generalized Fourier Transform and Its Applications*” Mathematical Notes Vol. 79, Issue 3. 537-550

101 Castleman, K. R. 1996 “*Digital Image Processing*” 2nd edition Prentice Hall, Upper Saddle River, N.J.

102 Pratt, W. K. 2001 “*Digital Image Processing*” 3rd edition, John Wiley & Sons, New York.

103 Engelberg, S. 2008 “*Edge detection using Fourier coefficients*” American Mathematical Monthly June – July. 115(6), 499-513

104 Tadmor, E and Zou, J 2008 “*Three Novel Edge Detection Methods for Incomplete and Noisy Spectral Data*” Journal of Fourier Analysis and Applications Vol. 14 DOI: 10.1007/s00041-008-9038-9. 744-763

105 Gelb, A. and Hines, T 2011 “*Detection of Edges from Nonuniform Fourier Data*” Journal of Fourier Analysis and Applications Vol. 17 DOI: 10.1007/s00041-011-9172-7. 1152-1179

106 Abdel-Qader, I., Abudayyeh, O., and Kelly, M. 2003 “*Analysis of Edge-Detection Techniques for Crack Identification in Bridges*” Journal of Computing in Civil Engineering, Vol. 17, Issue 4. 255-263

107 Gelb, A. and Cates, D 2009 “*Segmentation of Images from Fourier Spectral Data*” Communications in Computational Physics Vol. 5, No. 2-4. 326-379

108 Yang, L. Zhang, B. Ye, X 2004 “*Fast Fourier Transform and its Applications*” Opto-electronic Engineering 2004-S1. 31, 1-7

109 Shivakumara, P., Phan, T. Q., & Tan, C. L. 2011. “*A Laplacian approach to multi-oriented text detection in video.*” IEEE transactions on pattern analysis and machine intelligence, 33(2), 412-419.

110 Porco, C. C. and West R. A. 2004 “*Cassini Imaging Science: Instrument Characteristics and Anticipated Scientific Investigations at Saturn*” Space Science Reviews, vol. 115. 363-497

111 Bakir, T. and Reeves, J. S. 2000 “*A Filter Design Method for Minimizing Ringing in a Region of Interest in MR Spectroscopic Images*” IEEE Trans, Medical Image, vol. 19 no.6. 585-600

- 112 Nussbaumer, H. J. 2012. *"Fast Fourier Transform and convolution algorithms"* Vol. 2. Springer Science & Business Media.
- 113 Shan, Q. Jia, J. 2008 *"High-Quality Motion Deblurring From A Single Image"* ACM Transaction on Graphics (TOG) SIGGRAPH 2008 Vol. 27, No 3, 73.
- 114 Krahmer, F. Lin, Y. McAdoo, Bonnie. Ott, K. Wang, J. Widemann, D. and Wohlberg, B 2006 *"Blind Image Deconvolution: Motion Blur Estimation"* Institute for Mathematics and Its Applications, University of Minnesota.
- 115 Gonzalez R. Woods R. and Eddins, S. 2003 *"Digital Image Processing Using Matlab"* Prentice Hall.
- 116 Ren, J. Q 2010 *"Novel Adaptive Algorithm of Shadow Elimination for Video Moving Objects"* Computer Engineering and Applications vol. 46(36).
- 117 Alexander, P. T., Josef, C. and Zhanqing, L. 2002 *"Effects of spectral response function on surface reflectance and NDVI measured with moderate resolution satellite sensors"* Remote Sensing of Environment 81(2002) 1-18.
- 118 Boykov, Y., & Funka-Lea, G. 2006. *"Graph cuts and efficient N-D image segmentation."* International journal of computer vision, 70(2), 109-131.
- 119 Arbelaez, P., Maire, M., Fowlkes, C., & Malik, J. 2011. *"Contour detection and hierarchical image segmentation."* IEEE transactions on pattern analysis and machine intelligence, 33(5), 898-916.
- 120 Agarwal, P., Singh, S. P., & Kumar Pandey, V. 2014, November. *"Mathematical analysis of blackman window function in fractional Fourier transform domain."* IEEE In Medical Imaging, m-Health and Emerging Communication Systems (MedCom), 2014 International Conference 120-125
- 121 Gupta, H. R., Mehra, R., & Batan, S. 2013. *"Power spectrum estimation using Welch method for various window techniques."* International Journal of Scientific Research Engineering & Technology, 2(6), 389-392.
- 122 Podder, P., Khan, T. Z., Khan, M. H., & Rahman, M. M. 2014. *"Comparative Performance Analysis of Hamming, Hanning and Blackman window."* International Journal of Computer Applications, 96(18).
- 123 Rajput, S. S., & Bhadauria, D. S. 2012. *"Implementation of fir filter using efficient window function and its application in filtering a speech signal."*

International Journal of Electrical, Electronics and Mechanical Controls, 1(1).

124 Liu, Y., Yin, Y., & Zhang, S. 2012, “*Hand gesture recognition based on Hu moments in interaction of virtual reality.*” IEEE In Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2012 4th International Conference Vol. 1, 145-148.

125 Huang, M., Ma, Y. Q., Shu, H. Z., & Gong, Q. P. 2013. “*Trademark Recognition Based on Hu Modified Invariant Moments.*” In Applied Mechanics and Materials Vol. 397, 2313-2317. Trans Tech Publications.

126 Flusser, J. and Suk, T. 2006 “*Rotation Moment Invariants for Recognition of Symmetric Objects*”, IEEE Transactions on Image Processing, vol. 15, no. 12. 3784-3790

127 Neelima, B. and Punit, K. J. 2015 “*A Comparative Study on Feature Extraction using Texture and Shape for Content Based Image Retrieval*” International Journal of Advanced Science and Technology Vol.80. 41-52

128 Georgy, G 2012 “*CBIR: Texture Features*” from <https://www.cs.auckland.ac.nz/courses/compsci708s1c/lectures/Glect-html/topic4c708FSC.htm> The University of Auckland.

129 Chamorro-Martínez, J., & Martínez-Jiménez, P. 2009. “*A comparative study of texture coarseness measures.*” In 2009 16th IEEE International Conference on Image Processing (ICIP) 1337-1340

130 Klette, R. 2014 “*Concise computer vision an introduction into theory and algorithms*” Springer-Verlag, London.