The Effect of Changing Illumination Tilt on Texture Features

Michael Schmidt

October 2001

Contents

List of Figures

List of Tables

Acknowledgements

I would specially like to thank my supervisor Dr. Mike Chantler for his ideas, support and guidance throughout this project.

I also would like to thank Prof. Dr.-Ing. Eberhard Hänsler and Dipl.-Ing. Henning Puder from Darmstadt University of Technology and Dr. Mike Chantler from Heriot-Watt-University for making the ERASMUS exchange programme possible.

I am particularly grateful to Ged and all the other people in the Texture Lab for their help and patience in the lab and their entertainment during free time.

I especially would like to thank Ged for sharing his flat with me when I lost my room in the final phase of my project.

Special thanks to all my friends in Edinburgh and Germany and to my family for their support and encouragement.

Abstract

Texture recognition has been a very active research area during the last three decades. Although a large amount of progress has been made in many areas, there was not much work on the effect of varying light conditions. Especially the effects of changing illumination direction have received little attention. This work shows that a wide range of texture features have a sinusoidal behaviour due to illumination tilt rotation.

Starting from a linearised diffuse reflection model, we derive analytically a sinusoidal prediction for texture features which are generated by linear filters followed by an signal energy estimation. In addition we make further predictions for the behaviour of isotropic and unidirectional surfaces. We also show that the resulting curve of these features in the two-dimensional feature space in general is an ellipse. We define the boundary conditions for which the marginal cases of the ellipse, circle and line, occur.

We verify these predictions both by simulations and experiments. For simulation we use five different synthetic surfaces and their artificial illuminated images. To show the general validity of our sinusoidal prediction, we investigate the behaviour of 30 different real world textures.

Chapter 1

Introduction

1.1 Motivation

Texture analysis is a growing subject of interest and research. One problem of texture analysis is that the input of the classifier is usually an image of the surface instead of the intrinsic surface. This can be mastered if the image is only dependent on the surface height map and all other influences are negligible or constant. One of the main effects which can change the image of a surface dramatically are light conditions. Figure 1.1 on the following page shows two textures illuminated with different tilt angles. Although all other parameters are constant, the images look quite different. Thus illumination tilt seems to have a large influence on images of 3D textures.

The main motivation of this dissertation is to derive a prediction for the behaviour of texture features due to tilt rotating. To make this statement as general as possible, we derive this prediction using as few as possible restrictions for the textures and the feature generation.

1.2 Area of Research

Although the area of 3D surface recognition is a large research field, surprisingly there was not much work about the effect of illumination conditions on texture features.

[4] showed that classification is effected by the illumination spectral distribution, i.e. by the colour of the light. He presented illumination invariant features using a linear model for surface spectral reflectance.

[3] showed that the illumination direction has a deep impact on the image of a texture and thus on classification. Classifiers, which ignore the effect

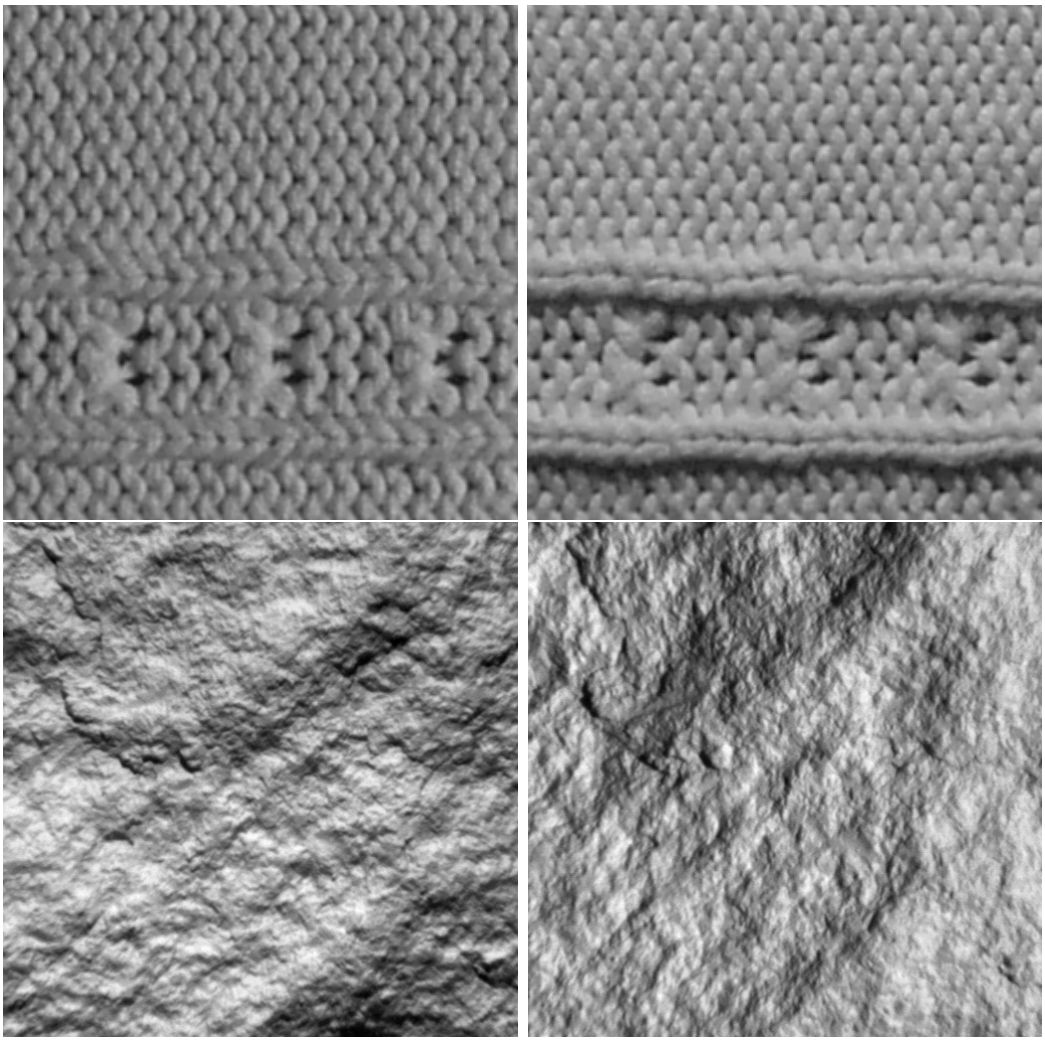


Figure 1.1: two textures illuminated with different tilt angles

of illumination changes, can fail catastrophically. He also introduces tilt-compensation filters to reduce the effect of illumination tilt angle changes.

Rather than compensating the effect, [2] used a model based approach to obtain a illumination independent classification in the case that the illumination angle is changed between training and classification: The physical surface is estimated from several images using photometric stereo. This surface is used for classification instead of the image.

1.3 Dissertation Organisation

This dissertation consists of six chapters. **Chapter 2** presents the background theory needed for this work. First the Two-Dimensional Fourier Transform and two one-dimensional simplifications, the polar and the radial plot, are introduced. Also the Gabor and Laws filters are described. Finally Kube's Linearisation Model, as a link between the texture surface and its image, is derived. **Chapter 3** introduces four synthetically generated surfaces which are used for simulation later. **Chapter 4** gives a short review of texture feature generation in general and a description of the feature generation used for this work in detail. In **chapter 5** the effect of changing tilt direction is derived theoretically. A sinusoidal prediction for texture features due to tilt rotation is derived. The behaviour of isotropic unidirectional surfaces is investigated more closely. In addition the behaviour in the two-dimensional feature space is examined. In **chapter 6** the analytically derived predictions of chapter 5 are verified using simulations and experiments. For simulation the synthetic surfaces presented in chapter 3 are used. For the experiments 30 different real world textures with different attributes are used. **Chapter 7** summarises the results of this work and gives interesting points for further research.

Chapter 2

Background Theory

In this section we give a short overview of the background theory which is used in this work. Since this work uses the Two-Dimensional Fourier Transform extensive, we first give its definition. We also introduce two groups of linear filters which are used in this work: Gabor and Laws filters. The polar and radial plot as one-dimensional plots of the two-dimensional Fourier Transform are described. Finally we introduce a link between the surface texture and its illuminated image in frequency domain.

2.1 Two-Dimensional Fourier Transform

In this work we use the Fourier Transform to investigate the relationship between between the space and frequency domain. Since images are defined as a two-dimensional signal, its Fourier Transform is also two-dimensional. Because images are made up of pixels, they are defined only on a discrete grid. For this reason we use the Discrete Fourier Transform. The Discrete Two-Dimensional Fourier Transformation and its inverse Transformation are defined as follows:

$$
\begin{aligned}
F(u,v) &= \frac{1}{M\,N}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1} f(x,y)e^{-i\frac{2\pi}{M}ux}e^{-i\frac{2\pi}{N}vy} \\
f(x,y) &= \sum_{u=0}^{M-1}\sum_{v=0}^{N-1} F(u,v)e^{i\frac{2\pi}{M}ux}e^{i\frac{2\pi}{N}vy}
\end{aligned}
$$

where

$f(x,y)$ is the signal in spatial domain
$F(x,y)$ is the signal in frequency domain

M x N is the size of the image

Although the 2-D signal f(x,y) is often considered to be a real value and the integer or floating point intensity value at location (x,y), it need not to be always so. In general the Fourier Transform is a transform between two complex functions. Although the one-dimensional Fourier transform is well known, the interpretation of its two-dimensional generalisation is often tricky. Details and more properties of the Two-Dimensional Fourier Transform can be found in [10].

2.2 Polar and Radial Plot

The spectrum using a 2-D Fourier Transformation is a function of two variables (in this work we use u, v- or ω, θ-coordinates). In the following chapters we often use polar and radial plots is a graphic simplification. Because they are only dependent on one variable they can be drawn as a 2-D plot.

2.2.1 The Polar Plot

The polar plot adds the magnitudes of all frequencies in one certain direction to produce a measure for the intensity in this direction. All these frequencies lie in a radial line. It is used for illustrating directional properties of the image. The definition of the polar plot is

$$P(\theta) = \int_0^\infty F(\omega, \theta)\, d\omega$$

The polar plot shows the distribution of signal energy over direction. Due to the fact that a Cartesian grid is used for the FFT, the polar plots in this dissertation are only approximations. In fact we work with discrete images; thus we can not extend the integration infinitely, as a result, we can not calculate the exact polar plot of our images. A rotating of the image causes a shift of its polar plot of the same angle. Figure 2.1 on the next page shows *wood* in spatial and frequency domain. It shows that there is a high amount of energy in the direction of the x-axis. Figure 2.2 on the following page shows the polar plot of *wood*. It shows the directionality of *wood*.

2.2.2 The Radial Plot

The radial plot averages the magnitudes of radial frequencies over all directions of the image. The definition of the radial plot is

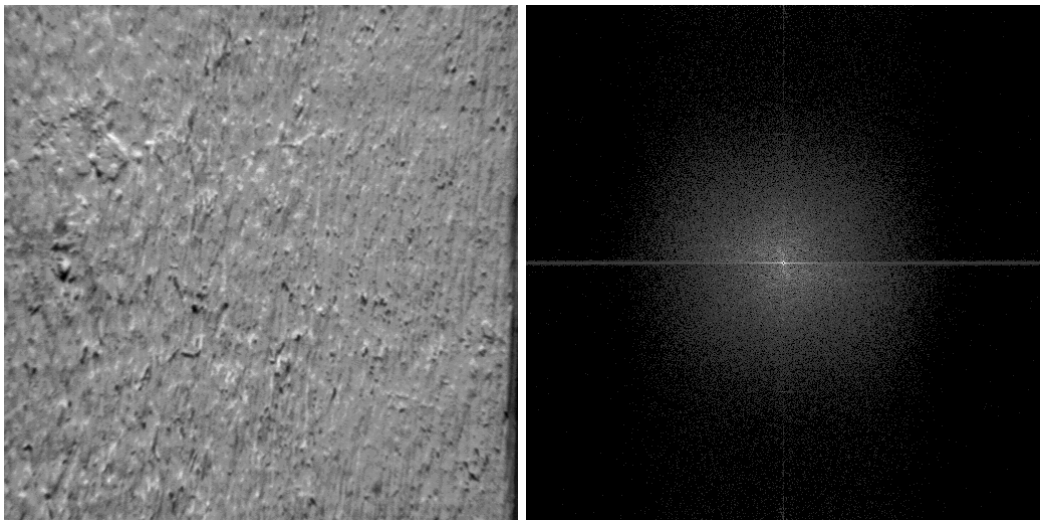


Figure 2.1: *wood* in spatial and frequency domain

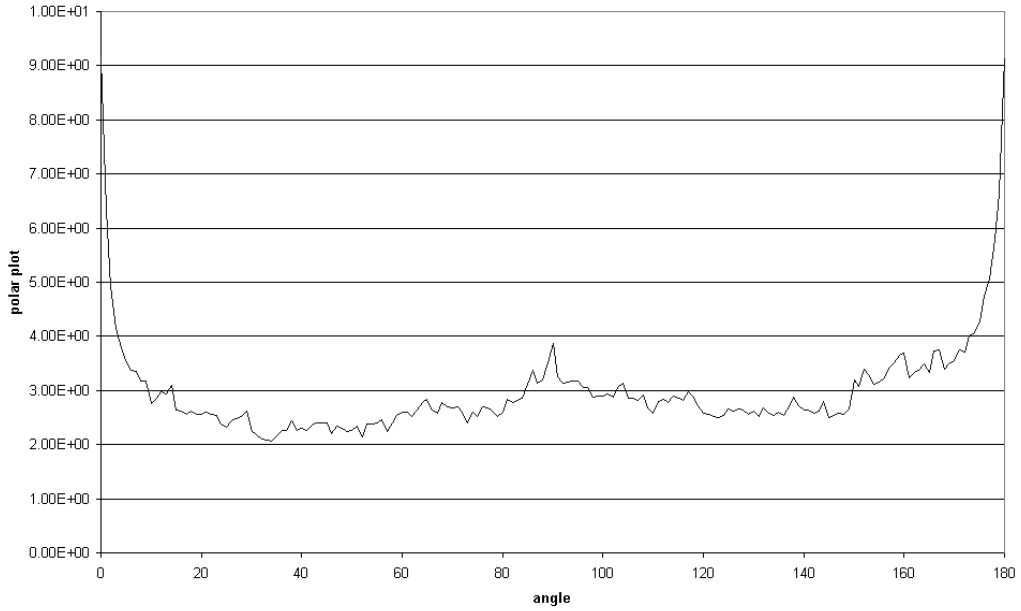


Figure 2.2: polar plot of *wood*

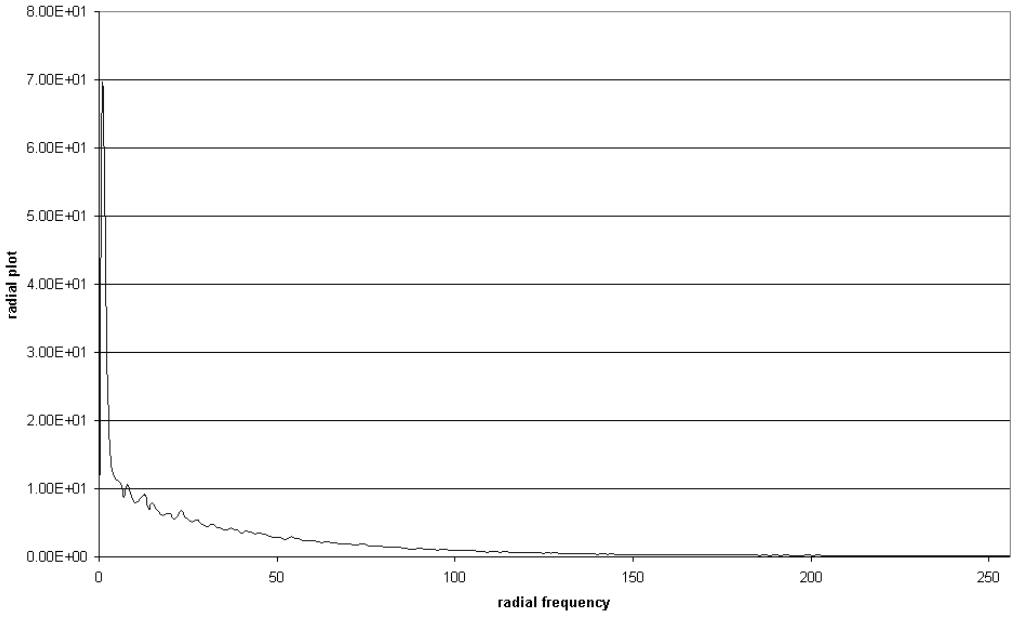


Figure 2.3: radial plot of *wood*

$$R(\omega) = \int_0^{2\pi} F(\omega, \theta)\, d\theta$$

The radial plot shows the radial shape of the frequency domain independently of the direction. Figure 2.3 shows the radial plot of *wood*. It shows that most of the image energy is in lower frequencies.

2.3 Gabor Filters

In this dissertation we use two groups of linear filters. We first introduce the Gabor filters which are very popular in the image processing society. One reason why these filters became quite popular is that the characteristics of certain cells in some mammals can be approximated by these filters. The one-dimensional spatial and spectral forms are

$$\begin{aligned} h(x) &= \exp\left(\frac{-x^2}{2\sigma^2}\right) \exp\left[j(2\pi\omega_0 x + \phi)\right] \\ G(\omega) &= A\left\{\exp[-2\pi\sigma^2(\omega-\omega_0)^2] + \exp[-2\pi\sigma^2(\omega+\omega_0)^2]\right. \end{aligned} \qquad (2.1)$$

$$
\begin{aligned}
&+ \quad \exp[-2\pi\sigma^2(\omega-\omega_0)^2] - \exp[-2\pi\sigma^2(\omega+\omega_0)^2]\Big\} \\
&= \quad 2A\,\exp[-2\pi\sigma^2(\omega-\omega_0)^2] \qquad (2.2)
\end{aligned}
$$

In spatial domain it consists of a Gaussian envelope modulated by a complex exponential. In frequency domain it consists of a Gaussian around the centre frequency ω_0 (equation 2.2). It can also be split into a real-even and a real-odd part (equation 2.1).

The spatial variance is dependent on the standard deviation σ of the Gaussian, which is in the denominator. The bandwidth in frequency domain is also dependent on σ, which is now in the enumerator. Thus a small spatial variance causes a large bandwidth and vice versa. For classification and segmentation problems both a small spatial variance and a small bandwidth is needed. This means that the value of σ is always a trade-off. It was shown that the complex Gabor filters have optimal localisation properties in both spatial and frequency domain and thus are ideal for these problems.

In the two-dimensional case Gabor filters have the following equation:

$$
h(x,y) = \exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y}\right)\right]\exp[j(2\pi\omega_0 x+\phi)]
$$

$$
\begin{aligned}
G(u,v) &= A\left\{\exp\left[-\frac{1}{2}\left(\frac{(u-u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right)\right]+\exp\left[-\frac{1}{2}\left(\frac{(u+u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right)\right]\right. \\
&+ \left.\exp\left[-\frac{1}{2}\left(\frac{(u-u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right)\right]-\exp\left[-\frac{1}{2}\left(\frac{(u+u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right)\right]\right\} \\
&= 2A\exp\left[-\frac{1}{2}\left(\frac{(u-u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right)\right]
\end{aligned}
$$

where

$$
\sigma_u = \frac{1}{2\pi\sigma_x} \qquad \sigma_v = \frac{1}{2\pi\sigma_y}
$$

A rotation of the x-y plane will result in Gabor filters at other orientations. Like the trade-off between spatial variance and bandwidth in the one-dimensional case, there exists also a trade-off between the spatial and spectral resolution. A large spatial variance causes a small bandwidth and vice versa. Instead of the whole complex filter sometimes only the real part of the pulse response is used:

$$
h(x,y) = \exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y}\right)\right]\cos(2\pi u_0 x+\phi)
$$

Because the pulse response is real and even, its transfer function is also real and even:

$$G(u,v) = A\left\{\exp\left[-\frac{1}{2}\left(\frac{(u-u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right)\right]+\exp\left[-\frac{1}{2}\left(\frac{(u+u_0)^2}{\sigma_u^2}+\frac{v^2}{\sigma_v^2}\right)\right]\right\}$$

This causes much less computation than using the whole complex filter. The disadvantage is that the trade-off between spatial variance and bandwidth is not longer optimal.

Examples of Gabor filters in frequency domain can be found in appendix C on page 74.

The previous Gabor filters filter have both a certain centre frequency and direction. In addition also isotropic Gabor filters are used. The transfer function in ω, θ-coordinates is

$$H(\omega) = \exp\left(-\frac{(\omega-\omega_0)^2}{\sigma^2}\right)$$

The usage of these filters and the advantages and disadvantages of isotropic filters are explained in chapter 4.

2.4 Laws Filters

In addition to the Gabor filters which where introduced in the previous section we use a second group of linear filters called Laws filters. [9] developed a set of two-dimensional masks derived from three simple one-dimensional filters. the basic one-dimensional filters are

L3 = (1,2,1) $\Longrightarrow$ "level detection"
E3 = (-1,0,1) $\Longrightarrow$ "edge detection"
S3 = (-1,2,-1) $\Longrightarrow$ "spot detection"

The magnitude frequency responses of these filters are:

$$\begin{aligned} |H_{L3}(\omega)| &= |e^{-j\omega}+2+e^{j\omega}| = 2(1+\cos\omega) \\ |H_{E3}(\omega)| &= 2\sin\omega \\ |H_{S3}(\omega)| &= 2(1-\cos\omega) \end{aligned}$$

Laws convolved these with each other to provide a set of symmetric and anti-symmetric centre-weighted masks with all but the level filters being zero sum. He found the most useful to be

L5 = L3 * L3 = (1,2,1) * (1,2,1) = (1,4,6,4,1) $\Longrightarrow$ "level detection"
E5 = L3 * E3 = (-1,-2,0,2,1) $\Longrightarrow$ "edge detection"
S5 = E3 * E3 = (1,0,-2,0,1) $\Longrightarrow$ "spot detection"
R5 = S3 * S3 = (1,-4,6,-4,1) $\Longrightarrow$ "ripple detection"

Because convoluting in time domain is equal to multiplying in frequency domain, the magnitude frequency responses of these filters can be obtained by multiplying the frequency responses of the previous filters:

$$\begin{aligned} |H_{L5}(\omega)| &= |2(1+\cos\omega)\,2(1+\cos\omega)| = 4(1+\cos\omega)^2 \\ |H_{E5}(\omega)| &= 4|\sin\omega(1+\cos\omega)| \\ |H_{S5}(\omega)| &= 4\sin^2\omega \\ |H_{R5}(\omega)| &= 4(1-\cos\omega)^2 \end{aligned}$$

These filters are convolved in turn with transposes of each other to give square masks. He found the most useful are

$$L5E5 = L5^T * E5 = \begin{pmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{pmatrix} * (-1,-2,0,2,1) = \begin{bmatrix} -1 & -2 & 0 & 2 & 1 \\ -4 & -8 & 0 & 8 & 4 \\ -6 & -12 & 0 & 12 & 6 \\ -4 & -8 & 0 & 8 & 4 \\ -1 & -2 & 0 & 2 & 1 \end{bmatrix}$$

$$E5S5 = \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \\ -2 & 0 & 4 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & -4 & 0 & 2 \\ 1 & 0 & -2 & 0 & 1 \end{bmatrix}$$

$$R5R5 = \begin{bmatrix} 1 & -4 & 6 & -4 & 1 \\ -4 & 16 & -24 & 16 & -4 \\ 6 & -24 & 36 & -24 & 6 \\ -4 & 16 & -24 & 16 & -4 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix}$$

$$L5S5 = \begin{bmatrix} -1 & 0 & 2 & 0 & -1 \\ -4 & 0 & 8 & 0 & -4 \\ -6 & 0 & 12 & 0 & -6 \\ -4 & 0 & 8 & 0 & -4 \\ -1 & 0 & 2 & 0 & -1 \end{bmatrix}$$

Since these two-dimensional Laws' masks are made up from separable one-dimensional filters, their frequency response may be obtained by multiplying in the frequency domain:

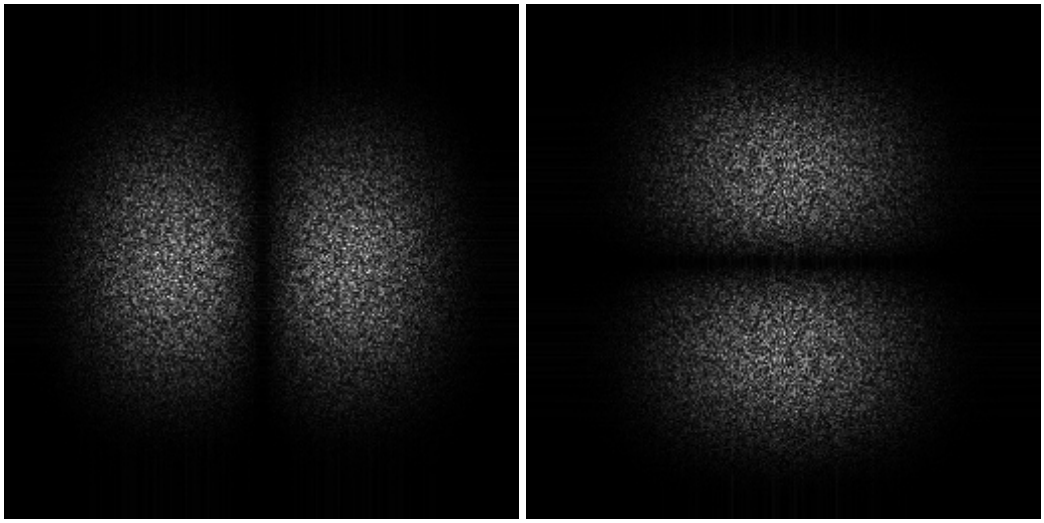


Figure 2.4: white noise in frequency domain after filtering with a L5E5 (left) and a E5L5 Laws filter (right)

$$
\begin{aligned}
|H_{L5E5}(u,v)| &= |H_{L5}(v)\, H_{E5}(u)| = 4(1+\cos v)^2\, 4\sin u(1+\cos u) \\
|H_{E5S5}(u,v)| &= 4\sin v\,(1+\cos v)\, 4\sin^2 u \\
|H_{R5R5}(u,v)| &= 4(1+\cos v)^2\, 4(1+\cos u)^2 \\
|H_{L5S5}(u,v)| &= 4(1-\cos v)^2\, 4\sin^2 u
\end{aligned}
$$

Figure 2.4(left) shows white noise in frequency domain after filtering with a L5E5 Laws filter. This matches the theoretically derived results very well. Changing the order of these filters (for example from L5E5 to E5L5) causes a 90° shift (figure 2.4 right).

2.5 Image Formation

To investigate the effect of changing illumination direction on rough surfaces the physical surface has to be linked to its resulting image. The first step is to link the surface derivatives to the image intensity. This is a local process applied to one facet of the surface.

2.5.1 The Lambertian Reflectance Model

To describe the relationship between the surface derivatives of a facet and the resulting image intensity the Lambertian model is used. [2] proofed that this model is a good approximation for a wide range of surfaces, especially for isotropic surfaces with low slope angles. This model is used for all surfaces used in this report. The Lambertian model is a model of diffuse reflection and deals with the local process of the light reflection of one facet. It states that the perceived intensity of a facet is dependent only on the relative geometry of the facet and the illumination direction. Although the radiance of the facet is constant with respect to the viewers position, the radiant intensity varies with the visible area of the facet. In the following an orthographic projection is assumed. Because the visible area is proportional to the cosine of the angle between the radiated light and the surface normal, the radiant intensity i can be expressed as the normalised dot product of the facet normal vector $\vec{s}$ with the illumination vector $\vec{l}$ times the albedo of the facet ρ:

$$i = \rho \frac{\vec{s} \cdot \vec{l}}{\left|\vec{s} \cdot \vec{l}\right|}. \tag{2.3}$$

2.5.2 Kube's Linear Model

In this section a transfer function in frequency domain between the surface and its resulting image is derived. We assume that the surface has Lambertian reflection and there are no significant self or cast shadowing or inter-reflection. Self shadowing means that the angle between the facet normal and the illumination vector becomes larger than 90°. In this case equation 2.3 becomes negative which has no physical sense. Cast shadowing means that a part of a surface is shadowed by another part of the surface. Inter-reflection means that light is reflected by one facet and reaches another facet. Adopting these assumptions equation 2.3 can be generalised to

$$i(x,y) = \rho(x,y)\frac{\vec{s}(x,y)\cdot\vec{l}(x,y)}{\left|\vec{s}(x,y)\cdot\vec{l}(x,y)\right|}. \tag{2.4}$$

$$\vec{s}(x,y) = \left[-\frac{\partial h}{\partial x} \quad -\frac{\partial h}{\partial y} \quad 1\right] = [-p(x,y) \quad -q(x,y) \quad 1] \tag{2.5}$$

where

$i(x,y)$ is the radiant intensity.
$\rho(x,y)$ is the albedo of the surface.
$\vec{s}(x,y)$ is the normal vector field of the surface height function $h(x,y)$.
$\vec{l}(x,y)$ is illumination vector field.
$p(x,y)$ is the slope in x direction.
$q(x,y)$ is the slope in y direction.

If the illumination is produced by a point source located in an infinite distance, $\vec{l}(x,y)$ can be simplified to

$$\vec{l}(x,y) = [k_1 \quad k_2 \quad k_3] \tag{2.6}$$

where

$$k_1 = \cos(\tau)\sin(\sigma) \qquad k_2 = \sin(\tau)\sin(\sigma) \qquad k_3 = \cos(\sigma)$$

τ tilt angle σ slant angle.

Using equation 2.5, 2.6 and $\left|\vec{l}(x,y)\right| = 1$ in 2.4 we get

$$i(x,y) = \frac{-k_1 p(x,y) - k_2 q(x,y) + k_3}{\sqrt{p^2(x,y) + q^2(x,y) + 1}} \tag{2.7}$$

For $p, q \ll 1$ we can linearise this equation to

$$i(x,y) = k_1 p(x,y) + k_2 q(x,y) + k_3$$

This equation can easily be transformed into frequency domain:

$$\mathcal{I}(u,v) = k_1\mathcal{P}(u,v) + k_2\mathcal{Q}(u,v) + k_3\delta(u,v)$$

Using polar coordinates we get:

$$p(x,y) = \frac{\partial h}{\partial x} \iff \mathcal{P}(u,v) = iu\mathcal{H}(u,v) = i\,\omega\,\cos(\theta)\mathcal{H}(u,v)$$

$$q(x,y) = \frac{\partial h}{\partial y} \iff \mathcal{Q}(u,v) = iv\mathcal{H}(u,v) = i\,\omega\,\sin(\theta)\mathcal{H}(u,v)$$

Ignoring the mean we get

$$\begin{aligned} \mathcal{I}(\omega,\theta) &= [k_1\,i\,\omega\,\cos(\theta) + k_2\,i\,\omega\,\sin(\theta)]\mathcal{H}(\omega,\theta) \\ &= i\,\omega\,\sin(\sigma)\cos(\theta-\tau)\mathcal{H}(\omega,\theta) \end{aligned} \tag{2.8}$$

Chapter 3

Synthetic Surfaces

3.1 Introduction

In this section we introduce some models for natural and machined surfaces. With these models we will be able to construct synthetic test surfaces for simulations. We will use these synthetic surfaces as a first step for investigating the effect of illumination tilt changing in section 6.2. Although simulations have disadvantages compared to experiments with real surfaces they are quite useful because many sources of error are eliminated. It is also possible to change the surface properties by changing the parameters, whereas in experiments we can only use a limited number of surfaces. After rendering the synthetic surface with a illumination algorithm we have both the illuminated surface image and the original surface height map. Using real surfaces we only get images of the surface but not the surface height map.

3.2 Generation of Synthetic Surfaces

Surfaces are in general two-dimensional random processes. Hence, they are described by means of their statistics. Because surfaces have a certain degree of spatial cohesion or correlation, their first order statistics give only a limited insight into their behaviours. It is customary and convenient to characterise surfaces by their second order statistics, the auto-correlation function (ACF). The following four models are described by the Fourier equivalent of the ACF, the power spectral density (PSD). To describe an image completely, both its PSD and its phase spectrum is needed. [8] and showed that although structured images have most of their information in the phase spectrum, unstructured images have all their information and characteristics in their PSD. All following models describe unstructured surfaces. Therefore

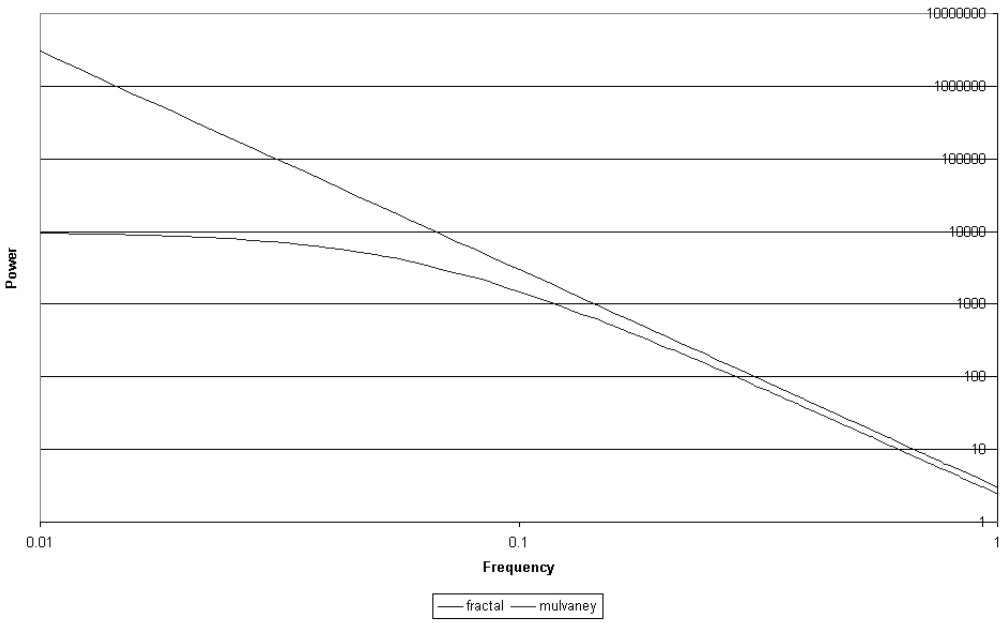


Figure 3.1: PSD of the fractal and Mulvaney surface models

we model for all models the phase spectrum as an uncorrelated random field with a normal distribution. The Inverse Fast Fourier Transform is used to generate the surface height map from the PSD and phase spectrum.

3.2.1 Fractal

We first consider a model for isotropic natural surfaces. Because the model is isotropic, the power spectrum is only dependent on the centre frequency ω. A large number of surfaces in nature have over several decades a power spectrum of the following form:

$$S(\omega) = \frac{k_1}{\omega^3} \tag{3.1}$$

This means that the Bode plot of the PSD is a line with a roll off of 3 (figure 3.1). Since the model is used in discrete form we assume the surface is band-limited above the Nyquist frequency.

Figure 3.2 on the next page shows the height map and the illuminated surface of a fractal which was modelled using equation 3.1. The illuminated surface is computed using Lambertian reflection (equation 2.7 on page 13) with an illumination slant angle of 45° and a tilt angle of 180°. Although the height map does not seem to be a real world natural surface, the illuminated

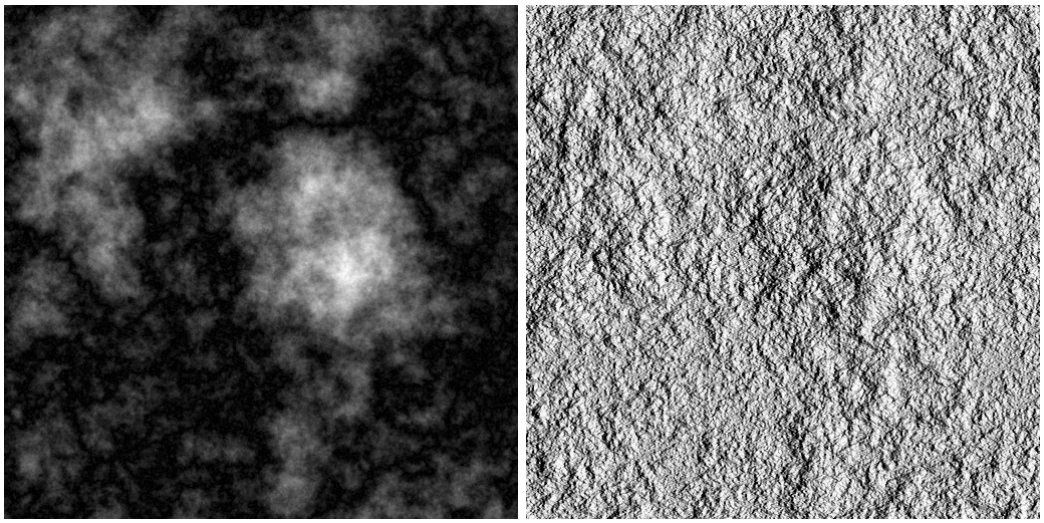


Figure 3.2: height map (left) and illuminated surface (right) of the fractal surface model

surface looks like a real surface such as rocks or stones.

3.2.2 Mulvaney

This section introduces the Mulvaney surface which models a machined surface. The fractal surface in the previous chapter models a natural surface. It is not only rough but also uneven because it has most of its energy in lower frequencies. This is volitional for natural surfaces but not for machined surfaces which are rough but even. Thus a model for machined surfaces has to suppress lower frequencies compared to the fractal model. Because the model is also isotropic, the power spectrum is again only dependent on the centre frequency ω. The PSD for the Mulvaney model is

$$S(\omega) = k_1 \left(k_2\omega^2 + 1\right)^{-3/2}. \tag{3.2}$$

Figure 3.1 on the preceding page shows the PSD of the Mulvaney model with the cut-off frequency $k_2 = 0.32$. For low frequencies it behaves like white noise (the PSD is a constant). For high frequencies it behaves like a fractal with a roll-off factor of 3. Figure 3.3 on the next page shows the height map and an illuminated image of the Mulvaney model. Because of the suppressed low frequencies it is flat compared to the fractal model.

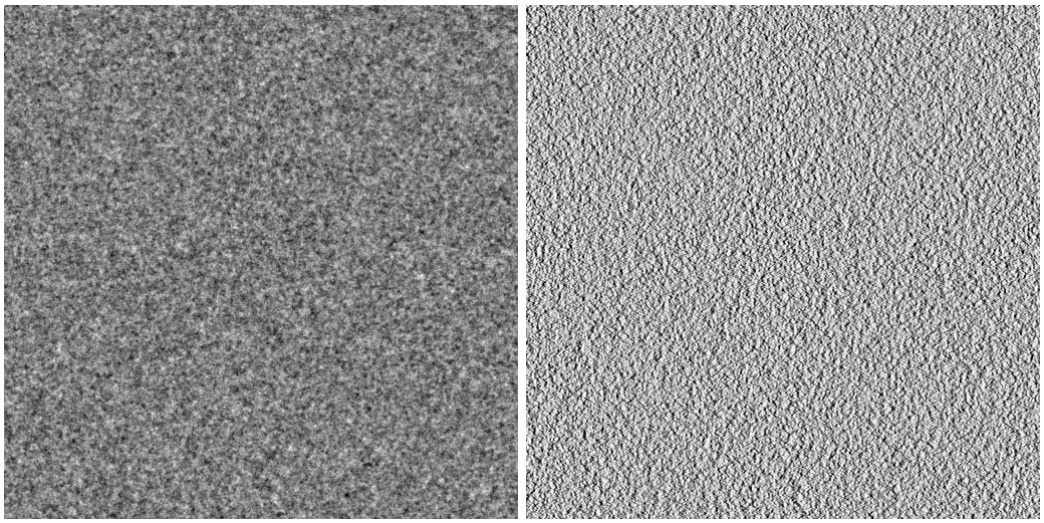


Figure 3.3: height map (left) and illuminated surface (right) of the Mulvaney surface model

3.2.3 Ogilvy

The previous two models are isotropic, i.e. their features do not vary with the direction. But many real world surfaces are directional. Most of these directional real world surfaces have a higher image energy in one direction which makes them directional. The Ogilvy surface models this directionality by using different cut-off frequencies for the u and v direction:

$$S(u,v) = k_1 \frac{1}{1/\lambda_1^2 + u^2} \frac{1}{1/\lambda_2^2 + v^2} \tag{3.3}$$

Above these cut-off frequencies the Ogilvy surface behaves like a fractal with a constant roll-off factor. Figure 3.4 on the following page shows an illuminated image and the PSD with different cut-off frequencies. For highly directional surfaces (left part) it is a proper model for real world surfaces. The main disadvantage can be seen on the right part of the figure: If the cut-off frequencies are similar in both directions the surface does not become a isotropic surface but a two-directional surface (the image energy is high in two directions). Thus the Ogilvy surface is not a proper model for a smooth transition between isotropic and directional surfaces. For this reason we use this model only as a highly directional surface.

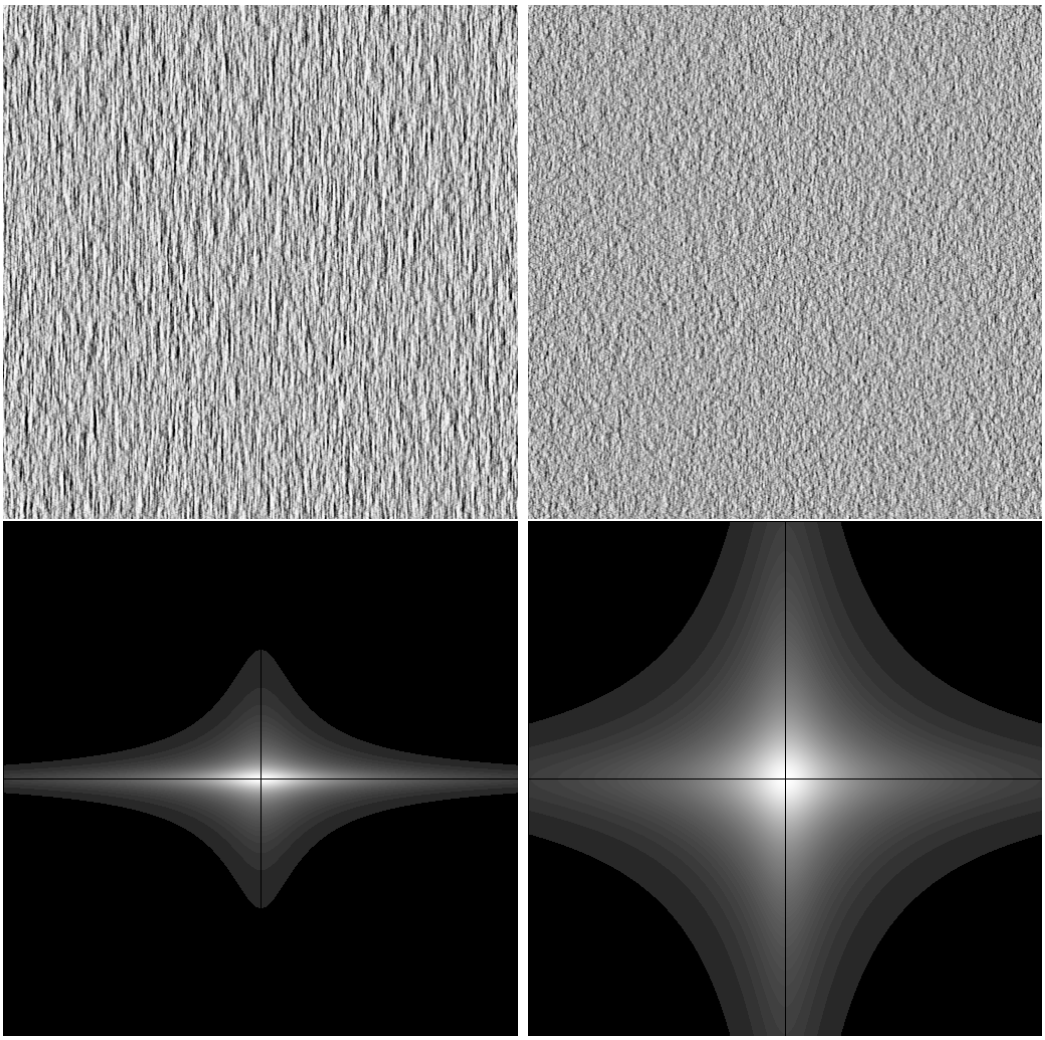


Figure 3.4: illuminated image and PSD of a Mulvaney surface with the parameters $\lambda_1 = 1/16$ $\lambda_2 = 1/64$ (left) and $\lambda_1 = 1/16$ $\lambda_2 = 1/16$ (right)

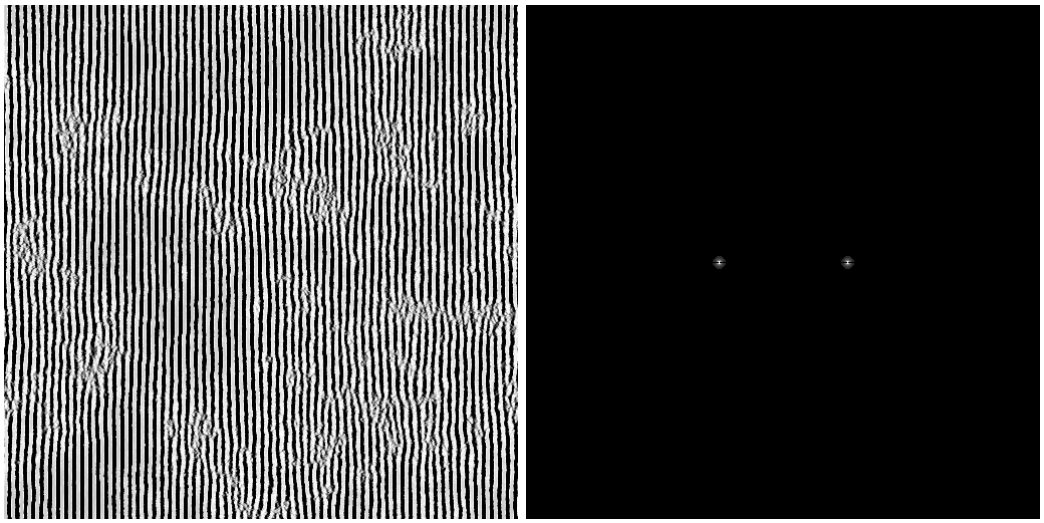


Figure 3.5: illuminated surface and PSD of the synthetic surface *sand ripples*

3.2.4 Sand Ripples

The last model introduced in this chapter is *sand ripples* which is an extreme case of a highly isotropic surface with nearly all image energy in one centre frequency. The PSD consists of two Dirac impulses:

$$S(u,v) = \delta(u - k_1, v - k_2) + \delta(u + k_1, v + k_2)$$

Thus the surface is approximately a sine wave. Figure 3.5 shows the illuminated surface and the PSD of this surface with the parameters $k_1 = 64$ and $k_2 = 0$. All the image energy is around the direction and frequency of the sine wave.

Chapter 4

Texture Feature Generation

4.1 Introduction

In the previous chapter we introduced the theoretical background needed for theoretical and experimental investigation into the effect of illuminant tilt changing on texture features. Because feature generation is a subprocess of the classification problem, this chapter first gives a short introduction into the idea of classification. After that we give a short overview of popular and useful texture feature extraction techniques. Finally we describe the feature generation using linear filters and image energy which is used for this dissertation.

4.2 Classifier

The combination if algorithms by which pixels are classified and the image segmented will be described as the *classifier*. The classifier must be able to detect and identify different textures and the boundaries between them. The whole process can be broken into three subprocesses: measurement, feature extraction and discrimination. Measurement in this case means the measurement of the physical signal. From this measure, features are derived which represent a numerical quality of the surface. Several features which are derived from the same measurement are treated as being orthogonal and grouped together as a feature vector. The feature vector is associated to a pixel and thus is a local process. The final step is to use a discriminant function to label every pixel according to the estimated class. The result is a segmented image where every pixel is labelled and thus belongs to one class.

4.3 Overview of Feature Extraction Techniques

The aim if this section is not to show any technique in detail but show the main groups of features. Feature extraction can be categorised in *model-based*, *structural* and *feature-based methods*.

Model-based features are derived from a surface model. This model has to fit the group of textures which are classified. The classifier computes the parameters of the model for every texture and uses this parameters as features. This model-bases textures can be divided into stochastic and fractal models. Although we do not use these features for classification in this dissertation, we used fractal models in chapter 3 to construct synthetic simulation surfaces.

Another group of texture features are derived using *structural methods*. These techniques assume that textures are composed of well-defined texture primitives. Although this assumption is only valid for a small range of textures, some of these techniques work very well for these group of textures. Because the textures used in this work do not meet this criterion we do not discuss these methods.

The last group we introduce are the widely used *feature-based methods*. This group again can be divided into its main groups statistically based and operator-based features. Statistically based features use histograms or one or second order statistics of the image grey levels. A wide range of these statistics are published, like average intensity, standard deviation, contrast, entropy, angular second moment, etc. . Examples for operator-based features are the convolution with different masks followed by an energy estimation or operators derived from co-occurrence matrices.

Texture features can also be divided into statistical methods and spatial-frequency or spatial/spatial-frequency. It was proofed that in the past statistical methods were superior compared to spatial-frequency methods. This was due to the lack if spatial locality in these early frequency analysis methods. Joint spatial/spatial-frequency methods are based on image representations that indicate the frequency content in localised regions. These methods can achieve a high resolution both in spatial and frequency domain and thus produce much better results.

The feature generation introduced in the next section uses the spatial/-spatial-frequency method. The aim of this work is to investigate the behaviour of texture features produced by this group of features due to illuminant tilt changing.

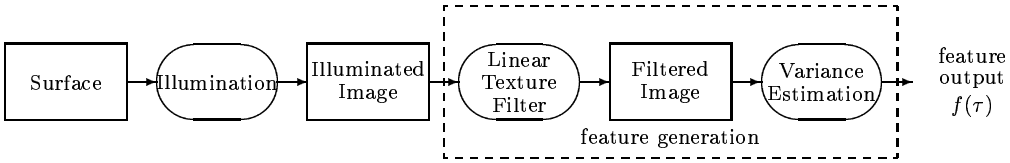


Figure 4.1: Feature generation

4.4 Texture Feature Generation using Linear Filters

As mentioned in the previous section the texture feature generator used in this dissertation uses a spatial/spatial-frequency domain method. Because we want to predict the behaviour of a wide range of texture features, we model this generator as general as possible.

Figure 4.1 shows the general structure of the classifier. Although we want to classify textures, the input of the feature generator is an illuminated image of the texture. The problem is that the image of one surface varies under different light conditions. The feature generator is a combination of a linear filter and a variance estimation. We will derive the theoretical behaviour of texture features without any further assumptions for the linear filter. The variance estimator computes the square (x^2) ,followed by a low-pass filter or averaging function. Sometimes the absolute value ($|x|$) is used as a cheaper approximation of the variance. Although we will only use the square for an analytical prediction of texture features, we will use also the absolute value in our experiments.

Chapter 5

The Behaviour of Texture Features due to Illumination Tilt Changing

5.1 Introduction

In this chapter we derive a analytical prediction for the behaviour of texture features due to tilt rotating. We use the group of features derived by the texture feature generator introduced in section 4.4. We assume that the surfaces reflections of textures used in this work can approximately be described by Kube's linear model (introduced in section 2.5.2). After we have derived a general prediction for the behaviour of texture features we derive further predictions for the features of isotropic and unidirectional textures. Finally we derive a analytical prediction for the behaviour of features in feature space.

5.2 Output of Linear Filters and their Features

In section 2.5.2 we derived a link between the surface and its image in the frequency domain. Using equation 2.8 on page 14 in terms of power spectrum the surface is linked to its image as follows:

$$I(\omega,\theta) = \omega^2 cos^2(\theta - \tau) sin^2(\sigma) H(\omega,\theta)$$

where

(ω,θ) is the polar form of spatial frequency

Figure 5.1: Feature generation with Kube's Linearisation model

(with $\theta = 0°$).
$I(\omega,\theta)$ is the image power spectrum.
$H(\omega,\theta)$ is the surface power spectrum.

The assumptions for this equation where

- Lambertian reflection
- no self or cast shadowing
- no inter-reflection
- small slopes
- orthographic projection.

For the group of classifiers described in section 4.4 the generated features can be described

$$f(\tau) = \mathcal{VAR}(o(x,y))$$

where $o(x,y)$ is the output of the linear texture filter. Because we treat the illumination process as a linear operation we can exchange the illumination process and the texture filter (5.1).

If the average of $o(x,y)$ is zero and $O(\omega,\theta)$ is the power spectrum of $o(x,y)$ then

$$\begin{aligned} f(\tau) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} O(\omega,\theta)\,dx\,dy \\ &= \int_{0}^{\infty}\int_{0}^{2\pi}\left|\frac{\partial(u,v)}{\partial(\omega,\theta)}\right| O(\omega,\theta)\,d\theta\,d\omega \\ &= \int_{0}^{\infty}\int_{0}^{2\pi}\omega\,O(\omega,\theta)\,d\theta\,d\omega \\ &= \int_{0}^{\infty}\omega^3 sin^2(\sigma)\int_{0}^{2\pi} cos^2(\theta-\tau)A(\omega,\theta)\,d\theta\,d\omega \end{aligned} \tag{5.1}$$

where $A(\omega, \theta)$ is the notional power spectrum of the output of the linear texture filter applied directly to the surface height function.

Using $cos^2(x) = 1/2\,(1 + cos(2x))$ and $cos(x - y) = cos(x)cos(y) + sin(x)sin(y)$ we get

$$\begin{aligned} f(\tau) &= \int_0^{\infty} \omega^3 sin^2(\sigma) \int_0^{2\pi} 1/2\,[1 + cos(2\theta)cos(2\tau) \\ &\quad + sin(2\theta)sin(2\tau)]A(\omega, \theta)\,d\theta\,d\omega \\ &= a + b\cos(2\tau) + c\sin(2\tau) \\ &= a + d\cos(2\tau + \phi) \end{aligned} \tag{5.2}$$

where

$$\begin{aligned} a &= 1/2\,sin^2(\sigma) \int_0^{\infty} \omega^3 \int_0^{2\pi} A(\omega, \theta)\,d\theta\,d\omega \\ b &= 1/2\,sin^2(\sigma) \int_0^{\infty} \omega^3 \int_0^{2\pi} cos(2\theta)A(\omega, \theta)\,d\theta\,d\omega \\ c &= 1/2\,sin^2(\sigma) \int_0^{\infty} \omega^3 \int_0^{2\pi} sin(2\theta)A(\omega, \theta)\,d\theta\,d\omega \end{aligned}$$

$$d = \sqrt{b^2 + c^2}, \qquad \phi = \arctan(c/b)$$

The parameters a, d and ϕ of equation 5.2 are independent of the tilt angle τ. Thus this equation predicts a sinusoidal behaviour with a period of 2τ for the features generated by this group of texture classifiers.

5.3 Behaviour of Isotropic, Directional and Unidirectional Surfaces

In this section we examine equations 5.1 and 5.2 more closely for the special cases of investigating an isotropic, a directional or a unidirectional surface. Equation 5.1 on the preceding page can also be written as

$$f(\tau) = \int_0^{\infty} \int_0^{2\pi} \omega\, H(\omega, \theta)\, K(\omega, \theta)\, F(\omega, \theta)\,d\theta\,d\omega \tag{5.3}$$

where

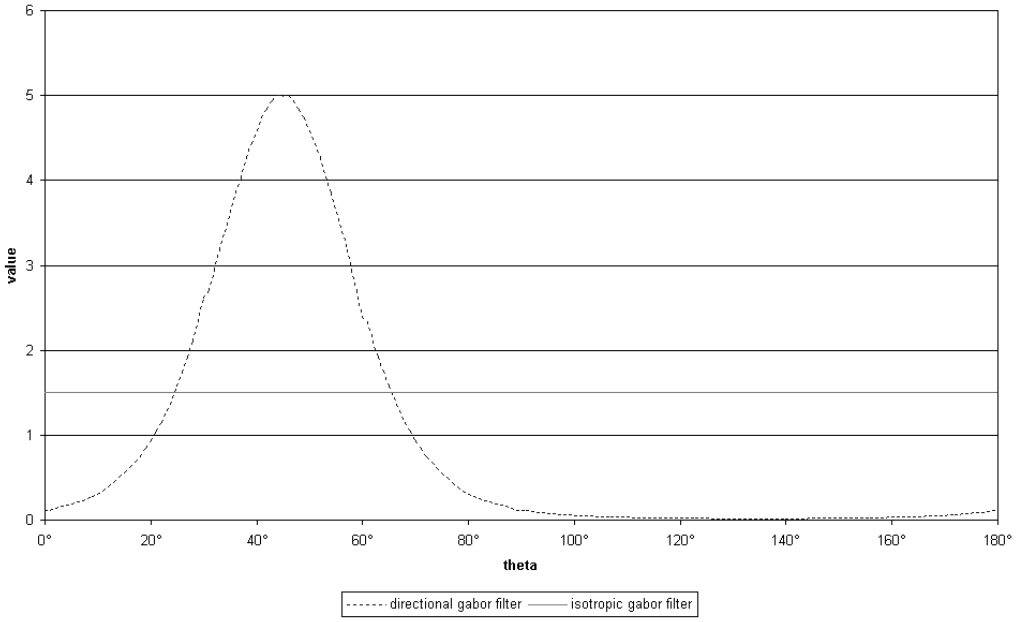


Figure 5.2: angular plot of an isotropic and a directional gabor filter

$H(\omega, \theta)$ is the surface PSD
$K(\omega, \theta)$ is Kube's model in terms of the PSD:
$K(\omega, \theta) = \omega^2 \, sin^2(\sigma) \, cos^2(\theta - \tau)$
$F(\omega, \theta)$ is the PSD of the texture filter.

Thus $f(\tau)$ can be imagined as the area of the angular plot after multiplying $H(\omega, \theta)$, $K(\omega, \theta)$ and $F(\omega, \theta)$. Figure 5.2 shows the angular plot of an isotropic and a directional Gabor filter as examples of an isotropic and directional filter. We define the maximum of the angular plot of a directional filter is at the filter direction α. A variation of α causes a shift of the figure according to the x-axis. Figure 5.3 on the following page shows the angular plot of Kube's model. The maximum is at $\theta = \tau$. A variation of the illumination tilt angle τ causes a shift of the figure according to the x-axis. Figure 5.4 on the next page shows the angular plot of an isotropic, a directional and an unidirectional surface. we define the maximum of a directional and unidirectional surface as the surface direction β.

5.3.1 Using Directional Filters

First we investigate the behaviour of $f(\tau)$ using identical filters except a different filter direction α. If the surface is isotropic, the area of the angular plot and thus $f(\tau)$ becomes maximal if $\tau = \alpha$. $f(\tau)$ becomes minimal if

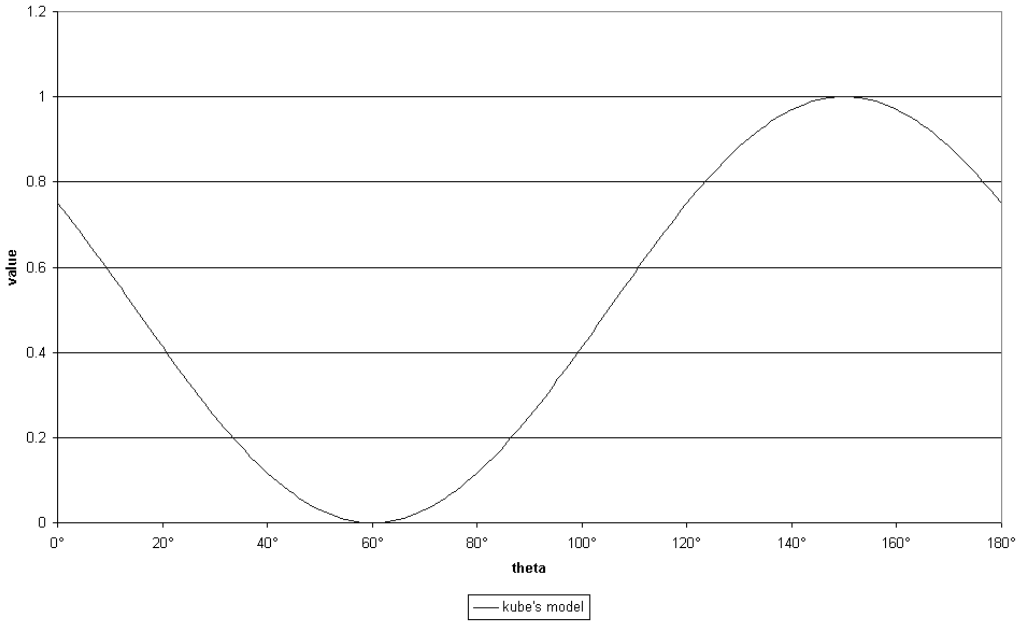


Figure 5.3: angular plot of Kube's model

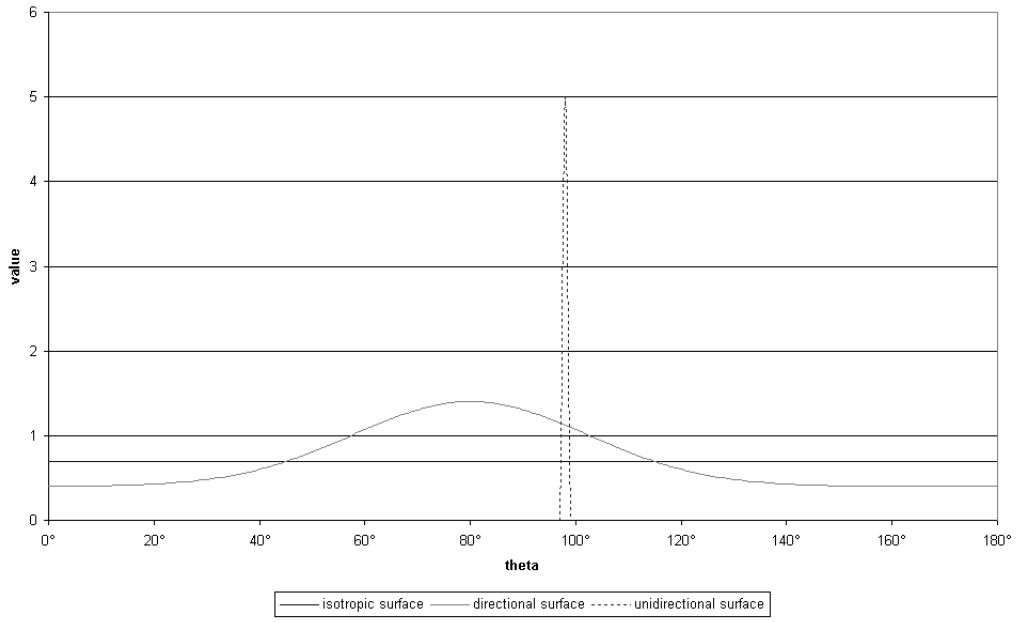


Figure 5.4: angular plot of an isotropic, a directional and an unidirectional surface

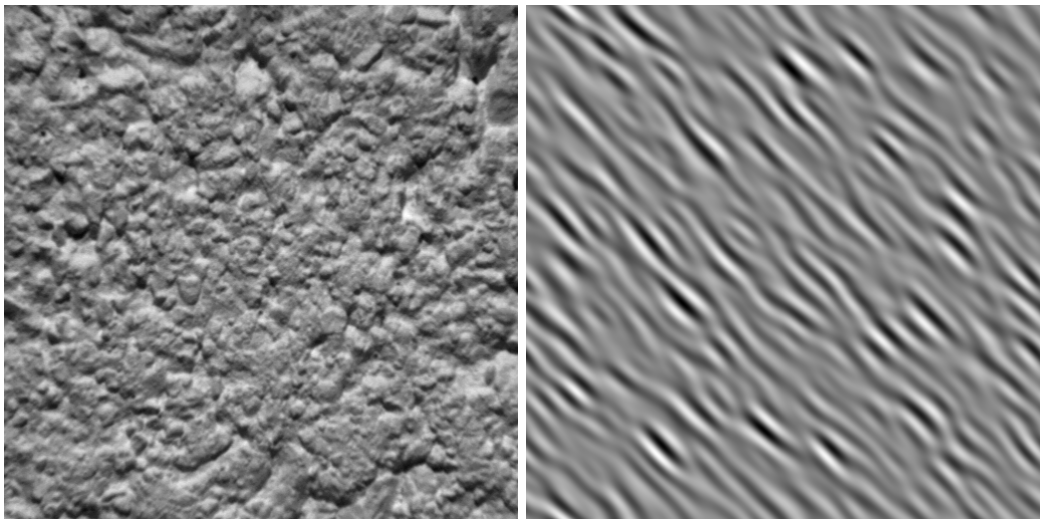


Figure 5.5: *slab45* before and after filtering with a F25A45 Gabor filter

$\tau - \alpha = 90°$. Because the value if the minimum and maximum is independent of the filter phase α, the parameters a and d are constant for every α. Thus we expect for the special case of an *isotropic surface*

$$f_{iso}(\tau, \alpha) = a + d\cos[2(\tau - \alpha)]. \tag{5.4}$$

This means that the phase of the oscillation of f_{iso} is always twice the direction of the filter.

If the surface is unidirectional, the area of the angular plot and thus $f(\tau)$ is maximal if $\tau = \beta$. The parameters a and d are dependent on the filter direction α and are maximal if $\alpha = \beta$. Thus the feature output of an *unidirectional surface* is

$$f_{uni}(\tau, \alpha, \beta) = a(\alpha) + d(\alpha)\cos[2(\tau - \beta)]. \tag{5.5}$$

This means that the phase of the oscillation of f_{iso} is always twice the direction of the surface.

Isotropic and unidirectional surfaces are the marginal cases of all possible surfaces. All other surfaces behave like a mixture of these to cases. There are no further simplifications for these *directional surfaces*:

$$f_{direct}(\tau, \alpha, \beta) = a(\alpha, \beta) + d(\alpha, \beta)\cos[2\tau + \phi(\alpha, \beta)] \tag{5.6}$$

The same formulas can also be derived in a more intuitive way. Figure 5.5 shows the surface **slab45** before and after filtering with a directional F25A45 Gabor filter.

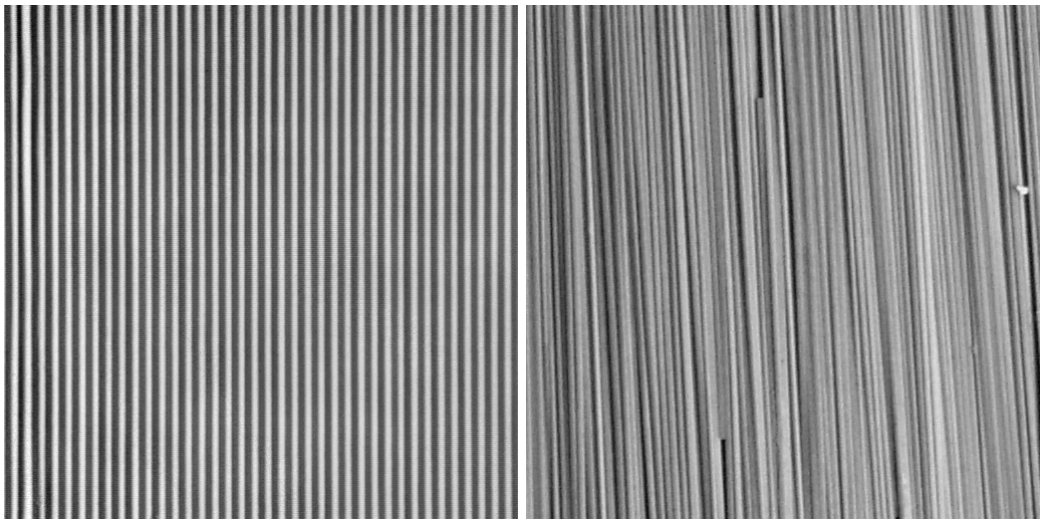


Figure 5.6: unidirectional surfaces: *card1* and *michael3*

It is obvious, that only frequencies similar to the filter centre frequency and direction can pass the filter. If the angle between illumination tilt and filter frequency $\tau - \alpha$ is small, these surface frequencies produce a high image contrast and with it a high image energy behind the filter. If the surface is isotropic the maximum of $f(\tau)$ is at $\tau = \alpha$. If $\tau - \alpha$ is similar to 90° the energy of the output image is small, because the surface frequencies which pass the filter are not illuminated. If the surface is isotropic, the minimum of $f(\tau)$ is at $\tau - \alpha = 90°$. The resulting image for a constant $\tau - \alpha$ is always the same except a rotation of the whole image. Thus the image energy is independent of $\tau - \alpha$. It follows that the parameters a and d of equation 5.2 on page 26 are independent of α which leads to formula 5.4 on the preceding page.

Figure 5.6 shows two approximately unidirectional surfaces. These surfaces contain only one direction of waves which can be illuminated. $f(\tau)$ is maximal if τ equals the surface direction β because in this case the illuminated surface waves produce a maximal image energy. Although the location of the maximum is independent of the filter direction α the value of the maximum varies with α. If $\alpha = \beta$ the energy which can pass the filter is maximal. This leads to formula 5.5 on the preceding page.

5.3.2 Using Isotropic Filters

Isotropic filters are used quite often in pattern recognition although the information of direction of the image is lost. The reason becomes obvious by

looking at equation 5.3 on page 26. If the used object is a 2-D object instead of a rough 3-D object, Kube's model is not applied. A rotation of the object results a shift according to the x-axis of its angular plot. But using an isotropic gabor filter this has no effect on the area of the resulting angular plot. Thus the features of all kind of 2-D objects (isotropic, directional and unidirectional) are rotation invariant.

But this is not true for rough 3-D surfaces. Because Kube's model is applied to the surface, the texture features are neither rotation invariant nor illumination tilt invariant. Only in the special case of an isotropic surface the features are both rotation and tilt invariant. This is because Kube's model is multiplied with both a constant because of the surface angular plot and a constant because of the isotropic gabor filter. In this case the resulting area of the angular plot does not change due to rotation or illumination tilt changes.

5.4 Behaviour in a Multi-Dimensional Feature Space

To classify or segment an image normally many features are generated from one image (see also section 4.2). In the case that features are generated using linear filters these different features are generated using the same group of filters but with different centre frequencies and directions. Because the classifier makes its decisions in this multi-dimensional feature space it is important to investigate the behaviour of texture features due to tilt changing in feature space.

In the general one-dimensional case, the sinusoidal behaviour of two different surfaces under varying illumination tilt can vary in the mean value, the oscillation amplitude and the phase:

$$f_1(\tau) = a_1 + b_1 \cos(2\tau + \phi_1)$$

$$f_2(\tau) = a_2 + b_2 \cos(2\tau + \phi_2)$$

If $b_1 + b_2 > |a_1 - a_2|$ then there is no boundary to distinguish between the two surfaces. In this case an illumination tilt independent classification or segmentation is not possible.

If two different features are derived from the same image, the results can be plotted in two dimensional feature space by plotting one feature against the x-axis and one feature against the y-axis. Using our sinusoidal prediction we get the general behaviour

$$x = f_1(\tau) = a_1 + b_1 \cos(2\tau + \phi_1)$$

$$y = f_2(\tau) = a_2 + b_2 \cos(2\tau + \phi_2)$$

If the surface is isotropic and the two filters a identical except a difference in direction of 90°, the mean value and the oscillation amplitude of the two features are the same and the phase difference becomes 180°:

$$x = f_1(\tau) = a + b \cos(2\tau) \tag{5.7}$$
$$y = f_2(\tau) = a + b \cos(2\tau + 180°) = a - b \cos(2\tau) \tag{5.8}$$

The range of values for both x and y are $a - b$ to $a + b$. By substituting equation 5.7 in 5.8 we get

$$y = 2a - x$$

This is a line which is symmetric to the bisector of the x- and y-axis end goes from the point $(a - b; a + b)$ to the point $(a + b; a - b)$. Thus we predict that the scatter plot for isotropic textures and two identical but orthogonal filters is a line.

If the surface is isotropic and the two filters a identical except a difference in direction of 45°, the mean value and the oscillation amplitude of the two features are the same and the phase difference becomes 90°:

$$x = f_1(\tau) = a + b \cos(2\tau) \tag{5.9}$$
$$y = f_2(\tau) = a + b \cos(2\tau + 90°) = a - b \sin(2\tau) \tag{5.10}$$

The range of values for both x and y again are $a - b$ to $a + b$. By substituting equation 5.9 in 5.10 we get

$$y = a - b \sin\left[\arccos\left(\frac{x - a}{b}\right)\right]$$

Using $\sin[\arccos(x)] = \sqrt{1 - x^2}$ we get

$$\sqrt{1 - \left(\frac{x - a}{b}\right)^2} = \frac{a - y}{b}$$

$$\Longleftrightarrow\ b^2 = (x - a)^2 + (y - a)^2$$

This is the equation of a circle with the radius b and the centre point $(a; a)$. Thus we predict that the scatter plot for isotropic textures and two identical filters except a difference in direction of 45° is a circle which is symmetric to the bisector of the x- and y-axis.

The line and the circle are the two marginal cases of all possible curves. In the general case of a directional surface and two different filters the curve is an ellipse. The proof is shown in appendix E.

Because the parameters of the ellipses can have any value, the ellipses can overlap. In this case there is no boundary to distinguish between the two surfaces. An illumination tilt independent classification or segmentation is not possible.

5.5 Summary

In this section we derived a prediction of the behaviour of texture features due to tilt variation. It predicts a sinusoidal behaviour of this features. The parameters of this equation $f(\tau) = a + d\,cos(2\tau + \phi)$ are dependent on the height map of the surface and thus can not be calculated from images. Although there is no further simplification for the general case we can make more predictions for special cases. In the case of applying filters, which are identical except the filter direction α, $f(\tau)$ behaves according to equations 5.4, 5.5 and 5.6 on page 29. Applying an isotropic filter we predict that $f(\tau)$ is a constant for isotropic surfaces. Finally we predicted that the resulting figure in the two-dimensional feature space is an ellipse.

Chapter 6

Simulations and Experiments

6.1 Introduction

In the previous chapter we derived the behaviour of texture features due to tilt angle changing in theory. Particularly we predicted a sinusoidal behaviour of texture features. Now we verify these predictions using simulations and experiments. This chapter consists of two parts: First we use simulations to compare features derived from synthetic surfaces with features derived from illuminated images. Then we use real world textures to compare our predictions with experimental results.

6.2 Simulations

We use a simulation as a first step to test our analytical results. Our prediction for the behaviour of texture features was as follows:

$$\begin{aligned} f(\tau) &= a + b\cos(2\tau) + c\sin(2\tau) \\ &= a + d\cos(2\tau + \phi) \end{aligned}$$

$$\begin{aligned} a &= 1/2\, sin^2(\sigma) \int_0^\infty \omega^3 \int_0^{2\pi} A(\omega,\theta)\, d\theta\, d\omega \\ b &= 1/2\, sin^2(\sigma) \int_0^\infty \omega^3 \int_0^{2\pi} cos(2\theta) A(\omega,\theta)\, d\theta\, d\omega \end{aligned}$$

$$c = 1/2\, sin^2(\sigma) \int_0^{\infty} \omega^3 \int_0^{2\pi} sin(2\theta) A(\omega, \theta)\, d\theta\, d\omega$$

$$d = \sqrt{b^2 + c^2}, \qquad \phi = \arctan(c/b)$$

The parameters a, b and c are only dependent on $A(\omega, \theta)$ which is the power spectrum of the surface multiplied with the power spectrum of the used linear filter. For a explicit computation of the parameters we need the surface of the texture. Using real world textures we can only get illuminated images of the textures instead of the intrinsic surfaces. For this reason we use the synthetic generated surfaces presented in chapter 3.

For this simulation we have to transform these equations from ω, θ-coordinates to u, v-coordinates. Using the Jacobian determinant $\frac{\partial(\omega,\theta)}{\partial(u,v)} = \frac{1}{\sqrt{u^2+v^2}}$ and $\sin(2x) = 2\sin(x)\cos(x)$, $cos(2x) = 2\cos^2(x) - 1$, $u = \omega\cos(\theta)$ and $v = \omega\sin(\theta)$ we get

$$\begin{aligned} a &= 1/2\, \sin(\sigma)^2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (u^2 + v^2)\, A(u, v)\, du\, dv \\ b &= 1/2\, \sin(\sigma)^2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (u^2 - v^2)\, A(u, v)\, du\, dv \\ c &= 1/2\, \sin(\sigma)^2 \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} 2\, u\, v\, A(u, v)\, du\, dv \end{aligned}$$

The simulation compares the texture features derived from the surface (using the sinusoidal prediction) with the features computed from illuminated images. The simulation setup is shown in figure 6.1 on the next page. First the surfaces of the four different synthetic surfaces introduced are computed. From this surfaces illuminated images are rendered using the Lambertian reflection (equation 2.3 on page 12). The illumination slant angle is 45° for every image; the tilt angle goes from 0° to 180° with 10° steps. These images are filtered with four different even Gabor filters. The centre frequency of the filters is always 25 circles per image; the directions are 0°, 45°, 90° and 135°. The images are filtered with a real Gabor filter, which means a multiplication with a real and even filter in frequency domain. In addition the features are computed from the PSD of the surfaces and the same Gabor filters using the sinusoidal prediction.

Figures 6.2 to 6.5 show the results of this simulation. They show that the prediction for the texture features fits the real behaviour of these features. Using the surface *sand ripples* only the 0°-filtered images have measurable energy. This happens because this surface is highly directional with the direction 0°. All other filters suppress nearly all the image energy.

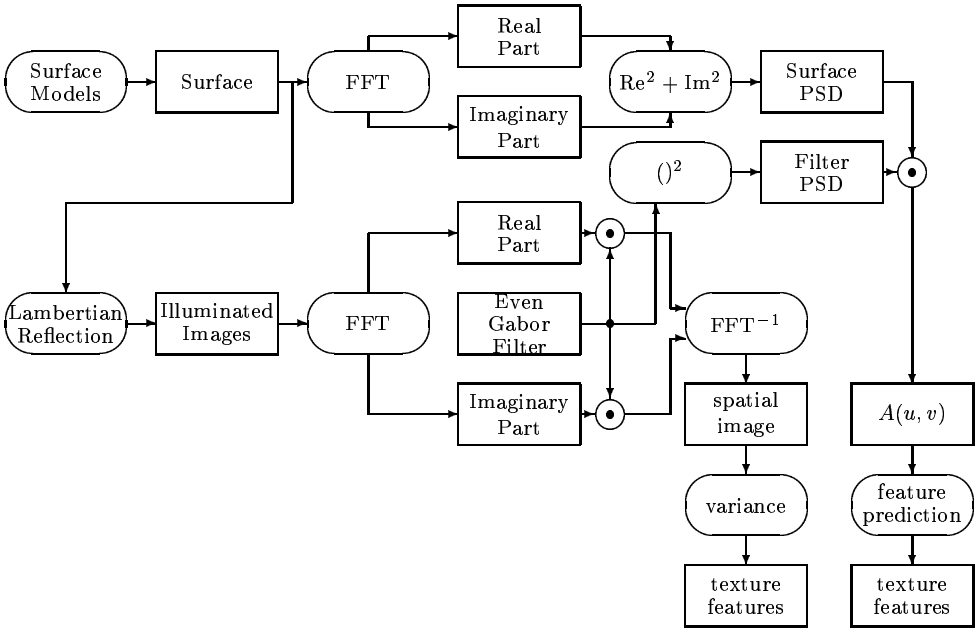


Figure 6.1: Simulation Setup

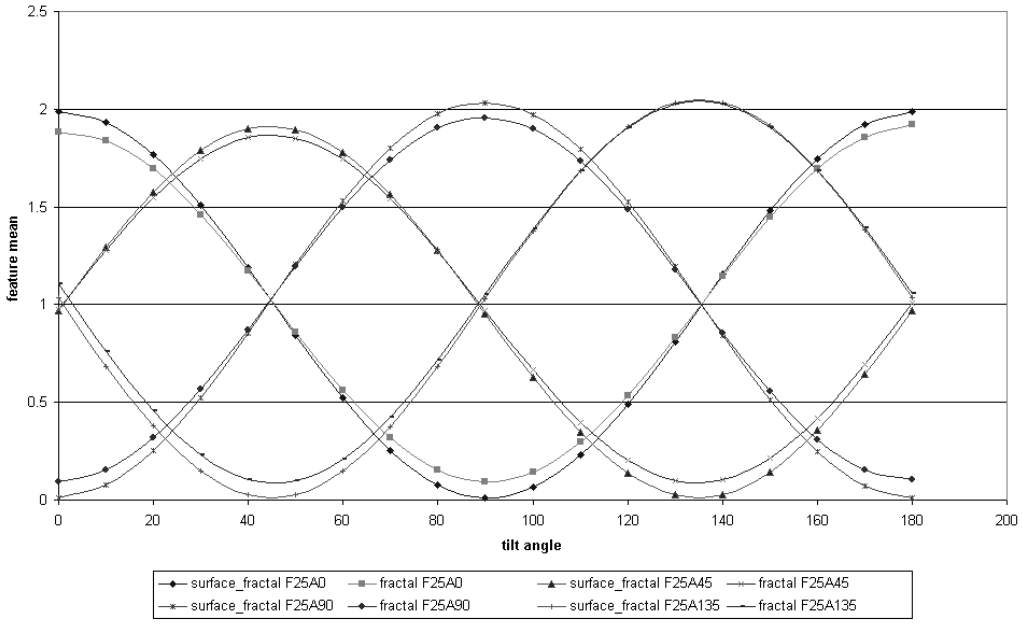


Figure 6.2: feature means of a filtered fractal derived from the surface and illuminated images

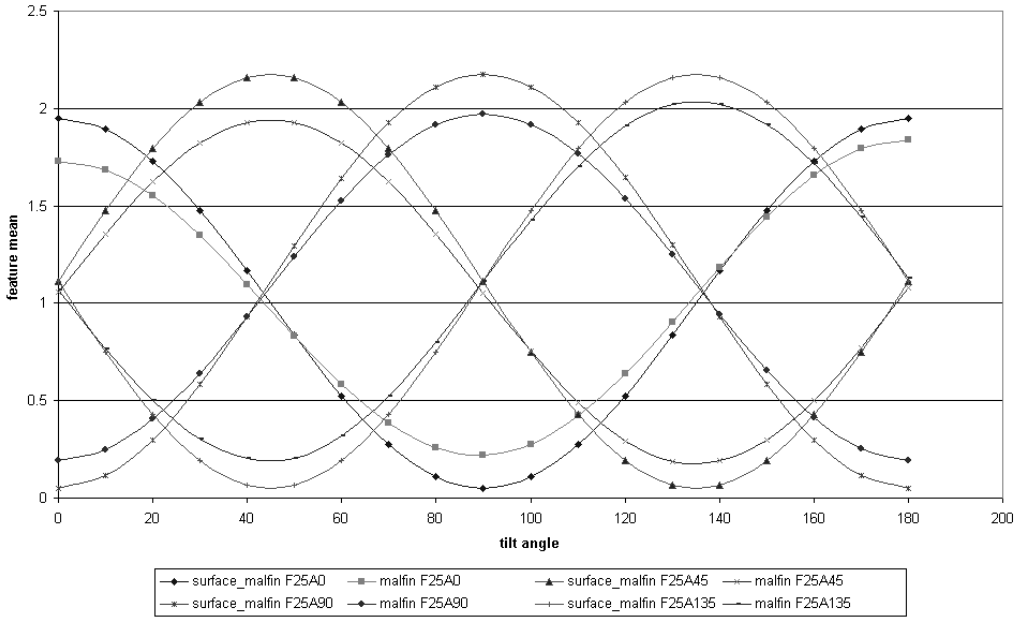


Figure 6.3: feature means of a filtered Mulvaney surface derived from the surface and illuminated images

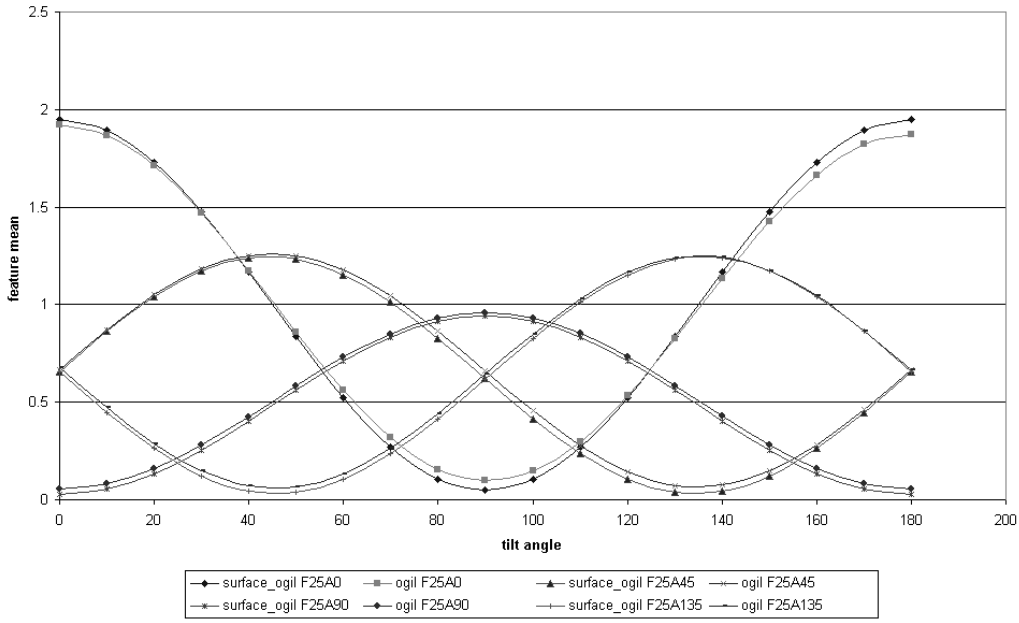


Figure 6.4: feature means of a filtered Ogilvy surface derived from the surface and illuminated images

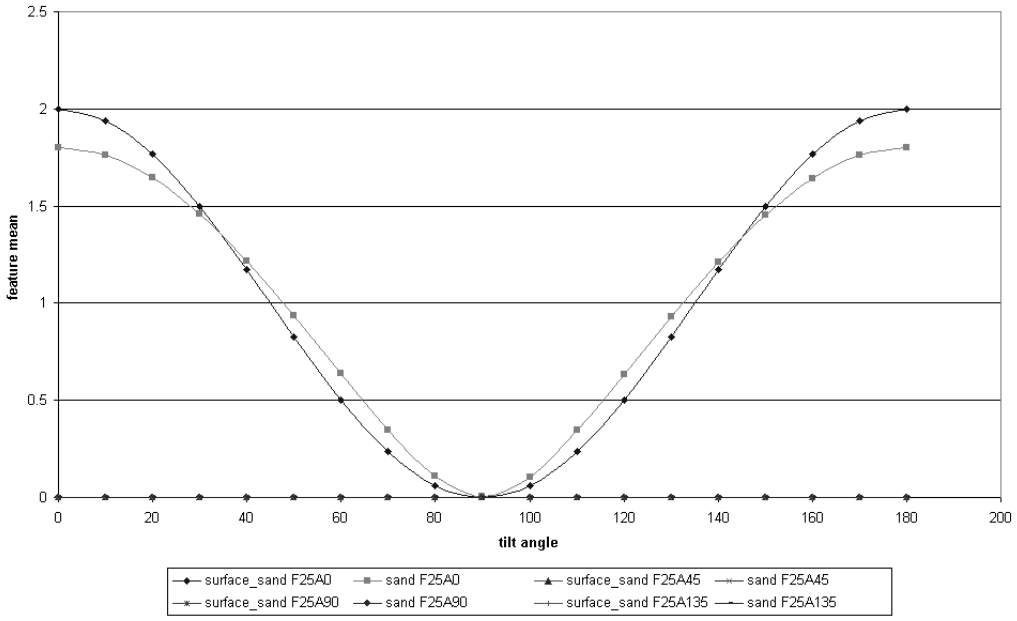


Figure 6.5: feature means of the filtered *sand ripples* surface derived from the surface and illuminated images

6.3 Experiments

6.3.1 Introduction

The experiments below are implemented using thirty different surfaces, each imaged with a illumination tilt angle between 0° and 180°. The difference of the illumination tilt angle of two consecutive images is 10° except the data sets *and1* to *and7*, which are imaged in 15° steps. All surfaces are illuminated at a slant angle of 45°, in addition some surfaces are also illuminated with a slant angle of 60°. One image of every surface can be found in appendix B on page 71.

To verify if the results correspond with our theoretical prediction $f(\tau) = a + d\cos(2\tau + \phi)$ a best fit curve for every data set output is calculated. These best fit curves $f_b(\tau) = a_b + b_b\cos(2\tau) + c_b\sin(2\tau)$ represent these functions which satisfy the theoretical prediction and are closest to the measured curves. One way to calculate the parameters a_b, b_b and c_b is to use an iterative nonlinear optimisation. A much easier non iterative way which produces also good results is as follows: The parameter a_b is calculated as the average value of one period. The parameters b_b and c_b are calculated as the amplitude of the fundamental oscillation using a *Discrete Fourier Transform*. The resulting formulae are as follows:

$$f_b(\tau) = a_b + b_b\cos(2\tau) + c_b\sin(2\tau) \tag{6.1}$$

$$a_b = \frac{1}{N}\sum_{k=0}^{N-1} f_k \tag{6.2}$$

$$b_b = \frac{2}{N}\sum_{k=0}^{N-1} (f_k - a_b)\cos(2\pi k\tau/N) \tag{6.3}$$

$$c_b = \frac{2}{N}\sum_{k=0}^{N-1} (f_k - a_b)\sin(2\pi k\tau/N) \tag{6.4}$$

N : number of measure points of one period

The average value is subtracted from every value before calculating the *DFT* to get a better accuracy.

To specify how close the measured curve is to the best fit curve we need a *measure of error*. An intuitive way to calculate an *absolute measure of error* M, is to sum the squared differences between the measured and best fit points:

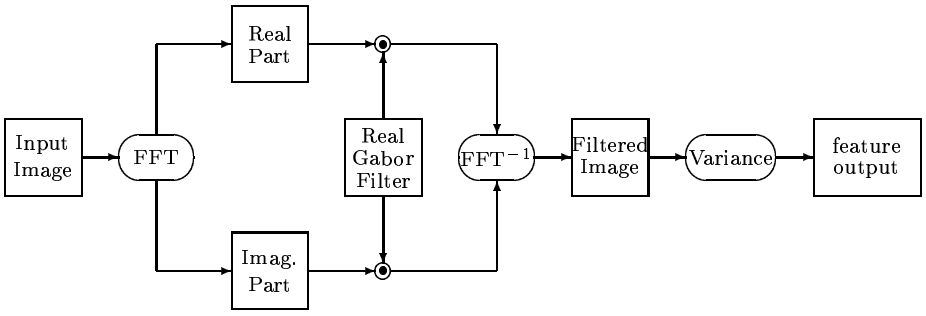


Figure 6.6: Feature generation using real Gabor filters

$$M = \frac{1}{N}\sqrt{\sum_{k=0}^{N-1}(f_k - f_{b_k})^2} \tag{6.5}$$

To calculate the *relative measure of error* we divide the absolute measure of error with the amplitude of the fundamental oscillation. Thus the relative measure of merit is calculated as follows:

$$m = \frac{M}{\sqrt{b_b^2 + c_b^2}} = \frac{1}{N}\sqrt{\frac{\sum_{k=0}^{N-1}(f_k - f_{b_k})^2}{b_b^2 + c_b^2}} \tag{6.6}$$

6.3.2 Gabor filters

Implementation

In the following all directional Gabor filters are specified by the expression $F\Omega A\Theta$, where Ω is the centre frequency in circles per image and Θ is the filter direction.

As our first experiment to verify our analytical predictions, we filter all surfaces with four complex Gabor filters with the parameters F25A0, F25A45, F25A90, F25A135, F50A45 and one real Gabor filter with the parameters F25A45. All filters are represented in the frequency domain in appendix C on page 74. The experiment implementation for real Gabor filters is straight forward (figure 6.6). Because real Gabor filters are real and even in spatial domain they are also real and even in frequency domain. Thus the real and imaginary part of the input image can be multiplied with the filter before the filtered image is transformed back to spatial domain.

Complex Gabor filters have a complex pulse response. This means that the result of a real image filtered with this filter is complex. The following symmetrical relationships are summarised in table 6.1 on the following page.

spatial domain	frequency domain
real	real-even + imaginary-odd
imaginary	real-odd + imaginary-even
real-even + imaginary-odd	real

Table 6.1: symmetrical relationships of the Fourier transform (from [10] page 13)

Because the complex Gabor filter consists only of a real-even and a imaginary-odd part in spatial domain, it consists only of a real-even and a real-odd part in frequency domain. Because the image is real in spatial domain, it consists of a real-even and an imaginary-odd part in frequency domain.

Before the filtered image can be transformed back to spatial domain, it has to be split into the part which is real in spatial domain and the part which is imaginary in spatial domain. These two parts can be transformed back separately. To get the real part in spatial domain, the real-even and the imaginary-odd part in frequency domain are needed. The real-even part in frequency domain is the multiplication of the real-even part of the image with the real-even part of the filter. The imaginary-odd part in frequency domain is the multiplication of the imaginary-odd part of the image with the real-even part of the filter. To get the imaginary part in spatial domain, the real-odd and the imaginary-even part in frequency domain are needed. The real-odd part in frequency domain is the multiplication of the real-even part of the image with the real-odd part of the filter. The imaginary-even part in frequency domain is the multiplication of the imaginary-odd part of the image with the real-odd part of the filter.

Figure 6.7 on the next page shows the implementation of the feature generation. The FFT^{-1} program generates only a real image in spatial domain:

$$F_{real} + jF_{imag} \subset f$$

By multiplying both sides with j the spatial domain becomes imaginary:

$$-F_{imag} + jF_{real} \subset jf$$

Thus the same program can be used to compute the imaginary spatial image, if the negative imaginary part is treated as the real part and the real part is treated as the imaginary part.

Results

Because there are far to many results it is not possible to show all the data set curves. In the following some examples with typical data set outputs are

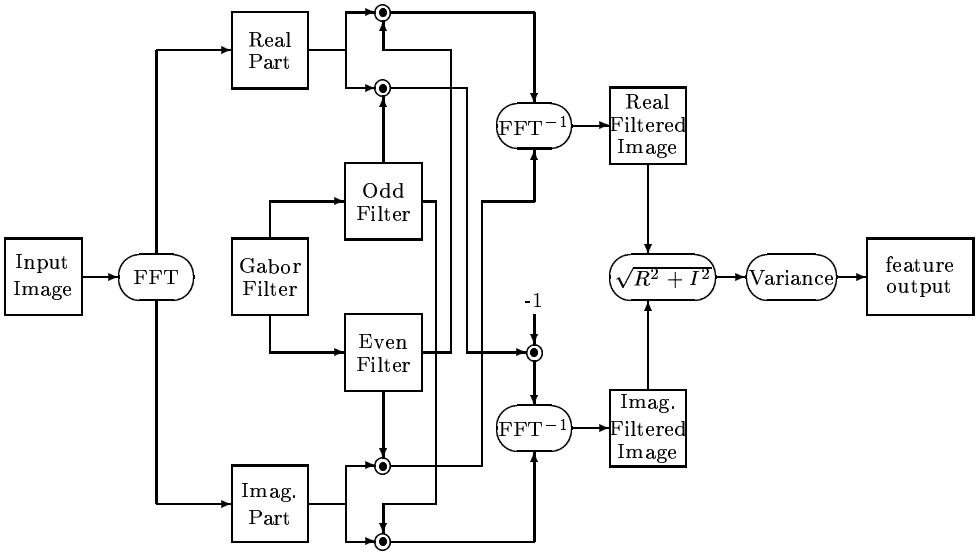


Figure 6.7: Feature generation using complex Gabor filters

shown.

To check if a high relative measure of error corresponds to a bad intuitive fit of the measured curve with the prediction $f(\tau) = a + d\cos(2\tau + \phi)$, the curves with the best, the worst and a mean relative measure of error are compared in figure 6.8 on the following page. It shows that our measure of error is accurate to calculate the sinusoidal fit of a curve.

The complete results are shown on the tables A.1 on page 67 and A.2 on page 68. Figure 6.9 on page 44 shows a graphic representation of the sinusoidal measure of error and the energy of the filtered images of all combinations of surfaces and filters.

The figure shows that nearly every combination of a surface and a filter produces a nearly sinusoidal behaviour of $f(\tau)$. Only if the energy of a filtered image is very low (because the images contains no information in the filter frequencies), does the measure of error become large. For example *card1* is a nearly unidirectional surface and contains only information in the direction 0°. Because of this the F25A90com filtered image contains nearly no energy; its measure of error becomes large. *Stones2* has most of its information in lower frequencies. Thus the high frequency filtered F25A50com image has few energy and a large measure of error.

Figure 6.10 on page 45 shows a comparison between the measure of error of images filtered with real and complex F25A45 Gabor filters. It shows that the behaviour of images filtered with real Gabor filters is as good as

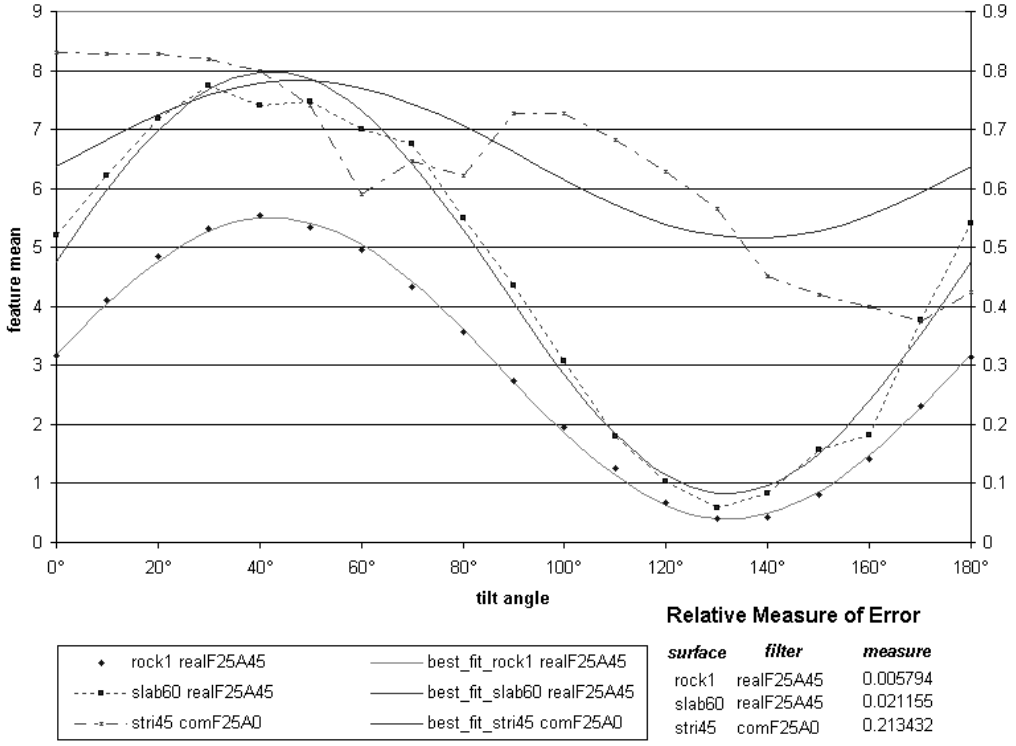


Figure 6.8: Accuracy of the measure of error

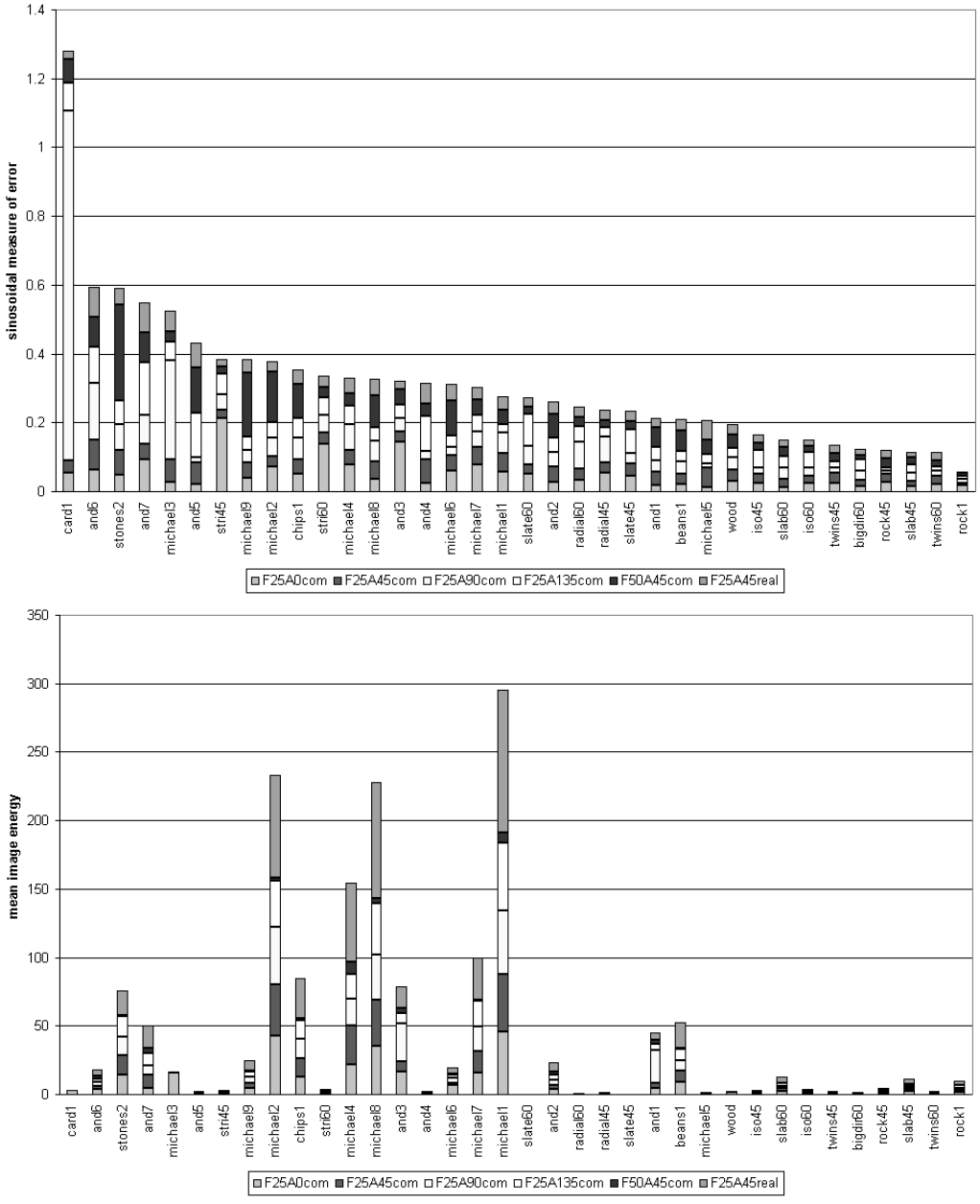


Figure 6.9: relative sinusoidal measure of error and image energy after filtering with Gabor filters

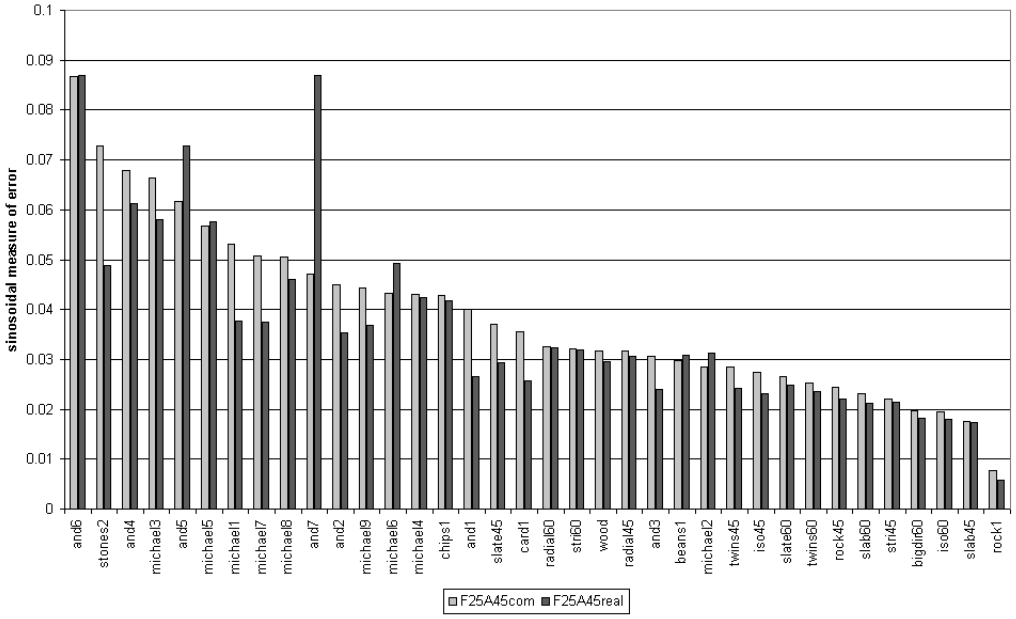


Figure 6.10: comparison of the sinusoidal measure of error between a real and a complex Gabor filter

images filtered with complex Gabor filters. The average measure of error of all F25A45real filters is 0.036424 compared to 0.038977 for F25A45com filters.

6.3.3 Behaviour of Isotropic, Directional and Unidirectional Surfaces

In section 5.3 we made further predictions for the special cases of using filters on isotropic or unidirectional surfaces. Some of the results of the previous experiment can be used to verify these predictions.

Figure 6.11 on the next page shows the feature mean dependent on the tilt angle for *rock1* which is approximately an isotropic surface. In agreement with equation 5.4 on page 29 the maximum of the oscillation is always at the filter direction α. The amplitude of the oscillation and the mean of the curve is approximately equal for every α. Figure 6.12 on the next page shows the feature mean dependent on the tilt angle for *michael3* which is an approximately unidirectional surface with the surface direction $\beta = 0°$. In agreement with equation 5.5 on page 29 the maximum of the feature size dependent on the tilt angle τ is always at $\tau = \beta$ and is independent of the filter angle α. The amplitude of the oscillation is the bigger the smaller the

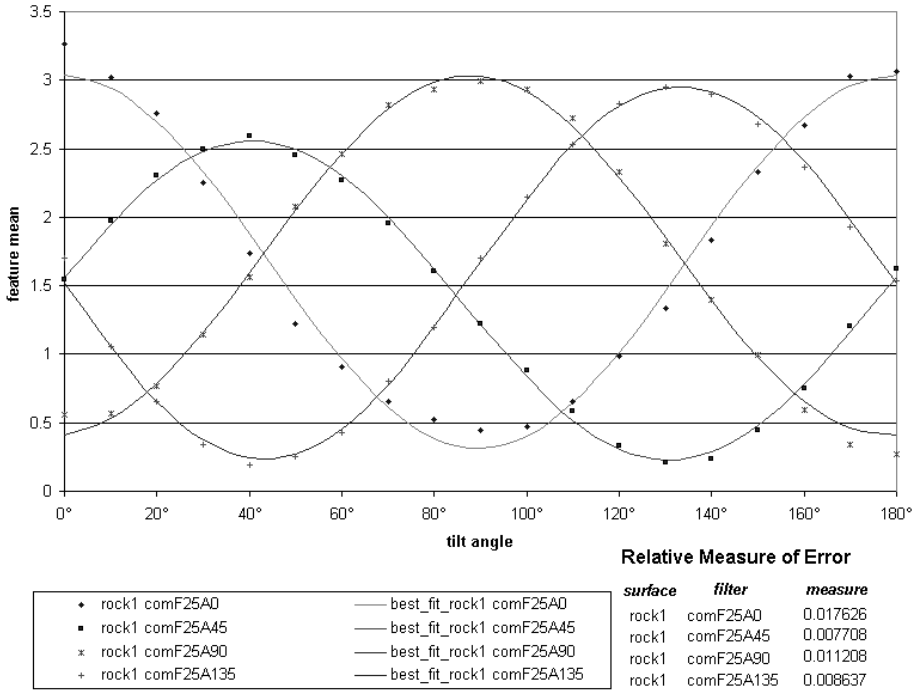


Figure 6.11: feature mean of an isotropic surface exemplified by *rock1*

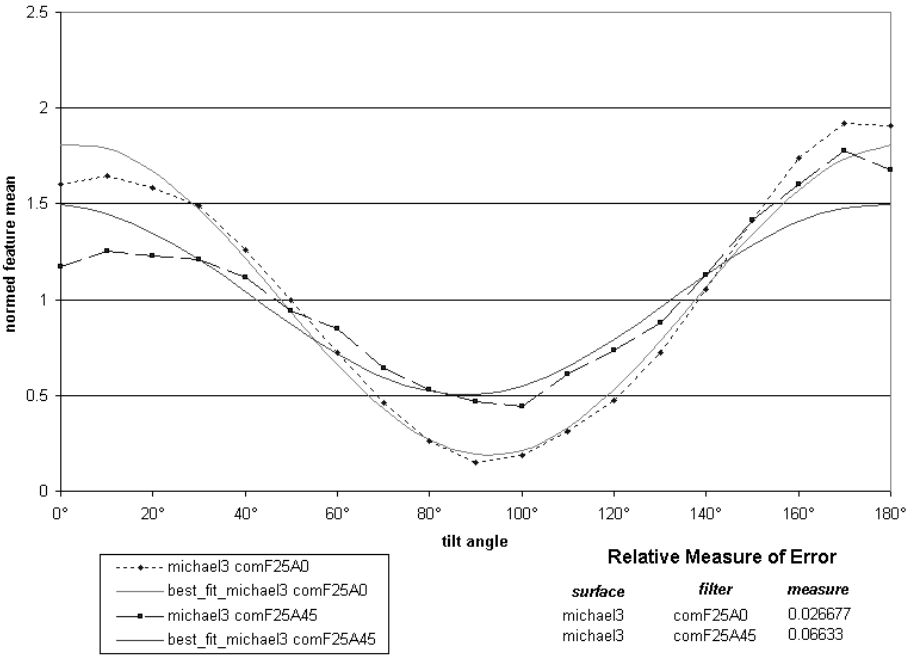


Figure 6.12: feature mean of an unidirectional surface exemplified by *michael3*

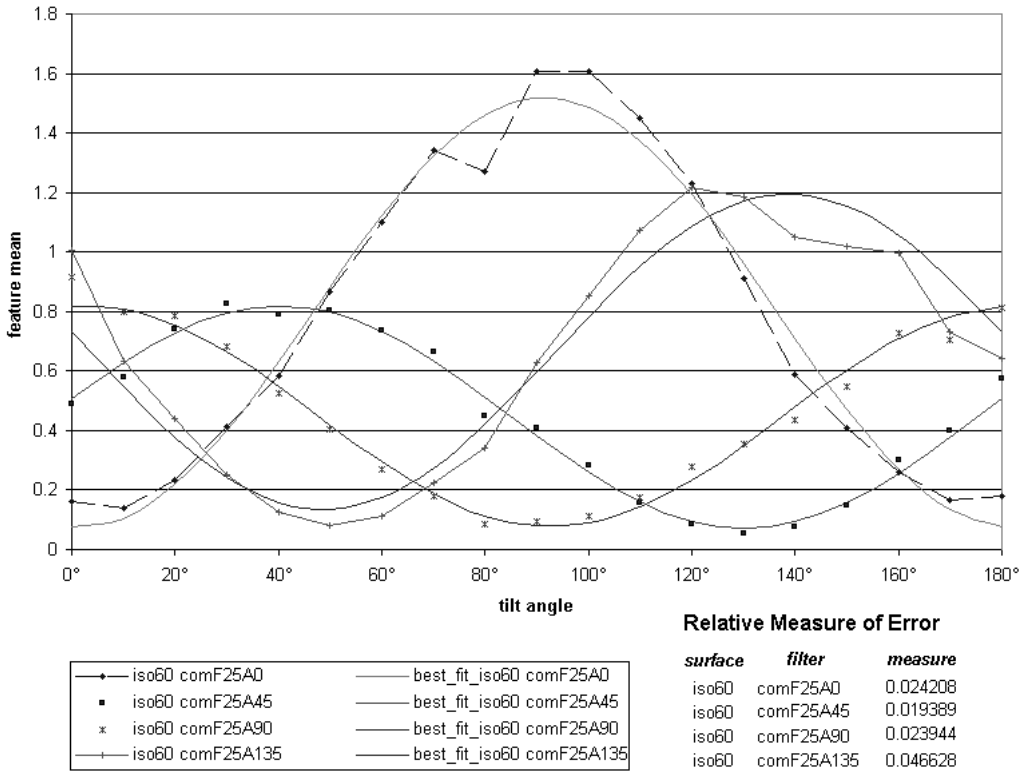


Figure 6.13: feature mean of an directional surface exemplified by *bigdir60*

difference $\tau - \beta$ is. The filters F25A90com F25A135com produce a output of nearly zero because the image has nearly no information in this direction and thus very little energy passes through the filters.

Figure 6.13 shows the feature means of the directional surface *bigdir60*. In general the maximum does not equal α or β. The amplitude of the oscillation varies dependent on α.

In section 5.3.2 we suggested that isotropic filters produce only rotation and tilt independent features in the case of isotropic surfaces. If a feature is tilt independent, $f(\tau)$ would be a straight line. To investigate the behaviour of these surfaces using isotropic filters, the texture feature means using a isotropic F25 Gabor filter are computed. Instead of computing a sinusoidal best fit curve, we now want to know how close the output curve is to a straight line. As a measure of error the relative variances of the features are computed:

$$v = \frac{1}{N} \frac{\sqrt{\sum_{k=0}^{N-1} (f_k - f_{b_k})^2}}{a_b} \tag{6.7}$$

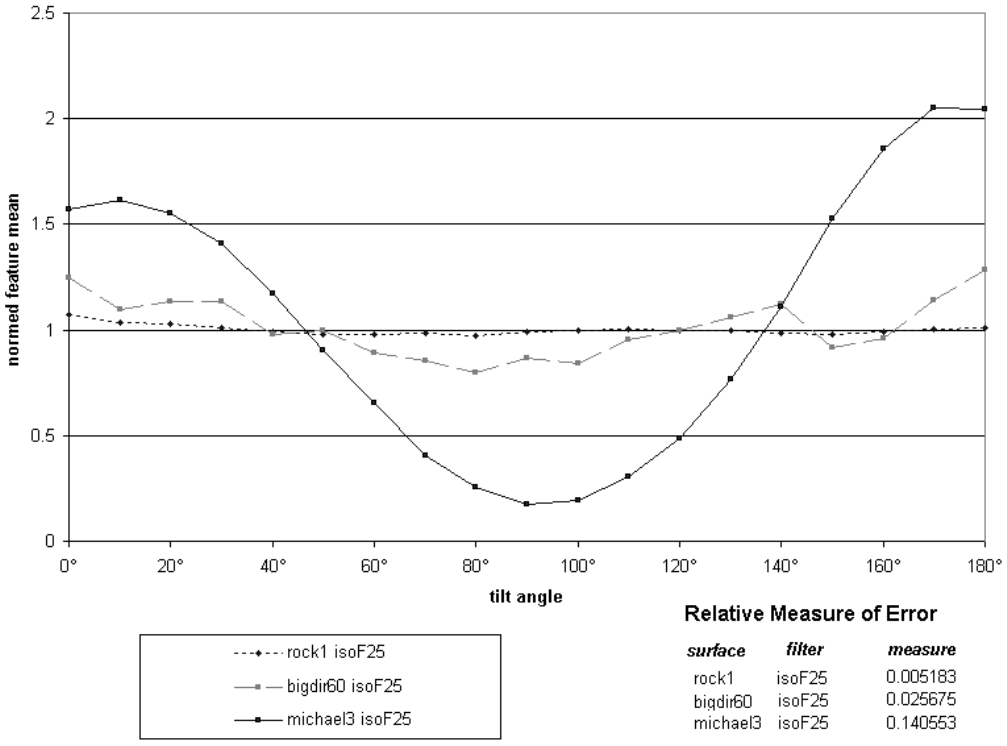


Figure 6.14: normalised feature mean of an isotropic, directional and unidirectional surface using an isotropic gabor filter exemplified by *rock1*, *bigdir60* and *michael3*

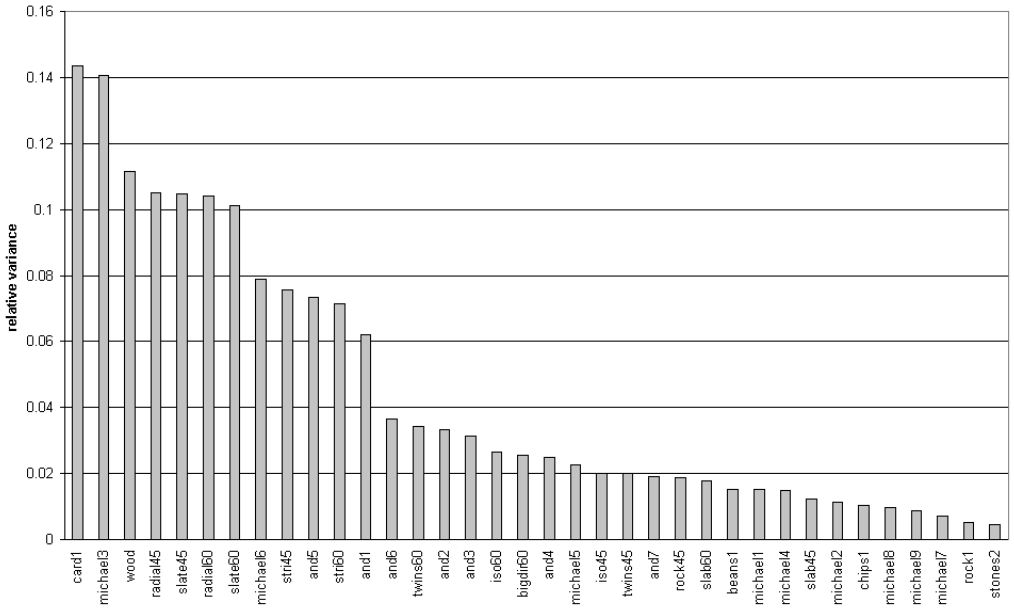


Figure 6.15: comparison between the relative variances of texture features using an isotropic Gabor filter

Figure 6.14 on the preceding page shows the normalised feature means[1] of the three surfaces investigated above. In agreement with the theory the output of the isotropic surface is nearly a line and the output of the unidirectional surface varies much. The behaviour of the directional surface is between these two marginal cases.

The relative variances of the feature means of all surfaces using an isotropic Gabor filter are shown on table A.3 on page 69. A chart of all relative variances is shown in figure 6.15. This measure of error is worst for the two unidirectional surfaces *card1* and *michael3* and best for isotropic surfaces. The more directional a surface is the worse is the relative variance of its texture feature.

6.3.4 Results of High Frequency Filters

Figure 6.9 on page 44 shows that the images filtered with the F50A45 filter have always less energy than images filtered with F25 filters. Now we analyse this effect more detailed. Figure 6.16 on the following page shows surface **slab45** in frequency domain. It can be seen that most of the energy and

[1] *mean* stands for the mean value over the whole image; *normalised* means that $f(\tau)$ is divided by its mean value

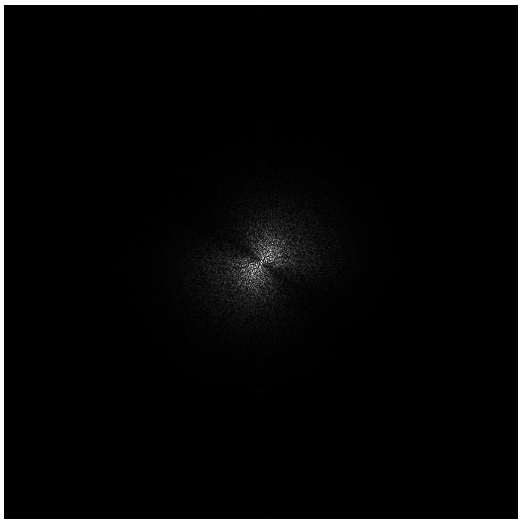


Figure 6.16: Norm of *slab45* in frequency domain

thus the information of the surface is situated in the lower frequencies.

Figure 6.17 on the next page shows gabor filters with a centre frequency of 20 and 150 circles per image. If the image is filtered with the F150A40 filter (this means it is multiplied in frequency domain) nearly all of the information of the image is lost.

Figure 6.18 on the following page shows the radial plot of *slab45* and the F25 and F150 filters.

Again it can be seen that nearly all of the energy is cut by the filter with 150 circles per image. The remaining energy is superposed by noise. This noise is mainly quantisation noise, which results from the eight bit quantisation of the digital camera. Because normally the full range of the camera is not used, one or two bits get lost. So the resulting quantisation is six or seven bit. Because of these problems the results produced with high frequency filters are not used in this work.

6.3.5 Using the Absolute Pixel Value

Quite often the average of the absolute pixel values $|x|$ is used as a estimation of the image energy instead of the variance. The advantage is that it is faster to compute. The disadvantage is that it is only a rough estimation of the image energy. Although the behaviour of the texture features due to tilt rotation can not be predicted analytically we compute the features and compare them with the best fit curves $f_1(\tau) = a + b\cos(2\tau + \phi)$ and $f_2(\tau) = \sqrt{a + b\cos(2\tau + \phi)}$. The best fit curve $f_2(\tau)$ is the theoretical be-

Figure 6.17: Gabor filter F25A45 and F100A45 in frequency domain

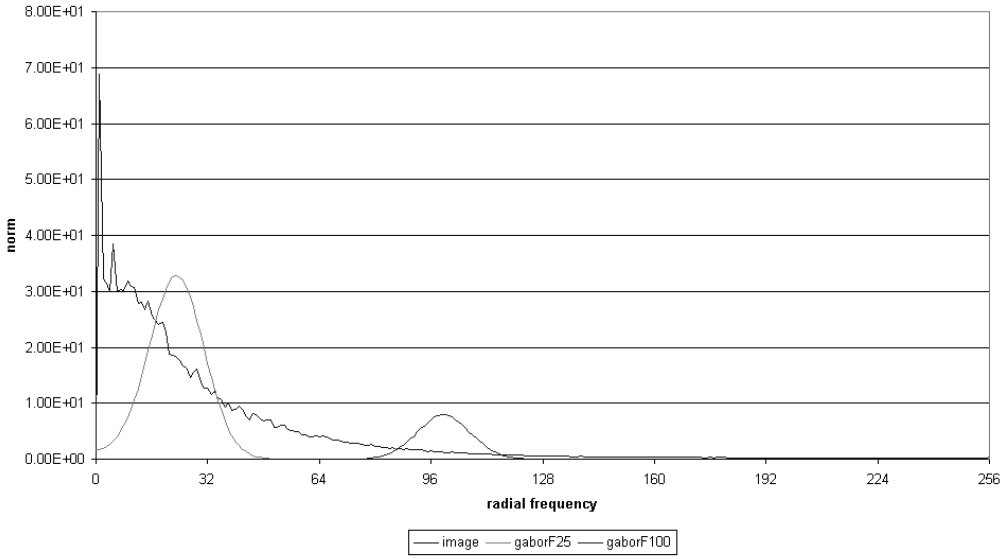


Figure 6.18: Polar plot of *slab45* and the filters in figure 6.17

haviour of the standard deviation. The results are shown in table A.4 on page 69. Figure 6.19 on the following page shows a comparison between the behaviour of these features. Although the absolute pixel value is different from the standard deviation surprisingly the results show that the best fit curve $f_2(\tau) = \sqrt{a + b\cos(2\tau + \phi)}$ is approximately as good for texture features using $|x|$ as the best fit curve $f_1(\tau) = a + b\cos(2\tau + \phi)$ for texture features using $(x)^2$. This is particular true for surfaces with a good fit (surfaces right of *michael7*).

6.3.6 Laws Filters

The statement of this work is that the behaviour of texture features derived from a linear filter followed by an energy estimation follows the equation $f(\tau) = a + b\cos(2\tau + \phi)$. This is true for every linear filter. Also it does not matter if the image is filtered by multiplying it in frequency domain or by convoluting it spatial domain. The previous experiments used Gabor filters in the frequency domain. One main reason to use Gabor filters was there popularity in the computer vision community. To show the general validity of this equation we investigate the behaviour of texture features using Laws filters. In contrast to Gabor filters we use Laws filters by convoluting the images in spatial domain. For this experiment we use the same surfaces as before and filter them with L5E5 and E5L5 filters (which are orthogonal). Figure 6.20 on page 54 shows the behaviour of the mean features derived from the surfaces *chips1* and *michael7*. It shows that the feature behaviour fits very good the prediction $f(\tau) = a + b\cos(2\tau + \phi)$. The results of all surfaces are shown in table A.5 on page 70. Figure 6.21 on page 55 shows a chart of the relative sinusoidal measures of error and the image energy after filtering with Laws filters. It shows that the results fit the theoretical prediction very well.

6.3.7 The phase of $f(\tau)$

In section 5.3.1 we predicted that for isotropic surfaces, the phase of the oscillation of f_{iso} is always twice the direction of the filter. We can use our previous experimental results to verify this prediction.

First we have to choose a set of isotropic surfaces. It is always difficult to decide if a surface is isotropic, because the image of an isotropic surface is directional, dependent on the illuminant direction. It is also difficult to decide the directionality from the visual appearance. All textures which are used in this section except *rock1* are made up of small randomly distributed objects and thus can be assumed as isotropic. In addition we use *rock1*

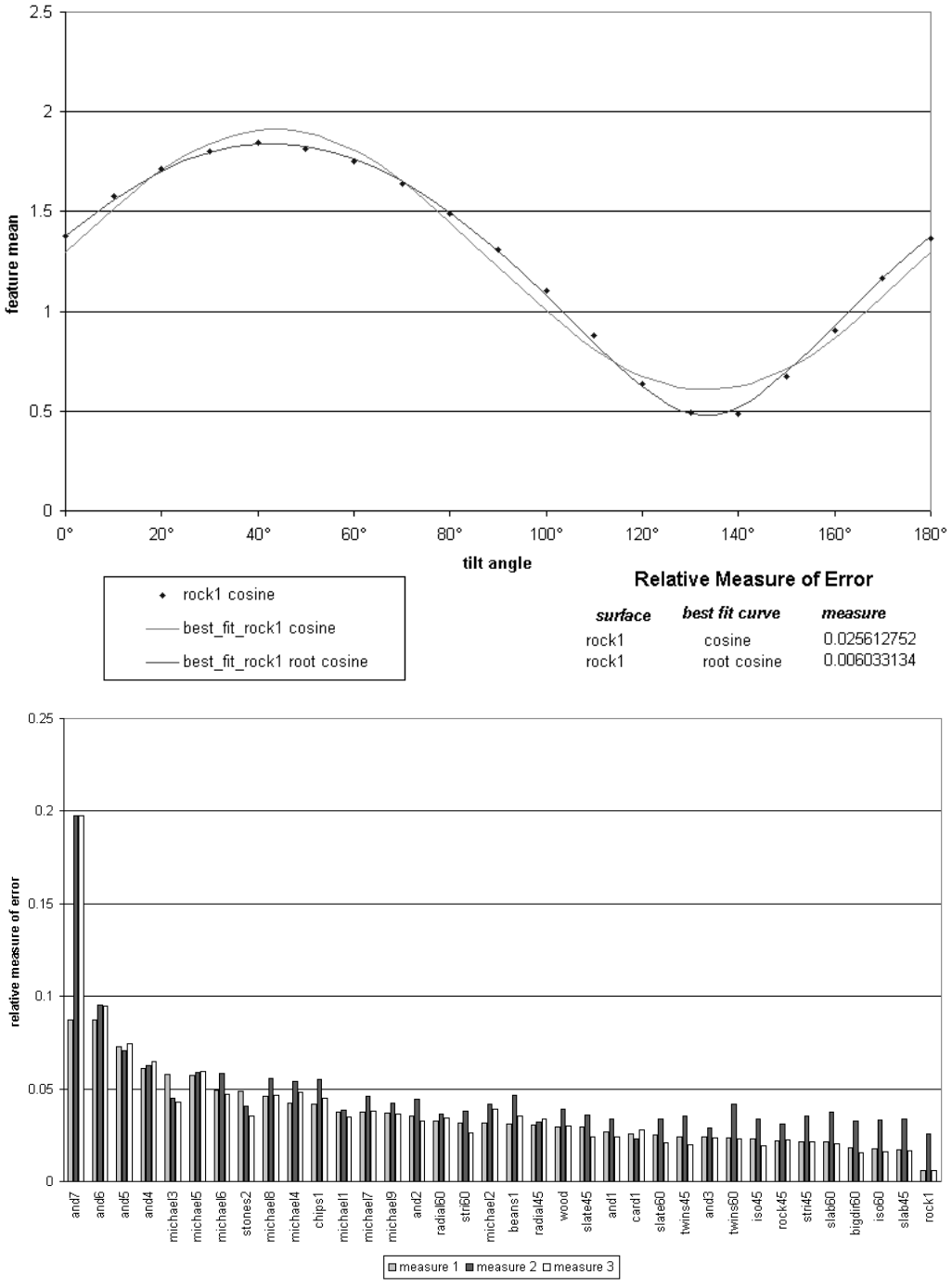


Figure 6.19: comparison between features calculated from the image variance and the mean absolute pixel value

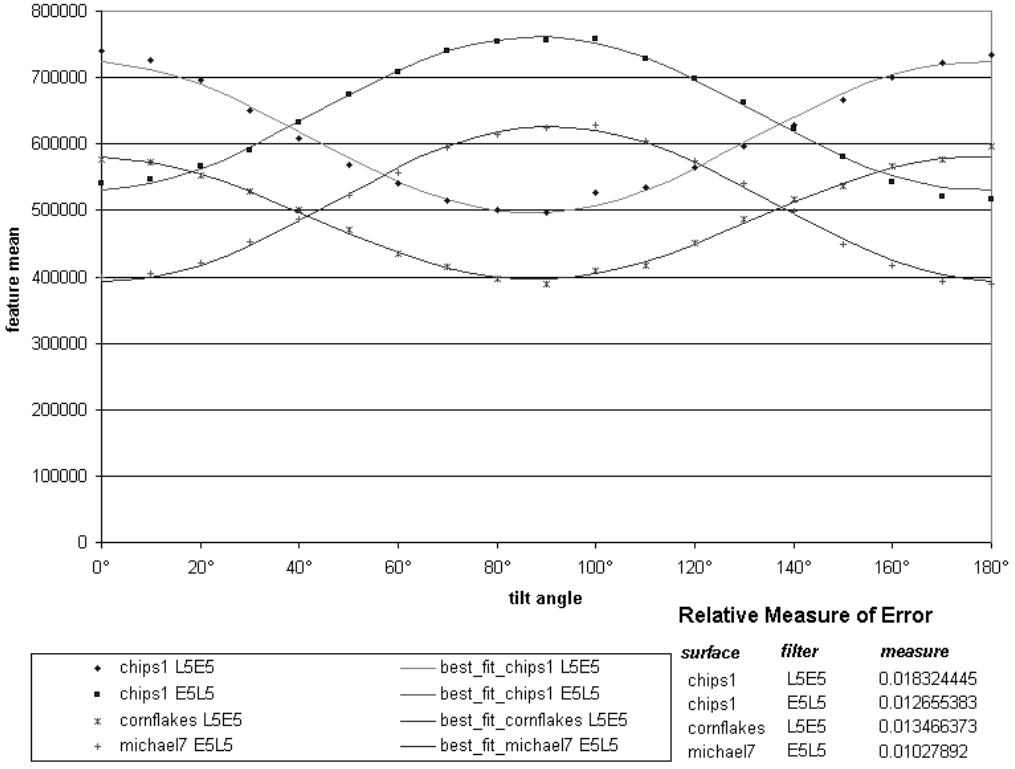
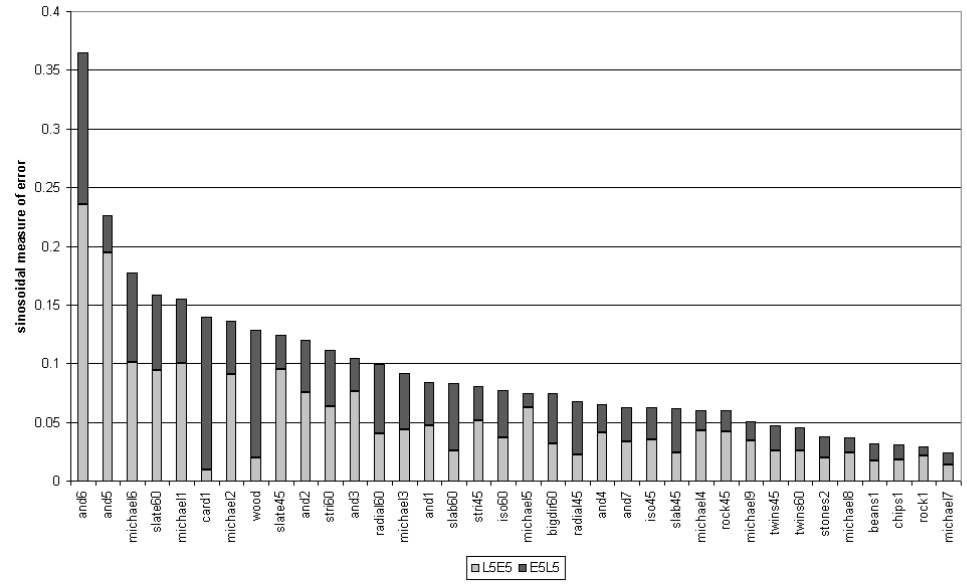


Figure 6.20: mean features using Laws filters



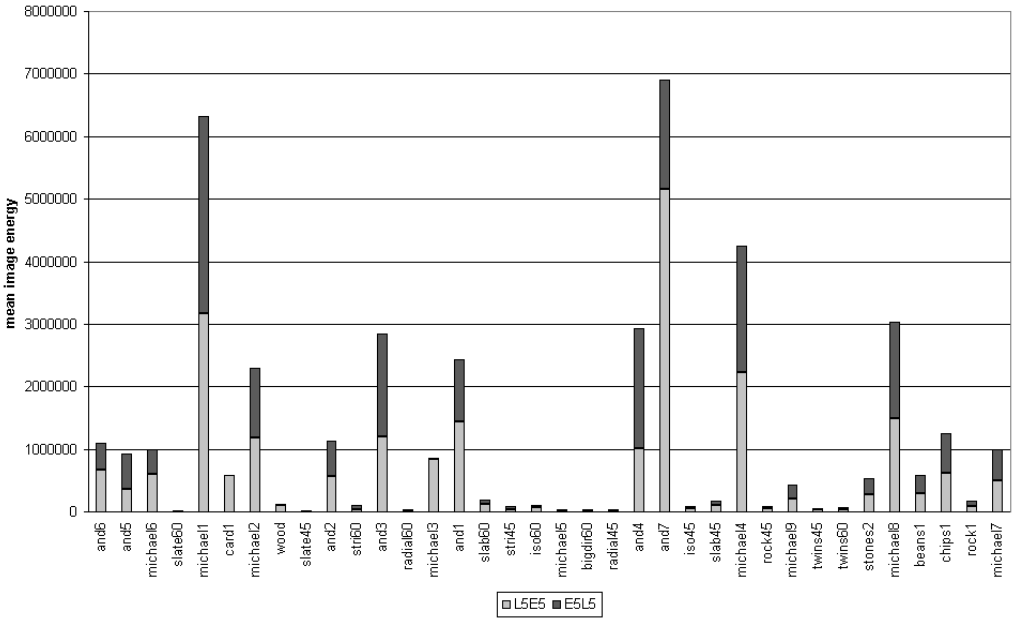


Figure 6.21: relative measures of error and image energy after filtering with Laws filters

filter	F25A0 complex	F25A45 complex	F25A90 complex	F25A135 complex	F50A45 complex	F25A45 real	L5E5	E5L5
theoretical phase of $f(\tau)$	0°	90°	180°	270°	90°	90°	0°	180°

Table 6.2: theoretical phase of $f(\tau)$ using isotropic surfaces

surface	F25A0 complex	F25A45 complex	F25A90 complex	F25A135 complex	F50A45 complex	F25A45 real	L5E5	E5L5	mean[a]
michael1	9.59°	1.60°	1.25°	-8.92°	-0.74°	2.86°	-23.60°	3.84°	6.55°
michael2	20.91°	3.28°	3.05°	-9.26°	-14.49°	4.22°	-15.33°	4.75°	9.41°
michael4	-20.17°	-7.58°	-5.03°	3.00°	-10.11°	-3.63°	11.39°	-11.91°	9.10°
michael7	-20.25°	7.42°	-7.87°	-3.19°	7.95°	8.29°	-4.61°	2.43°	7.75°
michael8	-0.30°	4.87°	-1.66°	-1.10°	13.27°	6.93°	-9.38°	6.14°	5.46°
michael9	-4.42°	1.47°	-3.05°	2.83°	6.97°	2.56°	-0.03°	4.87°	3.27°
chips1	-26.32°	-7.84°	-14.68°	-1.13°	-3.99°	-3.73°	-5.72°	-3.90°	8.42°
beans1	8.52°	-0.87°	-7.55°	1.83°	-4.95°	-3.21°	2.25°	-4.29°	4.18°
rock1	-1.24°	-8.60°	-4.46°	-3.31°	-3.04°	-5.11°	1.60°	-7.43°	4.35°
stones2	10.50°	-2.82°	20.04°	10.13°	-41.87°	-3.44°	-2.90°	1.45°	11.64°
mean[a]	12.22°	4.63°	6.86°	4.47°	10.74°	4.40°	7.68°	5.10°	7.01°

[a]mean of absolute value

Table 6.3: differences between the theoretical and experimental curve phases in degree

because we can assume its isotropy from previous experiments (figures 6.11 on page 46 and 6.15 on page 49). The theoretical phase of $f(\tau)$ for different filters is shown in table 6.2. The difference between the theoretical and the experimental phase of $f(\tau)$ is shown in table 6.3. The mean error over all results is only 7.1°, which confirms our prediction.

6.3.8 Behaviour in 1D and 2D Feature Space

In section 5.4 we investigated the behaviour of features in the one and two dimensional feature space analytically. We claimed that in general an illumination tilt independent classification or segmentation is not possible. We predicted an ellipse-shaped behaviour of features in the two dimensional feature space. In the special case of isotropic textures and two identical but orthogonal filters the curve is a line. In the special case of isotropic textures and two identical filters except a difference in direction of 45° the curve is a circle. We now verify these predictions using our experimental results.

Figure 6.22 on the following page shows the behaviour of features using a L5E5 Laws filter. It is obvious that there is no possible boundary between the feature curves. Even if the illuminant tilt angle is known we can not distinguish between the textures for particular illumination tilt angles, because the curves cross each other. E.g. if the illumination tilt angle is 120° we can

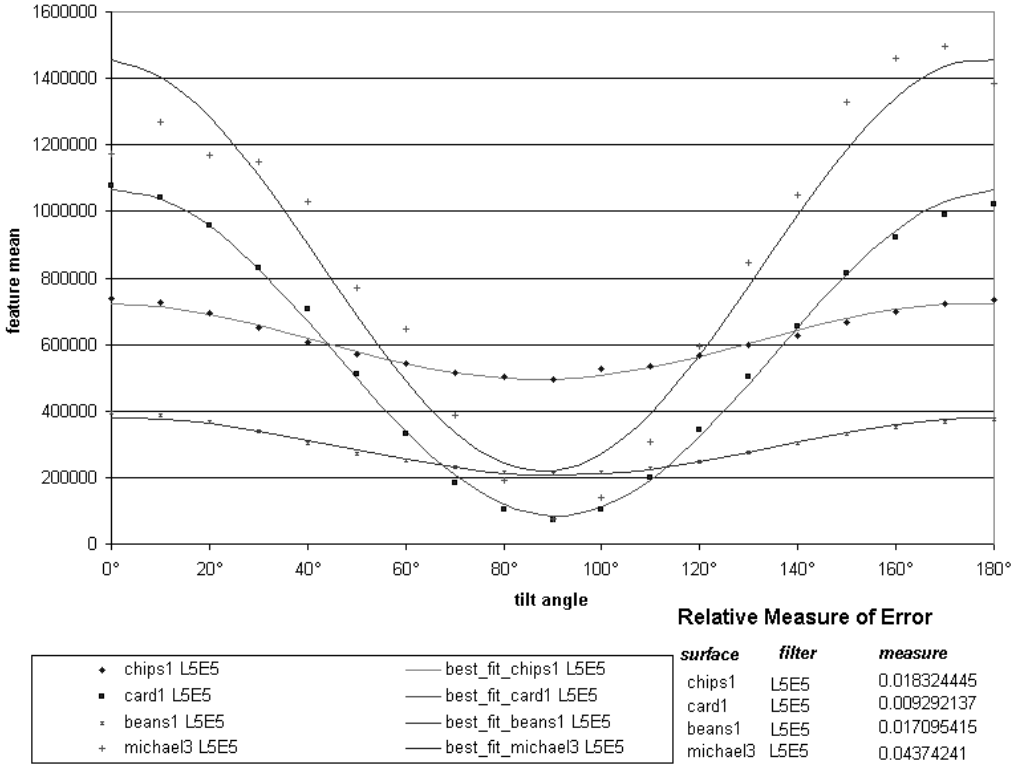


Figure 6.22: one dimensional feature space using a L5E5 Laws filter

not distinguish between *chips1* and *michael3*.

Figure 6.23 on the next page shows the behaviour of different surfaces in the two dimensional feature space together with their best fit ellipses using F25A0 and F25A45 Gabor filters. The best fit ellipses in two-dimensional feature space are computed by plotting the best fit sinusoidal curves in one-dimensional feature space (equation 6.1 on page 39) against each other. It can be seen that the ellipse is a good approximation for the behaviour in feature space. Because the filters are identical except a difference in direction of 45° the curve of the isotropic surface *rock1* is a circle. Figure 6.24 on page 60 shows features of different surfaces using L5E5 and E5L5 Laws filters. Because the filters are identical except a difference in direction of 90° the curve of the isotropic surface *bigdir60* is a line.

In both charts the ellipses of the different surfaces cross each other. Thus in general illumination tilt independent classification or segmentation is not possible.

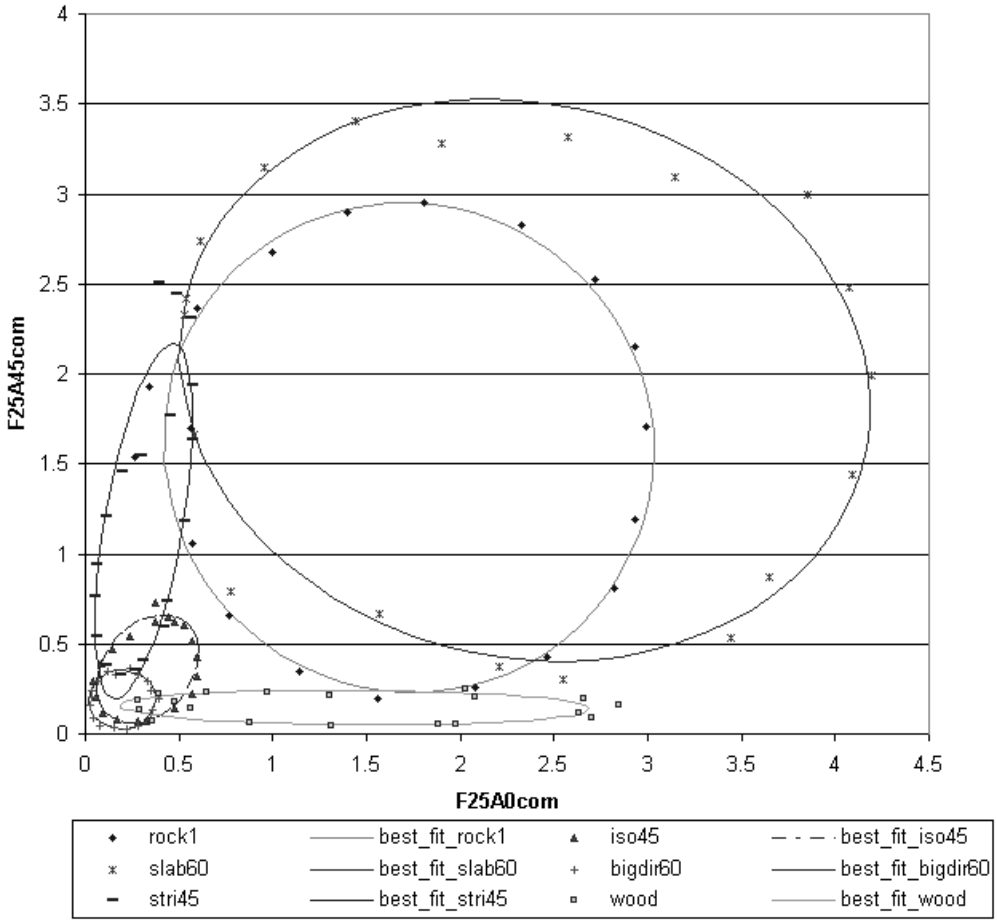


Figure 6.23: two-dimensional feature space using F25A0 and F25A45 Gabor filters

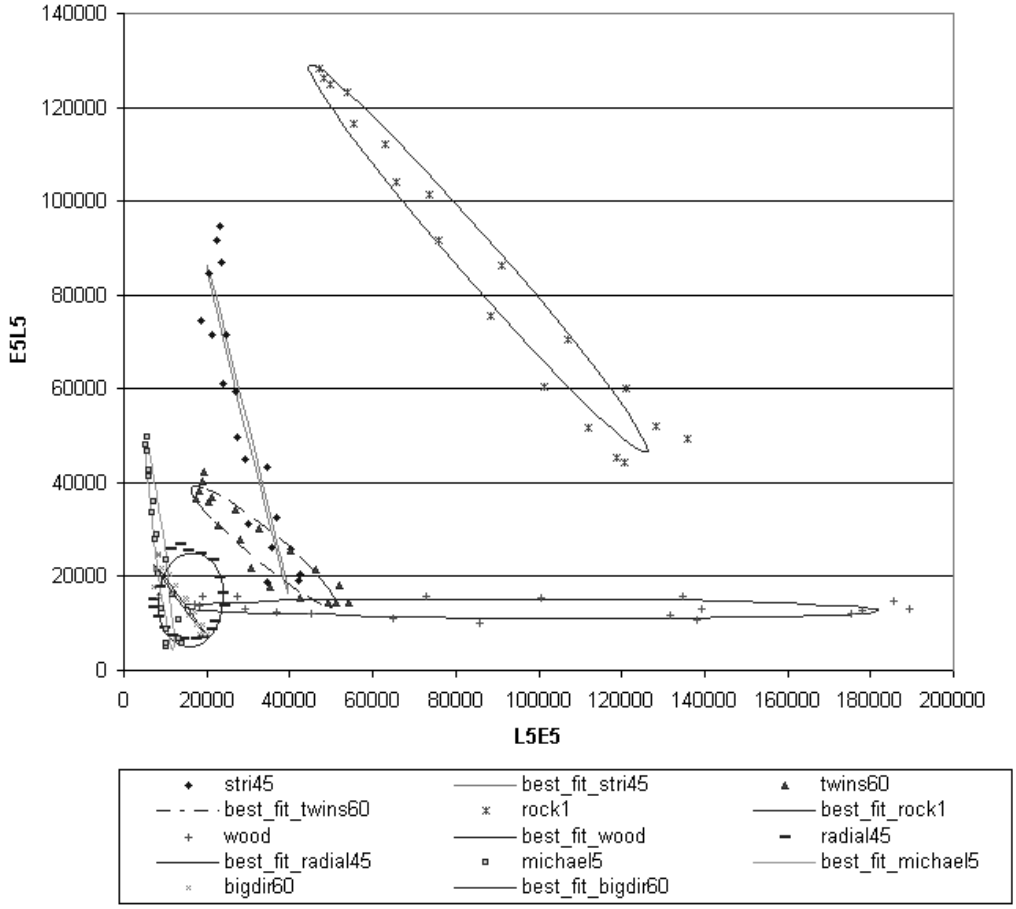


Figure 6.24: two-dimensional feature space using L5E5 and E5L5 Laws filters

Chapter 7

Summary and Further Research

7.1 Summary

The main result of this dissertation is the sinusoidal behaviour of texture features due to changing illumination tilt angle τ:

$$\begin{aligned} f(\tau) &= a + b\cos(2\tau) + c\sin(2\tau) \\ &= a + d\cos(2\tau + \phi) \end{aligned}$$

$$\begin{aligned} a &= 1/2\, sin^2(\sigma) \int_0^\infty \omega^3 \int_0^{2\pi} A(\omega,\theta)\, d\theta\, d\omega \\ b &= 1/2\, sin^2(\sigma) \int_0^\infty \omega^3 \int_0^{2\pi} cos(2\theta) A(\omega,\theta)\, d\theta\, d\omega \\ c &= 1/2\, sin^2(\sigma) \int_0^\infty \omega^3 \int_0^{2\pi} sin(2\theta) A(\omega,\theta)\, d\theta\, d\omega \end{aligned}$$

$$d = \sqrt{b^2 + c^2}, \qquad \phi = \arctan(c/b)$$

The parameters a, b and c are only dependent an the surface PSD and the filter PSD. This prediction is valid for all features derived using a linear filter followed by a signal energy estimator. It was derived presuming the following assumptions

- Lambertian reflection

- no self or cast shadowing
- no inter-reflection
- small slopes.
- orthographic projection

Although this equation can not be simplified in general, we made further predictions for special cases. If an isotropic texture is filtered with several filters which are identical except a different filter direction α, the parameters a and d are constant for every α. The maximum of $f(\tau)$ is always at $\tau = \alpha$ and the phase of $f(\tau)$ is always twice the filter direction α:

$$f_{uni}(\tau, \alpha, \beta) = a(\alpha) + d(\alpha)\cos[2(\tau - \beta)].$$

If a unidirectional texture with the surface direction β is filtered with several filters which are identical except a different filter direction α, the parameters a and d are maximal if $\alpha = \beta$. The maximum of $f(\tau)$ is always at $\tau = \beta$ and the phase of $f(\tau)$ is always twice the surface direction β:

$$f_{uni}(\tau, \alpha, \beta) = a(\alpha) + d(\alpha)\cos[2(\tau - \beta)].$$

In addition we analysed the behaviour of features in the 2D feature space as a function of illuminant tilt τ. We predicted an ellipse as the resulting figure. If the texture is isotropic and the two filters are identical except for a phase difference of 45°, the resulting figure is a circle. If the texture is isotropic and the two filters are identical except for a phase difference of 90°, the resulting figure is a line. We showed that in general these ellipses, circles and lines cross each other. Thus a illumination tilt independent classification is not possible.

We verified this predictions both by simulation and experiments: We computed the parameters a, b and c from the surfaces of synthetic generated surfaces and compared them with the texture features computed from illumated images. We used experiments to show the sinusoidal behaviour of 30 different real world textures using different Gabor and Laws filters.

7.2 Further Research

This work investigated only the effect of changing tilt. We think it is worth investigating the effect of changing illuminant slant angle, too. Equation 5.2

on page 26 can also be rearranged to predict the effect of illumination slant changes:

$$
\begin{aligned}
f(\sigma) &= sin^2(\sigma)\, C \\
C &= 1/2 \left[\int_0^{\infty} \omega^3 \int_0^{2\pi} A(\omega,\theta)\, d\theta\, d\omega \right. \\
&+ \cos(2\tau) \int_0^{\infty} \omega^3 \int_0^{2\pi} cos(2\theta) A(\omega,\theta)\, d\theta\, d\omega \\
&+ \left. \sin(2\tau) \int_0^{\infty} \omega^3 \int_0^{2\pi} sin(2\theta) A(\omega,\theta)\, d\theta\, d\omega \right]
\end{aligned}
$$

The parameter C is independent of the illumination slant angle σ. Thus we expect that $f(\sigma)$ is proportional to $\sin^2(\sigma)$. Unfortunatelly at the time of this work we did not have the equipment to take texture images with different defined illumination slant angles. Thus we can not verify this prediction.

We hope that this work will be usefull for future investigation. An interesting task for future research is to use these results to compensate the effect of illumination angle changes. One technique could be to develop illumination compensation filters as the first stage of a classifier.

Appendix A

Results

A.1 Feature Charts as a Function of Illumination Tilt

The underlying experimental results for this work are very numerous. We used 30 different real surfaces (6 of them with two different slant angles) and filtered them with seven different complex, real and isotropic Gabor filters and two Laws filters. This means a total of 324 resulting curves. This is fare to much to show it in this work, even in the appendix. To get a typical selection of all results we first compute the histogram of all 288 results using a directional filter (figure A.1 on the next page). The medium of these results is at 0.036408. This means that half of the results are better and half of the results worse. Figure A.2 on the following page shows the four curves which are nearest to the median. We treat them as typical results. They match the sinusoidal prediction quite good. Now we take all results which are better than the medium and compute the medium of these results. The result is 0.024118. One quarter of all results are better, three quarter worse. We treat these results as typical good results. Figure A.3 on page 66 shows the four curves which are nearest to this value. The same can be made with the bad half of the results (the result is 0.056864) to get typical bad results (figure A.4 on page 66). Even these bad results fit the sinusoidal behaviour quite good.

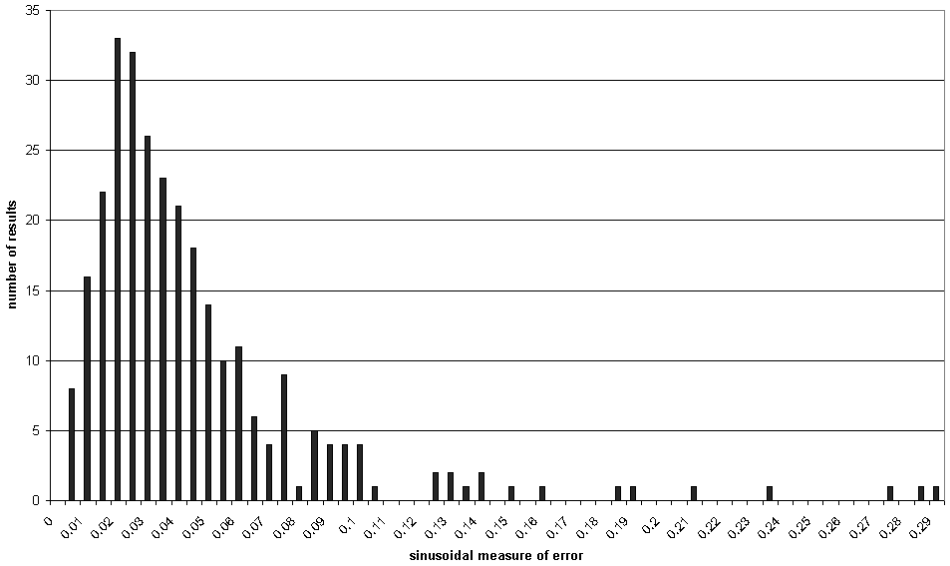


Figure A.1: Histogram of all results using directional filters

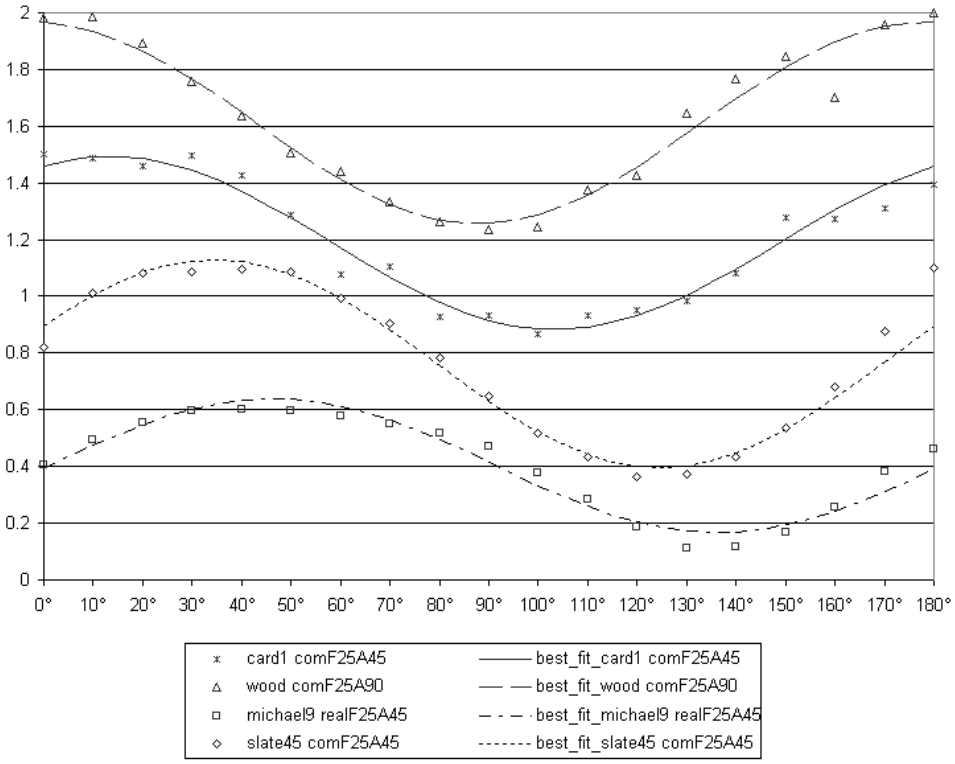


Figure A.2: Results with a typical measure of error (results which are nearest to the median)

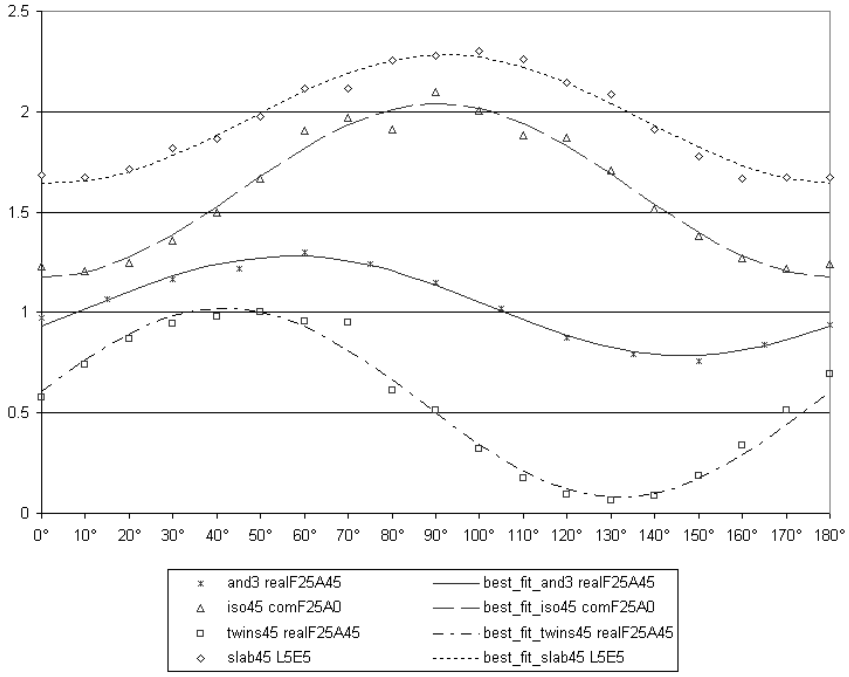


Figure A.3: Results with a good measure of error

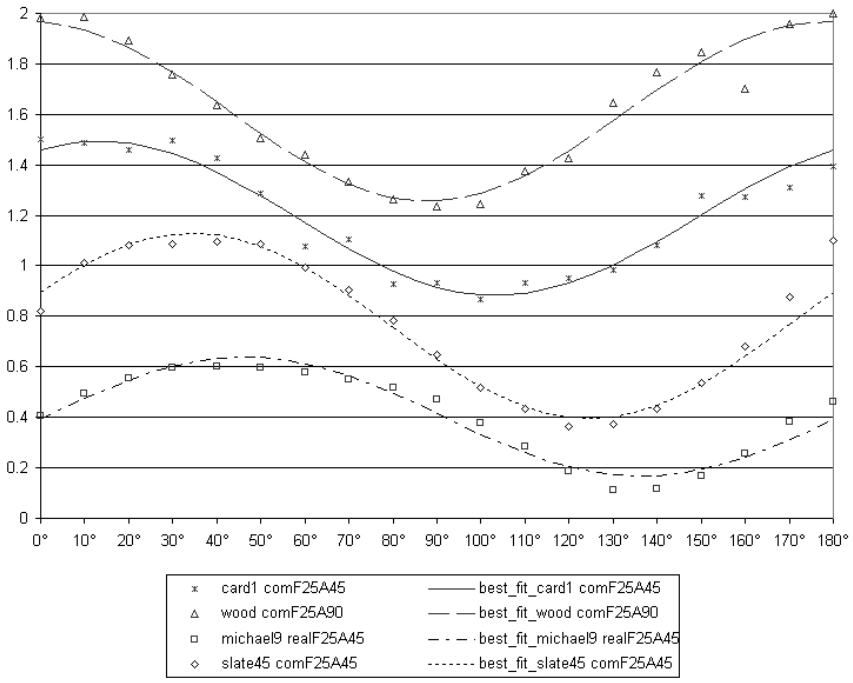


Figure A.4: Results with a bad measure of error

A.2 Measures of Error

surface	F25A0 complex	F25A45 complex	F25A90 complex	F25A135 complex	F50A45 complex	F25A45 real	**mean**
card1	0.054369	0.035566	1.017084	0.079127	0.068492	0.02574	0.213396
and6	0.063686	0.086828	0.164711	0.104357	0.088089	0.087006	0.099112
stones2	0.046747	0.072754	0.076088	0.068525	0.277682	0.048795	0.098431
and7	0.091667	0.047163	0.08257	0.154642	0.085483	0.087043	0.091428
michael3	0.026677	0.06633	0.28685	0.053984	0.031353	0.057988	0.087197
and5	0.022222	0.061727	0.014496	0.129018	0.132232	0.072793	0.072081
stri45	0.213432	0.022094	0.047372	0.05804	0.020312	0.021481	0.063788
michael9	0.039038	0.044338	0.035215	0.041467	0.185489	0.036915	0.063743
michael2	0.073287	0.028472	0.053759	0.046193	0.144608	0.031357	0.062946
chips1	0.051269	0.042766	0.060856	0.057672	0.099673	0.041786	0.059003
stri60	0.139012	0.032127	0.049806	0.051655	0.031095	0.031838	0.055922
michael4	0.078138	0.043032	0.072205	0.054501	0.038028	0.04246	0.054727
michael8	0.035249	0.050461	0.06259	0.037735	0.093974	0.045939	0.054324
and3	0.142825	0.030557	0.039851	0.0393	0.045594	0.023968	0.053682
and4	0.023614	0.067834	0.026939	0.101393	0.034108	0.06115	0.052506
michael6	0.061337	0.043344	0.023918	0.033608	0.100186	0.049304	0.051949
michael7	0.076699	0.050834	0.047165	0.047289	0.044746	0.037419	0.050692
michael1	0.056876	0.053088	0.061681	0.022284	0.042705	0.037679	0.045718
slate60	0.05154	0.026511	0.052377	0.095803	0.020354	0.02491	0.045249
and2	0.026698	0.044887	0.042821	0.04201	0.069327	0.035313	0.043509
radial60	0.033431	0.032491	0.07676	0.045987	0.026204	0.032428	0.041216
radial45	0.052869	0.031642	0.075244	0.026582	0.020868	0.030514	0.039619
slate45	0.043494	0.036971	0.031717	0.067654	0.025493	0.029427	0.039126
and1	0.018258	0.039975	0.032107	0.038667	0.056323	0.026643	0.035328
beans1	0.02202	0.029702	0.034145	0.030876	0.061106	0.030913	0.034793
michael5	0.01228	0.056852	0.01135	0.02733	0.040942	0.057527	0.034380
wood	0.030409	0.031727	0.035901	0.028741	0.039329	0.029562	0.032611
iso45	0.02408	0.027424	0.016261	0.051409	0.021048	0.023168	0.027231
slab60	0.011817	0.023054	0.032901	0.035221	0.026029	0.021155	0.025029
iso60	0.024208	0.019389	0.023944	0.046628	0.016805	0.017888	0.024810
twins45	0.025028	0.028458	0.016922	0.018028	0.02283	0.024156	0.022570
bigdir60	0.014462	0.019759	0.026188	0.032023	0.013179	0.018249	0.020643
rock45	0.027184	0.024394	0.009213	0.00938	0.02621	0.022093	0.019745
slab45	0.013637	0.017607	0.02174	0.02637	0.018404	0.017278	0.019172
twins60	0.021057	0.025308	0.013829	0.011721	0.01869	0.023593	0.019033
rock1	0.017626	0.007708	0.011208	0.008637	0.007395	0.005794	0.009728
mean	0.049062	0.038977	0.077438	0.050662	0.058177	0.036424	0.051790

Table A.1: relative sinusoidal measure of error using Gabor filters sorted by the mean measure of error of every surface

surface	F25A0 complex	F25A45 complex	F25A90 complex	F25A135 complex	F50A45 complex	F25A45 real	**mean** [a]
card1	3.29838	0.004985	0.014553	0.005558	0.007893	0.00973	0.213396
and6	3.989043	1.651365	3.567205	1.971362	2.683147	3.815013	0.099112
stones2	14.23342	14.54340	13.53327	14.9703	0.706138	17.82059	0.098431
and7	4.592971	9.825123	6.495529	8.975717	3.578888	16.97039	0.091428
michael3	15.94551	0.276344	0.113047	0.198883	0.047971	0.275129	0.087197
and5	0.426464	0.085127	1.277333	0.086995	0.084733	0.195629	0.072081
stri45	0.64887	0.312584	1.18271	0.31746	0.089843	0.618672	0.063788
michael9	4.378596	3.764813	4.285226	3.992625	0.611411	7.652123	0.063743
michael2	42.40027	37.59083	42.41390	33.29939	2.789578	74.81456	0.062946
chips1	12.72381	13.20491	14.63473	13.30125	1.619873	28.89644	0.059003
stri60	0.564672	0.425475	1.401759	0.400147	0.120075	0.846809	0.055922
michael4	22.04097	28.54557	18.97303	18.20272	9.250407	57.56914	0.054727
michael8	35.20553	34.05491	32.93947	37.19066	3.777672	84.82105	0.054324
and3	16.53467	7.476012	27.40703	7.961772	3.511898	15.73041	0.053682
and4	0.719646	0.135869	0.963437	0.1406	0.154217	0.292835	0.052506
michael4	6.415494	1.85276	3.394262	2.834938	0.770201	4.181116	0.051949
michael7	15.69859	15.90760	17.75771	18.67721	1.214948	30.75361	0.050692
michael1	46.02110	41.50654	46.31262	49.52092	7.423702	104.6494	0.045718
slate60	0.01997	0.026104	0.18633	0.069534	0.009062	0.050479	0.045249
and2	3.952079	2.916004	3.532771	3.794899	1.960165	6.860466	0.043509
radial60	0.131105	0.241141	0.111216	0.0517	0.084971	0.496054	0.041216
radial45	0.127728	0.244347	0.11877	0.051245	0.084718	0.504555	0.039619
slate45	0.014862	0.020207	0.132001	0.0529	0.006486	0.038352	0.039126
and1	4.779757	3.704695	23.64979	4.468808	3.336491	4.951399	0.035328
beans1	9.321997	8.102075	7.020569	8.77205	0.629097	18.58046	0.034793
michael5	0.615729	0.014274	0.534401	0.014914	0.0082	0.033138	0.034380
wood	1.432406	0.147783	0.096377	0.237811	0.034888	0.279384	0.032611
iso45	0.562585	0.332292	0.360078	0.506555	0.088466	0.79639	0.027231
slab60	2.340606	1.966591	1.736869	2.032821	0.22392	4.40865	0.025029
iso60	0.796156	0.444498	0.448454	0.664486	0.113707	1.058005	0.024810
twins45	0.479373	0.278629	0.245939	0.343577	0.061581	0.622172	0.022570
bigdir60	0.195587	0.191815	0.336983	0.203441	0.033031	0.438549	0.020643
rock45	0.744335	0.6374	0.929729	0.921882	0.106178	1.45596	0.019745
slab45	2.100186	1.715461	1.508594	1.724276	0.200342	3.754119	0.019172
twins60	0.582478	0.327927	0.290069	0.433404	0.07062	0.731837	0.019033
rock1	1.671747	1.391594	1.722263	1.592928	0.219182	2.951367	0.009728
mean	7.658520	6.496307	7.767446	6.610715	1.269825	13.831223	7.272339

Table A.2: mean image energy after filtering with Gabor filters sorted by the mean measure of error of every surface

[a] mean of the relative measure of error (see table A.1 on the page before)

surface	isotropic
card1	0.143657
michael3	0.140553
wood	0.111518
radial45	0.105104
slate45	0.104776
radial60	0.104213
slate60	0.101331
michael6	0.078729
stri45	0.075587
and5	0.073491
stri60	0.07145
and1	0.062104
and6	0.036658
twins60	0.034128
and2	0.033422
and3	0.031401
iso60	0.026466
bigdir60	0.025675
and4	0.024746
michael5	0.022489
iso45	0.020064
twins45	0.019909
and7	0.019148
rock45	0.018628
slab60	0.017729
beans1	0.015333
michael1	0.015124
michael4	0.014736
slab45	0.012351
michael2	0.011302
chips1	0.010494
michael8	0.00957
michael9	0.008569
michael7	0.006954
rock1	0.005183
stones2	0.004651
mean	0.044923

Table A.3: relative variance using an isotropic Gabor filter

energy estimation	$(x)^2$	$\|x\|$	
best fit curve	$a + b\cos(2\tau + \phi)$	$\sqrt{a + b\cos(2\tau + \phi)}$	
and7	0.087043	0.197465	0.197347
and6	0.087006	0.095079	0.094487
and5	0.072793	0.070531	0.074579
and4	0.061149	0.062610	0.064773
michael3	0.057987	0.045102	0.042570
michael5	0.057523	0.058642	0.059410
michael6	0.049304	0.058171	0.047209
stones2	0.048795	0.040807	0.035536
michael8	0.045938	0.055529	0.046726
michael4	0.042459	0.054139	0.048403
chips1	0.041785	0.054886	0.045014
michael1	0.037678	0.038665	0.034558
michael7	0.037419	0.045802	0.037797
michael9	0.036915	0.042459	0.036317
and2	0.035313	0.044248	0.032794
radial60	0.032427	0.036234	0.034423
stri60	0.031837	0.038167	0.026311
michael2	0.031356	0.041645	0.039140
beans1	0.030912	0.046794	0.035239
radial45	0.030514	0.032262	0.033584
wood	0.029562	0.039027	0.029835
slate45	0.029426	0.036116	0.023967
and1	0.026643	0.033965	0.024167
card1	0.025745	0.023234	0.028042
slate60	0.024911	0.033552	0.021124
twins45	0.024156	0.035570	0.019994
and3	0.023968	0.028875	0.023294
twins60	0.023593	0.041506	0.023116
iso45	0.023167	0.033869	0.019243
rock45	0.022092	0.030933	0.022309
stri45	0.021481	0.035134	0.021171
slab60	0.021154	0.037262	0.020600
bigdir60	0.018248	0.032892	0.015334
iso60	0.017888	0.033048	0.015973
slab45	0.017278	0.033583	0.016578
rock1	0.005794	0.025612	0.006033
mean	0.036424	0.047039	0.038805

Table A.4: relative sinusoidal measure of error of features derived from the image energy and the absolute value using a F25A45 Gabor filter

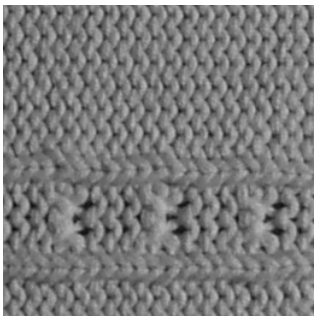
surface	measure of error			image energy	
	L5E5	E5L5	mean	L5E5	E5L5
and6	0.235421	0.129158	0.182289	675888.3	415584.0
and5	0.194427	0.031635	0.113031	356449.6	571387.1
michael6	0.101347	0.076138	0.088743	594409.1	397070.8
slate60	0.094415	0.063615	0.079015	4577.903	10294.56
michael1	0.099910	0.055526	0.077718	3163524	3163167
card1	0.009292	0.130257	0.069775	574511.1	2594.397
michael2	0.090369	0.045408	0.067889	1183644	1120102
wood	0.019517	0.109054	0.064286	98280.11	13219.89
slate45	0.095232	0.028654	0.061943	3267.912	7494.331
and2	0.075096	0.044822	0.059959	557249.6	581503.9
stri60	0.063488	0.047676	0.055582	40250.50	65905.26
and3	0.075953	0.028460	0.052207	1207397	1629082
radial60	0.040008	0.059646	0.049827	15874.56	14773.47
michael3	0.043742	0.047876	0.045809	838209.8	12192.95
and1	0.046851	0.037470	0.042161	1431479	996576.0
slab60	0.025889	0.056916	0.041402	113888.5	81652.81
stri45	0.051637	0.028923	0.040280	29831.72	51298.23
iso60	0.037187	0.039869	0.038528	65389.69	42686.50
michael5	0.062157	0.012675	0.037416	8656.350	25828.49
bigdir60	0.031550	0.042853	0.037201	13276.24	15305.63
radial45	0.022187	0.045648	0.033917	16129.09	15095.16
and4	0.040719	0.024714	0.032716	1014255	1921410
and7	0.033399	0.029372	0.031386	5156531	1752334
iso45	0.035241	0.027011	0.031126	48706.76	33872.63
slab45	0.024190	0.037185	0.030687	98869.76	71345.01
michael4	0.042802	0.017298	0.030050	2227981	2016206
rock45	0.041693	0.018217	0.029955	45195.40	41753.78
michael9	0.034670	0.016280	0.025475	204894.5	217041.9
twins45	0.025739	0.021065	0.023402	27221.34	22051.95
twins60	0.026029	0.019131	0.022580	33835.03	26468.79
stones2	0.020017	0.017592	0.018805	269805.2	260318.3
michael8	0.024224	0.012688	0.018456	1493259	1542111
beans1	0.017095	0.014526	0.015810	293904.3	280657.0
chips1	0.018324	0.012655	0.015489	610115.3	645875.8
rock1	0.021646	0.007758	0.014702	85366.25	87719.12
michael7	0.013466	0.010278	0.011872	488564.3	510302.4
mean	0.053748	0.040224	0.046986	641408.1	518396.8

Table A.5: relative sinusoidal measure of error and mean image energy using Laws filters sorted by the mean measure of Error of every surface

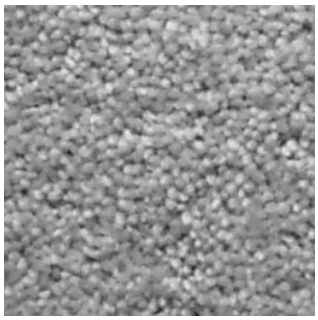
Appendix B

Texture Data Base

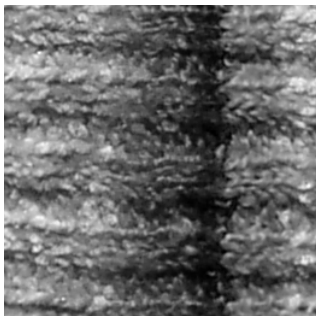
The experiments of this work are implemented using thirty different textures, each imaged with a illumination tilt angle between 0° and 180°. The difference of the illumination tilt angle of two consecutive images is 10° except the data sets *and1* to *and7*, which are imaged in 15° steps. All textures are illuminated at a slant angle of 45°, in addition some surfaces are also illuminated with a slant angle of 60°. In the following one image of every texture illuminated with 45° slant and 0° tilt is shown.



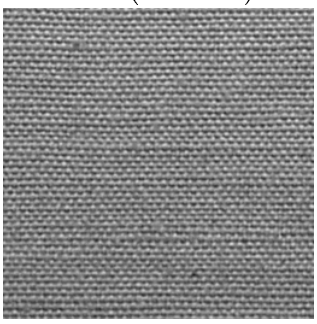
and1(zoomed)



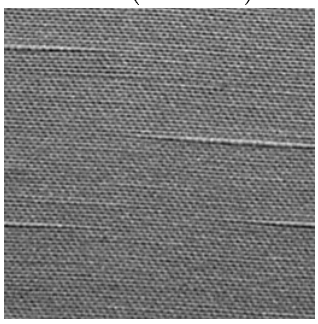
and2(zoomed)



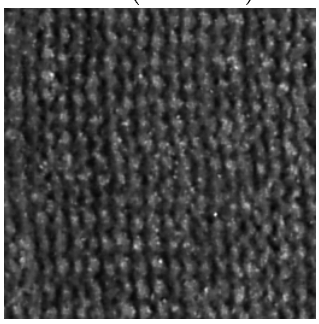
and3(zoomed)



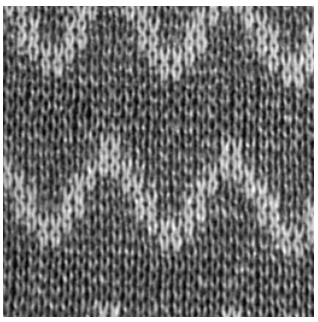
and4(zoomed)



and5(zoomed)



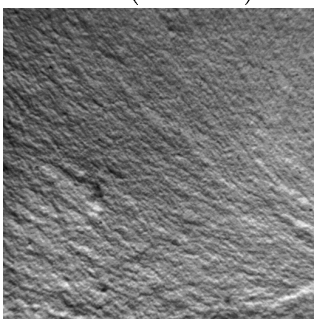
and6(zoomed)



and7(zoomed)

bigdir45

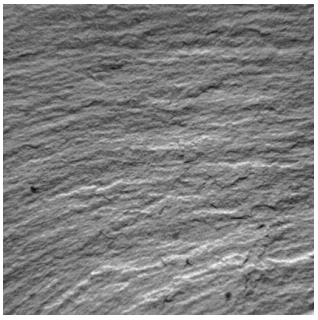
iso45



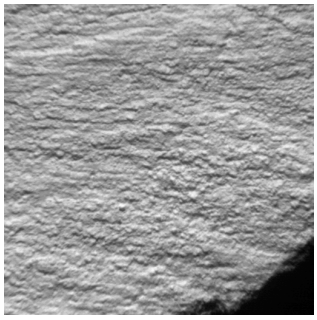
radial45

rock45

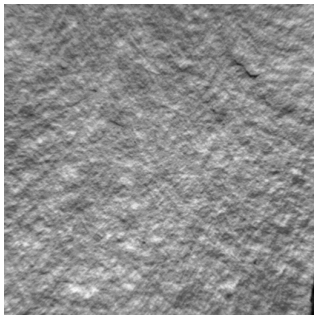
slab45



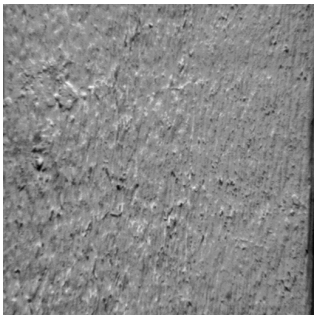
slate45



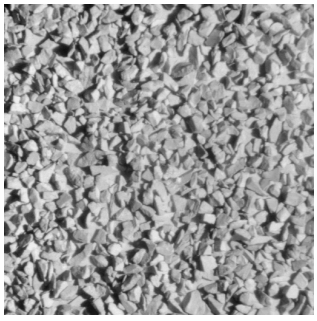
stri45



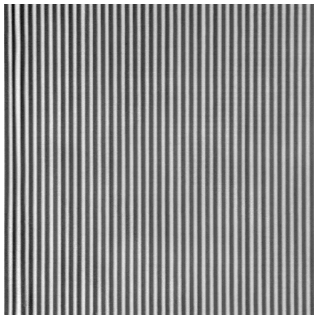
twins45



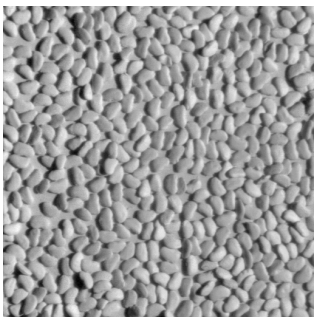
wood



chips1



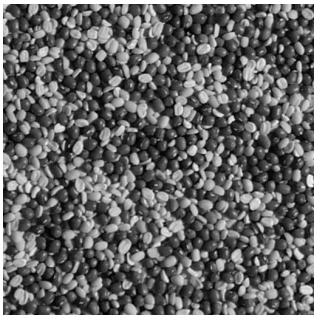
card1



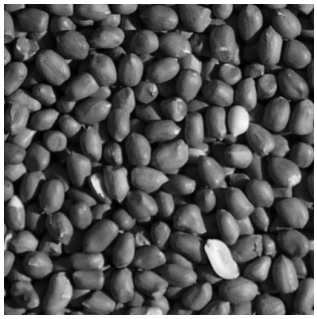
beans1

rock1

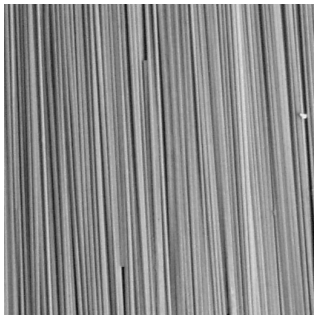
stones2



michael1

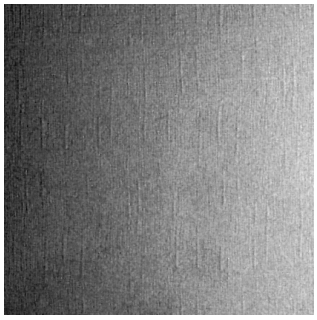


michael2



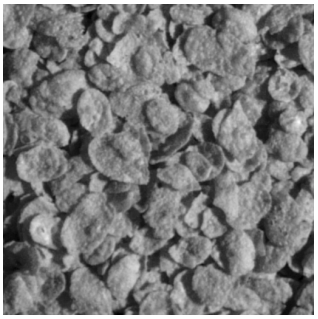
michael3

michael4

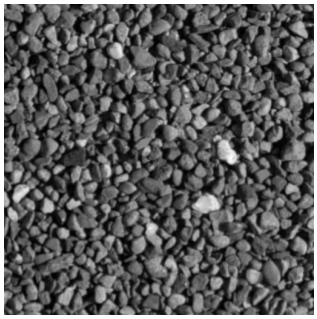


michael5

michael6



michael7



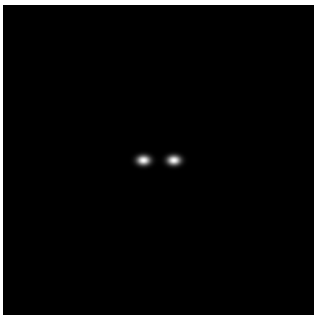
michael8

michael9

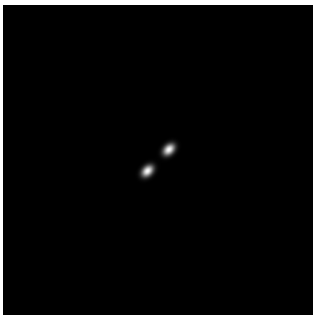
Appendix C

Filters

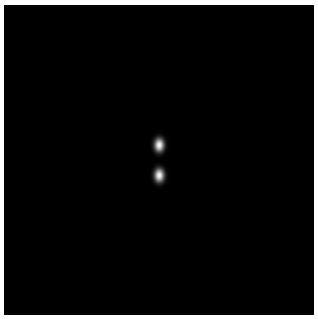
In the following the real part in frequency domain of all filters which are used in this work are shown.



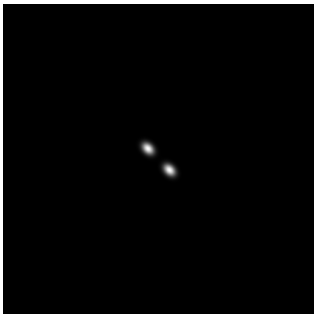
gaborF25A0



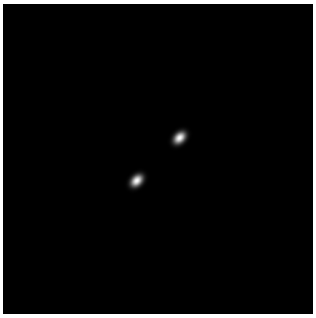
gaborF25A45



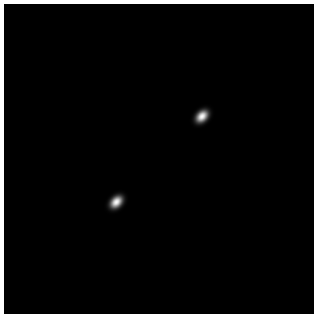
gaborF25A90



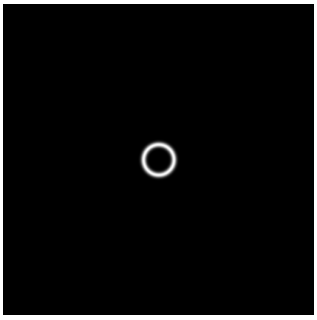
gaborF25A135



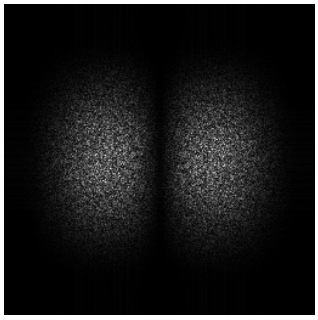
gaborF50A45



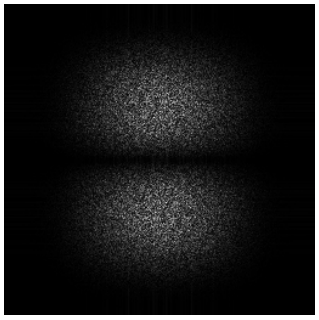
gaborF100A45



iso_gaborF25



L5E5



E5L5

Appendix D

Source Code

D.1 C-Programs

C-Programs which can be found in ~ceems4/cpp/.

variance.c

Computes the variance of the input image:

$$v = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} (p(i,j) - \bar{p})^2$$

Usage: variance image

The input must be a scope file of either float or integer format.

abs.c

Computes the mean absolute value of the input image:

$$a = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} |p(i,j)|$$

Usage: abs image

The input must be a scope file of either float or integer format.

vtau.c

Computes a sinusoidal prediction of the behaviour of texture features due to illumination tilt rotating from the surface filter power spectrum density (PSD).

Usage: vtau σ N A

σ	illumination slant angle
N	number of output points for one period
A	PSD of the surface times the PSD of the filter (scope file of either float or integer format)

Output: o_0 o_1 ... o_n a b c

o_i	sinusoidal prediction of texture features $o_i = a + b\,\cos(2\pi\, i/n) + c\,\sin(2\pi\, i/n)$
a, b, c	parameters of sinusoidal curve $a = 1/2\,\sin(\pi/180\,\sigma)^2 \sum_{i=1}^{N}\sum_{j=1}^{M}(i^2+j^2)\,A(i,j)$ $b = 1/2\,\sin(\pi/180\,\sigma)^2 \sum_{i=1}^{N}\sum_{j=1}^{M}(i^2-j^2)\,A(i,j)$ $c = 1/2\,\sin(\pi/180\,\sigma)^2 \sum_{i=1}^{N}\sum_{j=1}^{M}2\,i\,j\,A(i,j)$

best_fit_cosine.c

Calculates the parameters a,b and c of the function $a+b*cos(x)+c*sin(x)$ which fits the function given by the input values best. a is the mean, b and c are the coefficients of the fundamental oscillation. The input values must cover exactly one period.

Usage: best_fit_cosine p_0 p_1 ... p_n

p_i	points of the approximately sinusoidal curve

Output: o_0 o_1 ... o_n a b c

o_i	sinusoidal best fit curve $o_i = a + b\,\cos(2\pi\, i/n) + c\,\sin(2\pi\, i/n)$
a, b, c	parameters of best fit sinusoidal curve $a = 1/n \sum_{i=0}^{n} p_i$ $b = 2/n \sum_{i=0}^{n} \cos(2\pi\, i/n)\,(p_i - a)$ $c = 2/n \sum_{i=0}^{n} \sin(2\pi\, i/n)\,(p_i - a)$

best_fit_line.c

Calculates a best fit line of the approximately equal input values by calculating the average value.

Usage: best_fit_line p_0 p_1 ... p_n

p_i points of the approximately equal values

Output: o_0 o_1 ... o_{n+1}

o_i best fit line

best_fit_sqrt_cosine.c

Calculates the parameters a,b and c of the function $\sqrt{a + b * cos(x) + c * sin(x)}$ which fits the function given by the input values best. The input values must cover exactly one period.

Usage: best_fit_cosine p_0 p_1 ... p_n

p_i points of the approximately rooted sinusoidal curve

Output: o_0 o_1 ... o_n a_2 b_2 c_2

o_i rooted sinusoidal best fit curve

$$o_i = \sqrt{a_1 + b_1 \cos(2\pi\, i/n) + c_1 \sin(2\pi\, i/n)}$$

$$a_1 = 1/n \sum_{i=0}^{n} o_i^2$$

$$b_1 = 2/n \sum_{i=0}^{n} \cos(2\pi\, i/n)\, o_i^2$$

$$c_1 = 2/n \sum_{i=0}^{n} \sin(2\pi\, i/n)\, o_i^2$$

a_2, b_2, c_2 parameters of best fit sinusoidal curve

$$a_2 = 1/n \sum_{i=0}^{n} o_i$$

$$b_2 = 2/n \sum_{i=0}^{n} \cos(2\pi\, i/n)\, o_i$$

$$c_2 = 2/n \sum_{i=0}^{n} \sin(2\pi\, i/n)\, o_i$$

error_cosine.c

Calculates the absolute and relative error between an approximately sinusoidal curve and a sinusoidal curve. The absolute error is the mean of the squared difference of the two curves. The relative error is the absolute error divided by the amplitude of the oscillation.

Usage: error_cosine p_0 p_1 ... p_n q_0 q_1 ... q_n a b c

p_i	points of the approximately sinusoidal curve
q_i	points of the sinusoidal curve
a, b, c	parameters of the sinusoidal curve $(a_1 + b_1 \cos(2\pi\, i/n) + c_1 \sin(2\pi\, i/n))$

Output: M m

M	absolute measure of error $M = \frac{1}{N}\sqrt{\sum_{k=0}^{N-1}(f_k - f_{b_k})^2}$
m	relative measure of error $m = \frac{M}{\sqrt{b_b^2 + c_b^2}}$

error_line.c

Calculates the absolute and relative error of the approximately equal input values. The absolute error is the variance of the input values. The relative error is the variance divided by the average input value.

Usage: error_line p_0 p_1 ... p_n

p_i	points of the approximately equal values

Output: M m

M	variance
m	variance divided by the average input value

D.2 Shell Scripts

com_gabor

Filters an image with a complex Gabor filter and transforms the resulting image back into spatial domain. The input is the real and imaginary part of the image in frequency domain and the real and imaginary part of the Gabor filter in frequency domain. The output is the norm of the complex filtered image in spatial domain.

Usage: com_gabor image_real image_im gabor_real gabor_im result

```
# file paths
a=/depot/imaging/bin

# multiply with Gabor filters in frequency domain
$a/multimago $1 $3 real_even
$a/multimago $2 $3 imag_even
$a/multimago $1 $4 real_odd
$a/multimago $2 $4 imag_odd

# transform back into spatial domain
$a/swapshop real_even real_even real_even
$a/swapshop imag_even imag_even imag_even
$a/swapshop real_odd real_odd real_odd
$a/swapshop imag_odd imag_odd imag_odd
$a/ffti real_even imag_even spatial_real
$a/multim "-v-1.0" real_odd real_odd real_odd_inv
$a/ffti imag_odd real_odd spatial_imag

# compute norm of the filtered image
$a/multimago spatial_real spatial_real spatial_real
$a/multimago spatial_imag spatial_imag spatial_imag
$a/addimago spatial_real spatial_imag $5
$a/sqroot $5 $5 $5

rm real* imag* spatial*
```

shellscript_synth_surf

Compares the texture features which are derived from synthetic surfaces with features derived from illuminated images: First features are computed from the surface PSD and filter PSD using the sinusoidal prediction. Then features are computed by filtering their synthetic illuminated images with a real Gabor filter.

```
set -x # debug version

# file paths
a=/depot/imaging/bin
b=~/cpp

#init the file 'charts' which is read by Excel
echo \"`date`\" > charts #quotes are needed for Excel
echo paper/shellscript_new >> charts
```

```
#write header for file 'charts'
echo surface filter 0 10 20 30 40 50 60 70 80 90 100 110 120 130\
     140 150 160 170 180 a b c absolute relative >> charts

#Generating four synthetic surfaces
$a/greg2dfrac -b3.0 -n512 fractal
../Sand_ripples 64 0 2222 0.25
mv temp sand
$a/Ogil_dir 32 16 2222 0.125
mv temp ogil
../Malfin 32 4444 0.25
mv temp malfin

#creating Gabor filters in the frequency domain
for phase in 0 45 90 135
do
  ../gab4vinnie3-4 -f50 -a$phase -p1 -n512 gaborF25A${phase}
done

# loop over every surface
for surface in fractal sand ogil malfin
do

# empty feature arrays
  featureF25A0=""
  featureF25A45=""
  featureF25A90=""
  featureF25A135=""

# Compute the power spectrum of the surface
  $a/fft3 $surface fft_re fft_im
  $a/swapshop fft_re fft_re fft_re
  $a/swapshop fft_im fft_im fft_im

  $a/multimago fft_re fft_re fft_re
  $a/multimago fft_im fft_im fft_im
  $a/addimago fft_re fft_im power1

  # loop over different tilt angles
  for tilt in 180 170 160 150 140 130 120 110 100 90 80 70 60 50 40 30\
  20 10 0
  do

# Render surface using Lambertian reflection
    $a/lamb -s45 -t$tilt -o1 $surface rendered
    $a/divim -v240 rendered rendered rendered

# transform image into frequency domain
    $a/fft3 rendered fft_re fft_im
    $a/swapshop fft_re fft_re fft_re
    $a/swapshop fft_im fft_im fft_im

    #filter image with different gabor filters and compute features
    for dir in 0 45 90 135
    do
      ../real_gabor fft_re fft_im gaborF25A${dir} final
      eval featureF25A$dir=\"\$featureF25A$dir \`$b/variance final\`\"
    done

  done
```

```
  for dir in 0 45 90 135
  do

# Compute notional power spectrum of filtered surface
    $a/multimago power1 gaborF25A${dir} power2
    $a/multimago power2 gaborF25A${dir} power2

# Compute theoretical behaviour of features derived from the surface
    echo surface_$surface F25A$dir `$b/vtau 45 18 power2` 0 0 >> charts

# Compute features derived from the rendered images and compare them
# with features derived from the surface
    eval best_fit=\`$b/best_fit_cosine \$featureF25A$dir\`
    eval errorF25A$dir=\`$b/error_cosine \$featureF25A$dir $best_fit\`
    eval echo $surface F25A$dir \$featureF25A$dir 0 0 0 \$errorF25A$dir >> charts
    echo best_fit_$surface F25A$dir $best_fit 0 0 >> charts
  done

done

echo \"`date`\" >> charts

rm gabor* fft_* final rendered fractal sand ogil malfin power*
```

The following scripts only use the images of the surfaces *michael1* to *michael9*. The shellscripts using the other images are very similar; only the file format of the images is different.

gabor/shellscript_ms

Investigation into the effect of changing tilt direction on the feature output. For every surface the features of 19 images taken under different tilt angles are computed. Every image is filtered with different Gabor filters in the image domain, then transformed back into spatial domain. At least the mean of the variance of the filtered image is calculated as a feature of the image. The features are compared with the theoretical prediction $f(\tau) = a + b * cos(2\tau + \phi)$ (equation 5.2 on page 26), best fit curves are computed (equations 6.2 - 6.4 on page 39) and the sinusoidal measure of error (equations 6.5 and 6.6 on page 40) are calculated. All results are saved into the files results and charts. This files can be read by Excel to produce output figures.

```
set -x # debug version

# file paths
a=/depot/imaging/bin
b=~/cpp

#init the files 'results' and 'charts' which are read by Excel
echo \"`date`\" > results #quotes are needed for Excel
echo \"`date`\" > charts
echo paper/shellscript_paper_new >> results
echo paper/shellscript_paper_new >> charts

#write header for file 'results'
echo surface F25A0comabs  F25A0comrel F25A45comabs F25A45comrel\
     F25A90comabs F25A90comrel F25A135comabs F25A135comrel\
```

```
     F50A45comabs F50A45comrel F25A45realabs\
     F25A45realrel isotropicabs isotropicrel >> results

#write header for file 'charts'
echo surface filter 0 10 20 30 40 50 60 70 80 90 100 110 120 130\
     140 150 160 170 180 a b c absolute relative >> charts

#creating Gabor filters in the frequency domain
for phase in 0 45 90 135
do
  ../gab4vinnie3-4 -f25 -a$phase -p1 -n512 gaborF25A${phase}_re
  ../gab4vinnie3-4 -f25 -a$phase -p-1 -n512 gaborF25A${phase}_im
done

../gab4vinnie3-4 -f50 -a45 -p1 -n512 gaborF50A45_re
../gab4vinnie3-4 -f50 -a45 -p-1 -n512 gaborF50A45_im

# loop over every surface
for surface in 1 2 3 4 5 6 7 8 9
do

# empty feature arrays
  featurecomF25A0=""
  featurecomF25A45=""
  featurecomF25A90=""
  featurecomF25A135=""
  featurecomF50A45=""
  featurerealF25A45=""
  featureisoF25=""

  # loop over different tilt angles
  for tilt in 000 010 020 030 040 050 060 070 080 090 100 110 120 130\
      140 150 160 170 180
  do

    # prepare image for computing
    cp ~/images/michael$surface/michael$surface.$tilt.tif.gz .
    gunzip michael$surface.$tilt.tif.gz
    $a/tif2scope -x1792 -y1200 michael$surface.$tilt.tif dummy image
    rm michael$surface.$tilt.tif

    $a/cutim -X512 -Y512 -x640 -y344 image image

    # transform image into frequency domain
    $a/fft3 image fft_re fft_im
    $a/swapshop fft_re fft_re fft_re
    $a/swapshop fft_im fft_im fft_im

    #filter image with different gabor filters and compute features
    ../com_gabor fft_re fft_im gaborF25A0_re gaborF25A0_im final
    featurecomF25A0="$featurecomF25A0 `$b/variance final`"

    ../com_gabor fft_re fft_im gaborF25A45_re gaborF25A45_im final
    featurecomF25A45="$featurecomF25A45 `$b/variance final`"

    ../com_gabor fft_re fft_im gaborF25A90_re gaborF25A90_im final
    featurecomF25A90="$featurecomF25A90 `$b/variance final`"

    ../com_gabor fft_re fft_im gaborF25A135_re gaborF25A135_im final
    featurecomF25A135="$featurecomF25A135 `$b/variance final`"

    ../com_gabor fft_re fft_im gaborF50A45_re gaborF50A45_im final
```

```
    featurecomF50A45="$featurecomF50A45 `$b/variance final`"

    ../real_gabor fft_re fft_im gaborF25A45_re final
    featurerealF25A45="$featurerealF25A45 `$b/variance final`"

    ../iso_gabor fft_re fft_im 25 final
    featureisoF25="$featureisoF25 `$b/variance final`"

  done

  # compare results with theoretical prediction and write results into
  # files
  best_fit=`$b/best_fit_cosine $featurecomF25A0`
  errorcomF25A0=`$b/error_cosine $featurecomF25A0 $best_fit`
  echo michael$surface comF25A0 $featurecomF25A0 0 0 0 $errorcomF25A0 >> charts
  echo best_fit_michael$surface comF25A0 $best_fit 0 0 >> charts

  best_fit=`$b/best_fit_cosine $featurecomF25A45`
  errorcomF25A45=`$b/error_cosine $featurecomF25A45 $best_fit`
  echo michael$surface comF25A45 $featurecomF25A45 0 0 0 $errorcomF25A45 >> charts
  echo best_fit_michael$surface comF25A45 $best_fit 0 0 >> charts

  best_fit=`$b/best_fit_cosine $featurecomF25A90`
  errorcomF25A90=`$b/error_cosine $featurecomF25A90 $best_fit`
  echo michael$surface comF25A90 $featurecomF25A90 0 0 0 $errorcomF25A90 >> charts
  echo best_fit_michael$surface comF25A90 $best_fit 0 0 >> charts

  best_fit=`$b/best_fit_cosine $featurecomF25A135`
  errorcomF25A135=`$b/error_cosine $featurecomF25A135 $best_fit`
  echo michael$surface comF25A135 $featurecomF25A135 0 0 0 $errorcomF25A135 >> charts
  echo best_fit_michael$surface comF25A135 $best_fit 0 0 >> charts

  best_fit=`$b/best_fit_cosine $featurecomF50A45`
  errorcomF50A45=`$b/error_cosine $featurecomF50A45 $best_fit`
  echo michael$surface comF50A45 $featurecomF50A45 0 0 0 $errorcomF50A45 >> charts
  echo best_fit_michael$surface comF50A45 $best_fit 0 0 >> charts

  best_fit=`$b/best_fit_cosine $featurerealF25A45`
  errorrealF25A45=`$b/error_cosine $featurerealF25A45 $best_fit`
  echo michael$surface realF25A45 $featurerealF25A45 0 0 0 $errorrealF25A45 >> charts
  echo best_fit_michael$surface realF25A45 $best_fit 0 0 >> charts

  best_fit=`$b/best_fit_line $featureisoF25`
  errorisoF25=`$b/error_line $featureisoF25 $best_fit`
  echo michael$surface isoF25 $featureisoF25 0 0 0 $errorisoF25 >> charts
  echo best_fit_michael$surface isoF25 $best_fit 0 0 0 0 >> charts

  echo michael$surface $filter $errorcomF25A0 $errorcomF25A45 $errorcomF25A90\
  $errorcomF25A135 $errorcomF50A45 $errorrealF25A45 $errorisoF25\
  >> results

done

echo \"`date`\" >> results
echo \"`date`\" >> charts

rm gabor* fft_* final
```

isotropic_gabor/shellscript_ms

Investigation into the effect of changing tilt direction on the feature output. The setup is similar to the **gabor/shellscript_ms**. Instead of directional Gabor filters, isotropic Gabor filters are used; instead of the sinusoidal measure of error, the relative variance is computed.

```
set -x # debug version

# file paths
a=/depot/imaging/bin
b=~/cpp

#init the files 'results' and 'charts' which are read by Excel
echo \"`date`\" > results #quotes are needed for Excel
echo \"`date`\" > charts
echo paper/shellscript_paper_new >> results
echo paper/shellscript_paper_new >> charts

#write header for file 'results'
echo surface isotropicabs isotropicrel >> results

#write header for file 'charts'
echo surface filter 0 10 20 30 40 50 60 70 80 90 100 110 120 130\
     140 150 160 170 180 a b c absolute relative >> charts

# loop over every surface
for surface in 1 2 3 4 5 6 7 8 9
do

# empty feature array
  featureisoF25=""

  # loop over different tilt angles
  for tilt in 000 010 020 030 040 050 060 070 080 090 100 110 120 130\
      140 150 160 170 180
  do

    # prepare image for computing
    cp ~/images/michael$surface/michael$surface.$tilt.tif.gz .
    gunzip michael$surface.$tilt.tif.gz
    $a/tif2scope -x1792 -y1200 michael$surface.$tilt.tif dummy image
    rm michael$surface.$tilt.tif

    $a/cutim -X512 -Y512 -x640 -y344 image image

    # transform image into frequency domain
    $a/fft3 image fft_re fft_im
    $a/swapshop fft_re fft_re fft_re
    $a/swapshop fft_im fft_im fft_im

    ../iso_gabor fft_re fft_im 25 final
    featureisoF25="$featureisoF25 `$b/variance final`"

  done

  # compare results with theoretical prediction and write results into
  # files

  best_fit=`$b/best_fit_line $featureisoF25`
  errorisoF25=`$b/error_line $featureisoF25 $best_fit`
```

```
  echo michael$surface isoF25 $featureisoF25 0 0 0 $errorisoF25 >> charts
  echo best_fit_michael$surface isoF25 $best_fit 0 0 0 0 >> charts

  echo michael$surface $filter $errorisoF25\
  >> results

done

echo \"`date`\" >> results
echo \"`date`\" >> charts

rm fft_* final
```

shellscript_laws_ms

Every texture is filtered with two Laws filters (L5E5 and E5L5) under different illumination tilt angles. The results are compared with the theoretical predictions and the sinusoidal measure of error is computed.

```
set -x # debug version

# file paths
a=/depot/imaging/bin
b=~/cpp

#init the file 'charts' which are read by Excel
echo \"`date`\" > charts
echo laws/shellscript_new >> charts

#write header for file 'charts'
echo surface filter 0 10 20 30 40 50 60 70 80 90 100 110 120 130\
     140 150 160 170 180 a b c absolute relative >> charts

# loop over every surfe
for surface in 1 2 3 4 5 6 7 8 9
do

# empty feature arrays
  feature1=""
  feature2=""

  # loop over different tilt angles
  for tilt in 000 010 020 030 040 050 060 070 080 090 100 110 120 130\
      140 150 160 170 180
  do

    # prepare image for computing
    cp ~/images/michael$surface/michael$surface.$tilt.tif.gz .
    gunzip michael$surface.$tilt.tif.gz
    ../bmpbust -x1792 -y1200 michael$surface.$tilt.tif image image
    rm michael$surface.$tilt.tif
    $a/cutim -X512 -Y512 -x640 -y344 image image

    # filter images with Laws filters using convolution in spatial
    # domain and compute features
    $a/mask -f -m $a/masks/L5E5 image final
    feature1="$feature1 `$b/variance final`"
    $a/mask -f -m $a/masks/E5L5 image final
    feature2="$feature2 `$b/variance final`"
```

```
  done

  # compare results with theoretical prediction and write results into
  # 'charts'
  best_fit=`$b/best_fit_cosine $feature1`
  error=`$b/error_cosine $feature1 $best_fit`
  echo michael$surface L5E5 $feature1 0 0 0 $error >> charts
  echo best_fit_michael$surface L5E5 $best_fit 0 0 >> charts

  best_fit=`$b/best_fit_cosine $feature2`
  error=`$b/error_cosine $feature2 $best_fit`
  echo michael$surface E5L5 $feature2 0 0 0 $error >> charts
  echo best_fit_michael$surface E5L5 $best_fit 0 0 >> charts

done

echo \"`date`\" >> charts

rm final image
```

shellscript_abs_ms

Compares the behaviour of image features using the variance of the image (the image energy) and the mean absolute value. The results are compared with the best fit curves $f(\tau) = a + b * cos(2\tau + \phi)$ and $f(\tau) = \sqrt{a + b * cos(2\tau + \phi)}$.

```
set -x # debug version

# file paths
a=/depot/imaging/bin
b=~/cpp
c=~gmg/texture

#init the files 'results' and 'charts' which are read by Excel
echo \"`date`\" > results
echo \"`date`\" > charts
echo abs/shellscript_abs_gmg >> results
echo abs/shellscript_abs_gmg >> charts

#write header for file 'results'
echo surface F25A45realabs F25A45realrel F25A45realabs F25A45realrel \
F25A45realabs F25A45realrel >> results
echo surface var var abs_cosine abs_cosine abs_sqrt_cosine \
abs_sqrt_cosine >> results

#write header for file 'charts'
echo surface filter 0 10 20 30 40 50 60 70 80 90 100 110 120 130\
     140 150 160 170 180 a b c absolute relative >> charts

#creating Gabor filter in the frequency domain
../gab4vinnie3-4 -f25 -a45 -p1 -n512 gabor

# loop over every surface
for surface in 1 2 3 4 5 6 7 8 9
do

# empty feature arrays
  feature_abs=""
```

```
  feature_var=""

  # loop over different tilt angles
  for tilt in 000 010 020 030 040 050 060 070 080 090 100 110 120 130\
      140 150 160 170 180
  do

    # prepare image for computing
    cp ~/images/michael$surface/michael$surface.$tilt.tif.gz .
    gunzip michael$surface.$tilt.tif.gz
    $a/tif2scope -x1792 -y1200 michael$surface.$tilt.tif dummy image
    rm michael$surface.$tilt.tif

    $a/cutim -X512 -Y512 -x640 -y344 image image

    # transform image into frequency domain
    $a/fft3 image fft_re fft_im
    $a/swapshop fft_re fft_re fft_re
    $a/swapshop fft_im fft_im fft_im

    #filter image with Gabor filter and compute features using the
    #image variance and the absolute pixel value
    ../real_gabor fft_re fft_im gabor final
    feature_abs="$feature_abs `$b/abs final`"
    feature_var="$feature_var `$b/variance final`"

  done

  # compare variance feature with sinosoidal best fit curve
  best_fit=`$b/best_fit_cosine $feature_var`
  error_var=`$b/error_cosine $feature_var $best_fit`
  echo michael$surface var $feature_var 0 0 0 $error_var >> charts
  echo best_fit_michael$surface var $best_fit 0 0 >> charts

  # compare absolute value feature with sinosoidal best fit curve
  best_fit=`$b/best_fit_cosine $feature_abs`
  error_abs_cosine=`$b/error_cosine $feature_abs $best_fit`
  echo michael$surface abs_cosine $feature_abs 0 0 0 $error_abs_cosine >> charts
  echo best_fit_michael$surface abs_cosine $best_fit 0 0 >> charts

  # compare absolute value feature with rooted sinosoidal best fit curve
  best_fit=`$b/best_fit_sqrt_cosine $feature_abs`
  error_abs_sqrt_cosine=`$b/error_cosine $feature_abs $best_fit`
  echo michael$surface abs_sqrt_cosine $feature_abs 0 0 0 $error_abs_sqrt_cosine >> charts
  echo best_fit_michael$surface abs_sqrt_cosine $best_fit 0 0 >> charts

  echo michael$surface $error_var $error_abs_cosine $error_abs_sqrt_cosine >> results

done

echo \"`date`\" >> results
echo \"`date`\" >> charts

rm gabor* fft_* final
```

Appendix E

Proof: The scatter plot of two sinusoidal curves, which oscillate about an average value and have different average values, amplitudes and phases but the same period, is always an ellipse.

$$\begin{aligned} x &= a_1 + b_1 \cos(2\tau + \phi_1) \iff 2\tau = \arccos\left(\frac{x - a_1}{b_1}\right) - \phi_1 \\ y &= b_1 + b_2 \cos(2\tau + \phi_2) \end{aligned}$$

$$\Longrightarrow y = a_2 + b_2 \cos\left[\arccos\left(\frac{x - a_1}{b_1}\right) - \phi_1 + \phi_2\right]$$

Using $\cos(\alpha + \beta) = \cos(\alpha)\cos(\beta) - \sin(\alpha)\sin(\beta)$ we get

$$y = a_2 + b_2 \left\{\left(\frac{x - a_1}{b_1}\right)\cos(\phi_1 - \phi_2) - \sin\left[\arccos\left(\frac{x - a_1}{b_1}\right)\right]\sin(\phi_2 - \phi_1)\right\}$$

Using $\sin[\arccos(x)] = \sqrt{1 - x^2}$ and the abbreviations $c_1 = \frac{b_2}{b_1}\cos(\phi_2 - \phi_1)$ and $c_2 = b_2 \sin(\phi_2 - \phi_1)$ we get

$$y = a_2 + c_1 x - c_1 a_1 - c_2 \sqrt{1 - \left(\frac{x - a_1}{b_1}\right)^2}$$

$$\Longrightarrow \sqrt{1 - \left(\frac{x - a_1}{b_1}\right)^2} = \frac{a_2}{c_2} + \frac{c_1}{c_2}x - \frac{c_1}{c_2}a_1 - \frac{1}{c_2}y$$

Using the abbreviation $c_3 = \frac{a_2}{c_2} - \frac{c_1}{c_2}a_1$ we get

$$1 - \left(\frac{x - a_1}{b_1}\right)^2 = \left(c_3 + \frac{c_1}{c_2}x - \frac{1}{c_2}y\right)^2$$

$$\begin{aligned}\Longleftrightarrow\ 0 &= \left[\left(\frac{a_1}{b_1}\right)^2 + c_3^2 - 1\right] - 2\frac{c_3}{c_2}y + \left[2\frac{c_1}{c_2}c_3 - 2\frac{a_1}{b_1^2}\right]x + \frac{1}{c_2^2}y^2 \\ &+ \left[\left(\frac{c_1}{c_2}\right)^2 + \frac{1}{b_1^2}\right]x^2 - 2\frac{c_1}{c_2^2}xy\end{aligned}$$

This is equivalent to the general equation of conics in [11]:

$$0 = ax^2 + 2bxy + cy^2 + 2dx + 2ey + f$$

$$\begin{aligned}a &= \frac{1}{b_1^2}\left[\frac{1}{\tan^2(\phi_2 - \phi_1)} + 1\right] = \frac{1}{b_1^2 sin^2(\phi_2 - \phi_1)} \\ b &= \frac{1}{b_1 b_2 \sin(\phi_2 - \phi_1)\tan(\phi_2 - \phi_1)} \\ c &= \frac{1}{b_2^2 \sin^2(\phi_2 - \phi_1)} \\ d &= \frac{a_2}{b_1 b_2 \tan(\phi_2 - \phi_1)\sin(\phi_2 - \phi_1)} - \frac{a_1}{b_1^2}\left[\frac{1}{\tan^2(\phi_2 - \phi_1)} + 1\right] \\ &= \frac{1}{b_1 \sin(\phi_2 - \phi_1)}\left[\frac{a_2}{b_2 \tan(\phi_2 - \phi_1)} - \frac{a_1}{b_1 \sin(\phi_2 - \phi_1)}\right] \\ e &= \frac{1}{b_2 \sin(\phi_2 - \phi_1)}\left[\frac{a_1}{b_1 \tan(\phi_2 - \phi_1)} - \frac{a_2}{b_2 \sin(\phi_2 - \phi_1)}\right] \\ f &= \left(\frac{a_1}{b_1}\right)^2 - 1 + \frac{a_2^2 - 2a_2\frac{b_2}{b_1}\cos(\phi_2 - \phi_1) + \left(\frac{b_2}{b_1}\right)^2 a_1^2 \cos^2(\phi_2 - \phi_1)}{b_2^2 \sin^2(\phi_2 - \phi_1)}\end{aligned}$$

The determinant

$$B = \begin{vmatrix} a & b \\ b & c \end{vmatrix} = \frac{1}{b_1^2 b_2^2 \sin^4(\phi_2 - \phi_1)}[1 - \cos^2(\phi_2 - \phi_1)]$$

is always positive. The determinant

$$A = \begin{vmatrix} a & b & d \\ b & c & e \\ d & e & f \end{vmatrix}$$

is not zero. In this case the conic figure is always an ellipse.

Bibliography

[1] **M. J. Chantler, G. McGunnigle**, *The response of texture features to illuminant rotation*, 15th International Converence on Pattern Recognition, Vol. 3 pp. 943 - 946, September 2000

[2] **G. McGunnigle**, *The Classification of Textured Surfaces Under Varying Illuminant Directions*, PhD Theses, Department of Computing and Electrical Engineering, Heriot Watt University, April 1998

[3] **Michael J. Chantler**, *The effect of variation in illuminant direction on texture classification*, PhD Theses, Department of Computing and Electrical Engineering, Heriot Watt University, August 1994

[4] **G. Healey and L. Wang**, *Illumination-Invariant Recognition of Texture in Color Images*, Journal of the Optical Society of America A, pp. 1877-1883, September 1995

[5] **Todd R. Reed and J. M. Hans du Buf**, *A Review of Recent Texture Segmentation and Feature Extraction Techniques*, CVGIP: Image Understanding, V57, No.3, pp359–372, May 1993

[6] **Trygve Randen and John H. Husoy**, *Filtering for Texture Classification: A Comparative Study*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 21, No. 4, April 1999

[7] **Thomas P. Weldon and William E. Higgins**, *Integrated Approach to Texture Segmentation Using Multiple Gabor Filters*, International Conference on Image Processing, vol. 3 , pp. 955 -958, 1996

[8] **Christian V. Sinn**, *An investigation into the importance of phase in textured images*, Diploma Theses, Department of Computing and Electrical Engineering, Heriot Watt University, July 2000

[9] **K.I. Laws**, *Texture Energy Measures*, Proceedings in Image Understanding Workshop, No. 1979 pp. 47-51.

[10] **Ronald N. Bracewell**, *The Fourier Transform and its Applications 3rd ed.*, McGraw-Hill International Editions, 2000

[11] **John W. Harris and Horst Stocker**, *Handbook of Mathematics and Computational Science*, Springer Verlag New York, 1998, pp. 393