Segmenting Textured 3D Surfaces Using the Space/Frequency Representation

John Krumm and Steven A. Shafer

CMU-RI-TR-93-14

The Robotics Institute
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213

April 1993

This research was sponsored by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U. S. Air Force, Wright-Patterson AFB, OH 45433-6543 under Contract F33615-90-C-1465, Arpa Order No. 7597. This first author was supported by NASA under the Graduate Student Researchers Program, Grant No. NGT-50423.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

Table of Contents

Abstract

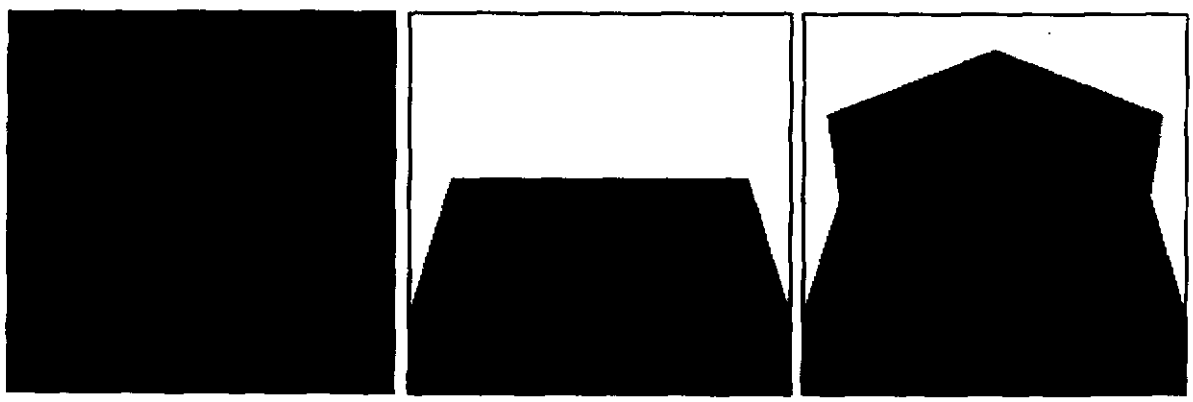
Segmenting 3D textured surfaces is critical for general image understanding. Unfortunately, current efforts at automatically understanding image texture are based on assumptions that make this goal impossible. Texture segmentation research assumes that the textures are flat and viewed from the front, while shape-from-texture work assumes that the textures have already been segmented. This deadlock means that none of these algorithms can be successfully applied to images of 3D textured surfaces.

We have developed an algorithm that can segment an image containing nonfrontally viewed, planar, periodic textures. We use the spectrogram to compute local surface normals from many different regions of the image. This algorithm does not require unreliable image feature detection. Based on these surface normals, we compute a "frontalized" version of the local power spectrum which shows what the region's power spectrum would look like if viewed from the front. If neighboring regions have similar frontalized power spectra, they are merged. To our knowledge, this is the first program that can segment 3D textured surfaces by explicitly accounting for shape effects.

1. Introduction

Automatic recognition and understanding of image texture is critical for machine understanding of general images. Almost every scene, either natural or man-made, contains some texture. In fact, everything is textured at some level of magnification. One reason for the importance of texture is that it can tell us much about a scene. Julesz[24] and Gibson[15] did early work that shows how humans use texture to segment images and to estimate surface normals, respectively. Both of these capabilities have been reproduced by computers. Unfortunately, many computer vision algorithms give disastrous results on texture. For instance, segmentation algorithms are usually based on an assumption of smoothly varying gray levels, which is not true for texture. Stereo matching often fails on repetitive texture. Thus, to avoid errors with other algorithms and to exploit what we can from texture, we need to explicitly account for it.

Past efforts at automatically understanding texture in images are inherently insufficient because of their assumptions about the underlying textured surfaces. The current state of the art is advancing on two distinct, mutually exclusive fronts (see Figure 1). One effort, corresponding to Julesz' theories, is aimed at segmenting images into regions of similar texture, where it is assumed the textures are flat and viewed frontally. Differences or similarities in some characteristic of the image texture are used to find texture boundaries or to group regions of similar texture. The other effort, based on Gibson's observations, is targeted at finding the shape of uniformly textured objects, assuming the objects themselves have been segmented. Here, changes in otherwise uniform texture are attributed to 3D effects and used to compute surface normals. The two efforts have conflicting assumptions that prevent their ever being applied to the same image. If the textures are not flat and viewed frontally, the image can't be segmented. If the texture is not segmented, its shape can't be found.



Traditional texture segmentation requires a flat, frontal view.

Traditional shape-from-texture must have only one texture in the image.

We solve the combined problem.

Figure 1: Combining old texture problems into a new one

One way around the problem of segmenting textures that are changing due to three-dimensional effects is to loosen the thresholds on the segmentation algorithm such that it allows for the variation. This is the approach taken by Voorhees and Poggio:

> ... if we did not ignore small differences in [texture] attribute values, a graded texture gradient, perhaps formed by the projection of a curved surface, would yield undesirably significant texture boundaries across its face.[44]

But, it is just these small differences that can be used to compute surface orientation, so it is undesirable to ignore them if the goal is to understand as much from the texture as possible.

Another problem with some texture analysis programs is their need for finding texture elements. Feature-finding by computer is never very reliable, and this is a problem for texture programs that rely on it. Blake and Marinos said in 1990:

> Our greatest practical problems arise from isolating independent oriented [texture] elements from an image.[5]

And Aloimonos said in 1988:

> There is no known algorithm that can successfully detect texels from a natural image.[2]

Not only are texture elements hard to find, it is not even clear what one is. Although Julesz has made great progress in differentiating between preattentive texture elements (textons) and focal-attentive texture elements, the distinction is still not fully understood. In addition, humans can also preattentively segment at least some random, gray-level textures as in Figure 2, for which texture elements do not exist. Thus, for machine understanding of general textures, it makes sense to develop methods that don't rely on finding texture elements.

Some years after the important observations of Julesz and Gibson, researchers are trying to explain human abilities in texture understanding in terms of local spatial frequency filtering. There has also been success in the computer vision community at using local frequency representations to do texture segmentation and shape-from-texture. These are attractive theories, because they postulate similar mechanisms for both tasks, because they admit to a quantitative formulation, and because they do not require feature detection.

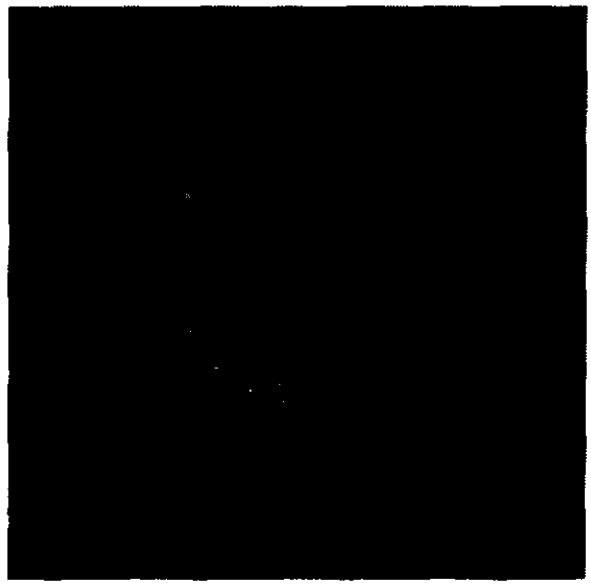


Figure 2: These random textures can be preattentively segmented. (Both textures have the same mean and variance in gray level. They are from Brodatz[7], D2 fieldstone and D12 bark of tree.)

We have developed a texture understanding program based on local spatial frequency that both overcomes the segmentation/shape deadlock and does not rely on finding texels. It is shown pictorially in Figure 3. Given an image with multiple, nonfrontal textures, our program can segment the texture and compute surface normals. We do this by computing 2D Fourier power spectra over small square patches in the image. These spectra show the local spatial frequency content of each part of the image. Our program works exclusively with the local spectra, so it does not ever require finding texture elements. The local spectra of distinct textures are different, so we can use this for segmentation. We show that the local spectra of similar textures are approximately equal to within an affine transformation that depends on the underlying surface normal. Our program works by growing hypotheses about various image regions. Each hypothesis covers a certain part of the image, and they each contain an estimate of what that region's frequency content would be if viewed frontally. This frontal view is based on a local estimate of the surface normal. Hypotheses with similar frontally-viewed frequency content are merged. To our knowledge, this is the first program that can segment nonfrontal textures by explicitly accounting for surface normals.

Using power spectra to analyze texture is effective, because uniform texture usually exhibits coherence in spatial frequency. It is important to use *local* spectra, however, to avoid Fourier transforming a region that contains a significant change in frequency. Such a change could be due to a texture boundary or due to the perspective effects of a nonfrontal surface.

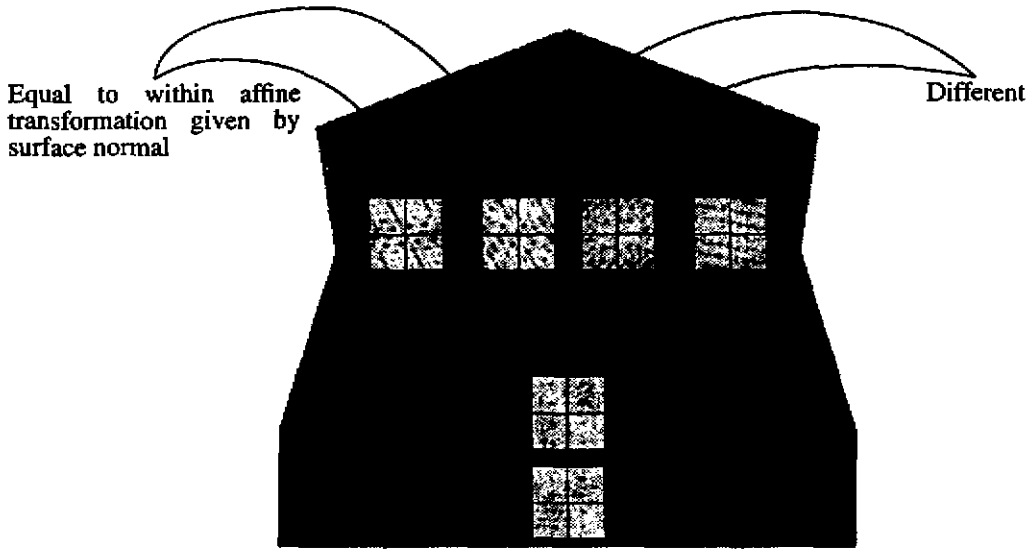


Figure 3: Local Fourier power spectra are used for segmentation and shape-from-texture.

2. The Space/Frequency Representation

Signals are traditionally analyzed in either the space (time) or frequency domain, but this dichotomy inadequate for texture segmentation. An example is shown in Figure 4. The distinct parts of this signal, *i.e.* the low frequency parts on the outside and the high frequency part in the middle, are characterized by their frequency. But, the power spectrum of the signal (with u as the frequency variable) shows only that the constituent frequencies exist somewhere in the signal, not where they are. We need a representation that shows both the spatial and frequency characteristics simultaneously. This "space/frequency" representation for a 1D signal is a 2D function that shows the instantaneous frequency distribution of every part of the signal. It is like having a little power spectrum plotted vertically at every point along the spatial axis. For image analysis, the input signal is 2D, and the resulting space/frequency representation is 4D (two spatial and two frequency variables).

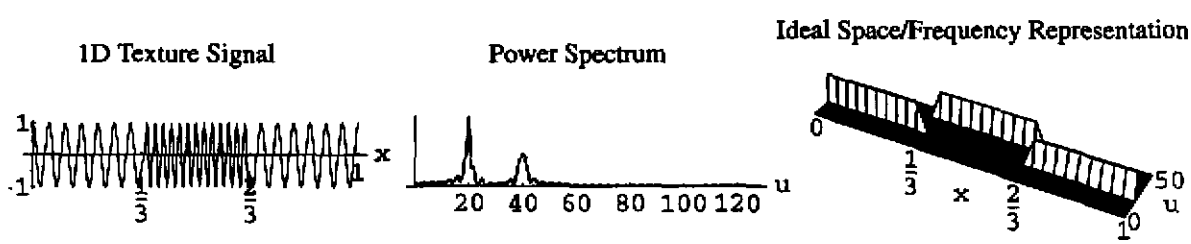


Figure 4: A signal, its power spectrum, and its space/frequency representation

The space/frequency representation shown in Figure 4 is ideal, and it cannot be computed by any commonly used techniques. We use the image spectrogram as our instantiation of the representation. For each point in the image, we extract a square block of surrounding pixels and multiply this block of intensities by a window function that falls off at the block's edges. We compute the two-dimensional Fourier transform of this product and take the squared magnitude as the local frequency representation, giving the local power spectrum. This is the image spectrogram $S(x, y, u, v)$, defined as

$$S(x, y, u, v) = \left| \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} w(x', y') f(x' - x, y' - y) e^{-j2\pi(ux' + vy')} dx' dy' \right|^2 \quad (1)$$

where $f(x, y)$ is the image and $w(x, y)$ is the window function. The frequency variables are (u, v), measured in cycles per pixel. This is what we used to compute the light-colored blocks in Figure 3.

Our particular window function is the "Blackman-Harris minimum 4-sample" window, recommended by experts[17][10] for Fourier analysis. Its equation is

$$w(l) = w_0 - w_1 \cos\left(\frac{2\pi}{L} l\right) + w_2 \cos\left(\frac{4\pi}{L} l\right) - w_3 \cos\left(\frac{6\pi}{L} l\right) \quad (2)$$

where L is the radius of the window, $0 \le l \le L$, and $l = \sqrt{x^2 + y^2}$. The coefficients are $(w_0, w_1, w_2, w_3) = (0.35875, 0.48829, 0.14128, 0.01168)$. This function is plotted in Figure 5.

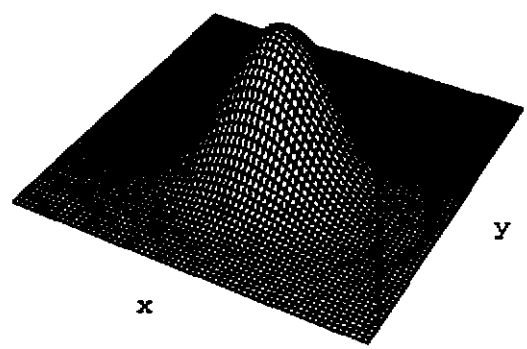


Figure 5: Blackman-Harris minimum 4-sample window function

For our analysis, we let $L = 64$. Any choice of window size is a compromise. A large window gives better frequency resolution for frontal textures. But when the texture is changing due to 3D effects, a large window will cover a larger variation in frequency. This causes smearing in the Fourier transform. A large window will also more likely contain a texture boundary, which makes it useless for both shape and segmentation.

Thinking in terms of basis functions is a good way to compare the spectrogram to other methods of computing the space/frequency representation. The real distinction between many of these methods is their basis functions. In each of these transforms, the basis functions are convolved with the image data, meaning they define what signal components the transform emphasizes. For instance, the basis functions of the spectrogram are complex sinusoids modulated by the window function $w(x, y)$. In Figure 6a we show a sampling of the basis functions from our spectrogram. They are sinusoids modulated by the Blackman-Harris window. Figure 6b shows some of the basis functions of a variable window spectrogram, where the window size is a constant multiple of the sinusoid's wavelength, giving smaller windows for higher frequencies. These smaller windows mean the high frequencies in the space/frequency representation are less likely to be corrupted by the window overlapping into two or more distinct regions (*e.g.* textures) of the signal. For our analysis, however, the higher frequencies are usually just overtones of the lower frequencies, so they usually have the same extent. Also, we look at all frequencies simultaneously, so a spectrum with only the lower frequencies corrupted is no better than a spectrum with all the frequencies corrupted. Therefore, by using constant-sized windows, we gain the advantage of higher frequency resolution at high frequencies (because of the larger windows) over the variable window spectrogram and other techniques.

Figure 6c shows some of the Gaussian-modulated sinusoids (an example of wavelets) used by Super and Bovik[42] for their work in shape-from-texture. These differ from the variable window spectrogram in that they are normalized to have equal energy. The important difference between their space/frequency representation and ours is that we compute a dense sampling in frequency, using about 2000 filters at each pixel, while they use only 72 for images the same size as ours. We find the dense sampling makes it easier to track small frequency shifts in the typically "peaky" Fourier transforms of periodic texture.

Figure 6d shows the filters used by Malik and Perona[30] for their work in modeling preattentive, frontal texture segmentation. These are not modulated sinusoids like the rest, but linear combinations of two or three Gaussians, meant to approximate the physiological mechanisms of early vision. They use 96 different filters and process their outputs nonlinearly. Their filters' sparse sampling and small size would give inadequate resolution in space and frequency for detecting small frequency shifts due to shape effects.

In summary, we chose the spectrogram because it gives an intuitive-looking picture, provides a dense sampling in space and frequency, and comes with the well-developed theory of Fourier transforms. We are not trying to mimic biological vision mechanisms, so we are free to choose the method that is best for machine implementation. The method of computing the representation is really only important at the algorithmic level of our development. The basic theory of projecting frequencies, which we cover in the next section, applies regardless of the particular representation.

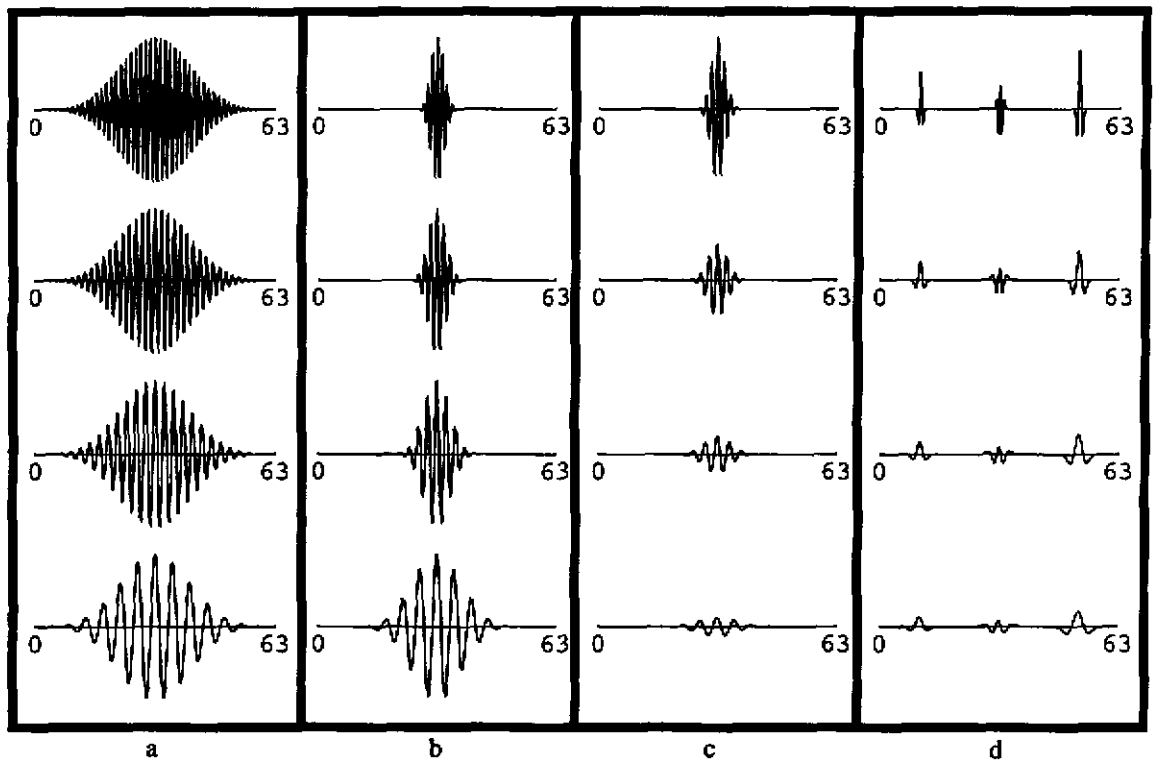


Figure 6: Space/frequency basis functions

a) Constant-sized windowed sinusoids that we use (spectrogram)
b) Window size a constant multiple of wavelength (variable window spectrogram)
c) Gabor functions used by Super & Bovik[42] for shape-from-texture (wavelets)
d) Linear combinations of Gaussians used by Malik & Perona[30] for texture segmentation

3. Periodic Texture in 3D

This section contains a derivation of the connection between the surface normal of a periodically textured surface and the local frequency of a projected sinusoid in an image. This is important because it relates a physical characteristic of a 3D scene to the measurable behavior of the projected frequencies in an image. We show how the local spatial frequencies in the image are approximately related by an affine transformation to the frontal texture's frequency. The affine parameters are functions of known camera parameters and the unknown depth and surface normal of the texture. From this we show that the frequencies of two image patches are also related by an affine transform. If we assume the two patches come from the same plane, then the depth variable drops out, leaving the surface normal as the only unknown. We exploit this fact in our shape-from-texture algorithm in Section 4.

3.1. Coordinate Frames

Figure 7 shows the coordinate frames used in the derivation. The camera's pinhole is at the origin of the (X, Y, Z) frame. This serves as the world coordinate frame, and points defined in it will be referred to with upper-case (X, Y, Z). The $-Z$ axis is coincident with the camera's optical axis and points into the scene being imaged. The image plane is the (x, y) frame with its origin on the optical axis at a distance d behind the pinhole. It is parallel to the XY plane.

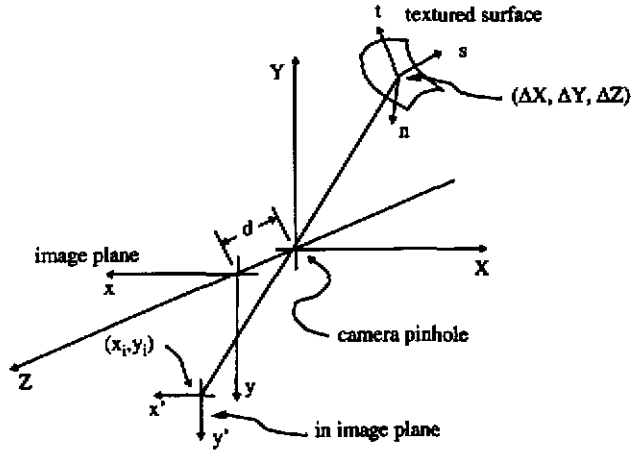


Figure 7: Coordinate frames used in derivation

We imagine that each point on the locally planar textured surface has its own coordinate frame (s, t, n), with the n axis coincident with the surface normal. The surface normal is defined with the gradient space variables (p, q), thus the unit vector along the n axis is $\hat{n} = \frac{1}{r}(p, q, 1)$, with $r = \sqrt{p^2 + q^2 + 1}$, in the world frame. The origin of this surface frame is $(\Delta X, \Delta Y, \Delta Z)$ with respect to the world frame.

The 4x4 homogeneous transformation matrix that locates and orients the surface frame with respect to the world frame is

$$\begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} \frac{p^2 + rq^2}{p^2 + q^2} & \frac{pq(1-r)}{p^2 + q^2} & p & r\Delta X \\ \frac{pq(1-r)}{p^2 + q^2} & \frac{rp^2 + q^2}{p^2 + q^2} & q & r\Delta Y \\ -p & -q & 1 & r\Delta Z \\ 0 & 0 & 0 & r \end{bmatrix}. \quad (3)$$

This was derived by making a single rotation of the (s, t, n) frame around the unit vector $(-q, p, 0) / (\sqrt{p^2 + q^2})$ by an angle ϕ with $\cos\phi = \frac{1}{r}$ and $\sin\phi = \frac{\sqrt{p^2 + q^2}}{r}$.

3.2. Projected Texture

This subsection concludes with an expression for a perspectively projected sinusoid. We begin by assuming the texture on the surface is "painted" on and not a relief pattern. It is locally characterized in the (s, t, n) surface frame as a pattern of surface markings given by $g(s, t)$. Points on this locally planar surface are given by coordinates $(s, t, 0)$. Applying the transformation matrix, the corresponding world coordinates are

$$\begin{aligned} X &= t_{11}s + t_{12}t + \Delta X \\ Y &= t_{21}s + t_{22}t + \Delta Y \\ Z &= t_{31}s + t_{32}t + \Delta Z \end{aligned} \tag{4}$$

Under perspective, these points project to the image plane at

$$\begin{aligned} x &= -d\frac{X}{Z} = -d\frac{t_{11}s + t_{12}t + \Delta X}{t_{31}s + t_{32}t + \Delta Z} \\ y &= -d\frac{Y}{Z} = -d\frac{t_{21}s + t_{22}t + \Delta Y}{t_{31}s + t_{32}t + \Delta Z} \end{aligned} \tag{5}$$

The origin of the (s, t, n) frame thus projects to $(x_i, y_i) = (-d\frac{\Delta X}{\Delta Z}, -d\frac{\Delta Y}{\Delta Z})$ on the image plane. In order to avoid carrying a coordinate offset through the calculations, we define another coordinate system, (x', y'), on the image plane that is centered at (x_i, y_i) with its axes parallel to those of the image plane. Given an (x, y) on the surface,

$$\begin{aligned} x' &= x - x_i = -d\frac{t_{11}s + t_{12}t + \Delta X}{t_{31}s + t_{32}t + \Delta Z} - x_i \\ y' &= y - y_i = -d\frac{t_{21}s + t_{22}t + \Delta Y}{t_{31}s + t_{32}t + \Delta Z} - y_i \end{aligned} \tag{6}$$

Solving these two equations for (s, t) will give equations that give a point in the surface frame for any corresponding point in the (x', y') frame. Doing so, using $(\Delta X, \Delta Y) = (-\frac{x_i \Delta Z}{d}, -\frac{y_i \Delta Z}{d})$ and the orthonormality relationships among the vectors in the transformation matrix, we have

$$\begin{aligned} s(x', y') &= -\frac{\Delta Z\,[d\,(y' t_{12} - x' t_{22}) + t_{32}\,(y' x_i - x' y_i)\,]}{d\,[t_{13}\,(x' + x_i) + t_{32}\,(y' + y_i) - d\Delta Z]} \\ t(x', y') &= \frac{\Delta Z\,[d\,(y' t_{11} - x' t_{21}) + t_{31}\,(y' x_i - x' y_i)\,]}{d\,[t_{13}\,(x' + x_i) + t_{32}\,(y' + y_i) - d\Delta Z]} \end{aligned} \tag{7}$$

Thus, if the brightness pattern on a locally planar patch on a textured surface is $g(s, t)$, then the projected pattern on the image plane is a nonlinear warping of the pattern given by $g(s(x', y'), t(x', y'))$.

To simplify the frequency analysis, we will linearize this warping using a truncated Taylor series around $(x', y') = (0, 0)$. The approximation is justified since we are only examining a relatively small window of intensities around the point of interest. We have

$$\begin{aligned} s(x', y') &\approx s_x x' + s_y y' \\ t(x', y') &\approx t_x x' + t_y y' \end{aligned} \tag{8}$$

with

$$\begin{aligned} s_x &= \frac{\partial}{\partial x'} s(x', y')\big|_{(x', y') = (0, 0)} = \frac{\Delta Z\,[d\,(rp^2 + q^2) - q y_i\,(p^2 + q^2)\,]}{d\,(p^2 + q^2)\,(p x_i + q y_i - d)} \\ s_y &= \frac{\partial}{\partial y'} s(x', y')\big|_{(x', y') = (0, 0)} = \frac{\Delta Z q\,[dp\,(r - 1) + x_i\,(p^2 + q^2)\,]}{d\,(p^2 + q^2)\,(p x_i + q y_i - d)} \\ t_x &= \frac{\partial}{\partial x'} t(x', y')\big|_{(x', y') = (0, 0)} = \frac{\Delta Z p\,[dq\,(r - 1) + y_i\,(p^2 + q^2)\,]}{d\,(p^2 + q^2)\,(p x_i + q y_i - d)} \\ t_y &= \frac{\partial}{\partial y'} t(x', y')\big|_{(x', y') = (0, 0)} = \frac{\Delta Z\,[d\,(p^2 + rq^2) - p x_i\,(p^2 + q^2)\,]}{d\,(p^2 + q^2)\,(p x_i + q y_i - d)} \end{aligned} \tag{9}$$

where we have substituted the values of t_{ij} from Equation (3).

The projected version of $g(s, t)$ is then approximately $g(s_x x' + s_y y', t_x x' + t_y y')$, which is just an affine transformation (without translation) of the coordinates.

3.3. Relation Between Projected Sinusoids

If we show how the projection affects a single, sinusoidal texture pattern, we can easily see what happens to periodic textures, because they are just summed sinusoids (according to the Fourier series). Suppose the brightness pattern on a textured surface is given by $\cos(2\pi(u_0 s + v_0 t))$, then the corresponding projected textures from two different points on this surface would be given by

$$\cos(2\pi((s_{x_1} x' + s_{y_1} y')u_0 + (t_{x_1} x' + t_{y_1} y')v_0))$$
$$\cos(2\pi((s_{x_2} x' + s_{y_2} y')u_0 + (t_{x_2} x' + t_{y_2} y')v_0))$$

where we have started subscripting with "1" and "2" to indicate two distinct points on the image plane. The frequencies of the sinusoids are

$$\begin{bmatrix} u_1 \\ v_1 \end{bmatrix} = \begin{bmatrix} s_{x_1} & t_{x_1} \\ s_{y_1} & t_{y_1} \end{bmatrix} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix}$$
$$\begin{bmatrix} u_2 \\ v_2 \end{bmatrix} = \begin{bmatrix} s_{x_2} & t_{x_2} \\ s_{y_2} & t_{y_2} \end{bmatrix} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \tag{10}$$

Some linear algebra shows that the frequencies of the two projected sinusoids are themselves related by an affine transform (without translation):

$$\begin{bmatrix} u_2 \\ v_2 \end{bmatrix} = \begin{bmatrix} a_1 & b_1 \\ a_2 & b_2 \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \end{bmatrix} = \frac{1}{s_{x_1} t_{y_1} - s_{y_1} t_{x_1}} \begin{bmatrix} s_{x_2} t_{y_1} - s_{y_1} t_{x_2} & s_{x_1} t_{x_2} - s_{x_2} t_{x_1} \\ s_{y_2} t_{y_1} - s_{y_1} t_{y_2} & s_{x_1} t_{y_2} - s_{y_2} t_{x_1} \end{bmatrix} \begin{bmatrix} u_1 \\ v_1 \end{bmatrix} \tag{11}$$

To get the full relation in terms of quantities we know, we plug in for the s's and t's from Equation (9). We assume the two points on the textured surface are both on the same plane, thus $p_1 = p_2 = p$, $q_1 = q_2 = q$, and

$$\frac{\Delta Z_2}{\Delta Z_1} = \frac{d - px_1 - qy_1}{d - px_2 - qy_2} \quad . \tag{12}$$

Then

$$\begin{aligned} a_1 &= \frac{(d - px_1 - qy_1)\,(d - px_1 - qy_2)}{(px_2 + qy_2 - d)^2} \\ b_1 &= \frac{p\,(d - px_1 - qy_1)\,(y_2 - y_1)}{(px_2 + qy_2 - d)^2} \\ a_2 &= \frac{q\,(d - px_1 - qy_1)\,(x_2 - x_1)}{(px_2 + qy_2 - d)^2} \\ b_2 &= \frac{(d - px_1 - qy_1)\,(d - px_2 - qy_1)}{(px_2 + qy_2 - d)^2} \end{aligned} \tag{13}$$

where (x_1, y_1) and (x_2, y_2) are the two points on the image plane being compared.

We conclude that the frequencies of a single sinusoid projected from the same plane to two different points in the image are approximately related by an affine transformation. The affine parameters are functions of the position of the two points on the image, the camera's pinhole-to-sensor distance, and the plane's surface normal. In the next section, we show how to exploit this relationship to find the surface normal.

4. Shape from Periodic Texture

This section presents our algorithm for finding the surface normal of a plane with periodic texture using local spatial frequency. We presented the theory for general textures in [26]. We concentrate on periodic textures here for the sake of simplicity, speed, and noise immunity. This shape algorithm is an integral part of our segmentation algorithm.

4.1. Periodic Texture Representation

If we assume the texture on the plane is periodic, then any physically realizable such texture can be represented by a Fourier series. Thus, we assume the frontal texture brightness pattern is given by

$$g(s,t) = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} c_{mn} \exp\left[j2\pi\left(mu_0 s + nv_0 t\right)\right], \tag{14}$$

where we are unconcerned with the values of the fundamental frequency (u_0, v_0) and the complex Fourier series coefficients c_{mn}. Using upper-case letters to represent Fourier transforms of their corresponding lower-case functions in space, along with this definition of the Fourier transform,

$$F(u,v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x,y)\, e^{-j2\pi(ux+vy)}\, dxdy, \tag{15}$$

we have

$$G(u,v) = \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} c_{mn}\delta(u - mu_0, v - nv_0). \tag{16}$$

This is a grid of delta functions, with each delta at one component frequency. For example, a periodic cotton canvas (Brodatz[7] D77) and its power spectrum are shown in Figure 8. We note that the delta functions are slightly spread. This is because we are computing the Fourier transform with only local support.

We showed in Section 3 that the local brightness pattern from a surface patch in the scene undergoes approximately an affine transformation when it is projected onto the image plane. Since an affine transformation in space corresponds to an affine transform in frequency[14], the Fourier transform of the projected texture patch will be a scaled and skewed grid of delta functions, with each delta representing one frequency component.

In order to represent the spectrogram more efficiently and to speed subsequent computations, we only store the peak frequencies from each power spectrum patch. Our spectrogram preprocessor finds the peaks in each patch in order of size. It keeps looking until the current peak is less than 20% of the magnitude of the largest peak, or until it finds six peaks, whichever comes first. It also ignores peaks below a frequency of 0.03 cycles/pixel. This helps eliminate low frequencies due to shading.

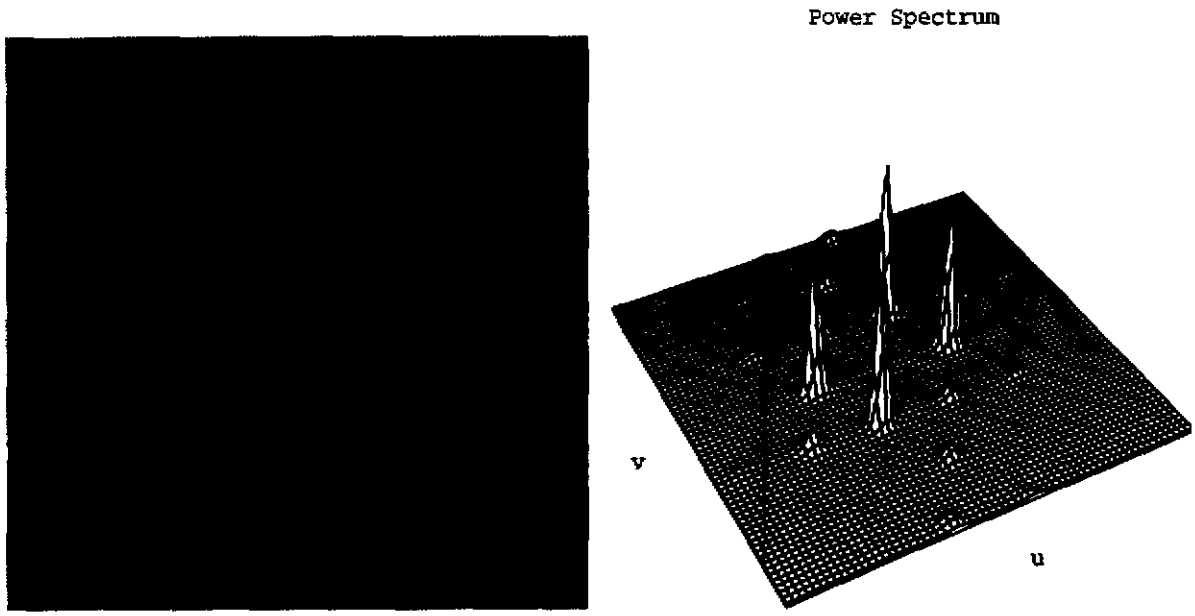


Figure 8: The Fourier transform of this periodic cotton canvas is composed of *delta functions*.

In order to track frequency shifts for computing surface normals, we need to know which peaks in one patch correspond to those in neighboring patches. Our preprocessor matches peaks between every patch and its two neighboring patches to the right and below. We do this pairwise matching by considering every possible match combination between the two sets of peaks, including leaving some peaks unmatched. We pick the combination that has simultaneously the most matches and no match errors that exceed a threshold based on the largest surface normal we expect in the scene. For a maximum (p, q) of $(1.5, 1.5)$, this threshold prevents matching peaks that are more than about 0.05 cycles/pixel apart.

After this preprocessing step we do not need the original spectrogram for any of the subsequent operations. It is adequately represented by the peaks and peak matches.

4.2. Computing Surface Normals

We compute surface normals by finding the (p, q) that best accounts for the observed frequency shifts between neighboring patches. At its most basic, this computation involves just two adjacent patches centered at (x_1, y_1) and (x_2, y_2) on the image plane. The sets of m matching peaks from the two patches are (u_{1_0}, v_{1_0}), (u_{1_1}, v_{1_1}), (u_{1_2}, v_{1_2}), ... $(u_{1_{m-1}}, v_{1_{m-1}})$ and (u_{2_0}, v_{2_0}), (u_{2_1}, v_{2_1}), (u_{2_2}, v_{2_2}), ... $(u_{2_{m-1}}, v_{2_{m-1}})$. If we write the affine parameters from Equation (13) as functions of the surface normal, we have

$$e_{ssd}(p,q) = \sum_{i=0}^{m-1} \left\| \begin{bmatrix} u_{2_i} \\ v_{2_i} \end{bmatrix} - \begin{bmatrix} a_1(p,q) & b_1(p,q) \\ a_2(p,q) & b_2(p,q) \end{bmatrix} \begin{bmatrix} u_{1_i} \\ v_{1_i} \end{bmatrix} \right\|^2 . \quad (17)$$

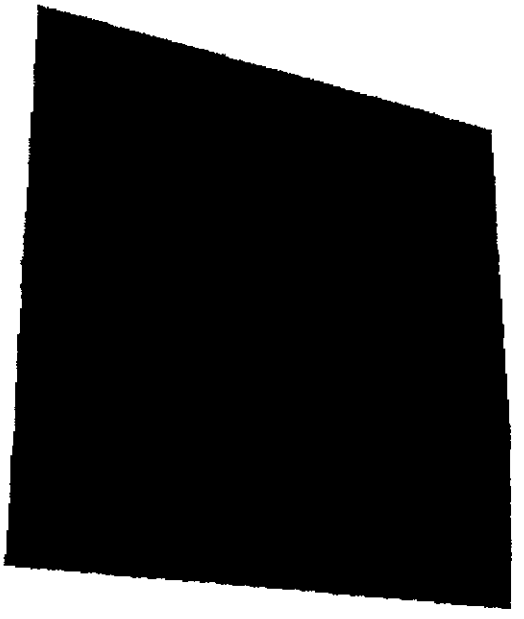
This will be small if we have the correct surface normal and the correct matches among the peaks. We perform an exhaustive search over a grid in (p, q) and take the surface normal that minimizes e_{ssd} as the solution. If we have more than two patches to use, we find the surface normal that minimizes the sum of the e_{ssd}'s for all unique, adjacent pairs of patches in the region. We only consider adjacent pairs of patches, that is, the patches that have had their frequency peaks matched by the preprocessor. This algorithm is similar to one developed by Super and Bovik[41]. One difference is that ours uses multiple frequency peaks from a single texture, while theirs uses a single, dominant frequency at each point.

4.3. Results

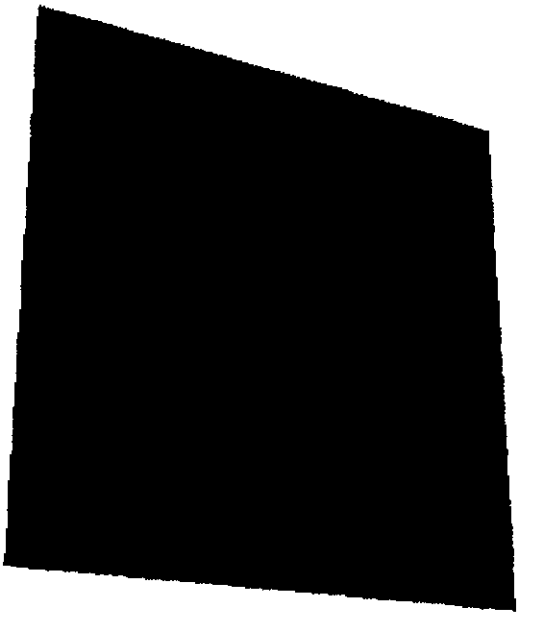
Two important parameters that affect the accuracy of our solution are the number of patches used to compute the surface normal and the center-to-center spacing of the power spectrum patches. For a given center-to-center spacing, we would like to use as many patches as possible, as long as they all fall on the same textured plane, in order to have more data contributing to the solution. We would also like to avoid small center-to-center distances, because the shape-induced frequency shifts would be dominated by noise and approximation errors.

Figure 9 shows four identical plates with different Brodatz[7] textures mapped onto them using a computer graphics program. The actual surface normal is $(p, q) = (0.614, 0.364)$. We tested our algorithm on these images using different numbers of patches and different center-to-center spacing. In each trial, the center-to-center spacing was equal in x and y. We let this parameter vary from 5 to 50 pixels in increments of 5. For each center-to-center distance, we computed shape using as many unique n x n squares of adjacent patches as would fit on the textured part of the image, starting with $n = 2$.

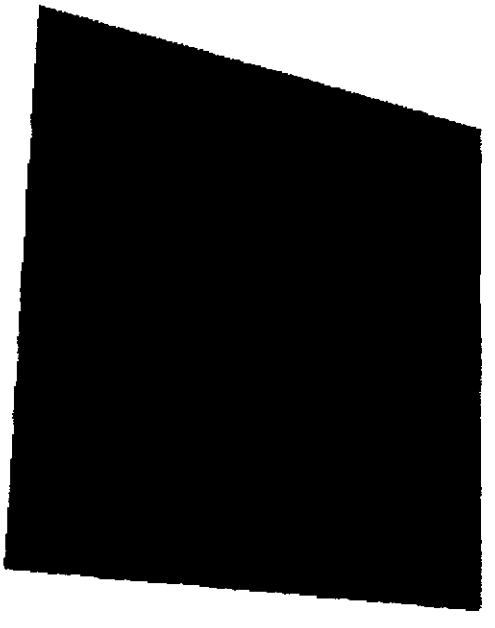
Figure 10a shows the average errors in degrees of our surface normal estimates for different numbers of patches and different center-to-center spacings. The average was taken over all four images and over all the n x n squares of patches that would fit on the texture. As expected, the error decreases for larger numbers of widely spaced patches, with the best estimates being in error by about six degrees. Our shape-from-texture algorithm succeeds in giving good results on periodic textures without the need for image feature detection. Since it uses the space/frequency representation, it is possible to integrate it into a segmentation algorithm that works on 3D textured, planar surfaces.



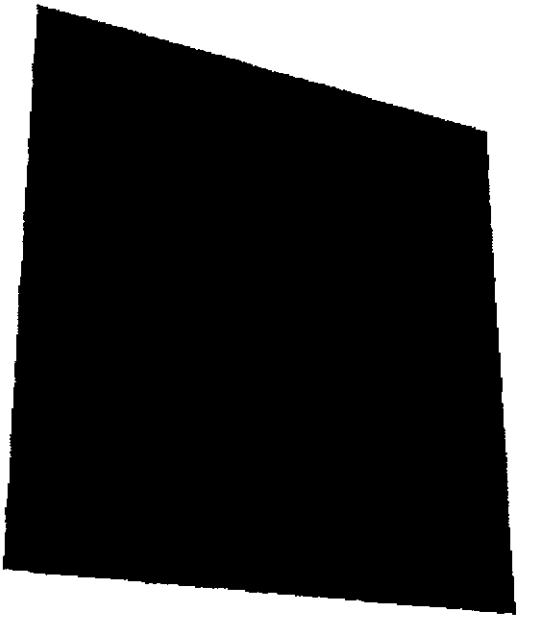
Woven aluminum wire (D6)



French canvas (D21)



Oriental straw cloth (D53)



Cotton canvas (D77)

Figure 9: Images used for testing surface normal computation. These are all from the Brodatz[7] book of textures, and the book's designations are given in parentheses.

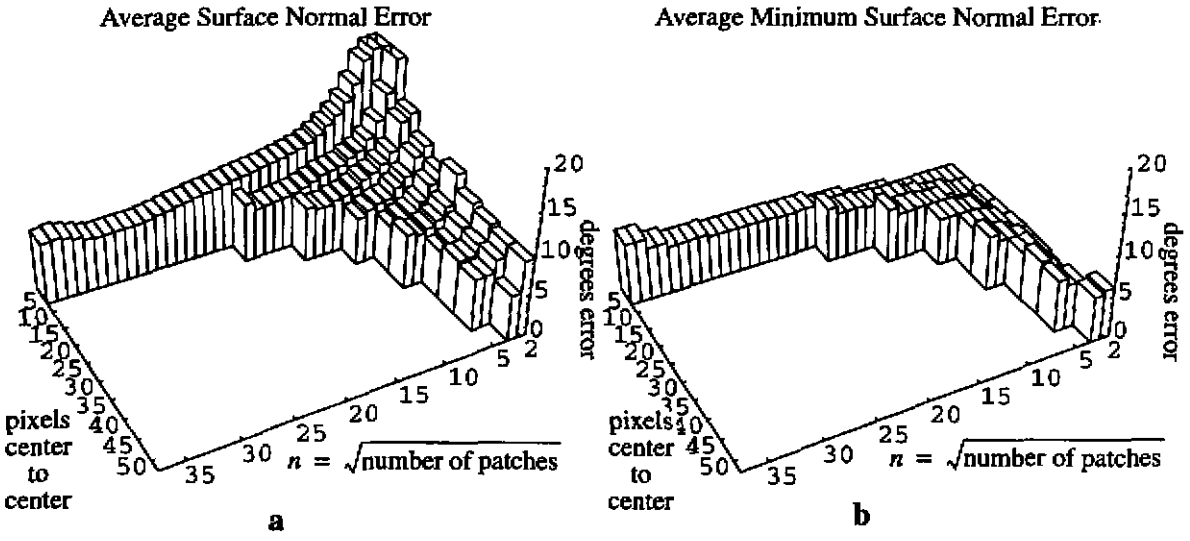


Figure 10: Average errors in surface normal from the four test images for different patch center-to-center distances and different numbers of patches.

Unfortunately the need for accuracy conflicts with the requirements of our segmentation algorithm in terms of the number of patches and center-to-center spacing. Our segmentation algorithm begins by estimating surface normals using small parts of the image. Using small support for these estimates is important, because we do not want the support to overlap texture boundaries. This means we have to keep n and the center-to-center spacing small, which tends to compromise accuracy according to Figure 10a. Fortunately, though, some of the estimates from the n x n squares are still good, even with small support and small n. Figure 10b shows the average minimum error in surface normal, where the minimum is taken over all the n x n squares and the average over the four images. In almost every case, at least one of the n x n squares gave a fairly accurate surface normal. Since we start our segmentation with many seed regions, we are likely to have some that are "good", even with small support. For the segmentation algorithm discussed in the next section, we chose $n = 2$ and a center-to-center spacing of 15 pixels. Since we do not allow interleaved regions, we computed the spectrogram with the same center-to-center spacing.

5. Segmenting Textured 3D Surfaces

Our segmentation procedure is a region-growing algorithm that merges regions based on similarities in their local power spectra. The problem with applying such a procedure naively to an image of 3D textured surfaces is that the power spectra on identically textured surfaces will change due to 3D effects. And while a generous tolerance may still allow such regions to be merged, this may well allow different textures to be merged also. Thus, we need to explicitly account for the 3D effects. We do this by computing the surface normal of each region (using the algorithm in the previous section) and then "frontalizing" the frequencies to show what the power spectra of the texture would look like if viewed from the front. If adjacent regions have similar frontalized frequency content, they are merged. A detailed description of the segmentation algorithm follows.

5.1. The Data Structures

The smallest elements of our image representation are the power spectrum patches, represented by their peaks. Since we segment based on 4-connectedness, each patch has a list of its 4-connected neighbors. Each patch also contains the indices of the matched peaks in the patch to the right and the patch below.

Sets of merged patches are called hypotheses. Each hypothesis contains the usual records needed for region growing, *i.e.* the constituent patches, neighboring patches, and neighboring hypothesis. We also use the constituent patches to compute the surface normal using our shape-from-texture algorithm. This surface normal is used to compute a frontalized version of the frequency peaks for each constituent patch. Each group of matching frontalized peaks is represented in the hypothesis in terms of its mean frequency. These mean frequencies give an idea of the power spectrum of the region if it were viewed frontally. The surface normal is also used to compute frontalized versions of the four-connected neighboring patches of the hypothesis. If these frontalized neighbors are from the same texture on the same plane, they will be similar to the frontalized hypothesis.

5.2. Frontalization of Frequency Peaks

This section describes our frequency peak frontalization algorithm. Our goal is to determine what a group of frequency peaks on different patches would be if we viewed the texture from the front. We know from Equation (10) that a frequency (u_0, v_0) on a non-frontal textured surface in the scene is related by an affine transformation to a frequency (u_i, v_i) on the image plane. In matrix form, this is

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} s_{x_i} & t_{x_i} \\ s_{y_i} & t_{y_i} \end{bmatrix} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = S_i \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \tag{18}$$

We cannot simply invert this relationship for the frontalization, because we don't know the ΔZ_i coordinate of the surface, and this is required to compute the matrix S_i. In fact, we can never compute $[u_0, v_0]^T$, because we never know the depth of the patch.

Imagine we have a frontalized reference patch ($(p, q) = (0, 0)$) with a depth of ΔZ_{ref} from the same plane and with the same texture. The 4x4 homogeneous transformation locating the surface patch's local coordinate frame would be

$$\begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \Delta X_{ref} \\ 0 & 1 & 0 & \Delta Y_{ref} \\ 0 & 0 & 1 & \Delta Z_{ref} \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{19}$$

Using these transformation parameters and solving Equation (6) for s and t gives

$$\begin{aligned} s_{ref}(x', y') &= \frac{-x'\Delta Z_{ref}}{d} \\ t_{ref}(x', y') &= \frac{-y'\Delta Z_{ref}}{d} \end{aligned} . \tag{20}$$

Then the projected frequency from this frontal patch will be approximated as before as an affine transformation of the scene frequency. The affine transformation parameters come from the first partial derivative terms of the Taylor series of $s_{ref}(x', y')$ and $t_{ref}(x', y')$. The frontalized frequency is then

$$\begin{bmatrix} u_{frontal} \\ v_{frontal} \end{bmatrix} = \begin{bmatrix} \frac{-\Delta Z_{ref}}{d} & 0 \\ 0 & \frac{-\Delta Z_{ref}}{d} \end{bmatrix} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = S_{ref} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} . \tag{21}$$

Solving Equation (18) for $[u_0, v_0]^T$ and inserting this into Equation (21) gives

$$\begin{bmatrix} u_{frontal} \\ v_{frontal} \end{bmatrix} = S_{ref} S_i^{-1} \begin{bmatrix} u_i \\ v_i \end{bmatrix} = F_i \begin{bmatrix} u_i \\ v_i \end{bmatrix} \tag{22}$$

When F_i is multiplied out, it elements become

$$\begin{aligned} f_{11} &= \frac{\Delta Z_{ref}[d(p^2 + rq^2) - px_i(p^2 + q^2)]}{dr(p^2 + q^2)\Delta Z_i} \\ f_{12} &= \frac{\Delta Z_{ref} p[dq(1-r) - y_i(p^2 + q^2)]}{dr(p^2 + q^2)\Delta Z_i} \\ f_{21} &= \frac{\Delta Z_{ref} q[dp(1-r) - x_i(p^2 + q^2)]}{dr(p^2 + q^2)\Delta Z_i} \\ f_{22} &= \frac{\Delta Z_{ref}[d(rp^2 + q^2) - qy_i(p^2 + q^2)]}{dr(p^2 + q^2)\Delta Z_i} \end{aligned} \tag{23}$$

This still contains the unknown depth value ΔZ_i. But, since the reference patch is on the same plane, then we have from Equation (12):

$$\frac{\Delta Z_{ref}}{\Delta Z_i} = \frac{d - px_i - qy_i}{d - px_{ref} - qy_{ref}}. \tag{24}$$

Putting this ratio into Equation (23) gives the affine frontalization parameters for an arbitrary patch i in terms of known quantities:

$$
\begin{aligned}
f_{11} &= \frac{[d(p^2+rq^2) - px_i(p^2+q^2)]\ (px_i+qy_i-d)}{dr(p^2+q^2)\ (px_{ref}+qy_{ref}-d)} \\
f_{12} &= \frac{p[dq(1-r)-y_i(p^2+q^2)]\ (px_i+qy_i-d)}{dr(p^2+q^2)\ (px_{ref}+qy_{ref}-d)} \\
f_{21} &= \frac{q[dp(1-r)-x_i(p^2+q^2)]\ (px_i+qy_i-d)}{dr(p^2+q^2)\ (px_{ref}+qy_{ref}-d)} \\
f_{22} &= \frac{[d(rp^2+q^2) - qy_i(p^2+q^2)]\ (px_i+qy_i-d)}{dr(p^2+q^2)\ (px_{ref}+qy_{ref}-d)}
\end{aligned}
\tag{25}
$$

The frontalization step works this way: For a group of patches hypothesized to be on the same plane, we arbitrarily pick one patch as the reference patch. In our case we pick the first in the list. The affine frontalization transformation is then computed for each patch according to Equation (25), and each peak frequency is transformed accordingly. This does not tell us what the true frontalized frequencies are, but it tells us what the frequencies would be if all the patches had the same depth as the reference patch, which is good enough for segmentation.

5.3. Initial Hypotheses

Region-growing begins with a conservative set of small hypotheses. Each of these initial hypotheses is made up of four adjacent power spectrum patches arranged in a square. We check each possible 2x2 set of patches as an initial hypothesis. In order to qualify, the set of four patches must meet three criteria:

1. They must all have the same number of peaks.
2. All possible peak matches among the four patches must exist.
3. There can be no inconsistent match loops, where a set of peak-to-peak matches would result in two peaks in the same patch being matched (see Figure 11).

These initial hypotheses are allowed to overlap. The centers of the initial hypotheses for the image in Figure 3 are shown in Figure 12.

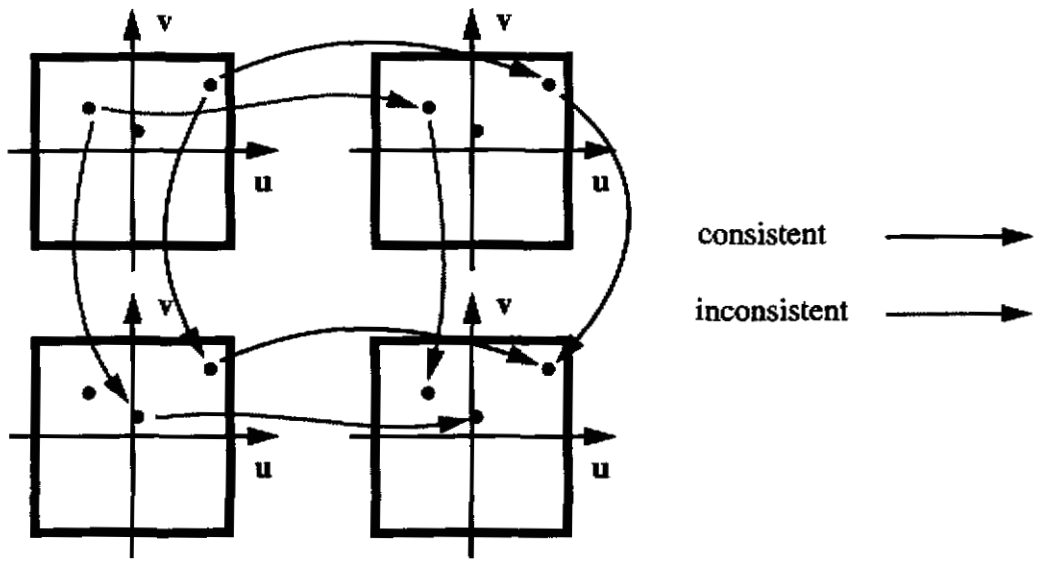


Figure 11: Inconsistent match loops are not allowed in the initial hypotheses.

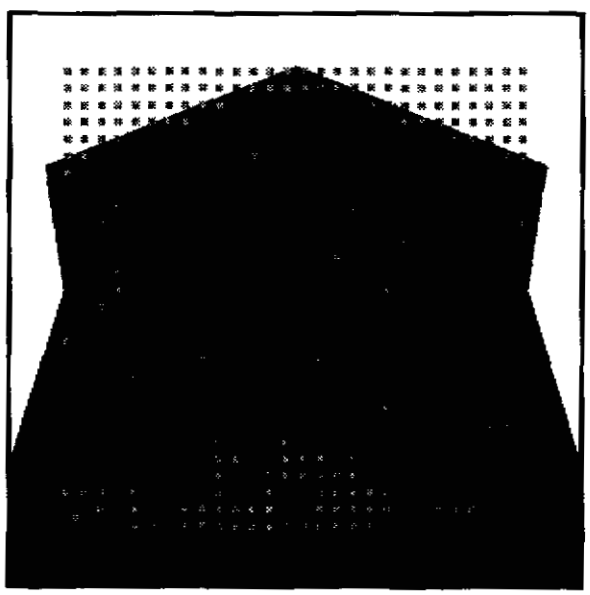


Figure 12: Centers of the initial 2x2 hypotheses

5.4. Hypothesis Growing

Growing the initial 2x2 hypotheses proceeds in three stages. In the first stage, each 2x2 hypothesis is merged with neighboring patches that have the same number of peaks as the hypothesis. If the average deviation between frontalized peaks is more than $\Delta u = 0.01$ cycles/pixel, then the merge does not take place. Overlapping hypotheses are allowed in this stage. This makes the algorithm more robust, in that the constituent patches of a bad initial hypothesis can be taken over by a good hypothesis. If any hypothesis contains over half the patches of another hypothesis, the two hypotheses are merged.

The second stage begins by deassigning each patch that belongs to more than one hypothesis. Then each unassigned patch is assigned, in raster order, to the best neighboring hypothesis. The best hypothesis is chosen by creating a frontalized version of the patch with respect to each neighboring hypothesis. We match peaks between the frontalized patches and the hypotheses using the same peak-matching routine as in the spectrogram preprocessing program. The best hypothesis is the one with the most matches. Ties are broken by taking the hypothesis with the smallest sum of squared differences between the matched peaks. (If no matches are found for any of the candidate hypotheses, this patch becomes its own hypothesis.) This stage ends by splitting all noncontiguous hypotheses. The output is a set of contiguous regions with every patch assigned to one and only one region.

The final stage merges similar hypotheses. Each hypothesis maintains a list of four-connected neighboring hypotheses along with frontalized versions of their peaks. Two neighboring hypotheses are merged if the average deviation between the matched peaks on their common border is less than $\Delta u = 0.01$ cycles/pixel, and if they have "enough" matched peaks between their constituent patches on their common border. "Enough" means that of all possible peak matches between the two, at least 60% must be matched. This helps avoid merges between hypotheses that have a few, lucky, well-matched peaks.

5.5. Result

We tested our segmentation program on the image in Figure 3. This image was produced with a computer graphics program, mapping Brodatz[7] textures onto flat plates. Figure 13 shows the edges of the final hypotheses for the underlying image. The three textures are clearly outlined. This demonstrates an advantage of region-growing over edge-finding, in that all the edges are closed, and there is no "leaking" from one region to another. This is critical to the shape-from-texture computation that is an integral part of the region-growing, which is in turn a necessary component of successfully understanding as much as we can from the image. Figure 13 also shows the surface normals computed for each region. The average error for the three textured regions is 8.4 degrees.

The preliminary segmentation demonstration still has problems due to the coarse spatial sampling we use to compute the spectrogram. The blockiness could be solved by increasing the spatial resolution at the cost of increased computation time. The shrinkage in the regions is caused by patches that overlap texture boundaries or that butt up against the edge of the image. One solution to the texture boundary problem is to find and split these patches once we have an idea of what the frontal textures look like. Another solution might be to simply find and eliminate them, letting the overlapping "pure" patches take over the region left behind.

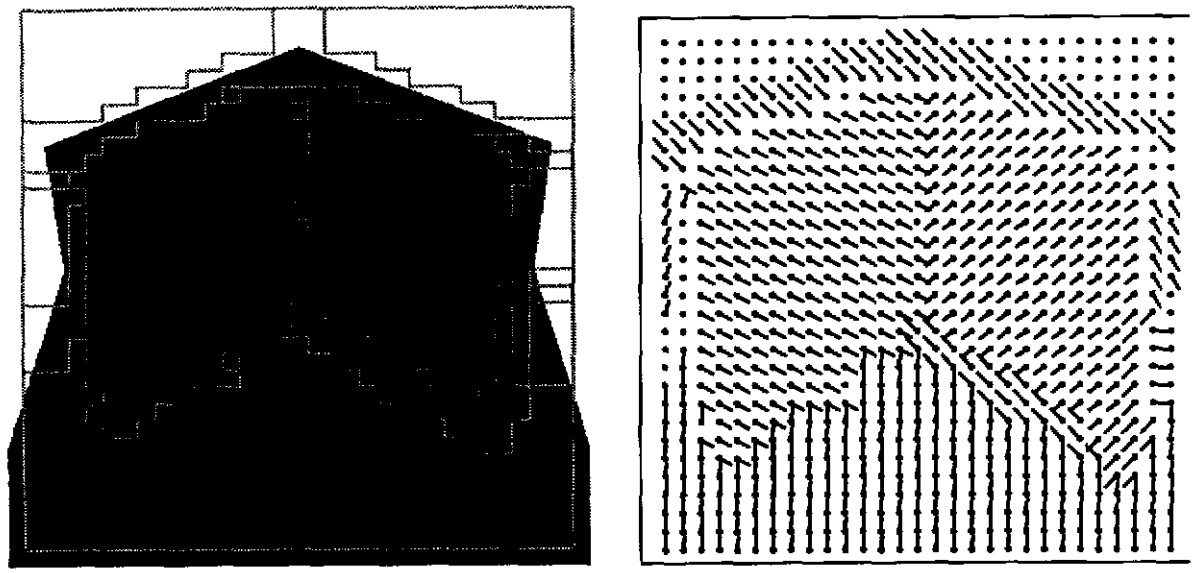


Figure 13: Edges of regions and needle diagram of computed surface normals of texture plates

6. The Future of Space/Frequency and Computer Vision

We have shown how the space/frequency representation is useful for solving the combined problem of segmentation and shape-from-texture. This should not be a surprise, because the representation has already been used to solve both problems separately, as shown in Figure 14. The space/frequency representation is the natural choice for solving the combined problem.

All the work cited in Figure 14 is computer vision research based on either the Fourier transform of the whole image or the space/frequency representation. Our earlier work in moire patterns[27] was based in the frequency domain, and this meant we were prepared to account for aliasing in the shape-from-texture algorithm we presented in [28]. This represents another unification of algorithms based on the space/frequency representation. Since so many other algorithms are based on the same representation, we predict a gradual unification of all these algorithms in terms of the space/frequency representation. We give this final theory the grand title of "The Unified Theory of Spatial Vision".

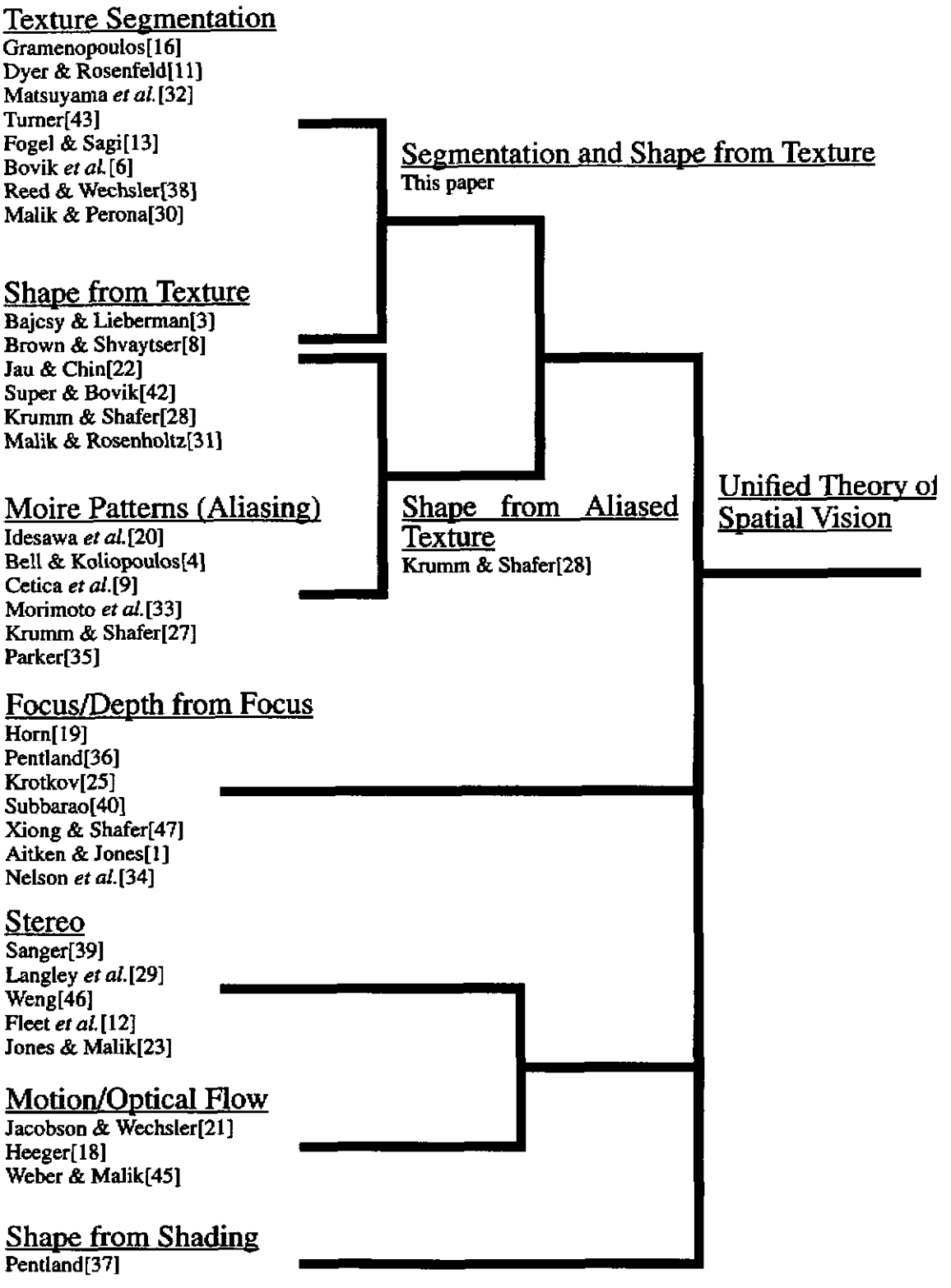


Figure 14: This work in computer vision has used spatial frequency or local spatial frequency representations, and indicates that many different algorithms can be unified because of their common representation.

References

[1] Aitken, G.J.M. and P.F. Jones. "Three-Dimensional Image Capture by Volume Imaging." In *Proceedings of the SPIE, Sensing and Reconstruction of Three-Dimensional Objects and Scenes*, vol.1260, 2-9, 1990.

[2] Aloimonos, J. "Shape from Texture." *Biological Cybernetics* 58 (1988): 345-360.

[3] Bajcsy, Ruzena and Lawrence Lieberman. "Texture Gradient as a Depth Cue." *Computer Graphics and Image Processing* 5 (1976): 52-67.

[4] Bell, Bernard W. and Chris L. Koliopoulos. "Moire Topography, Sampling Theory, and Charged-Coupled Devices." *Optics Letters* 9 (no. 5, May 1984): 171-173.

[5] Blake, Andrew and Constantinos Marinos. "Shape from Texture: Estimation, Isotropy and Moments." *Artificial Intelligence* 45 (1990): 323-380.

[6] Bovik, Alan Conrad, Marianna Clark, and Wilson S. Geisler. "Multichannel Texture Analysis Using Localized Spatial Filters." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12 (January 1990): 55-73.

[7] Brodatz, Phil. *Textures: A Photographic Album for Artists and Designers*, New York: Dover, 1966.

[8] Brown, Lisa Gottesfeld and Haim Shvaytser. "Surface Orientation from Projective Foreshortening of Isotropic Texture Autocorrelation." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12 (June 1990): 584-588.

[9] Cetica, Maurizio, Franco Francini, and Duilio Bertani. "Moire With One Grating and a Photodiode Array." *Applied Optics* 24 (no. 11, 1 June 1985): 1565-1566.

[10] DeFatta, David J., Joseph G. Lucas, and William S. Hodgkiss. *Digital Signal Processing: A System Design Approach*. New York: John Wiley & Sons, 1988, p. 270.

[11] Dyer, Charles R. and Azriel Rosenfeld. "Fourier Texture Features: Suppression of Aperture Effects." *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6 (October 1976): 703-705.

[12] Fleet, David J. and Allan D. Jepson, and Michael R. M. Jenkin. "Phase-Based Disparity Measurement." *CVGIP: Image Understanding* 53 (no. 2, March 1991): 198-210.

[13] Fogel, I. and D. Sagi. "Gabor Filters as Texture Discriminator." *Biological Cybernetics* 61 (June 1989): 103-113.

[14] Gaskill, Jack D. *Linear Systems, Fourier Transforms, and Optics*. New York: John Wiley & Sons, 1978.

[15] Gibson, James T. "The Perception of Visual Surfaces." *The American Journal of Psychology* 63 (July 1950): 367-384.

[16] Gramenopoulos, Nicholas. "Terrain Type Recognition Using ERTS-1 MSS Images." In *Symposium on Significant Results Obtained from the Earth Resources Technology Satellite-1*, Vol. 1, *Technical Presentations*, Section B, 1229-1241, 1973.

[17] Harris, Fredric, J. "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform." *Proceedings of the IEEE* 66 (January 1978): 51-83.

[18] Heeger, David J. "Optical Flow Using Spatiaotemporal Filters." *International Journal of Computer Vision* 1 (no. 4, January 1988): 279-302.

[19] Horn, Berthold. "Focusing." *Massachusetts Institute of Technology Artificial Intelligence Memo*, No. 160, May 1968.

[20] Idesawa, Masanori, Toyohiko Yatagai, and Takashi Soma. "Scanning Moire Method and Automatic Measurement of 3-D Shapes." *Applied Optics* 16 (no. 8, August 1977): 2152-2162.

[21] Jacobson, Lowell and Harry Wechsler, "Derivation of Optical Flow Using a Spatiotemporal-Frequency Approach." *Computer Vision, Graphics, and Image Processing* 38, (1987): 29-65.

[22] Jau, Jack Y. and Roland T. Chin. "Shape from Texture Using the Wigner Distribution." *Computer Vision, Graphics, and Image Processing* 52 (1990): 248-263.

[23] Jones, D. G. and J. Malik. "A Computational Framework for Determining Stereo Correspondence from a Set of Linear Spatial Filters." *European Conference on Computer Vision*, 395-410, 1992.

[24] Julesz, B. and J. R. Bergen. "Textons, The Fundamental Elements in Preattentive Vision and Perception of Textures." *The Bell System Technical Journal* 62 (no. 6, July-August 1983): 1619-1645.

[25] Krotkov, Eric. "Focusing." *International Journal of Computer Vision* 1 (no. 3, 1987): 223-237.

[26] Krumm, John and Steven A. Shafer. "Local Spatial Frequency Analysis for Computer Vision." Carnegie Mellon University Robotics Institute Technical Report No. CMU-RI-TR-90-11, May 1990.

[27] Krumm, John and Steven A. Shafer. "Sampled-Grating and Crossed-Grating Models of Moire Patterns from Digital Imaging." *Optical Engineering* 30 (no. 2, February 1991): 195-206.

[28] Krumm, John and Steven A. Shafer. "Shape from Periodic Texture Using the Spectrogram." *IEEE Conference on Computer Vision and Pattern Recognition*, 284-289, 1992.

[29] Langley, K., T. Atherton, R. Wilson, and M. Larcombe. "Vertical and Horizontal Disparities from Phase." *European Conference on Computer Vision*, 315-325, 1990.

[30] Malik, Jitendra and Pietro Perona. "Preattentive Texture Discrimination with Early Vision Mechanisms." *Journal of the Optical Society of America - A* 7 (no. 5, May 1990): 923-932.

[31] Malik, Jitendra and Ruth Rosenholtz. "A Differential Method for Computing Local Shape-From-Texture for Planar and Curved Surfaces." *IEEE Conference on Computer Vision and Pattern Recognition*, to appear, 1993

[32] Matsuyama, Takashi, Shu-Ichi Miura, and Makoto Nagao. "Structural Analysis of Natural Textures by Fourier Transformation." *Computer Vision, Graphics and Image Processing* 24 (1983): 347-362.

[33] Morimoto, Yoshiharu, Yasuyuki Seguchi, and Toshihoko Higashi. "Application of Moire Analysis of Strain Using Fourier Transform." *Optical Engineering* 27 (no. 8, August 1988): 650-656.

[34] Nelson, Brad, Nikolaos Papanikolopoulos, and Pradeep Khosla. "Dynamic Sensor Placement Using Controlled Active Vision." *International Federation of Automatic Control 1993 World Congress*, to appear, 1993.

[35] Parker, David H. "Moire Patterns in Three-Dimensional Fourier Space." *Optical Engineering* 30 (no. 10, October 1991): 1534-1541.

[36] Pentland, Alex P. "A New Sense for Depth of Field." In Aravind Joshi, Editor, *Ninth International Joint Conference on Artificial Intelligence*, 988-994, 1985.

[37] Pentland Alex P. "The Transform Method for Shape from Shading." M.I.T Media Lab Vision Sciences Technical Report 106, July 15, 1988.

[38] Reed, Todd R. and Harry Wechsler. "Segmentation of Textured Images and Gestalt Organization Using Spatial/Spatial-Frequency Representations." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12 (January 1990): 1-12.

[39] Sanger, Terence D. "Stereo Disparity Computation Using Gabor Filters." *Biological Cybernetics* 59 (1988): 405-418.

[40] Subbarao, Murali. "Parallel Depth Recovery by Changing Camera Parameters." *Second International Conference on Computer Vision*, 149-155, 1988

[41] Super, Boaz J. and Alan C. Bovik. "Three-Dimensional Orientation from Texture Using Gabor Wavelets." *SPIE Conference on Visual Communications and Image Processing*, November 1991.

[42] Super, Boaz J. and Alan C. Bovik. "Shape-from-Texture by Wavelet-Based Measurement of Local Spectral Moments." *IEEE Conference on Computer Vision and Pattern Recognition*, 296-301, 1992

[43] Turner, M.R. "Texture Discrimination by Gabor Functions." *Biological Cybernetics* 55, (October 1986): 71-82.

[44] Voorhees, Harry and Tomaso Poggio. "Computing texture boundaries from images." *Nature* 333 (26 May 1988): 364-367.

[45] Weber, Joseph and Jitendra Malik. "Robust Computation of Optical Flow in a Multi-Scale Differential Framework." University of California Berkeley, Computer Science Division, Technical Report No. UCB/CSD 92/709, November 1992.

[46] Weng, Juyang. "A Theory of Image Matching." *Third International Conference on Computer Vision*, 200-209, 1990

[47] Xiong, Yalin and Steven A. Shafer. "Depth from Focusing and Defocusing." *IEEE Conference on Computer Vision and Pattern Recognition*, to appear, 1993