To appear in **Spatial *Vision*** 1994, special
issue on Texture Perception **and** Attention

Segmenting Textured 3D Surfaces Using the Space/Frequency Representation

John **Krumm and** Steven A. Shafer

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

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.

Abstract

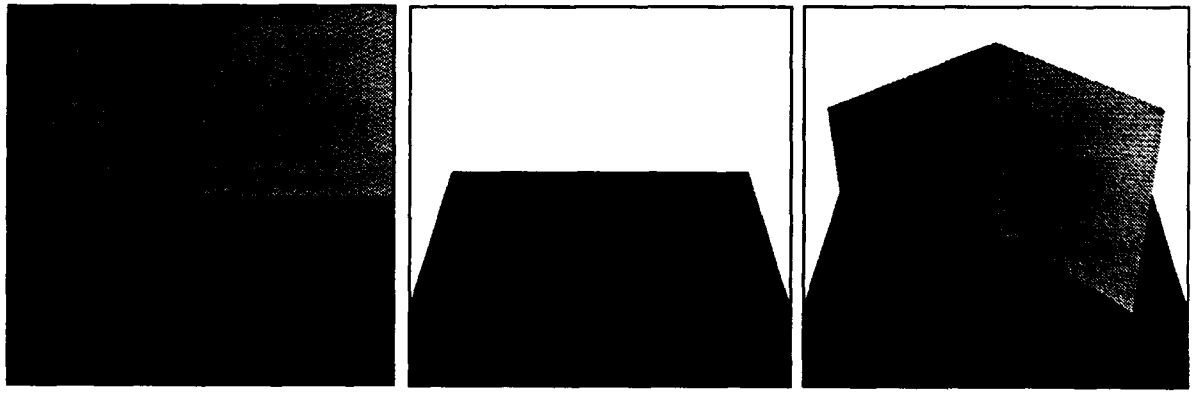
Segmenting 3D textured surfaces is critical for general image understanding. Unfortunately, current efforts at automatically understanding image texture are based on assumptions that make this goal impossible. Texture segmentation research assumes that the textures are flat and viewed from the front, while shape-from-texture work assumes that the textures have already been segmented. This deadlock means that none of these algorithms will work reliably on 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 (local power spectrum) to compute local surface normals from small 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. The merge criteria is based on a description length formula. We demonstrate the segmentation on images with real textures. To our knowledge, this is the first program that can segment 3D textured surfaces by explicitly accounting for shape effects.

1 The Problem and What We're Doing About It

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. Texture can tell us much about a scene. Julesz[28] and Gibson[19] 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' observations, 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

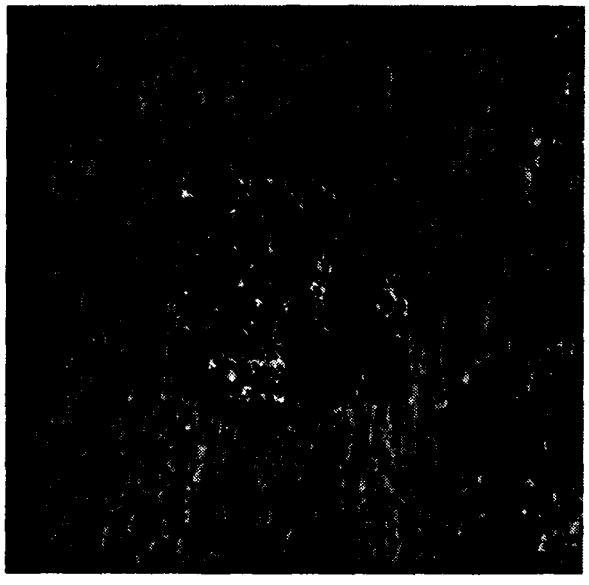


Figure 2: These random textures can be preattentively discriminated even though there are no texture elements. (Both textures have the same mean and variance in gray level. They are from Brodatz[9], D2 fieldstone and D12 bark of tree.)

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 (*e.g.* Malik and Perona[38]). These theories stand in sharp contrast to Julesz' texton theory. His response has been to point out that textons can still be preattentively discriminated by humans even after filtering out the spatial frequencies that appear to have the most discriminatory power[30]. We suspect this difference will be resolved someday with a local spatial frequency theory that subsumes Julesz' theory, whose observations will serve as a critical test.

Our goal is not to explain human vision, but to program a computer to imitate a human's abilities. And while this lets us develop algorithms with no psychophysical justification, some computer vision researchers are advocating the same types of local spatial frequency mechanisms as the psychophysicists. In particular, local spatial frequency can be used in computer vision for texture segmentation and shape-from-texture, as we show in this paper. These are attractive theories, because they use the same representation for both tasks, because they admit to a quantitative formulation, and because they do not require feature detection.

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).

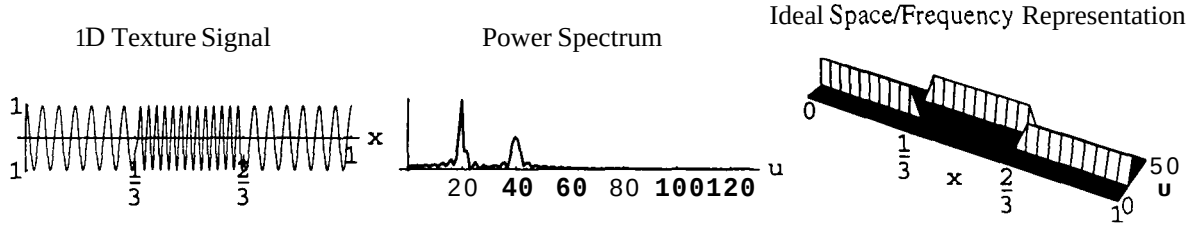


Figure 4: A signal, its power spectrum, and its spacdfrequency representation

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[21][12] for Fourier analysis. Its equation is

$$w(l) = w_0 + w_1 \cos\left(\frac{2\pi}{L-1} l\right) + w_2 \cos\left(\frac{4\pi}{L-1} l\right) + w_3 \cos\left(\frac{6\pi}{L-1} l\right) \quad (2)$$

Figure 6c shows some of the Gaussian-modulated sinusoids (an example of wavelets) used by Super and Bovik[51] 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, effectively 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[38] 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.

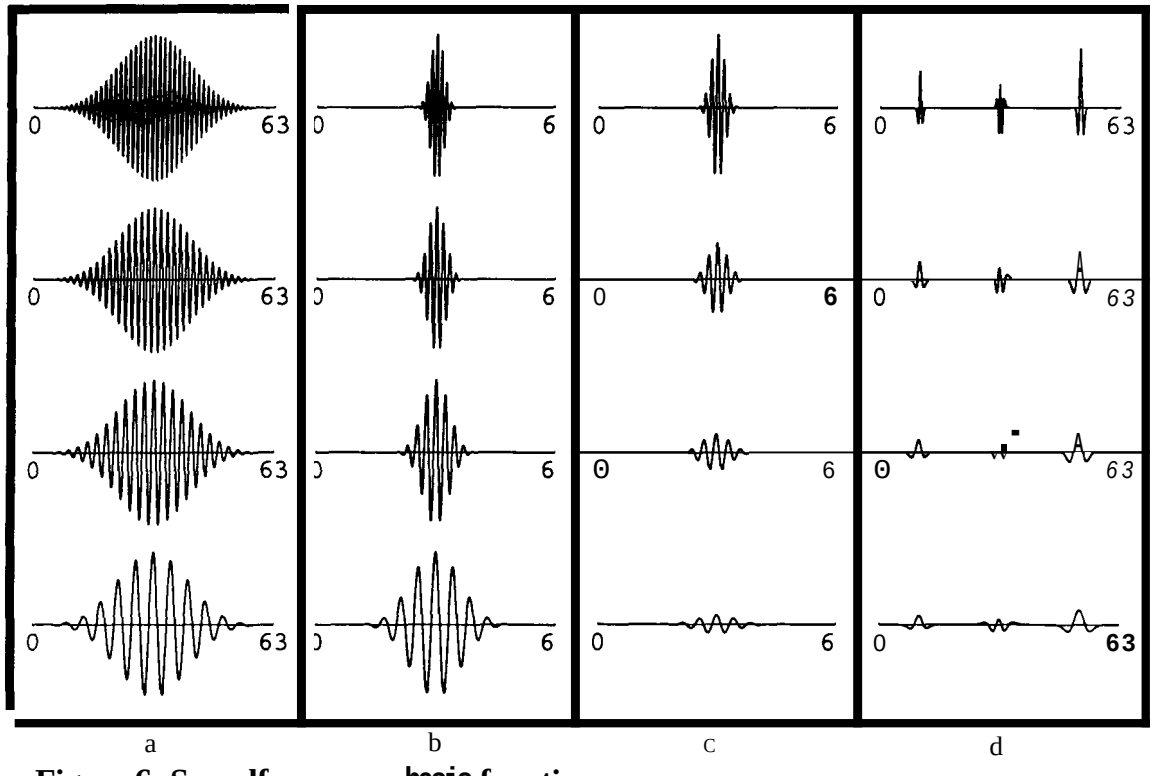


Figure 6: Spacdfrequency basis functions

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[51] for shape-from-texture(wavelets)
d) Linear combinations of Gaussians used by Malik & Perona[38] for texture segmentation

'A Fourier transform over L^2 pixels gives L^2 separate frequency components, each of which can be thought of as the result of one filter. Since the Fourier transform of a real signal is Hermitian symmetric, this reduces the number of independent frequencies to $L(L/2+1)$ if L is even.

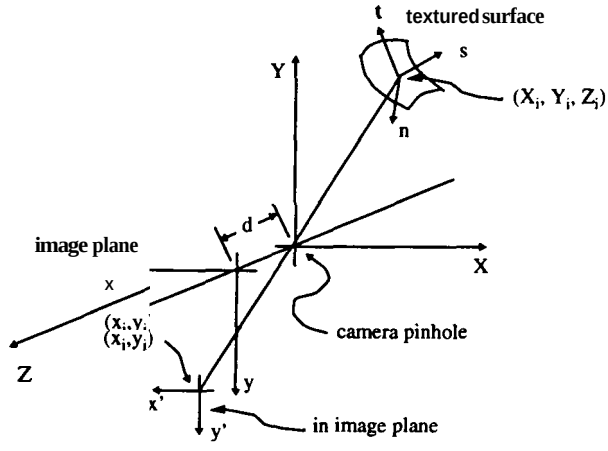


Figure 7: Coordinate frames used in derivation

$$\begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} \frac{p^2 + rq^2}{p^2 + q^2} & \frac{pq(1-r)}{p^2 + q^2} & p & rX_i \\ \frac{pq(1-r)}{p^2 + q^2} & \frac{rp^2 + q^2}{p^2 + q^2} & 4 & rY_i \\ -p & -4 & 1 & rZ_i \\ 0 & 0 & 0 & r \end{bmatrix} \tag{3}$$

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

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

Under perspective, these points project to the image plane at

with

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

where

$$
A = \frac{Z_i}{d\,(p^2 + q^2)\,(px_i + qy_i - d)}
$$

and we have substituted the value 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

$$
\begin{aligned}
&\cos\,(2\pi\,(\,(s_{x_1} x' + s_{y_1} y')\,u_0 + (t_{x_1} x' + t_{y_1} y')\,v_0\,)\,) \\
&\cos\,(2\pi\,(\,(s_{x_2} x' + s_{y_2} y')\,u_0 + (t_{x_2} x' + t_{Y2} y')\,v_0\,)\,)
\end{aligned}
$$

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

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

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[9] 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[18], 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.

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

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[50]. 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 could then be dominated by noise and approximation errors.

Figure 9 shows four identical plates with different Brodatz[9] 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 (p,q) 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. Each 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.

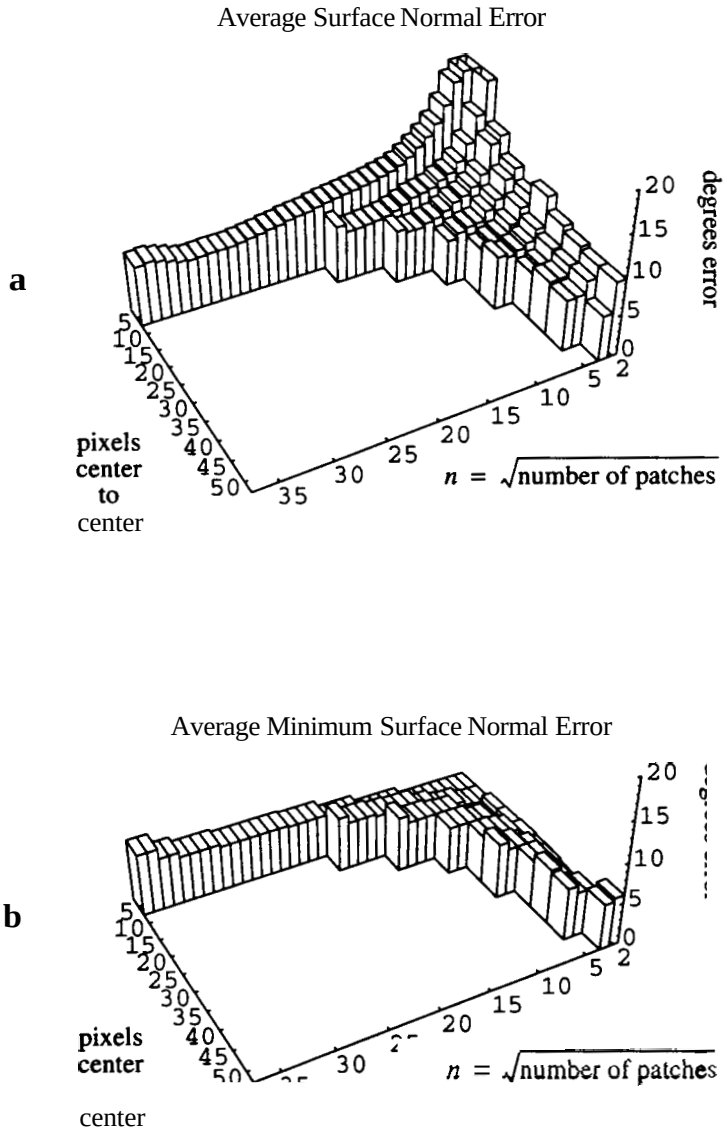


Figure 10: Average errors in surface normal from the four test images for different patch center-to-center distances and different numbers of patches.

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 $n \times n$ 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 is taken 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

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

We cannot simply invert this relationship for the frontalization, because we don't know the Z_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 Z_{ref} from the same plane and with the same texture. The 4x4 homogeneous transformation locating the surface patch's local coordinate frame would be

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

Using these transformation parameters and solving Equation (16) for s and t gives

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

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

(21)

$$\begin{bmatrix} u_{frontal} \\ v_{frontal} \end{bmatrix} = \begin{bmatrix} -Z_{ref}/d & 0 \\ 0 & -Z_{ref}/d \end{bmatrix} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} = S_{ref} \begin{bmatrix} u_0 \\ v_0 \end{bmatrix}$$

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

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 Creating a Dendrogram of Hypotheses

Our segmentation algorithm consists of building a dendrogram of the image. A dendrogram, defined by Duda and Hart[13], is a hierarchical clustering represented by a tree with individual feature vectors as the leaves. At the beginning, each feature vector is its own cluster. We can build a dendrogram by successively merging the two clusters that are most similar (by some measure) into a new cluster. Different levels of the dendrogram represent different clusterings. In our algorithm, clusters correspond to hypotheses. After each merge, the number of hypotheses is reduced by one. We only allow merges between four-connected hypotheses, and at all times each patch is a member of only one hypothesis. Different levels of our dendrogram correspond to different segmentations of the images. We show selected levels of three dendrograms in the results in Figure 11, where each of the 25 squares is a little version of the image with merged patches shaded the same. The first level has every patch as a separate region, while the last level has every patch in the same region. For our three example images, the best segmentation occurs when there are four regions.

Building a good dendrogram (one that shows a good segmentation at some level) depends critically on the merge criterion. For simple feature vectors, a suitable merge criteria is Euclidean or Mahalanobis distance. Our problem is more complex, and we have chosen a description length criterion instead. The idea, whose primary advocate is Jarma Rissanen[46], is that the best grouping of data is the one that requires the least number of bits to describe. If two hypotheses are similar, an efficient coding scheme can describe their union plus their deviations from each other in fewer bits that it can describe them separately. This means they should be merged. The description length criteria is desirable because it has only a few arbitrary parameters, and because it formalizes the trade-off between the complexity of the model and the quality of the fit.

The first addend of this formula gives the number of bits it takes to label each peak with an integer giving its cluster number. The second addend is Rissanen's estimate of the number of bits to describe the mean and standard deviation vectors of each cluster. The third addend is the number of bits it takes to describe the deviation of the peaks from their cluster centers. It is based on a Gaussian probability distribution, and is essentially $-\log_2 P(\text{peaks}|\mu, \sigma)$. The standard deviation for this term could be computed from the clusters themselves, however small clusters will give unstable values. We set the standard deviation to 0.01 for our experiments. This value is reasonable, since the Nyquist-limited range of frequencies is [-0.5,0s].

The dendrogram of hypotheses is built this way: At the beginning, each patch is its own hypothesis. To compute the next level in the dendrogram, all pairs 4-connected hypotheses are temporarily merged into new, trial hypotheses. Each trial merge involves computing the surface normal of the merged region, frontalizing the peaks, and building a dendrogram of the frontalized peaks. We take the merge that gives the largest reduction in description length. The "magic numbers" are Leclerc's value of b for the chain-code description length, the standard deviation of the peaks, and the level in the dendrogram of hypotheses to choose as the final segmentation. The first two parameters are easy to choose. The final dendrogram level is more difficult. As our algorithm stands now, we must choose this manually, although some measure based on description length would be appropriate here.

5.4 Results

The results of our segmentation algorithm on three different images are shown in Figure 11. These images were generated with the same program used for the images in Figure 9 using Brodatz textures. Selected levels of the dendrograms are in the left column and the edges corresponding to the four-region level of the dendrogram are in the right column. The algorithm clearly finds the correct regions. The regions are blocky because we compute the local frequency spectra on 15-pixel centers, rather than centered at every pixel. There are some errors in the edges near the texture boundaries due mostly to patches that overlap into two or more textures. These results demonstrate 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. In Figure 12 we show the surface normals associated with each of the final hypotheses. The average error in surface normal for the three images is 6.7°, 3.8°, and 3.4° respectively. (We take the uniform backgrounds to have a surface normal of $(p,q) = (0,0)$.)

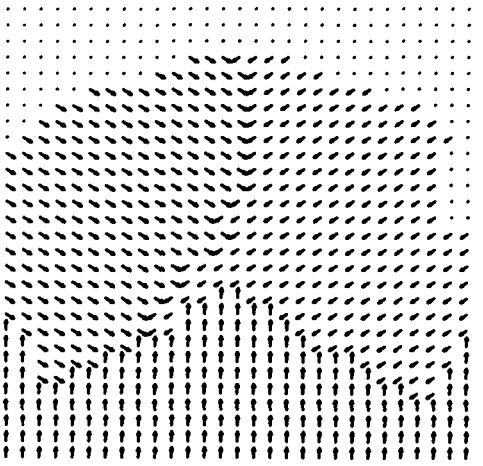


image 1

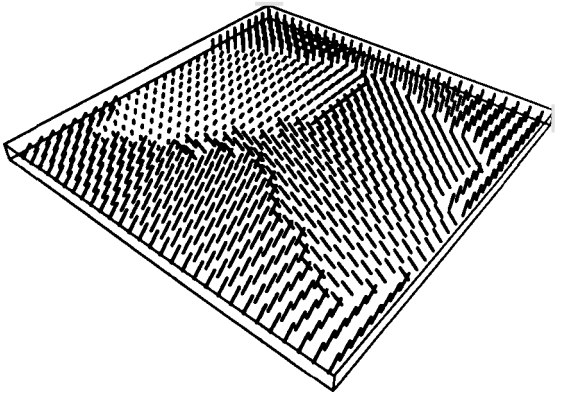


image 1

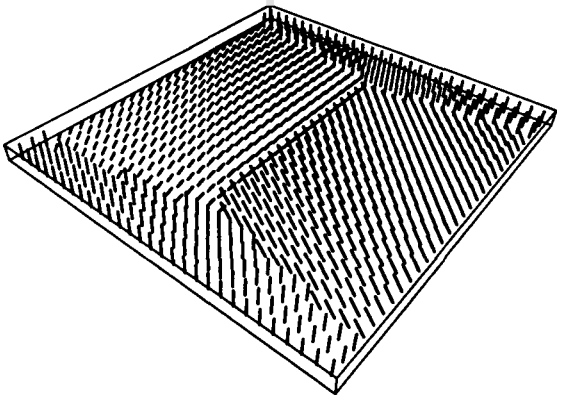


image 2

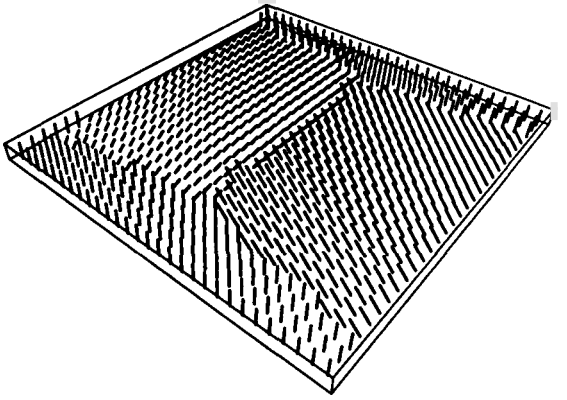


image 3

Figure 12: Surface normals of segmented images. The average error in surface normal in the three images is 6.7'. 3.8'. and 3.4" respectively.

It could be argued that accounting for the effect of surface normals is not necessary, because the differences in frequency content of distinct textures will be enough to overshadow the deformations caused by shape on a single textured surface. Using our algorithm, this is apparently not true. The segmentation shown in Figure 13 was computed by making one small change to our program: we altered the surface normal subroutine such that it always returned $(p,q) = (0,0)$. This meant that all the textures were considered to be already frontalized, so the deformations caused by 3D effects were not factored out. The resulting four regions give a segmentation that is quite poor. The center region contains significant amounts of two distinct textures. None of the other levels in the dendrogram is any better.

It should not be a surprise that local spatial frequency is a good way to do shape and segmentation simultaneously, 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. In fact, we expect the space/frequency representation to help in solving many combined problems in computer vision. 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[33] 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 [34]. This represents another unification of algorithms. 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".

References

[1] Aitken, G.J.M. & Jones, P.F. (1990). 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.

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

[3] Aloimonos, J. & Swain, M. (1988). Shape From Patterns: Regularization. International Journal of Computer Vision, 2 (no. 2, September), 171-187.

[4] Bajcsy, R. & Lieberman, L. (1976). Texture Gradient as a Depth Cue. Computer Graphics and Image Processing, 5, 52-67.

[5] Bell, B. W. & Koliopoulos, C. L. (1984). Moire Topography, Sampling Theory, and Charged-Coupled Devices. Optics Letters, 9 (no. 5, May), 171-173.

[6] Bergen, J. R. & Adelson, E. H. (1988). Early Vision and Texture Perception. Nature 333 (26 May), 363-364.

[7] Blake, A. & Marinos, C. (1990). Shape from Texture: Estimation, Isotropy and Moments. Artificial Intelligence, 45,323-380.

[8] Bovik, A. C., Clark, M., & Geisler, W. S. (1990). Multichannel Texture Analysis Using Localized Spatial Filters. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12 (January), 55-73.

[9] Brodatz, P. (1966). Textures: A Photographic Album for Artists and Designers. New York: Dover.

[10] Brown, L. G. & Shvaytser, H. (1990). Surface Orientation from Projective Foreshortening of Isotropic Texture Autocorrelation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12 (June), 584-588.

[11] Cetica, M., Francini, F., & Bertani, D. (1985). Moire With One Grating and a Photodiode Array. Applied Optics, 24 (no. 11, 1 June), 1565-1566.

[12] DeFatta, D. J., Lucas, J. G., & Hodgkiss, W. S. (1988). Digital Signal Processing: A System Design Approach. New York: John Wiley & Sons, p. 270.

[13] Duda, R. O. & Hart, P. E. (1973). Pattern Classification and Scene Analysis. New York: John Wiley & Sons, p. 229.

[14] Dyer, C. R. & Rosenfeld, A. (1976). Fourier Texture Features: Suppression of Aperture Effects. IEEE Transactions on Systems, Man, and Cybernetics, SMC-6 (October), 703-705.

[15] Fleet, D. J., Jepson, A. D., & Jenkin, M. R. M. (1991). Phase-Based Disparity Measurement. CVGIP: Image Understanding, 53 (no. 2, March), 198-210.

[16] Fogel, I. & Sagi, D. (1989). Gabor Filters as Texture Discriminator. Biological Cybernetics, 61 (June), 103-113.

[17] Garding, J. (1992). Shape from Texture for Smooth Curved Surfaces in Perspective Projection. Journal of Mathematical Imaging and Vision, 2 (no. 4, December), 327-350.

[18] Gaskill, J. D. (1978). Linear Systems, Fourier Transforms, and Optics. New York: John Wiley & Sons, p. 308.

[37] Leclerc, Y. G. **(1989).** Constructing Simple Stable Descriptions for Image Partitioning. International Journal of Computer Vision, 3, 73-102.

[38] Malik, J. & Perona, P. **(1990).** Preattentive Texture Discrimination with Early Vision Mechanisms. Journal of the Optical Society of America - A, 7 (no. 5, May), **923-932.**

[39] Malik, J. & Rosenholtz, R. **(1993).** A Differential Method for Computing Local Shape-From-Texture for Planar and Curved Surfaces. IEEE Conference on Computer Vision and Pattern Recognition, **267-273.**

[40] Matsuyama, T., Miura, S, & Nagao, M. **(1983).** Structural Analysis of Natural Textures by Fourier Transformation. Computer Vision, Graphics and Image Processing, **24,** 347-**362.**

[41] Morimoto, **Y.,** Seguchi, Y. & Higashi, T. **(1988).** Application of Moire Analysis of Strain Using Fourier Transform. Optical Engineering, **27** (no. **8,** August), **650-656.**

[42] Nelson, B., Papanikolopoulos, N., and Khosla, P. **(1993).** Dynamic Sensor Placement Using Controlled Active Vision. International Federation of Automatic Control **1993** World Congress.

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

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

[45] Pentland A. P. **(1988).** The Transform Method for Shape from Shading. M.I.T Media Lab Vision Sciences Technical Report **106** (July **15).**

[46] Rissanen, J. **(1983).** A Universal Prior for Integers and Estimation by Minimum Description Length. The Annals of Statistics, 11 (no. 2), 416-431.

[47] Reed, T. R. & Wechsler, H. **(1990).** Segmentation of Textured Images and Gestalt Organization Using Spatial/Spatial-Frequency Representations. IEEE Transactions on Pattern Analysis and Machine Intelligence, **12** (January), 1-**12.**

[48] Sanger, T. D. **(1988).** Stereo Disparity Computation Using Gabor Filters. Biological Cybernetics, **59,405-418.**

[49] Subbarao, M. **(1988).** Parallel Depth Recovery by Changing Camera Parameters. Second International Conference on Computer Vision, **149-155.**

[50] Super, B. J. and Bovik, A. C. **(1991).** Three-Dimensional Orientation from Texture Using Gabor Wavelets. SPIE Conference on Visual Communications and Image Processing (November).

[51] Super, B. J. & Bovik, A. C. **(1992).** Shape-from-Texture by Wavelet-Based Measurement of Local Spectral Moments. IEEE Conference on Computer Vision and Pattern Recognition, **296-301.**

[52] Turner, M. R. **(1986).** Texture Discrimination by Gabor Functions. Biological Cybernetics, 55 (October), 7**1-82.**

[53] Voorhees, **H.** & Poggio, T. **(1988).** Computing Texture Boundaries from Images." Nature, **333 (26** May), **364-367.**