

Recovering Object Surfaces From Viewed Changes in Surface Texture Patterns

Peter N. Belhumeur

Dept. of Electrical Engineering
Yale University
New Haven, CT 06520

Alan L. Yuille

Division of Applied Science
Harvard University
Cambridge, MA 02138

Abstract

This paper explores the reconstruction of object surfaces from viewed changes in surface texture patterns. Our approach differs from those in the past in that instead of simply producing local estimates of the surface orientation, our algorithm recovers complete surfaces. Past approaches [1, 16, 4, 9, 8, 11] only found the surface orientation locally and, therefore, did not take advantage of the surface integrability constraint. Unlike [16, 7, 3], our algorithm does not assume that the surface texture pattern is isotropic; and unlike [14, 13], our algorithm does not assume that the viewed surface is at some point fronto-parallel. Furthermore, our algorithm has mechanisms for handling texture boundaries and, consequently, does not produce erratic results in the regions abutting these boundaries. Results on real images are presented demonstrating the potential of our approach.

1 Introduction

Why should computer vision researchers care about recovering surfaces from variations in texture? Are there really that many instances in which texture variations give us enough information to recover surfaces? Gaze across a grassy lawn, stare at the bark on a towering elm tree, or examine the ribbed pile of corduroy slacks: If viewed monocularly, only texture provides the cue for recovering the underlying surface structure. Still, in spite of notable recent advances, shape-from-texture algorithms have not yet proven to be practical. Researchers have produced accurate results by testing their algorithms on highly constrained scenes – scenes in which texture homogeneity assumptions hold.

We feel, however, that in order for shape-from-texture algorithms to become practical three obstacles must be overcome. First, techniques are needed for making texture measurements near texture boundaries. Since texture measurements can not be defined at a point, but must be defined within some neighborhood of a point, if this neighborhood includes a texture boundary the measurement is invalid – i.e. the texture homogeneity assumption is violated. Second, a shape-from-texture algorithm should attempt to recover more than simply the local surface orientation, it should recover complete surfaces, i.e. the relative depth of points within a homogeneously textured re-

gion. Most past approaches to shape-from-texture have focused on recovering the local surface orientation; yet, simply recovering estimates of the local surface orientation does not ultimately yield surface reconstructions. Third, techniques are needed to determine when homogeneity assumptions are valid and, therefore, allow for the recovery of surfaces. The majority of images contain only select regions in which texture homogeneity assumptions hold. Thus, for shape-from-texture algorithms to produce reliable results on these images, mechanisms in the algorithm must exist for suspending the process in regions of non-homogeneous texture.

This paper takes steps toward solutions for the first two of the above listed problems and suggests a solution to the third. First, we develop a method for making texture measurements in regions adjoining texture boundaries. To do this, we use the texture boundaries provided by a texture segmentation algorithm to guide the measurement process. Our texture measurement is based on a local weighted average of the matrix $\nabla I \nabla I^T$, where $I = I(x, y)$ is the image luminance. The averaging is weighted according to a deformable Gaussian window, where the boundary information is used to determine the position and shape of the window. In the middle of the textured region, the Gaussian window is circularly symmetric. However, near a texture boundary the Gaussian window flattens, instead of crossing over the boundary. Thus, our texture measurements can be made arbitrarily close to texture boundaries.

Second, for each textured region in the image, we *directly* estimate the corresponding surface in space. This differs from past approaches [1, 16, 4, 9, 8, 11] which only recover the surface orientation locally. Because our shape-from-texture algorithm recovers complete surfaces, it takes advantage of the surface integrability constraint – i.e. if depth is denoted by $d = d(x, y)$, then $\partial^2 d / \partial x \partial y = \partial^2 d / \partial y \partial x$. Furthermore, unlike [16, 7, 3], our algorithm does not assume that the surface texture pattern is isotropic; and unlike [14, 13], our algorithm does not assume that the viewed surface is at some point fronto-parallel.

Third, we suggest a measure for determining when a prespecified region in the image does in fact satisfy

the homogeneity assumption. Without a measure of this type, there is no way to gauge the validity of the surface reconstructions.

2 Texture Measurement Method

Much of the difficulty in recovering surfaces from texture variations stems from the fact that texture can not be defined at a point, but rather must be defined as a measurement over region in the image – a region roughly the size of the period of the texture. Many people have proposed various methods for locally recovering surface orientation from texture: Bajcsy and Lieberman [1] used the peaks in the FFT spectrum, Witkin [16] and Blake and Marinos [4] used second order statistics of line elements, Garding [8] used second order moments of the FFT spectrum, Super and Bovik [14] used second order moments of narrow band Gabor wavelets, and Malik and Rosenholtz [11] used the whole FFT spectrum. A disadvantage to all of these methods was that no mechanism was proposed for adjusting the measurement region so that it does not cross over texture boundaries.

We propose a method which exploits the texture boundaries found by a texture segmentation algorithm, before recovering both depth and surface orientation. Our texture measurement is based on a local weighted average of the matrix $\nabla I \nabla I^T$, where $I = I(x, y)$ is the image luminance. While the use of this matrix as a texture measurement matrix is not novel, what separates our measurement method is that the averaging in the matrix is weighted by a deformable Gaussian window. In the middle of an object, the Gaussian window is circularly symmetric. However, along a texture boundary the Gaussian window elongates and displaces to avoid crossing the boundary. Our method for gathering measurements using deformable Gaussian shaped windows is a modification to the nonlinear smoothing operation described by Nitzberg and Shiota [12]. Our method differs in that the shape of the smoothing kernel is determined not by the gradient of the image intensity, but rather by pre-computed image texture boundaries. (This method could be improved to use image texture gradients instead of pre-computed texture boundaries.)

Our texture measurement, computed at each point (x, y) in the image, is given by the following matrix

$$F = \begin{pmatrix} \frac{\Psi \circ (\frac{\partial I}{\partial x})^2}{\Psi \circ (I - \Psi \circ I)^2} & \frac{\Psi \circ (\frac{\partial I}{\partial x} \frac{\partial I}{\partial y})}{\Psi \circ (I - \Psi \circ I)^2} \\ \frac{\Psi \circ (\frac{\partial I}{\partial x} \frac{\partial I}{\partial y})}{\Psi \circ (I - \Psi \circ I)^2} & \frac{\Psi \circ (\frac{\partial I}{\partial y})^2}{\Psi \circ (I - \Psi \circ I)^2} \end{pmatrix} \quad (1)$$

where I is the image luminance; $\Psi \circ (I - \Psi \circ I)^2$ is a normalizing factor to correct for image luminance gradients; Ψ is a nonlinear local averaging operator defined by

$$\Psi \circ f = \iint_W e^{-\frac{1}{2}(\vec{x}' - \vec{\mu})^T \Sigma^{-1}(\vec{x}' - \vec{\mu})} f(\vec{x}') d\vec{x}'$$

$\vec{x}' = (x', y')^T$; $\vec{\mu} = \vec{\mu}(x, y)$ specifies the displacement of the Gaussian window function; $\Sigma = \Sigma(x, y)$ species

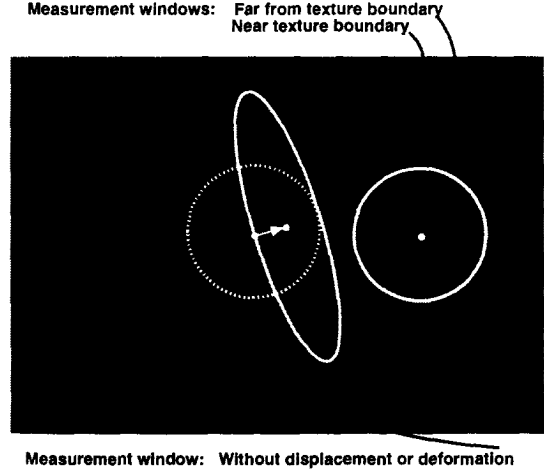


Figure 1: **Texture Measurement Window:** In the center of the textured region the Gaussian window is circularly symmetric. At texture boundaries the window deforms and displaces so as not to cross over the texture boundary. The solid curves indicate the actual shapes of the windows used. The dotted curve indicates the shape and location of the texture measurement window abutting the boundary, before it is displaced and deformed; the white arrow indicates the direction and size of the displacement.

the shape and orientation of the window function; and W specifies the region of integration, i.e. the size of the window, and is taken as $W = \{\vec{x}' \mid (\vec{x}' - \vec{\mu})^T \Sigma^{-1}(\vec{x}' - \vec{\mu}) \leq 1\}$. Note that the texture measurement matrix $F = F(x, y)$ is a function of image position.

In the center of the textured region

$$\vec{\mu} = (x, y)^T$$

$$\Sigma = \sigma^2 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

so the Gaussian window function is not displaced and is circularly symmetric. In a region abutting a texture boundary, $\vec{\mu}$ is displaced in the direction of the boundary normal toward the interior of the textured region; and, Σ is flattened in the direction of the boundary normal, while preserving $\|\Sigma\| = \sigma^2$ (for an illustration see Fig. 1 and for implementation details see [12]). The constant σ must be chosen to be at least as large as the period of the texture pattern. If σ is smaller than this, then the texture measurement window is too small and the texture measurements are, consequently, unreliable. If σ is chosen significantly larger than this, then the texture measurements will ultimately produce a flattening of the surface curvature. For long planar surfaces this is not a problem, but for surfaces with sharp curvature the flattening is noticeable. Naturally, this trade-off is inherent in all shape-from-texture algorithms.

It has been known for quite sometime in the signal processing community that the local averaging of

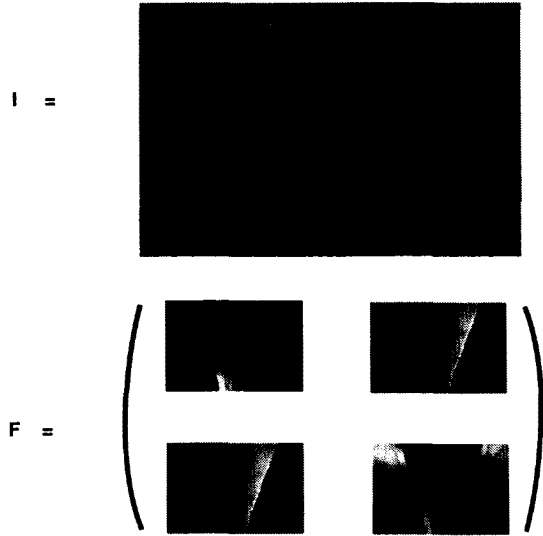


Figure 2: **Texture Measurements:** The top illustration shows an image of a tree trunk in front of a dried up lake bed. The bottom illustration shows the texture measurement matrix F resulting from applying Eq. 1 to the image at the top of the figure.

the matrix $\nabla I \nabla I^T$ is nearly equivalent to the second moment matrix of the localized Fourier Transform. Consequently, local averaging of $\nabla I \nabla I^T$ is also nearly equivalent to Super and Bovik's 2nd order local spectral moments [14], computed using Gabor wavelets. Krumm and Shafer [10] and Rosenholtz and Malik [11] differ in that they use the whole FFT spectrum, not simply the second order moments. Naturally, there are advantages and disadvantages to using the whole FFT spectrum. The principal advantage is that all of the texture information is used when the whole FFT spectrum is computed; the principal disadvantage is that window size needed to reliably compute the FFT spectrum is sizable – so much so that a flattening of surface curvature will result.

We should point out that these are certainly not the only type of texture measurement methods. Witkin [16], Blostein and Ahuja [6], Kanatani and Chou [9], and Blake and Marinos [3] measure texture by measuring the density of texels or edges. These methods have the advantage of being slightly “invariant” to image focus.

However, this paper is not intended as a study of texture measurement methods for shape-from-shading, as we are not – at this point – certain as to what is the best choice for the texture measurement. However, we are certain that whatever measurement is used, it must respect texture boundaries.

To illustrate our method, we have computed the matrix F for an image with two homogeneous textures. In the top of Figure 2, we have included an image of a tree trunk in front of a dried up lake bed. At the bottom of the figure, we have included the texture measurement matrix F resulting from applying Eq. 1 to the image at the top of the figure. Note

that the measurement matrix F respects the texture boundaries.

3 Estimating Surfaces

It is easy to see that if the texture on the surface of an object is roughly homogeneous, then changes in the texture measurement matrix F are caused by changes in the object depth and surface orientation. Yet, how does this measure vary with changes in the distance and orientation of the observed surface?

One can show that if we know the texture measurement matrix F for a specific depth and surface orientation, then we can calculate – in the absence of noise – what the measurement matrix should be for any depth and surface orientation. In particular, if $F = M$ when the depth d equals the focal length f , and surface orientation $\nabla d = 0$, then for any arbitrary depth and surface orientation

$$F \simeq A M A^T \quad (2)$$

where A is a 2×2 symmetric matrix with components

$$\begin{aligned} a_{11} &= \frac{d_x^2 \sqrt{d_x^2 + d_y^2 + (d/f)^2} + d_y^2 (d/f)}{d_x^2 + d_y^2} \\ a_{12} &= \frac{d_x d_y \sqrt{d_x^2 + d_y^2 + (d/f)^2} - d_x d_y (d/f)}{d_x^2 + d_y^2} \\ a_{22} &= \frac{d_y^2 \sqrt{d_x^2 + d_y^2 + (d/f)^2} + d_x^2 (d/f)}{d_x^2 + d_y^2} \end{aligned}$$

$d_x = \partial d / \partial x$; and $d_y = \partial d / \partial y$. The reader has been spared the derivation of the A matrix.

Equation 2 is approximate for several reasons. First, it assumes that the surface in space is locally planar. Second, it ignores the fact that we have deliberately altered the measurements near texture boundaries. Third, it assumes that the small angle approximation is valid, i.e. the angle a ray originating from (x, y) makes with the optical axis is small.

The matrix A is an affine transformation specifying – to first order – how points on the tangent plane of the object's surface are mapped to the image plane. This matrix A is a natural extension of Super and Bovik's orthographic projection matrix [14] to perspective projection, and it differs from Garding [8] in that A is defined relative to a fixed image plane coordinate system.

3.1 Without Surface Integrability

Garding [8] effectively found the local surface orientation and depth by assuming the matrix M was known and then solving

$$F = A M A^T \quad (3)$$

for $\nabla d = \{d_x, d_y\}^T$ and d . As Garding pointed out, however, ∇d in Eq. 3 can only be determined up to a sign, i.e. the sign of the tilt direction of the surface is ambiguous. To get around this, Garding used depth

estimates at neighboring points to correct for the sign ambiguity.

Rosenholtz and Malik [11] extended Garding's techniques, recovering not only local surface orientation, but surface shape (i.e. curvature) as well. To overcome the sign ambiguity in Eq. 3 and to recover surface shape, they incorporated additional information by considering the texture variation for neighboring points in a number of different directions. They tested their model by applying it at a single point in the image. Although, they did not attempt to recover the complete surface in space, they could have applied their model repeatedly – as Garding did – to recover depth and surface orientation estimates for points in the image. Yet, in doing this, all of the depth and surface orientation estimates would be done independent of one another.

A disadvantage of both methods – apart from the fact that they have no mechanisms for computing texture measurements near boundaries – is that because they only recover surface information locally, they do not take advantage of the surface integrability constraint. Each estimate of the depth d and the surface orientation ∇d is found independently; yet, the surface orientation ∇d should actually be determined by the relative depths at neighboring points.

3.2 Requiring Surface Integrability

We argue that estimates of d and $\nabla d = \{d_x, d_y\}$ at a point in the textured region must be coupled to the estimates for neighboring points. Thus, we require that ∇d at a point in the image be determined by the depth estimates at neighboring points. This requirement effectively enforces a surface integrability constraint.

Because neighboring points are coupled together, we find the collection of depth estimates d which best agrees with Eq. 3 for each point (x, y) . To do this, we use a least squares optimality criterion, choosing d and M as those which minimize

$$E[d, M] = \iint_X \sum_{ij} (F_{ij} - \{AM A^T\}_{ij})^2 dx dy \quad (4)$$

where the notation $\{\cdot\}_{ij}$ denotes the ij^{th} element of a matrix, and X is a region of homogeneous texture.

The strength of this technique is fourfold. First, the method has a mechanism for recovering the portions of surfaces near texture boundaries. Second, the method recovers complete surfaces, not simply the local surface orientation. This differs from past approaches [1, 16, 4, 9, 8, 11] which only find the surface orientation locally and, therefore, do not take advantage of the surface integrability constraint – $\partial^2 d / \partial x \partial y = \partial^2 d / \partial y \partial x$. Third, because we recover complete surfaces, as a byproduct we recover the surface shape as well. Measurements for the resulting surface curvature can be easily made from the reconstructed surface. Thus, we circumvent all of the machinery in [11] needed for estimating curvature. Fourth, because the tilt ambiguity is implicitly resolved, we eliminate – in most instances – the complications caused by Garding's [8] "phantom surfaces."



Figure 3: **Phantom Surfaces:** Each of the sketched surfaces would produce nearly identical texture measurements F . The only cue to differentiating between them is small perspective changes. Thus, if the first surface on the left is the true surface, than either the middle or right surface may – in presence of noise – actually minimize Eq. 4.

3.3 Requiring Surface Smoothness

Without the information given by perspective, the shape-from-texture problem is underconstrained. Since we rely on perspective information to resolve the tilt ambiguity, i.e. the sign of ∇d , if perspective information is not strong enough or the noise in the image is great enough, then there are many different possible surface reconstructions.

This fact is easiest to illustrate in 1-D. Figure 3 contains the sketches of three surfaces. Each of the sketched surfaces would produce nearly identical texture measurements F . The only cue to differentiating between them is small perspective changes. Thus, if the first surface on the left is the true surface, than either the middle or right surface may – in presence of noise – actually minimize Eq. 4. Naturally, these are not the only surfaces for which this may be true: we can construct an infinite number of possible "phantom surfaces" (the term belongs to Garding [8]) by inverting the sign ∇d at random points along the surface.

To get around this, we constrain the space of possible surfaces by putting a constraint on the smoothness of the reconstructed surfaces. To do this we use a thin plate type surface prior (see Blake and Zisserman [5] and Szeliski [15]) which biases in favor of surfaces with small curvature. The optimality criterion in Eq. 4 becomes

$$E[d, M] = \iint_X \sum_{ij} (F_{ij} - \{AM A^T\}_{ij})^2 dx dy + \alpha \iint_X \nabla^2 d dx dy \quad (5)$$

where $\nabla^2 d = \partial^2 d / \partial x^2 + \partial^2 d / \partial y^2$.

3.4 Verifying Texture Homogeneity

Because the majority of images contain only select regions in which texture homogeneity assumptions hold, mechanisms are needed for suspending the shape-from-texture reconstruction process in regions of non-homogeneous texture. Initial experiments have

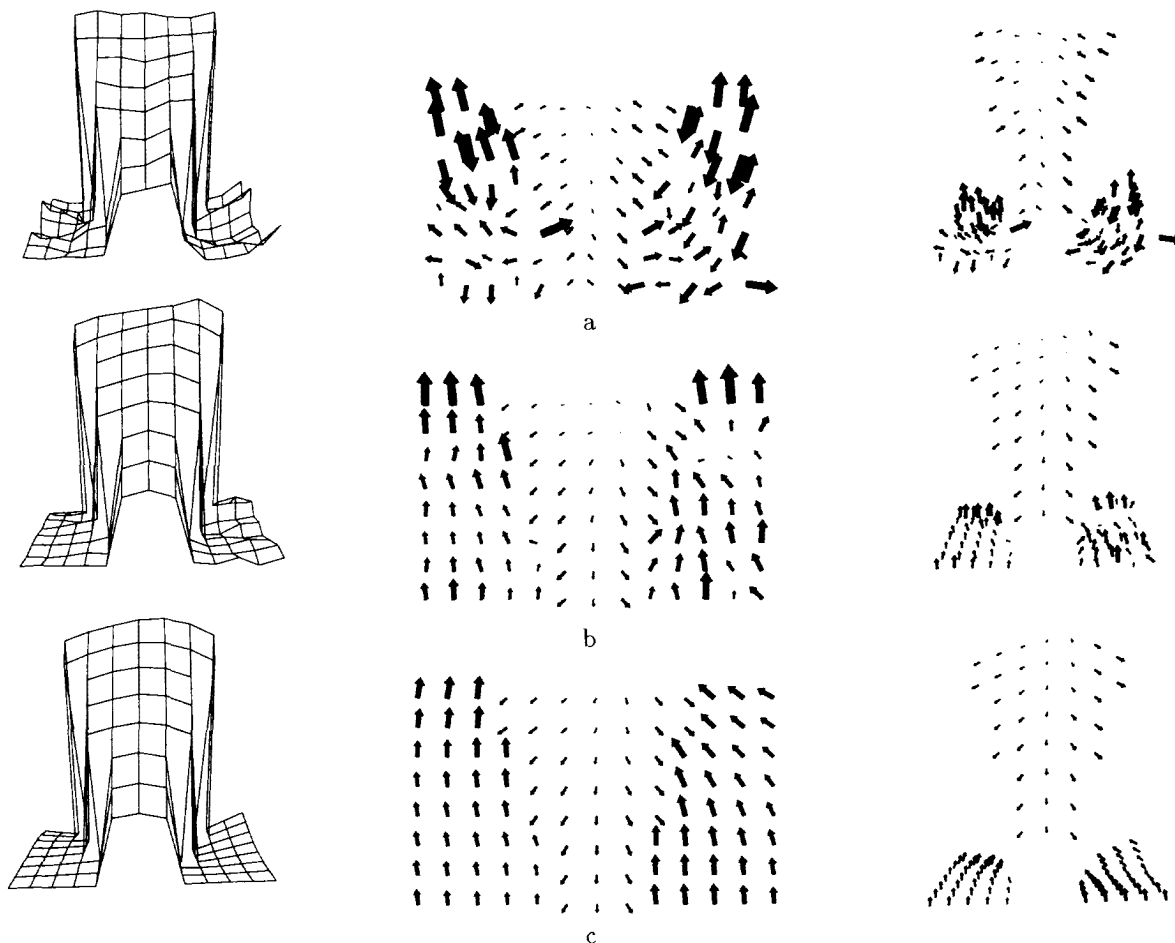


Figure 4: **Results:** (a) Without surface integrability constraint. (b) With surface integrability constraint. (c) With surface smoothness prior.

shown that the optimality criterion in Eq. 5 serves as a good metric for determining the validity of homogeneity assumptions. Surfaces on which the texture patterns are far from homogeneous have large minimum values for Eq. 5. Thus, the magnitude of the minimum value for $E[d, M]$ may prove to be a nice indicator of the reliability of the surface reconstructions.

4 Results

In this paper, we have discussed three methods for shape-from-texture surface reconstructions. In this section, we compare the results obtained by applying each of the three methods to the data in Fig. 2. Details of each implementation can be found in [2].

Figure 4a is the result of solving Eq. 3 for ∇d . Here the surface orientation is recovered locally, without using a surface integrability constraint. The left image of Figure 4a is a wire frame reconstruction of d . The center image of Figure 4a is a vector field corresponding to ∇d . The length and width of the arrows are proportional to $\|\nabla d\|$. The right image of Figure 4a is the vector field ∇d mapped onto the reconstructed surface. We have not attempted to correct for the tilt

ambiguity (as Garding did), because careful examination of the wire frame will show that this correction would be as often wrong as it was right.

For this result, we have assumed that the M matrices for the two homogeneously textured regions in Fig. 2 have already been computed. An effective method for computing M is developed by Rosenholtz and Malik [11]. Note that because the image textures in Fig. 2 are not isotropic, we can not assume that the M matrices are scaling of the identity matrix. Thus, the algorithms proposed in [16, 7, 3] could not handle scenes of this type. And, because neither the lake bed nor the tree trunk is anywhere fronto-parallel, the algorithms in [14, 13] could not be initialized.

Note that because M must be estimated along with the depth, the d can only be determined up to a scaling factor. Therefore, we can not actually recover the depth difference between the tree trunk and the lake bed. Yet, we have chosen the scaling to aid in the interpretation of the results.

Figure 4b is the result of solving Eq. 4 for d , ∇d , and M . Here the depth and surface orientation is recovered using a surface integrability constraint. The

left image of Figure 4b is a wire frame reconstruction of d . The center image of Figure 4b is a vector field corresponding to ∇d . The length and width of the arrows are proportional to $\|\nabla d\|$. The right image of Figure 4b is the vector field ∇d mapped onto the reconstructed surface. Note that the wire frame reconstruction is significantly closer to the actual surface than the reconstruction obtained without the integrability constraint. Furthermore, in most instances the tilt ambiguity – as seen in the vector field of the center image in Fig. 4b – has been resolved.

Figure 4c is the result of solving Eq. 5 for d , ∇d , and M . Here the depth and surface orientation is recovered using a surface integrability constraint and a smoothness constraint. The left image of Figure 4c is a wire frame reconstruction of d . The center image of Figure 4c is a vector field corresponding to ∇d . The length and width of the arrows are proportional to $\|\nabla d\|$. The right image of Figure 4c is the vector field ∇d mapped onto the reconstructed surface. Note that the results have improved yet again – particularly in places where the perspective cues are weak.

5 Summary and Conclusions

In spite of a considerable body of work on the shape-from-texture problem, none of the published algorithms, to our knowledge, are truly robust, general purpose solutions. While we believe that the techniques presented in this paper are a step in the right direction; clearly much work needs to be done before surface reconstructions from texture prove to be practical.

We must investigate – *in depth* – when the techniques outlined in this paper work and when they break down. We must investigate combining the information provided by shape-from-texture with other visual cues. Certainly one can not hope to construct a vision system which considers only texture cues. We must investigate ways of automating the process of finding texture boundaries. In this paper, we have assumed that they are known, but clearly this is an equally open area of research. Finally, we must test our suggested scheme for determining the reliability of the surface reconstructions. Because the majority of images contain only select regions in which texture homogeneity assumptions hold, mechanisms in the algorithm must exist for suspending the process in regions of non-homogeneous texture.

Acknowledgements

We are indebted to David Mumford, Tai Sing Lee, and Jitendra Malik for their many ideas and suggestions. This research was partially supported by the Brown/Harvard/MIT Center for Intelligent Control Systems with U.S. Army Research Office grant number DAAL03-86-K-0171.

References

- [1] R. Bajcsy and L. Lieberman. Texture gradient as a depth cue. *CGIP*, 5:52–67, 1976.
- [2] P. N. Belhumeur and A. L. Yuille. An algorithm for recovering object surfaces from viewed variations in surface texture patterns. Technical report, Yale University, 1994.
- [3] A. Blake and R. Cipolla. Robust estimation of surface curvature from deformation of apparent contours. In O. Faugeras, editor, *Proc. 1st European Conference on Computer Vision*, pages 465–474. Springer-Verlag, 1990.
- [4] A. Blake and C. Marinos. Shape from texture: estimation, isotropy and moments. *J. Artificial Intelligence*, in press, 1989.
- [5] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, USA, August 1987.
- [6] D. Blostein and N. Ahuja. Representation and three-dimensional interpretation of image texture: an integrated approach. Technical report, University of Illinois at Urbana-Champaign, 1987.
- [7] L.S. Davis, L. Janos, and Dunn S.M. Efficient recovery of shape from texture. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 5(5):485–492, 1983.
- [8] J. Garding. Shape from texture for smooth curved surfaces. In *Proc. European Conf. on Computer Vision*, pages 630–638, 1992.
- [9] L. Kanatani and T. Chou. Shape from texture: general principle. *Artificial Intelligence*, 38:1–48, 1989.
- [10] J. Krumm and S. Shafer. Shape from periodic texture using spectrogram. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 284–301, 1992.
- [11] J. Malik and R. Rosenholtz. A differential method for computing local shape-from-texture for planar and curved surfaces. In *Proc. Conf. Computer Vision and Pattern Recognition*, New York, New York, 1993.
- [12] M. Nitzberg and D. Mumford. The 2.1D sketch. In *Proc. Int. Conf. on Computer Vision*, pages 138–144, 1990.
- [13] K. Sakai and L. Finkel. A shape-from-texture algorithm based on human visual psychophysics. In *Proc. Conf. Computer Vision and Pattern Recognition*, pages 527–532, 1994.
- [14] J.S. Super and A.C. Bovik. Shape-from-texture by wavelet-based measurement of local spectral moments. In *Proc. Conf. Computer Vision and Pattern Recognition*, Champaign, IL, 1992.
- [15] R. Szeliski. *A Bayesian Modeling of Uncertainty in Low-level Vision*. Kluwer Academic Press, Boston, 1989.
- [16] A.P. Witkin. Recovering surface shape and orientation from texture. *Artificial Intelligence*, 17:17–45, 1981.