Correspondence

Surface Orientation from Projective Foreshortening of Isotropic Texture Autocorrelation

LISA GOTTESFELD BROWN AND HAIM SHVAYTSER

Abstract—**A new method for determining local surface orientation from the autocorrelation function of statistically isotropic textures is introduced. It relies on the foreshortening that occurs in the image of an oriented surface, and the analogous foreshortening produced in the texture autocorrelation function. This method assumes textural isotropy, but does not require the texture to be composed of texels or assume other texture regularities. The technique was applied to natural images of planar textured surfaces and found to give good results. The simplicity of the method and its use of information from all parts of the image are emphasized.**

Index Terms—**Autocorrelation, foreshortening, isotropy, shape-from-texture, surface-orientation, surface-texture.**

I. Introduction

An important task that arises in many computer vision systems is the reconstruction of three-dimensional depth information from two-dimensional images. Of the many potential depth cues discussed in the literature, texture might be expected to play a particularly central role in the processing of certain classes of images such as those of natural outdoor scenes. Indeed, a variety of shape-from-texture algorithms have been proposed. See [1].

The determination of surface orientation from textural cues has been based on two general techniques. Gradient methods, first suggested by Gibson [2], rely on changes in texture properties such as the density of texels as the surface recedes from the observer. It is also possible to deduce surface orientation from purely local properties of the observed texture. Our work follows this latter approach, and bears many similarities to the algorithm presented in the pioneering paper of Witkin [3].

The basic assumption in Witkin's work is that the texture is isotropic, that is, that statistically speaking the texture has no inherent directionality. The distribution of edge directions obtained from such a texture will therefore be flat. If, however, the textured surface is viewed obliquely, projective foreshortening—a purely local phenomenon—distorts this distribution in a well-defined way. Witkin thus proposes that a histogram of edge directions constructed from the image in question be used to determine surface orientation via a maximum-likelihood fit.

The method proposed here is also based on the assumption of textural isotropy, but uses the projective distortion of the texture autocorrelation function as an orientation cue rather than the projective distortion of Witkin's edge-direction histogram. The autocorrelation of an oriented texture is foreshortened in a way identical to the foreshortening of the image itself, and the amount and direction of the foreshortening are conveniently measured by the moments of the autocorrelation function. Potential advantages of this approach are the elimination of the arbitrariness associated with the choice of edge-detection algorithm and the fact that it uses information from all parts of the texture rather than just the edges. Because this method, like Witkin's, computes statistically a projective distortion from local properties, it is more suitable for natural imagery. Gradient methods typically cannot be applied to natural imagery because of assumptions about texture density, uniformity, or regular geometry.

Section II contains a quantitative treatment of projective foreshortening of texture autocorrelation, and culminates in a formula expressing surface orientation as a function of the moments of the texture autocorrelation. Section III provides the practical details of an algorithm based on this approach, and describes the results it yielded when applied to textures found in common outdoor scenes.

Manuscript received May 31, 1988; revised October 23, 1989. Recommended for acceptance by O. D. Faugeras. This work was supported in part by the Defense Advanced Research Projects Agency under Grant N00039-84-C-0165.

L. G. Brown is with the Department of Computer Science, Columbia University, New York, NY 10027.

H. Shvaytser is with the David Sarnoff Research Center, Princeton, NJ 08543.

IEEE Log Number 9034235.

II. Foreshortening of Texture Autocorrelation

Let us begin with the terminology to be used in this and the subsequent section. The orientation of a surface (with respect to the line of sight) will be given in terms on the slant and tilt parameters (σ, τ) as used by Witkin [3]. The slant of a surface σ is defined as the angle between the normal to the surface and the line of sight. Thus, σ is the amount the surface slants away from being parallel to the image plane. We will take σ to run between 0° and 90°. The tilt τ specifies the direction in which the surface is slanted, and is defined as the angle between the x-axis of the image plane and the projection into the image plane of the normal to the surface. We will take τ to run between $-180°$ and 180°. For example, a surface parallel to the image plane has zero slant and an undefined tilt.[1] See Fig. 1.

Let an image be specified by a gray-scale function $F(\vec{r})$, where $\vec{r} = (r_1, r_2)$ denotes a point in the image plane. The image of a textured plane that is viewed head on, *i.e.*, that is perpendicular to the line of sight, will be denoted by $F_\perp(\vec{r})$. $F_{(\sigma,\tau)}(\vec{r})$ is then defined as the image produced by the same plane when it is given orientation (σ, τ) with respect to the line of sight. In this notation, $F_\perp(\vec{r})$ is equivalent to $F_{(0,*)}(\vec{r})$ since the slant is zero and the tilt undefined.

The autocorrelation of an image $A(\vec{r})$ is defined as

$$A(\vec{r}) = \int \left(F(\vec{r}\,') - \bar{F}\right)\left(F(\vec{r}\,' + \vec{r}) - \bar{F}\right) d\vec{r}\,', \qquad (1)$$

where $\bar{F}$ is the mean of F. The autocorrelation is conventionally normalized such that $A(0) = 1$, but our purposes do not require a specific choice of normalization to be made. The headon autocorrelation $A_\perp$ and the oriented autocorrelation $A_{(\sigma,\tau)}$ correspond to the images $F_\perp$ and $F_{(\sigma,\tau)}$, respectively.

Finally, we introduce the autocorrelation moment matrix,

$$\mu = \begin{pmatrix} \mu_{11} & \mu_{12} \\ \mu_{21} & \mu_{22} \end{pmatrix},$$

defined as

$$\mu_{ij} \equiv \int r_i r_j A(\vec{r})\, d\vec{r}. \qquad (2)$$

[1]The orientation of a surface is given with respect to the viewer and not a reference coordinate system. In this notation, a planar surface has different orientations at different locations depending on their relation to the viewer.

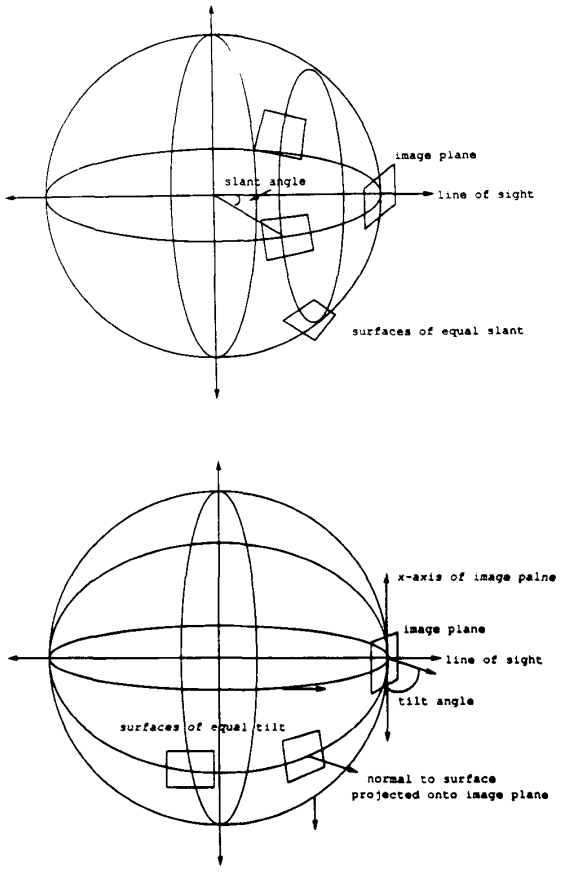

Fig. 1. Orientation of a surface is given with respect to the line of sight. This is depicted on the Gaussian sphere. Each surface is represented by the point on the sphere which has the same normal as the surface. The slant is the degree of inclination of the surface, given by the angle between the line of sight and the normal to the surface. The tilt specifies the direction of inclination and is defined as the angle between the x axis of the image plane and the projection of the normal to the surface onto the image plane.

Finally, we introduce the autocorrelation moment matrix. The moment matrix is symmetric; $\mu_{12} = \mu_{21}$. Since the region of integration in (2) is notionally infinite (although it corresponds in practice to a finite sum over pixels) we require $A(\vec{r})$ to fall off sufficiently rapidly at large distance that the moments be well-defined. The randomness found in natural textures suffices to produce the desired behavior. The autocorrelation moment matrices derived from the image of a textured plane viewed headon and the same plane with orientation (σ, τ) will be denoted $\mu_\perp$ and $\mu_{(\sigma,\tau)}$, respectively.

We now discuss how projective distortion affects the autocorrelation moment matrix of an image, and how this information can be used to determine the orientation of a textured plane. Let $F_\perp(\vec{r})$ be the head on image of a textured plane. Now consider $F_{(\sigma,\tau)}(\vec{r})$, the image produced by the same planar surface when viewed obliquely with slant and tilt parameters (σ, τ). We make the simplifying assumption that the distance between the plane and observer is very large in comparison to the linear size of the portion of the plane under consideration. The image of the oriented plane is therefore given by orthographic projection:

$$F_{(\sigma,\tau)}(\vec{r}) = F_\perp(\boldsymbol{M}^{-1}(\sigma, \tau)\,\vec{r}), \tag{3}$$

where

$$\boldsymbol{M}(\sigma, \tau) = \begin{pmatrix} 1 + (\cos\sigma - 1)\cos^2\tau & (\cos\sigma - 1)\cos\tau\sin\tau \\ (\cos\sigma - 1)\cos\tau\sin\tau & 1 + (\cos\sigma - 1)\sin^2\tau \end{pmatrix}. \tag{4}$$

Notice that

$$\boldsymbol{M}(\sigma, \tau) = \boldsymbol{R}_\tau \boldsymbol{M}(\sigma, 0)\,\boldsymbol{R}_{-\tau}, \tag{5a}$$

with

$$\boldsymbol{R}_\tau = \begin{pmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{pmatrix}, \tag{5b}$$

and

$$\boldsymbol{M}(\sigma, 0) = \begin{pmatrix} \cos\sigma & 0 \\ 0 & 1 \end{pmatrix}. \tag{5c}$$

Therefore, $\boldsymbol{M}(\sigma, \tau)$ is the matrix which foreshortens the vector $\vec{r}$ by a factor of $\cos\sigma$ along the direction of tilt τ, while leaving the direction perpendicular to τ unchanged. We should emphasize that $\boldsymbol{M}(\sigma, \tau + 180°)$ or equivalently $\boldsymbol{M}(-\sigma, \tau)$ and $\boldsymbol{M}(\sigma, \tau)$ are equal. Therefore, our method will yield surface orientation only up to an overall $\tau \leftrightarrow \tau + 180°$ ambiguity.

Our algorithm is based on the important observation that under orthographic projection, the image autocorrelation transforms identically to the image:

$$A_{(\sigma,\tau)}(\vec{r}) = A_\perp(\boldsymbol{M}^{-1}(\sigma, \tau)\,\vec{r}). \tag{6}$$

This is the analogous relation for the autocorrelation as expressed in (3) for the image. To relate $\mu_{(\sigma,\tau)}$ to $\mu_\perp$ we evaluate the integral in (2) via the change of variables $\vec{r} \to \vec{r}\,' = \boldsymbol{M}(\sigma, \tau)\,\vec{r}$, and (using the result that the $\det \boldsymbol{M}(\sigma, \tau) = \cos\sigma$) obtain

$$\mu_{(\sigma,\tau)} = \cos\sigma\, \boldsymbol{M}(\sigma, \tau)\,\mu_\perp\,\boldsymbol{M}(\sigma, \tau).$$

This equation shows the relation between $\mu_{(\sigma,\tau)}$ and $\mu_\perp$. In general, since 3-D information cannot be extracted from a single view, the slant and tilt parameters cannot be obtained from $\mu_{(\sigma,\tau)}$ without knowledge of $\mu_\perp$. However, $\mu_\perp$ can be determined using the assumption of textural isotropy. This assumption, when applied to texture autocorrelation can be stated as:

> The autocorrelation of a planar surface is isotropic, i.e., $A_\perp(\vec{r}_1) = A_\perp(\vec{r}_2)$ whenever $|\vec{r}_1| = |\vec{r}_2|$.

In other words, our assumption is that the autocorrelation of an image of an isotropic texture viewed headon, will be circularly symmetric. This implies that the headon autocorrelation moment matrix will be a multiple of the identity, i.e., $\mu_\perp = \xi\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Thus, the autocorrelation moment matrix for an oriented surface is given by:

$$\mu_{(\sigma,\tau)} = c\boldsymbol{M}^2(\sigma, \tau), \tag{7}$$

where the constant is given by $c = \xi\cos\sigma$.[2]

Equation (7), which relates the autocorrelation moment matrix of the image of a (directionally homogeneous) textured surface to the slant and tilt parameters that specify its orientation, forms the basis of our shape-from-autocorrelation algorithm. We complete our discussion of the underpinnings of this algorithm by giving the slant and tilt parameters as explicit functions of the matrix $\mu_{(\sigma,\tau)}$. From (5c), $\cos^2\sigma$ is given by the ratio of the smaller to the larger eigenvalue of $\sigma(\sigma, \tau) \propto \boldsymbol{M}^2(\sigma, \tau)$:

$$\sigma = \arccos\sqrt{\frac{\mu_{11} + \mu_{22} - \sqrt{(\mu_{11} - \mu_{22})^2 + 4\mu_{12}}}{\mu_{11} + \mu_{22} + \sqrt{(\mu_{11} - \mu_{22})^2 + 4\mu_{12}}}}. \tag{8a}$$

[2]It should be clear that the assumption of isotropy need not be strictly satisfied. It suffices to make the weaker assumption that $\mu_\perp = \xi\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$, that is, that any directional inhomogeneities the texture may have are not of the sort that mimic an oblique viewing angle, and hence do not show up in the moment matrix.

The expression for the tilt parameter τ follows from (4):

$$\tau = \frac{1}{2} \arctan \frac{2\mu_{12}}{\mu_{11} - \mu_{22}}. \quad (8b)$$

The relationship of the autocorrelation moments to the surface orientation can be understood more intuitively by considering their relation to the shape of the autocorrelation image. Because of the assumption of isotropy, the autocorrelation image of a textured planar surface viewed headon is composed of concentric iso-contours which are circular. When the surface is oriented the concentric iso-contours become elliptic. The surface orientation can be determined from the shape of these ellipses via the autocorrelation can be determined from the shape of these ellipses via the autocorrelation moments. This is completely analogous to what one can do when viewing the foreshortening of a flat circular object laying on an oriented planar surface. In this framework, we see that the tilt is the direction of the minor axis of the elliptic iso-contours and the slant is related to the ratio of the minor and major axes. Any statistical measurement independent of directionality, such as the autocorrelation used here or the histogram of edge directions used by Witkin, could be similarly utilized.

To summarize: (8), our principal result, gives the orientation of a textured surface as a function of its autocorrelation moment matrix, provided

- textural isotropy is assumed,
- the autocorrelation falls off sufficiently rapidly with distance,
- orthographic projection is assumed, and
- the tilt τ is assumed to be less than 180°.

III. Implementation and Results

To test our method, images were made of outdoor scenes of texture that seemed reasonably isotropic. In each case a single planar surface was photographed from a sufficient distance that orthographic projection was a good approximation, and that the entire image consisted of an image with a single orientation. We used textures commonly found in natural outdoor scenes, such as grass, rocks, dirt and leaves. Because of their practical importance in navigation, images of roads, sidewalks, and pebbled pathways were also included.

The following simple scheme permitted the actual orientation to be conveniently determined. Two photographs of each surface were taken, identical except that in one of the pictures a flat circular object was placed on the surface. The picture without the circular object was used as input to our shape-from-autocorrelation algorithm, while direct measurements of the foreshortened image of the circular object in the second picture gave the "true" surface orientation.

The photographs were digitized to yield 256 × 256 8-bit gray-scale images, and the autocorrelation was computed as the Fourier transform of the power spectrum of the image [5]. Since the Fourier transform can be computed with $O(n \log n)$ cost, where n is the number of pixels in the image, and the moments are computed with $O(n)$ cost, the entire procedure has a complexity of $O(n \log n)$. Thus, for a small cost above the optimal, the method uses all the information in the image.

When computing the moments, the integration in (2) is replaced by a summation, with the autocorrelation nominally being summed over all possible separations. To the extent that the autocorrelation falls off rapidly, the result should be insensitive to the precise region of summation. However, realistic images do not obey this assumption in a robust way, and statistical noise in the autocorrelation at large separations can cause excessive error in the momemts values. In the derivation of (8a) and (8b) in Section II, we saw that the foreshortening of the image causes an identical foreshortening of the autocorrelation, i.e., the (average) distance between correlated pixels gets foreshortened. Since, in general, adjacent pixels tend to be highly correlated while distant pixels have very little correlation, measuring the foreshortening of the autocorrelation for distant pixels is unreliable. Unfortunately, small perturbations in the autocorrelation of distant pixels can cause excessive errors in the (second order) moments values. To alleviate this problem we compute the moments using only autocorrelation values of highly correlated pixels. Experiments in image compression (e.g., [5]) indicate that pixels are uncorrelated over a distance of more than 16 pixels in a typical 256 × 256 image. In order to avoid these uncorrelated pixels without biasing the autocorrelation moments, we sum only over autocorrelations which exceed a threshold value, effectively restricting the moment sum to relatively small separations. The threshold value is chosen to be the value of the autocorrelation averaged over a ring with an empirically determined radius. In our experiments, a radius of 10 pixels was found to give good results. Fig. 2 shows how this radius affects the slant and tilt estimates for four typical images (D, H, I, and K). For each image the "O" marks the radius value in which the computed (σ, τ) matches the actual value. As seen from this figure, the threshold radius should be chosen intelligently, but the choice is not particularly delicate—any threshold radius in the range 5 ~ 25 would yield similar results. We believe the abrupt change which occurs in the curve for Image D is due the fact that most of the information contained in the autocorrelation is very close to the origin. This can be seen in Fig. 7.

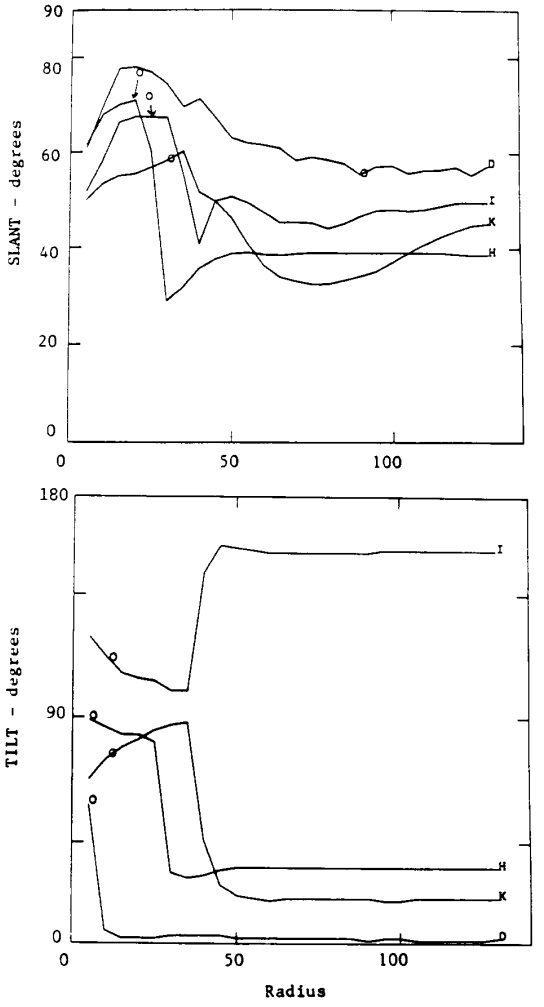

Fig. 2. Effects of thresholding radius on the elimination of uncorrelated values for four typical images. The small circle marks the radius value in which the computed slant or tilt matches the actual value.

A summary of the results is shown in Fig. 3. In this plot, each orientation is represented by a point in polar coordinates where the slant is the distance from the center and the tilt is the angle. For each image (A–K) the error between the actual measurement of the surface orientation and the value predicted by the method is shown.

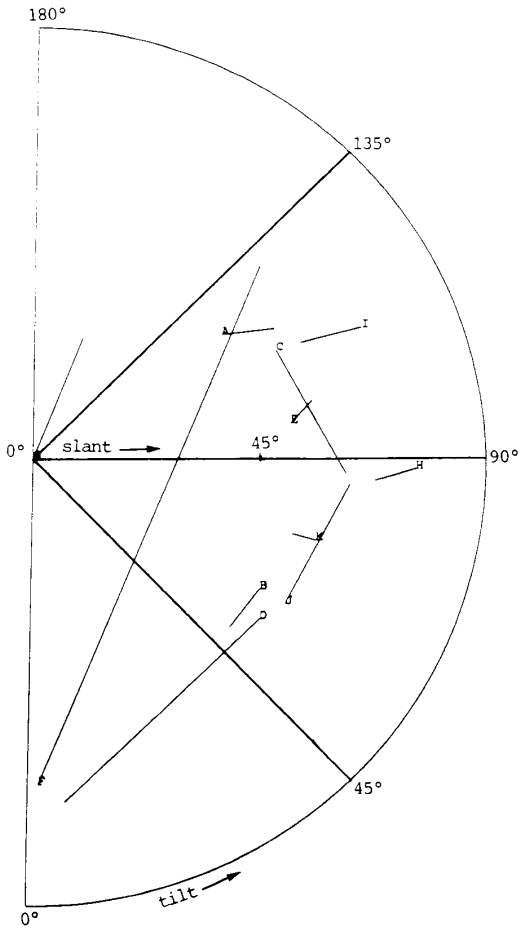

Fig. 3. Polar representation of error between actual (marked with letter) and computed surface orientation for eleven pictures. The slant is the distance from the center and the tilt is the angle.

The actual measurement is marked by the letter. Considering the difficulty in discerning orientations of homogeneously textured surfaces without perspective or world-knowledge cues, all of the results except images *F* and *D* were very reasonable.

The four representative images illustrate the technique and its behavior in more detail in Figs. 4–7. The large picture in upper left-hand corner of each figure shows a 512 × 512 color image of the textured surface with the flat circular object laying upon it. As was just discussed, from the foreshortening of the circular object in this image, the actual surface orientation was determined. The picture in the upper right-hand corner shows the 256 × 256 black and white subimage of the larger image without the circular object. *It is the autocorrelation of this image that was used to compute an estimate of the surface orientation.* The actual and computed orientations are given at the bottom of each picture. The picture in the lower right corner shows the central 64 × 64 region of the autocorrelation. The predicted foreshortening of the autocorrelation is clearly seen, and the elliptic shape of the foreshortened autocorrelation corresponds closely to the foreshortened image of the circular object placed on the surface. It should be observed, that from the pictures used for the computation (not the picture with the circular object), it is virtually impossible for most people to perceive the surface orientation (particularly without higher level knowledge) while nevertheless the method is successful. Even in Fig. 7, where the predicted orientation is poor, the autocorrelation shows the expected properties. We believe that the inaccuracies in this case are caused by the inability of the autocorrelation moments

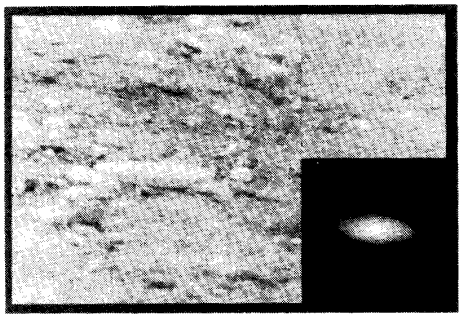

Fig. 4. Results of shape-from-autocorrelation method. Image H: actual slant = 76, tilt = 89; computed slant = 68, tilt = 86.

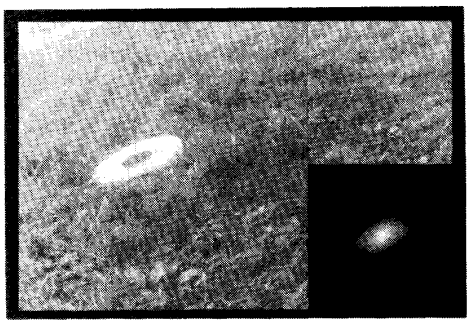

Fig. 5. Results of shape-from-autocorrelation method. Image I: actual slant = 71, tilt = 113; computed slant = 58, tilt = 115.

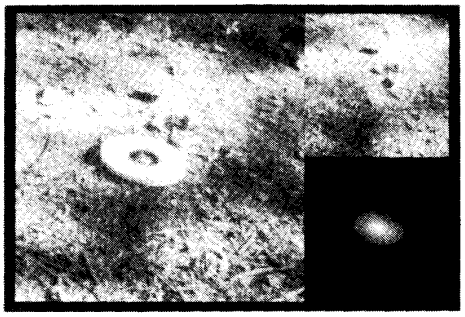

Fig. 6. Results of shape-from-autocorrelation method. Image K: actual slant = 58, tilt = 74; computed slant = 53, tilt = 73.

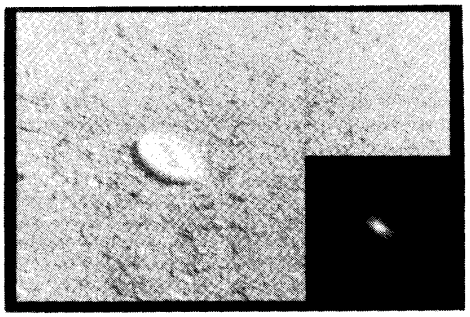

Fig. 7. Results of shape-from-autocorrelation method. Image D: actual slant = 55, tilt = 55; computed slant = 70, tilt = 5.

to accurately capture the ellipse parameters. Although the texture is indeed isotropic, its fine granularity causes all the information in the autocorrelation to be clustered close to the origin. The result is not enough information can be extracted by the autocorrelation moment matrix to accurately determine the surface orientation. The algorithm works well when the texture contains a range of texture

element sizes and an assortment of texture element types. Presumably this is because the assumption of isotropy is ensured and the autocorrelation contains redundant information spread out over a sufficient distance.

IV. Concluding Remarks

We have developed a new shape-from-texture algorithm that uses the effects of foreshortening on the autocorrelation of an image. The method assumes textural isotropy, that is, that the statistical properties of the texture have no inherent directionality, and that the texture is orthographically projected onto the image plane. The performance of this algorithm is quite good; the empirical tests presented in Section III show that this algorithm gives good estimates of surface orientation when applied to a variety of natural textures. This method is simple and well-defined, and avoids, for example, assumptions concerning texels and the arbitrariness associated with edge-detection. A particular advantage of this approach is its use of information from all parts of the image; all pixels contribute to the final orientation estimate.

Future extensions of this method might include weakening the assumption of textural isotropy through the use of multiple views, techniques involving gradients of the texture autocorrelation, or use of a knowledge system to provide *a priori* information about the headon texture autocorrelation. Surface curvature should be measurable by applying the method patch-wise over an image. A parallel implementation using image shifting to compute the relevant portion of the autocorrelation should lead to substantial increases in the speed of the algorithm.

Acknowledgment

We would like to thank M. Hanvey for the use of her photographic equipment and expertise. H. Shvaytser held a Weizmann fellowship during the course of this work.

References

[1] J. Aloimonios, "Shape from texture," *Biol. Cybern.*, vol. 58, 1988.
[2] J. J. Gibson, *The Perceptions of the Visual World.* Cambridge, MA: Riverside Press, 1950.
[3] A. P. Witkin, "Recovering surface shape and orientation from texture," *Artificial Intell.*, vol. 17, pp. 17-45, 1981.
[4] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 179-187, Feb. 1962.
[5] A. Rosenfeld and A. C. Kak, *Digital Picture Processing*, vol. I. Orlando, FL: Academic, 1982.

Image Seaming for Segmentation on Parallel Architecture

MING-HUA CHEN AND THEO PAVLIDIS

***Abstract*—We discuss some basic problems encountered when we assemble the results of image analysis on architectures with coarse parallelism. The emphasis is on strategies that minimize the distortion in the final result caused by processing image tiles independently. Methods are provided that can be used to reduce the disparity between the result when each tile is processed independently and when is processed as part of the whole image. A seaming algorithm has been constructed to seam the tiles with the results of region segmentation using the gray level mean difference or maximum-minimum criteria. Experimental results, obtained on both SEQUENT machine and SUN 3/160 workstation, are given to illustrate the performance of the algorithm.**

***Index Terms*—Image processing, image seaming, image tiling, parallel architectures, split-and-merge segmentation.**

Manuscript received December 21, 1988; revised November 17, 1989. Recommended for acceptance by A. K. Jain. This work was supported in part by a grant from the Digital Image Processing Laboratory of the Lockheed Palo Alto Research Laboratories and by the National Science Foundation under Grant IRI-8702168 and Equipment Grant DMC-8705290. The SEQUENT machine used in this work was supported by the National Science Foundation under Grant CCR87-05079.

M.-H. Chen is with the Department of Electrical Engineering, State University of New York, Stony Brook, NY 11794.

T. Pavlidis is with the Department of Computer Science, State University of New York, Stony Brook, NY 11794.

IEEE Log Number 9034373.

I. Introduction

In many image analysis applications, there is a need to divide the original large image into a number of small subimages (image tiles) and perform the required processing on each tile independently [1]. This will be the case if the whole image does not fit in the memory of the processor. Another case is when we have an array of processors and we want to assign a subimage for each processor, then perform our algorithm on the subimages in parallel [2]. While the processors could communicate, such a communication is costly and we may wish to avoid it. It is obvious that if we want to obtain the result for the whole image, we are faced with two questions:

A. If we put all the tile results together directly, is the final result equal to the result obtained on the whole image by the same algorithm?
B. If *A* is not true, is there any way to modify the tiling or the assembling strategy so that, when the individual results are put together, they will equal (or be close to) the result on the whole image?

Clearly, except for some local operations, most algorithms give a negative answer to question *A*. In this correspondence we will show that it is often possible to provide a positive answer for *B*. We define *seaming* as the process of assembling the results of individual tiles processed. In order that the result of seaming be correct, we must either employ operations that satisfy the local knowledge principle (defined in the next section), or find an efficient way to make the seamed result close to the result of the original whole image ("true result").

In Section II, we describe requirements that ensure the correctness of the seamed result. Then we will analyze some factors that will cause a disparity between the tile-results and the corresponding parts of the whole image result for the split-and-merge segmentation algorithm, which is used here as an example of those nonlocal, order-dependent image processing algorithms. In Section IV we construct a general purpose seaming algorithm for the region segmentation and give some experimental results. Our analysis assumes that there is no communication between processors during the processing of image tiles. Such communication is always costly and for coarse parallelism architecture, slows down the operation considerately. It also require a drastic redesign of the algorithms. The results of our correspondence suggest that some care in the partition of the image and in the assembling of the separate results on tiles may allow us to avoid interprocessor communication without sacrificing the quality of the results. Our discussion deals with two tiles only. This is not a real limitation because finer tiling can be done recursively, as we show in an example of implementation.

II. Conditions for Independent Processing of Tiles

We assume that the whole image $I(x, y)$ is defined on a region D^*, where $D^* \subset Z^2$. Suppose a subset $D \subset D^*$ is divided into two parts (tiles), so that $D = D_1 \cup D_2$, and $D_1 \cap D_2 = \varnothing$. If Ψ is a transform (an operation, an algorithm) on the image, then the basic