Differential Geometry of Surfaces in Range Imagery: A Computer Program

by
Dr. Griff L. Bilbro
and
Prof. Wesley E. Snyder

Center for Communications and Signal Processing
North Carolina State University
Box 1792
Raleigh, NC 27695

CCSP-TR-85/6
February 12, 1985

Abstract

A computer program is developed that uses the techniques of differential geometry to determine several local properties of simple curved surfaces found in range images. The computations are performed in frequency space, allowing excellent control of bandwidth and noise sensitivity in the computation of surface pratical derivatives. The algorithm not only has theoretical significance but also appears to have immediate application in the quality control of such surfaces, particularly since the use of the FFT makes possible implementations in fast, off-the-shelf hardware.

Differential Geometry of Surfaces in Range Imagery: A Computer Program

1. Introduction

Range Imagery is a new technology for computer vision. It is rapidly evolving out of the earlier technology of Luminance Imagery. A range image is an array of regularly sampled distances rather than regulary sampled brightnesses. It is a simple and direct encoding of the apparent surface geometry of the viewed object.

In contrast, a luminance image encodes a variety of local surface peculiarities including texture, illumination, and albedo. No range information is included in this encoding and the geometry of the surface cannot be directly determined from its luminance image. Thus the determinination of shape from shading is the primary problem in luminance imagery.

2. Range images as parametrized surfaces

An important early problem in range imagery is to develop a mathematical formalism for images that can (1) represent any range image of any object, and (2) analytically relate the actual solid object to its image. We[2,3] have developed such a formalism: The actual surface can be regarded as a parametrized surface sampled by the range array. The study of parametrized surfaces is well understood[1] and provides a natural representation for range image data.

In the analysis of luminance images, the difficulty was the determination of shape or location from the ambiguous clues of brightness. A major difficulty in range imagery is the less subtle problem of computational complexity. The caculations of differential geometry are done in three dimensions and entail substantial computation even for a 128×128 image. Furthermore, the simplest properties of surfaces are unfortunately dependent on orientation, scaling, or translation. The first invariant properties depend on second derivatives of real data. Such measurements are notoriously sensitive to noise and require elaborate preprocessing.

There do exist formidible problems in range imagery that deal with occluded features, solid object reconstruction, and global models for surfaces and symmetries. But as in the infancy of luminance imagery, there are numerous applications of this new technology that involve no such obscurities. One straightforward application is the quality control of simple curved surfaces. This requires the efficient determination of surface properties from data produced by the currently available first generation of range sensors. The problem is complicated by the data degradation anticipated on a production line and also by the requirement of real time speed.

3. Computer Vision

A general computer vision system operates in three phases: an image phase, a knowledge phase, and a database phase. The image phase deals with the raw image data and typically involves a vast number of logically simple, typically local calculations. In this phase, all possible neighboring pairs of pixels (or larger

groups) are examined to determine certain quantities or features that characterize the neighborhood in useful ways. These features may include the area of the surface intersected by the pixel cell, local surface normal, local curvature, or even local estimates of the confidence in other features. In addition the image may be processed, for instance, to improve signal to noise ratio.

In the second phase, regularities of the features are exploited to integrate the information of the image into a functional abstraction. In the third phase the abstraction of the image is applied to a database of objects or instructions to generate a response to the image. Contol may flow sequentially thru the three phases or vacillate in pursuit of successive approximations to a final answer. Control may originate in the pixel phase as in Bottom Up strategies or in the database phase as in Top Down strategies.

Especially in the Bottom Up strategy, the entire image may be processed to extract a certain class of generally useful measurements. This would be done in the absence of apriori information restricting the content of the image. If rapid robust means of extracting curvatures and other differential geometric measures of surfaces are developed, they would be the first objectives of such general processing.

4. Differential Geometry

Although the classical study of this field in recent years deals not only with local properties of surfaces but also with certain global properties that are related to topological notions of closed surfaces, we are concerned here only with the

former.

The lowest order property of a point on the surface of an object is its location, which is a 3-vector. Neither the direction nor even the length of that vector is truely characteristic of the object. The vector is a function of the coordinate system and the pose of the object in that system. It is not invariant to viewpoint changes and contains no invariants. It locates the point but does not further characterize it.

The tangents and normals to the surface are first order properties of the surface, since they depend on first partial derivatives of the location vector near the point in question. These vectors are normalized to unity because any length that could be assigned to them depends on details of the parametrization of the surface and not on intrinsic properties of the surface. The direction of these vectors also fails to be invariant under changes in the observation angle.

The curvature tensor is a simple matrix that describes how the surface normal varies as a function of path length in the vicinity of the point. Its components depend on the coordinate system, but its trace and its determinant do not. These true invariants are called the mean and Gauss curvatures and they do characterize the surface at that point. The curvature tensor is a second order property of the surface. It depends on the second partial derivatives of the location vector of the surface.

There are alternative approaches that do not require the computation of local invariants and that still fit into the three phase scheme outlined above. One is

template matching. Another is the set of Hough like techniques. Both approaches use lower order surface properties that can also be computed by the algorithms developed in this paper.

5. Partial derivitives

The main problem in the estimation of partial derivatives is treating noise reliably and rapidly. If the noise contains only a small amplitude component, then linear filters are appropriate. If it also contains a sparse large amplitude component, preprocessing with a median filter is essential. Even a sprinkling (one percent of total pixels) of small outliers (a few percent relative error) can ruin an attempt to linearly extract second derivatives.

Estimating the second partials from data with relatively low amplitude noise can be done with linear operators. At least conceptually, the data must be first smoothed then differentiated. We have experimented with two algorithms. Both use Fourier transforms to smoothe the data, by lowpass filtering the spatial transform of the image. The first algorithm then computes the derivatives in the Fourier domain before returning to direct space. The second uses Sobel-like operators on the smoothed direct image as discussed in a previous paper[2] with the the simplest 3×3 kernels discussed there. It was found that smoothing the data could be done more efficiently and more flexibly with the FFT than with the larger convolution kernels indicated[1] in the literature.

Fourier transform techniques are attractive for two reasons. First, they are relatively fast and immensely acceleratable with special purpose hardware.

Secondly, they are completely general and contain all the information in the original image, a desirable feature at the current state of the art in range imagery. With the exception of the Fourier Transform and its inversion, all other calculations are completely parallel, and would require no dataflow even between neighboring pixels if treated by multiple processors. So the DFT approach is attractive on grounds of real time processing.

On a smoothed image, Sobel-like operators also permit the estimation of second derivatives. The attraction of this scheme is reduced storage: all the derivatives can be calculated on the fly, saving the memory necessary to simultaneously hold the original image and five derivative images. Larger kernels can be used to simultaneously smoothe the data, but the cost of computation is high on a sequential machine. Special purpose hardware exists for the computation of these convolutions, but the gains in storage efficiency obtain only when the derivatives at each point are used immediately rather than accumulated into derivative images.

It appears that in the next few years, commercially available range sensors will require an image capture time on the order of a millisecond per pixel, or a few hundred milliseconds for even a tiny servo selected patch of the total scene. The operational definition of "real time" in the immediate future is on this order; this amounts to a several tenths of a second for a 128×128 image. The phase one image processing must be fast enough to permit some phase two abstraction and phase three searching.

Currently, we are using a gaussian lowpass filter to smoothe the image by fltering the Fourier dual image. We are investigating other means, but this arrangement allows us to explicitly specify the resolution of the process. Resolution is the diameter of the smallest feature of interest, measured in pixels widths. Features smaller that the resolution are treated as noise. Larger features are resolved in the analysis.

6. Equations

The first commercial generation of range sensors produce an array of integer distances. These distances may be measured along a ray from the (possibly hypothetical) focal point to the viewed object. This array represents a "range image," and is designated r_{ij}. The indices typically represent sampled distances in Cartesian directions (usually $i=x, j=y$). On the other hand, the elements of the array may represent perpindicular distances from the focal plane (as in an orthographic projection). This "depth image" is designated Z_{ij}, and is related to the range image by

$$z_{i,j} = \frac{\lambda}{\sqrt{i^2+j^2+\lambda^2}}\, r_{i,j} \ , \tag{1}$$

where λ is the focal length of the sensor and determines the unit of length reported by the sensor. The reduced depth, w, is then

$$w_{i,j} = \frac{1}{\lambda}\, z_{i,j} \ . \tag{2}$$

Analytically, the "reduced depth array" is the most natural to work with in the following development. The conventional Cartesian 3-vector locating that portion

of the surface visible thru the pixel at i,j is

$$\mathbf{x}_{i,j} = \left[iw_{i,j} \ , \ jw_{i,j} \ , \ \lambda w_{i,j} \right]^T \ . \tag{3}$$

The Cartesian vectors corresponding to each pixel jointly comprise a sampling of a surface parametrized by the focal plane array indices.

The discrete Fourier transform for $N \times N$ values on a regular lattice in direct space is

$$W_{p,q} = \sum_{i,j} e^{-\frac{2\pi\iota}{N}(pi+qj)} w_{i,j} \ , \tag{4}$$

where p and q will refer to indices in the Fourier image and i and j will refer to their conjugate indices in the direct image. The square root of minus one is represented by ι. The direct array can be expressed in terms of the Fourier array as

$$w_{i,j} = \frac{1}{N^2} \sum_{p,q} e^{\frac{2\pi\iota}{N}(pi+qj)} W_{p,q} \ . \tag{5}$$

Arrays on the direct lattice will be represented by lower case letters. Their conjugate arrays on the Fourier lattice will be represented by the corresponding capital letters. The indices in all sums in both direct space and in the dual space range from $-N/2$ to $N/2$. This is not only a typographical simplification, but also will simplify the specification of the subsequent treatment of the Fourier coefficients. It is a trivial matter to relabel the focal plane array indices to be centered around the origin in direct space. Centering the indices in the dual space depends on the periodicity of the exponential for the discrete Fourier transform.

$$e^{\frac{2\pi\iota}{N}(p\pm N)i} = e^{\pm\frac{2\pi\iota}{N}N}\, e^{\frac{2\pi\iota}{N}pi} = e^{\frac{2\pi\iota}{N}pi}, \tag{6}$$

since $e^{2\pi\iota}$ is unity. This translational symmetry is formally identical to the translational symmetry in direct space implicit in the DFT. But it is more useful than merely a relabeling of indices. We will use the DFT to estimate the first few partial derivatives of the reduced depth. Clearly the granularity of the discrete lattice should be minimal in this case for the smooth functions of interest. Although the DFT expressions for finite differences, such as $\Delta_i x_{i,j} = x_{i+1,j} - x_{i,j}$, are strictly invariant under translations such as $i := i \pm N$, the analogous expressions for derivatives are not. It is only sensible to differentiate the DFT in the centered representation, baseband, which is the complete set with lowest absolute spatial frequencies. On a discrete lattice all complete sets are equivalent, but in between lattice points the lowest set is distinguished as smoothest. This set describes the smoothest interpolation between the discrete elements of the array. the smoothest set is distinguished. Those points in between lattice points are implicitly required by derivitives.

A smoothing of the original data is obtained by inverting the lowpass filtered transform of that data:

$$w'_{i,j} = \sum_{p,q} \phi_{p,q}\, e^{\frac{2\pi\iota}{N}(pi+qj)} W_{p,q}, \tag{7}$$

where the smoothing filter with a psatial resolution of d pixels is a gaussian centered about the origin

$$\phi_{p,q} = e^{-(\frac{2d}{N})^2 (p^2+q^2)}. \tag{8}$$

In the centered representation, no special care is needed to preserve any special symmetry of the transform. Though, strictly speaking, each of the terms on the boundary of the region of summation in Fourier space should only be counted fractionally. Ordinarily these Nyquist terms can be neglected entirely without noticable error: the low frequency DFT approximation of derivations is not valid if the high frequency amplitudes are significant.

For low order derivatives of smooth functions, the granularity of the DFT is negligible. We will assume the convention that $w(i,j)$, written as a function, is the continuous function underlying the discetely sampled array, which will be written as a matrix, $w_{i,j}$. The partial derivatives of $w(i,j)$ will be important in the formulation of curvature. They can be approximated from the DFT of a smooth enough array. Practically, the lowpass smoothing described above is necessary to permit this. We will subsume the smoothing function, $\phi_{p,q}$ into $w_{p,q}$ so as not to obscure new features in the following equations.

The n^{th} partial with respect to the first argument of the smooth function underlying $w_{i,j}$ can be approximated as

$$\frac{\partial^n w(i,j)}{\partial i^n} = \sum_{p,q} e^{\frac{2\pi\iota}{N}(pi+qj)} (\frac{2\pi\iota}{N} p)^n \; W_{p,q} \; . \tag{9}$$

The corresponding partial with respect to the second index is

$$\frac{\partial^n w(i,j)}{\partial i^n} = \sum_{p,q} e^{\frac{2\pi\iota}{N}(pi+qj)} (\frac{2\pi\iota}{N} q)^n \; W_{p,q} \; . \tag{10}$$

The first Cartesian partials are then obtained by the chain rule

$$\frac{\partial \mathbf{x}}{\partial i} = \left[i\frac{\partial w}{\partial i} + w \ , \ j\frac{\partial w}{\partial i} \ , \ \lambda\frac{\partial w}{\partial i} \right]^T \ , \tag{11}$$

and

$$\frac{\partial \mathbf{x}}{\partial j} = [i\frac{\partial w}{\partial j} \ , \ j\frac{\partial w}{\partial j} + w \ , \ \lambda\frac{\partial w}{\partial j}]^T \ , \tag{12}$$

The second partials are

$$\frac{\partial^2 \mathbf{x}}{\partial i^2} = [i\frac{\partial^2 w}{\partial i^2} + 2\frac{\partial w}{\partial i} \ , \ j\frac{\partial^2 w}{\partial i^2} \ , \ \lambda\frac{\partial^2 w}{\partial i^2}]^T \ , \tag{13}$$

$$\frac{\partial^2 \mathbf{x}}{\partial i \partial j} = [i\frac{\partial^2 w}{\partial i \partial j} + \frac{\partial w}{\partial j} \ , \ j\frac{\partial^2 w}{\partial i \partial j} + \frac{\partial w}{\partial i} \ , \ \lambda\frac{\partial^2 w}{\partial i \partial j}]_T \ , \tag{14}$$

and

$$\frac{\partial^2 \mathbf{x}}{\partial j^2} = [i\frac{\partial^2 w}{\partial j^2} \ , \ j\frac{\partial^2 w}{\partial j^2} + 2\frac{\partial w}{\partial j} \ , \ f\frac{\partial^2 w}{\partial j^2}]^T \ . \tag{15}$$

From these relations, the unit normal to the surface, the First Fundamental Coefficients (E, F, G), and the Second Fundamental Coefficients (e, f, g), can be calculated at each point[2] from which the two classical curvatures, k_G and k_M.

The area intercepted by each pixel can also be calcuated as follows[1] in terms of the First Fundamental Coefficients,

$$a_{i,j} = \sqrt{EG - FF} \ . \tag{16}$$

The principal curvatures can be derived from the classical curvatures as the two roots of the eigenvalue equation[3] of the curvature tensor:

$$k_{\pm} = k_M \pm \sqrt{k_M^2 - k_G} \ . \tag{17}$$

7. Algorithms

Since the calculation of the curvatures near a pixel uses the value of the area intercepted by that pixel, the two calculations will be presented as a single pro-

gram, *geom*. The program is written in C and uses the standard C math library routines. In addition, *geom* calls four routines from PIRANA, the portable CCSP standard image analysis programming environment. These PIRANA routines provide both a convenient means of image input and output, and a one dimensional fast Fourier transform. All PIRANA routines use a common C structure of type *data* to maintain a desription of an image and its location in memory. *Pixin* reads a stored image and its desription from a specified disk file and copies it into allocated main memory. *Pixout* writes an image from memory to disk. *Nrows* and *ncols* return array limits for a stored image. *Fftj* performs a one dimensional discrete fast Fourier transform. The PIRANA library thus provides a convenient environment for image analysis. It was designed, implemented, and is maintained by CCSP. It has been proven portable and is available to CCSP member companies as a benefit of membership.

Conceptually, CURVS performs a sequence of eleven operations. These operations are reflected directly in its flow of control.

(1) Invocation: *Geom* reads and stores the name of the PIRANA file containing the image to be processed. Filenames are similarly recorded for the resultant images of Gaussian curvature, mean curvature, and surface area intercepted by each pixel. The desired resolution and the focal length of the range sensor configuration are read and stored. The invocation is tested for validity; on failure, execution is halted with an informative message to the user.

(2) Input: The specified image is read into a PIRANA buffer and copied to work arrays internal to *geom*.

(3) Conversion to depth: The range image is converted to a depth image as a first step in the *curvs* subroutine, which directs the actual computation of the resultants.

(4) Fourier transform: The depth image is Fourier transformed in place along both of its dimensions.

(5) Lowpass: The desired resolution is embodied in a symmetric, separable, Gaussian lowpass filter and applied to the image in the Fourier array.

(6) Derivatives: The two first partials are derived from the filtered Fourier array by *ddi*, which produces the Fourier array of the partial with respect to the first direct index, and *ddj*, which does the same thing with respect to the second direct index. The derived arrays are then passed thru *ddi* and *ddj* again to produce the three second partials.

(7) Inverse Fourier transform: The Fourier arrays for the smoothed depth and the five partial derivatives are then transformed back into the direct arrays. At this point the imaginary parts of the various arrays return to zero and are subsequently unused.

(8) Chain rule: The real partial arrays of the depth are converted by use of the chain rules of equations 11 through 15 into Cartesian vector partials and are assigned to C structures of type *vector*. Only the first partials are used for the calculation of pixel area. All the partials contribute to the curvatures.

(9) Curvatures and area: The Fundamental Coefficients are computed from the Cartesian partials by simple vector operations as outlined in equations ??. The area is easily determined from the first three coefficients. The classical curvatures are then computed from the area and the Fundamental Coefficients.

(10) Scaling: The resultant floating point arrays for the curvatures and area are then scaled into 8 bit integer in the range from 0 to 255 and copied into PIRANA buffers. The 0,0 pixel is overwritten with the logarithm of the scaling factor.

(11) Output: Finally the images and their descriptions are written by PIRANA to disk as specified by the invocation.

8. Program

A listing of *geom* and a machine readable copy of the application and its environment is available from CCSP.

---

1. M. DoCarmo, *Differential Geometry of Curves and Surfaces,* Prentice-Hall, Englewood Cliffs, NJ (1976.).

2. W. Snyder and G. Bilbro, "Segmentation of Range Images," *Technical Report* **TR-84-7**Center for Communications and Signal Processing, NCSU, (1984).

3. W. Snyder and G. Bilbro, "Segmentation of Range Images," *IEEE Conf on Automation and Robotics,* (April, 1985).