SURFACE CHARACTERIZATION FOR THREE-DIMENSIONAL OBJECT RECOGNITION IN DEPTH MAPS[1]

Paul Besl

Ramesh Jain

Department of Electrical Engineering and Computer Science

The University of Michigan

Ann Arbor, Michigan 48109-1109

December 1984

CENTER FOR RESEARCH ON INTEGRATED MANUFACTURING

Robot Systems Division

COLLEGE OF ENGINEERING

THE UNIVERSITY OF MICHIGAN

ANN ARBOR, MICHIGAN 48109-1109

[1]This work was supported in part by IBM Corporation, Data Systems Division, Kingston, New York under the monitorship of Dr. Jack Contino and in part by the Air Force Office of Scientific Research under contract F49620-82-C-0089.

TABLE OF CONTENTS

Abstract

A matrix of distance values, known as a depth map or range image, can be considered as noisy, discrete samples of some surface described by a function of the type z=f(x,y). This paper addresses the problem of characterizing this underlying surface using only its samples in a manner suitable for use by a view-independent three-dimensional object recognition algorithm. *Mean curvature* and *Gaussian curvature* are identified as local second order surface characteristics which possess desirable invariance properties. The importance of surface curvature is emphasized using theorems from differential geometry. First order surface characteristics, such as *critical points* and large metric determinant points (which occur at depth-discontinuities), can also be used to complement the surface curvature information. Experimental surface characterization results are shown for real and synthetic depth maps which indicate the usefulness of these quantitative features.

Index Terms: surface characterization, surface matching, depth maps, surface curvature, 3-D object recognition, computer vision

1. INTRODUCTION

In recent years digitized range data has become available from both active and passive sensors, and the quality of this data has been steadily improving. Range data is usually obtained in the form of a rectangular array of numbers, referred to as a *depth map* or *range image*, where the numbers quantify the distances from the sensor to the surfaces of objects within the field of view along rays emanating from a regularly spaced rectangular grid. Depth maps are obtained in a variety of different ways [1,23,24,25,32,43,45,49,50]. The three-dimensional *shape* of depth map regions directly approximates the three-dimensional shape of the corresponding *visible* object *surfaces* in the field of view. Due to the *explicitness* of this type of information, the process of recognizing objects by their geometric shape *should* be less difficult using depth maps than the intensity images typically used in computer vision work. This paper examines the three-dimensional object recognition problem using depth maps as sensor input. Our approach, which utilizes the concepts of surface characterization and surface matching, is presented for solving this problem. The mathematics of differential geometry is used to motivate and justify our selection of surface curvature characteristics. An approach for computing surface curvature using window operators is discussed and experimental results of this approach are included which demonstrate its performance on real and synthetic depth maps.

2. OVERVIEW OF OBJECT RECOGNITION APPROACH

Three-dimensional object recognition is an important problem in computer vision which has been addressed by many researchers [4]. A robust general purpose vision system which addresses this problem would find many applications. If such a system could work well with sensor data taken from only a single view, this would be even more desirable. We feel that it has been established over the last twenty years that it is extremely difficult to develop a general purpose vision system which uses single arbitrary view intensity images as sensor input. Given that high quality range data will be widely available in the near future, it makes sense to explore the use of depth maps as sensor input to an object recognition system. Several approaches to this problem have already been developed [5,7,11,22,29,40,42,47,48]. A literature survey [4] discusses these approaches and points out that no general purpose vision systems have been developed which utilize depth map input in a completely satisfactory manner.

We have decided to address the depth map object recognition problem with the aim to develop a *general purpose* approach. In the next section, we give a fairly precise verbal definition of an important version of the general object recognition problem that we hope to solve. Subsequently, we give a mathematical interpretation of this problem definition as it applies to depth maps which identifies the mathematical role of surface characterization. We assume that we are given discrete samples of an underlying depth map surface function with some noise added. First, we decide what kind of mathematical features characterize the underlying depth map function in ways that are *invariant to changes*

in viewpoint which preserve the overall visibility of the surface. This notion of invariance is extremely important and is sometimes confusing in the vision context. Therefore, we elaborate on this topic.

The term "invariant" is usually used to refer to a quantity which does not change under some group of transformations. Physical objects do not in general possess explicit features which are visible from any viewing angle. There are almost always degenerate viewing angles in which object features are radically different. Consider, for example, an object as simple as a cylinder. One sees a flat planar surface with a circular boundary when one looks down the axis of a cylinder. On the other hand, one sees a curved surface with a rectangular projected boundary when one looks perpendicular to the axis direction. See Figure 1 for the two views under consideration. There are *no explicit* invariant features even in this simple case. (For example, we do not consider the minimum projected silhouette area as an explicit feature.) The *roundness* of the cylinder manifests itself in both views, but in very different ways: in the first case, we get an

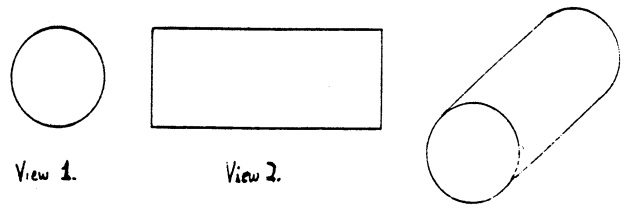

Figure 1. Two Views of a Cylinder with No Common Features

explicit *circular arc* depth-discontinuity boundary surrounding a flat region while in the second case, we get a constant-negative-mean-curvature, zero-Gaussian-curvature surface bounded by a projected rectangular depth-discontinuity boundary. Some papers in the computer vision literature seem to use the term invariant to mean "invariant when visible." We propose that the term "visible-invariant" should be used for this meaning. Thus, a "visible-invariant" surface characteristic is a quantitative feature of a surface region that does not change under viewpoint transformations which do not affect the visibility of that region. A general purpose vision system must incorporate the fact that some key features may not be visible even when an object is present and visible in the field of view. In situations where it is not possible to take intelligent actions based on what is currently visible, the general purpose system should automatically request the acquisition of image data from new vantage points [28].

The visible-invariant surface characteristics which we have decided to use are the *Gaussian curvature (K)* and the *mean curvature (H)* which are referred to collectively as *surface curvature.* We abbreviate this term as *S-curvature.* Both of these surface characteristics are invariant to changes in surface parameterization *and* to translations and rotations of object surfaces. In addition, mean curvature is an extrinsic surface property whereas Gaussian curvature is intrinsic. These terms are defined later. Differential geometry emphasizes that these are very reasonable surface features to consider. Since we must deal with noisy discrete samples, we have determined a method for computing S-curvature

from discrete data.

Since we can seldom obtain perfect sensor data from the real world, it is desirable to compute a "rich" characterization of the surface which preserves the surface structure information and is insensitive to noise. The noise insensitivity could possibly be achieved by computing redundant or at least "overlapping" information about a surface. In order to have a very rich geometric representation then, we propose to combine surface critical points (local maxima, minima and saddle points) and large metric determinant points (depth-discontinuities) with the surface curvature information to characterize a depth map surface in more detail. While these first order characteristics are *not* visible-invariants in general, they provide useful complementary information and can be computed for a small additional cost. Given a depth map surface characterization, we propose to match depth map surface region characteristics against pre-computed model object surface region characteristics guided by depth-discontinuity and critical point information.

The matching algorithm of a robust 3-D object recognition system must be view-independent. One could use multiple view ideas similar to those of Koenderink and Van Doorn ("Visual Potential") [26][27] or Chakravarty and Freeman ("Characteristic Views") [8], but we have some preliminary ideas for a new, more compact, scheme which does not increase its storage requirements so dramatically as object complexity increases. The complete details and implementation of the matching algorithm based on our surface characterization ideas are a subject for future research. We note that identification of translation and

orientation parameters should be performed during the matching process in addition to object identification. After the matching algorithm has produced a list of possible objects and their respective locations and orientations, we can use a depth-buffer algorithm to create a synthetic depth map using a world model. We may do verification matching directly between the synthetic depth map and the sensor data, or we may run the surface characterization algorithm on the synthetic data to yield a synthetic scene description which could be matched against the surface characterization scene description computed from the sensor data. If major discrepancies exist, the system should try to remedy the problems in its understanding automatically. This *feedback* element is another subject which we shall address in further study.

It may be necessary to compute our surface characterization using different window sizes ("scales") and correlate features in this scale-space dimension to help overcome the effects of noise. A discussion of this topic is beyond the scope of this paper.

3. DEFINITION OF OBJECT RECOGNITION PROBLEM

Three-dimensional object recognition is a somewhat nebulous term. A survey of the literature on this subject is sufficient proof of this statement [4]. Therefore, we attempt to give a reasonably precise verbal definition to the object recognition problem we hope to solve. The problem which we present is more general and more useful than many other recognition problems addressed in the literature, and yet it should still be solvable using current technology.

First, we present a brief qualitative summary concerning desirable visual capabilities which motivates the detailed definition that follows.

The *real world* which humans commonly perceive both visually and tactilely is primarily composed of solid *objects.* When humans are given a new object they have never seen before, they are typically able to gather information about that object from many different *viewpoints.* The process of gathering detailed object information and then storing that information in some format is referred to as *model formation.* Once human beings are familiar with many objects, they can identify those objects from an *arbitrary single stationary viewpoint* without further investigation in most cases. In particular, humans can *identify, locate, and qualitatively describe the orientation* of objects in black-and-white photographs. The black-and-white photograph capability is significant because it only involves the spatial variation of a *single* scalar parameter within a framed rectangular region corresponding to a fixed single view of the real world whereas human color vision generally involves a three-parameter color variation within a large, almost hemispherical solid angle corresponding to a continually changing viewpoint. Since we are interested in an automatic, computerized recognition process, we must restrict allowable input data to be compatible with digital computers. The term *digitized sensor data* will be used to refer to any input matrix of numerical values (which can represent intensity, range, or some other scalar parameter) and associated auxiliary information concerning how the matrix of values was obtained.

The above paragraph motivates the following definition of the autonomous single arbitrary view three-dimensional object recognition problem:

(1) Given any collection of labeled rigid solid objects, these objects *may be examined* in any way desired as long as the objects are not deformed.

(2) *Models* for the labeled objects *may be formed* using information from this examination in any way desired and given the appropriate object's label.

(3) (i) Given digitized sensor data corresponding to one particular, but arbitrary, field of view of the real world as it existed at the time of data acquisition;

(ii) given any data stored previously during the model formation process; and

(iii) given a list of labels of distinguishable solid objects;

answer the following questions for each object in the list using the capabilities of an autonomous processing unit:

a) Does the given labeled object appear in the digitized sensor data? (That is, does it permit a consistent interpretation of the sensor data.)

b) If it does, how many times does the object occur?

c) For each occurrence of a given object, determine the *location* of that object within the sensor data and, if it is possible using the particular type of sensor data, determine the three-dimensional location of that object with respect to some convenient coordinate system.

d) Also, if possible, determine the three-dimensional *orientation* of each occurrence of a given object with respect to some convenient

coordinate system.

(4) Finally, if there exist regions within the sensor data which do not correspond to any of the objects in the list, characterize these regions in a way that they can be recognized if they occur again in any future images.

We refer to the problem of successfully completing these assigned tasks using real world sensor data while obeying the given constraints as the 3-D object recognition problem. This problem is *not* successfully addressed in many of the object recognition systems discussed in our literature review [4]; more constrained problems are usually addressed which are limited to particular applications. If our stated 3-D object recognition problem were solved successfully by some system, that system would be extremely useful in a wide variety of applications, including automatic inspection and assembly and autonomous vehicle navigation. The problem is posed so that it is feasible to use computers to solve the problem, and it is also clearly solvable by human beings.

4. MATHEMATICAL FORMULATION OF DEPTH MAP OBJECT RECOGNITION PROBLEM

The three-dimensional object recognition problem was defined informally above. It is often advantageous to define a problem in a stricter mathematical form because it helps to eliminate possible problem ambiguities. Therefore, we now redefine *depth map object recognition* in precise mathematical terms as a *generalized inverse set mapping.* Since there is no general theory available on how to efficiently compute such a mapping, we put forth a computational

theory based on surface characterization and surface matching. We turn to the differential geometry of surfaces in search of answers based on the implicit decision to use features of the underlying depth map surface function. We believe that it is practically necessary to perform the matching required by the recognition process on data with lower dimensionality and with more explicit visible-invariance properties than the discrete depth map surface data itself.

First, we consider world modeling issues. We approximate the world as consisting of N_{tot} objects. The number of distinguishable objects is N_{obj}. Hence, $N_{obj} \leq N_{tot}$. Two objects are distinguishable if a human being can tell them apart without reading surface markings of any sort. For example, a wrench and a hammer are distinguishable whereas two nails of the same size look exactly the same unless marked in some way. We give each distinguishable object an index i, and we refer to that object as A_i. The number of occurrences, or instances, of that object is denoted as N_i. This yields the relationship

$$N_{tot} = \sum_{i=1}^{N_{obj}} N_i \ .$$

The number of objects which can be recognized by human beings is enormous and is dependent on human experience. The number of objects to be recognized by an object recognition system depends upon the application and the appropriate system training.

It is sometimes difficult to decide what is an object and what is an assembly of objects. Each object should therefore be considered as possessing its own

coordinate system and list of sub-objects. We currently consider only simple objects with no sub-parts and with only one occurrence, or instance, in this discussion. The general case of multiple instances of objects with sub-parts is not conceptually much more difficult with respect to surface characterization problems. However, it does present notation and implementation difficulties. We define the origin of the object coordinate system at the center of mass of the object with three orthogonal axes aligned with the principal axes of the object because these parameters can be uniquely determined for any given solid object.

Each object occupies space, and at most one object can occupy any given point in space. We need to describe the spatial relationships between each object and the rest of the world. One way to describe spatial relationships is through the use of coordinate systems. We define a world coordinate system for reference purposes which can be located at any convenient location. The center of the earth or the center of a table with three suitable directions might serve as such a reference. Then each object can be positioned in space relative to this coordinate system using translation and rotation parameters. We refer to the translation parameters of an object as the vector $\underline{\alpha}$. We refer to the rotation parameters of an object as the vector $\underline{\theta}$. The number of parameters for each vector depends on the dimension of the depth map recognition problem. For the *two-dimensional* case, we could write the parameters as follows:

$$\underline{\alpha} = (\alpha,\gamma) \qquad \underline{\theta} = (\theta) \qquad (2\text{-}D) .$$

We do *not* treat the 2-D case explicitly in this paper, but we mention it as a

part of our formalism for three reasons: (1) it points out how the number of spatial degrees of freedom doubles with an increase of only one dimension (from 2-D to 3-D), (2) if the 2-D problem were addressed correctly, it might give insight into how the 3-D problem could be addressed, and (3) the solution to the 3-D problem should reduce to a meaningful solution for the 2-D case. For the *three-dimensional* case, we write the parameters as follows:

$$\underline{\alpha} = (\alpha,\beta,\gamma) \qquad \underline{\theta} = (\theta,\phi,\psi) \qquad (3\text{-}D) .$$

We consider γ as the depth parameter and θ as the rotation angle in the α–γ plane for the 2-D case. See Figure 2 for the graphical illustration of these parameters.

We can now precisely define our world model W as a set of ordered triples (object,translation,rotation) in the world reference coordinate system:

Figure 2. 2- & 3-Dimensional Translation and Rotation Parameters

---

$$W = \left\{ (A_i, \underline{\alpha}_i, \underline{\theta}_i) \right\}_{i=0}^{N} .$$

We consider object A_0 to be the sensor object with position $\underline{\alpha}_0$ and orientation $\underline{\theta}_0$. If we want a time-varying world model, we can let all parameters be functions of time. For our current purposes of single static view object recognition, we need only concern ourselves with static parameter values. We denote the set of all objects as $L = \{ A_i \}$. This set is also referred to as the *object list.* The set of all translations is denoted R^t where t=2 in the 2-D problem and t=3 in the 3-D problem. The set of all rotations is denoted R^r where r=1 in the 2-D problem and r=3 in the 3-D problem. R is the set of all real numbers.

A depth map sensor obtains a depth map projection of a scene. We model this projection as a mathematical operator P which maps the set $\Omega = L \times R^t \times R^r$ into the set of all scalar functions of $t - 1$ variables which we denote as F:

$$P : \Omega \rightarrow F$$

These real-valued functions are referred to as depth map functions. This projection can be one of two types: *orthographic* or *perspective.* We consider only the orthographic projection because perspective range images can theoretically be transformed directly into orthographic range images using the spherical-to-Cartesian coordinate transformation, surface interpolation, and resampling. We write the projection as

$$f(\underline{x}) = g_{A,\underline{\alpha},\underline{\theta}}(\underline{x}) = P(A,\underline{\alpha},\underline{\theta})$$

where $\underline{x}$ is the vector of $t-1$ spatial variables of the focal plane of the sensor. The spatial parameters of the sensor object (the location $\underline{\alpha}_0$ and the orientation $\underline{\theta}_0$) are *implicitly* assumed arguments of the projection operator. This is done to simplify our expressions because we only have one sensor in this formalism. We refer to the depth map function as f when the identity of the object and its parameters are unknown. The symbol g with subscripts refers to the depth map function of a known object at a known location and orientation. This formalism points out that the set of depth map functions associated with one object is an *infinite family of functions.* The rotation parameters $\underline{\theta}$ have a particularly profound effect on this family of functions: the shape of the depth map function changes as the object rotates. Translation parameters have no effect on shape whatsoever under orthographic projection, and they have minimal effect under the perspective projection unless the sensor is very close to the object of interest (e.g., closer than 10 times the maximum object width).

Since objects do not take up all of space, we need a convention for the value of the depth map function for values of the spatial vector $\underline{x}$ which do not correspond to object surface points. If the point $(\underline{x}, f(\underline{x}))$ cannot lie on an object surface, we assign the value of $-\infty$ to $f(\underline{x})$. Hence, we can write the projection of a set of M objects as

$$f(\underline{x}) = \max_{1 \le i \le M} g_{A_i,\underline{\alpha}_i,\underline{\theta}_i}(\underline{x})$$

The depth map object recognition problem can be rephrased as follows: Given a depth map function $f(\underline{x})$ which results from depth map projection of a 3-D world scene, determine the sets of possible objects with the corresponding sets of translation and rotation parameters which could be projected to obtain the given depth map function. That is, determine the set of all ω_J, subsets of Ω, such that $\omega_J = \left\{(A_j, \underline{\alpha}_j, \underline{\theta}_j)\right\}_{j \in J}$ projects to the depth map $f(\underline{x})$ where J is an index set which depends on the possible valid interpretations of the depth map.

We can write these ideas more mathematically using inverse set mappings. For every *single object* depth map function, there exists a corresponding inverse set mapping which yields all single objects which could have created that object depth map function. We denote the inverse set mapping of P as P^{-1} where

$$P^{-1}(f(\underline{x})) = \left\{(A, \underline{\alpha}, \underline{\theta}) \in \Omega \mid P(A, \underline{\alpha}, \underline{\theta}) = f(\underline{x})\right\}$$

An inverse set mapping takes sets from the power set of the range into sets in the power set of the domain:

$$P^{-1} : 2^F \rightarrow 2^\Omega$$

For our purposes, we restrict the input sets in 2^F to be singletons; therefore, we can replace 2^F with F. For multiple object depth map functions, we must generalize P^{-1} one step further. A generalization is necessary due to the possible combinations of objects. See Figure 3 for a simple example of this

combinational issue. Hence, given $f(\underline{x}) \in F$, we seek a "generalized" inverse set mapping $\mathbf{P}^{-1}$ such that

$$\mathbf{P}^{-1}\{ f(\underline{x}) \} = \left\{ \omega_J \subseteq 2^{\Omega} \mid \max_{j \in J} P(A_j, \underline{\alpha}_j, \underline{\theta}_j) = f(\underline{x}) \right\} .$$

Thus the mapping we seek takes elements of the depth map function space into the power set of the power set of Ω:

$$\mathbf{P}^{-1} : F \rightarrow 2^{2^{\Omega}} .$$

The dimensionality of such a range space is huge even for relatively small finite sets, and the task of computing such an inverse for infinite sets is formidable. We know that human beings can perform such "computations" quickly, easily, and accurately using very compact biological equipment. This "computation" is

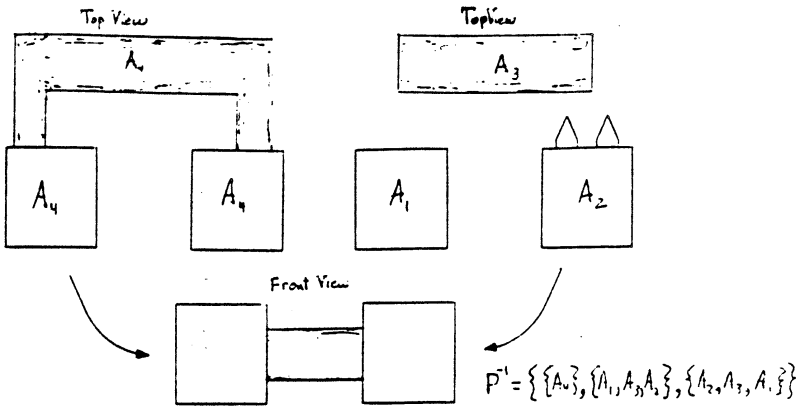

Figure 3. Different Object Interpretations of a Given Image

possibly being performed at several different levels in humans using a variety of features. Knowing that the problem is solvable by some means, we proceed to search for methods in which computers can perform such tasks.

The depth map object recognition problem can now be stated in terms of the modeled world as follows: given the world model W with N_{obj} simple objects and given any realizable depth map function $f(\underline{x})$, compute the inverse projection set mapping $\mathbf{P}^{-1}(f(\underline{x}))$ to obtain all possible explanations of the function $f(\underline{x})$ in terms of the world model. Usually, there should only be one valid scene interpretation; nonetheless, ambiguous situations, such as that in Figure 3, should be recognized as such when given only a single view. Since there is no general theory regarding how this mapping should be computed, we choose an approach that looks reasonable to us. We propose that the generalized inverse projection set mapping should be computed by recognizing the *individual surface regions* associated with the $g_{A,\underline{\alpha},\underline{\theta}}(\underline{x})$ *depth map surface function families* of the individual objects. That is, we propose that the *object recognition* problem be solved by decomposing it into a *surface characterization* problem combined with a *surface matching* algorithm constrained by known object structures. Isolated surface regions can be matched against the families of different surfaces associated with an object such that view-dependent effects are accounted for. We search for visible-invariant features of the object depth map surface families which can be used to reduce matching computations. The surface matching algorithm is an important part of our future research, but we feel that the issue of surface characterization for recognition purposes in depth maps

is close to being resolved.

Before proceeding, it should be noted that the formalization above can be augmented to state the object recognition problem for intensity images. By adding an illumination-reflectance operator I to the depth map function f, we obtain an intensity image $\iota(\underline{x})$ given by

$$\iota(\underline{x}) = I\left(\max_i \; P\left(A_i, \underline{\alpha}_i, \underline{\theta}_i\right)\right).$$

The intensity image object recognition problem is generally more difficult from a mathematical point of view due to the additional inversion of the I operator. Note that the *max* computation is the multiple object occlusion operator. To expand our world model for understanding intensity images, we would also need to add objects which generate light in addition to the single sensor object which receives light. Shape from shading [20], shape from photometric stereo [10][53], and shape from texture [52] techniques attempt to uniquely invert the I operator producing the depth map function f.

It is also interesting to note that human beings can understand images even when other operators besides illumination-reflectance are used to "color" visible surfaces. For example, people can correctly interpret photographic negatives or pseudo-colored images where the usual color and/or light-dark relationships are completely changed.

5. REVIEW OF DIFFERENTIAL GEOMETRY

In the previous section, we discussed how depth map object recognition can be considered a surface recognition problem. But how are surfaces to be recognized? We assume that surfaces can be recognized by their characteristics. But what do we mean by the term "surface characteristic" ? We define a *characteristic* of a mathematical entity, such as a function, to be any well-defined feature which can be used to distinguish among different mathematical entities of the same type. We may consider characteristics which uniquely determine a corresponding entity or characteristics which are many-to-one although the former are usually more desirable. One simple example of a characteristic which uniquely determines a function is a detailed description of the function itself. Another simple example of a many-to-one characteristic is the following: A circle and an ellipse are *round* figures. This round characteristic distinguishes them from rectangles, triangles, and other polygons; however, it does not distinguish between circles and ellipses. In this section, we aim to find a good mathematical characterization of depth map function surfaces. We motivate the discussion by generalization from lower dimensions.

5.1. POINTS

Perhaps the simplest geometric notion is that of a point in a space. Points are considered to be zero-dimensional entities without size. We can represent points in 2-D Euclidean space by dots on a sheet of paper. We can isolate points in 3-D Euclidean space by actually pointing to them. If we can-

not graphically represent a point, we usually resort to the use of coordinate systems for representation purposes. A coordinate system, or frame, consists of three orthogonal directions and an origin. Once these four items are specified, most spatial concerns can be handled relative to that coordinate system. A point in a coordinate system (CS) is determined by its coordinates which can be considered as measurements along the directions of the CS. When we rotate and translate a CS to get a new CS, we can similarly rotate and translate point coordinates to get the new coordinate description of the point in the new CS. There is nothing geometrically invariant about a single point description that allows us to distinguish it from other points. This is not true when we consider sets of points.

5.2. SPACE CURVES

Curves are a natural geometric generalization of points. The simplest kind of curve is a line segment between two points. General curves can be represented parametrically as a function of an interval of the real line. We write a curve C as follows:

$$C = \left\{ \underline{x}(s) \mid a \leq s \leq b \right\} .$$

where $\underline{x}(s)$ is a vector function of a scalar argument s which is not assumed to be arc length. (The under bar denotes "vector.") We consider only *smooth* curves where the components of the $\underline{x}$ vector have continuous second derivatives. The tangent unit vector function $\underline{t}(s)$ is then defined as follows:

$$\underline{t}(s) = \frac{d\underline{x}\ (s)/ds}{||d\underline{x}(s)/ds||} \quad .$$

The normal unit vector function $\underline{n}(s)$ is defined as follows:

$$\underline{n}(s) = \frac{d\underline{t}\ (s)/ds}{||d\underline{t}(s)/ds||} \quad .$$

The binormal vector function $\underline{b}(s)$ is defined as follows to complete a right-handed coordinate system:

$$\underline{b}(s) = \underline{t}(s) \times \underline{n}(s) \ .$$

This coordinate system is shown in Figure 4. The *speed* of a curve C is defined as

$$\nu(s) = ||d\underline{x}(s)/ds|| \ .$$

The *curvature* of a curve C can be defined as

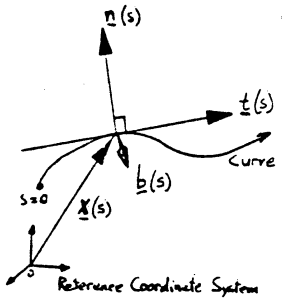

Figure 4. Tangent-Normal-Binormal Coordinate System at Point on Curve

$$\kappa(s) = \frac{\underline{n}\ (s) \cdot d\underline{t}\ (s)/ds}{\nu(s)} \ .$$

The *torsion* of a curve C can be defined as

$$\tau(s) = -\frac{\underline{n}\ (s) \cdot d\underline{b}\ (s)/ds}{\nu(s)} \ .$$

We have now defined all of the main concepts associated with space curves.

Given any smooth curve C described by $\underline{x}(s)$ with respect to some coordinate system, we can compute these functions: $\underline{t}(s),\underline{n}(s),\underline{b}(s),\kappa(s),\tau(s),\nu(s)$. Several questions arise: (1) Given two curves and the associated computed functions, are the functions different when the curves are different? (2) Given only these functions, or some subset of them, do they uniquely determine the curve? (3) What function information corresponds to shape? translation? rotation? (4) How do the functional descriptions change when a change in coordinate system occurs?

Fortunately, all these questions have already been answered. From the definition information above, it is possible to derive the Frenet-Serret equations using vector calculus. We write these equations as a single differential matrix equation:

$$\frac{d}{ds}\begin{bmatrix} \underline{t}(s) \\ \underline{n}(s) \\ \underline{b}(s) \end{bmatrix} = \begin{bmatrix} 0 & \kappa(s)\nu(s) & 0 \\ -\kappa(s)\nu(s) & 0 & \tau(s)\nu(s) \\ 0 & -\tau(s)\nu(s) & 0 \end{bmatrix} \begin{bmatrix} \underline{t}(s) \\ \underline{n}(s) \\ \underline{b}(s) \end{bmatrix} \ .$$

This state equation determines how the tangent-normal-binormal coordinate

frame evolves as a function of the parameter s given the initial coordinate frame. The *speed, curvature, and torsion* functions of the state matrix completely determine what happens to the coordinate frame once the initial conditions are given. If we were to integrate this first-order linear space-varying matrix ordinary differential equation over the entire length of the curve, we would only obtain what the coordinate frame does along the curve, not the curve itself. We only need to integrate the tangent function given the starting point to obtain a complete description of the original curve. The general space curve reconstruction formula is

$$\underline{x}(s) = \int_0^s \underline{t}(s)\nu(s)\, ds + \underline{x}(0) \qquad \underline{x}(s) \in \mathbf{R}^3$$

Thus, we can decompose the curve C into different functional components: $\underline{x}(0)$ is the starting point vector of three numbers which determine the starting point of the curve. $\underline{t}(0)$ is the initial tangent vector which can be completely determined by two numbers (it is a unit vector). $\underline{n}(0)$ is the initial normal vector which can be completely determined by one additional number. $\underline{b}(0)$ is the initial binormal vector which is completely determined by the specification of the initial tangent and normal vectors. $\nu(s)$ is a normalization factor which accounts for the 1-D intrinsic geometry of the curve parameterization. $\kappa(s)$ is the scalar function which determines the instantaneous radius of curvature at each point on the curve. $\tau(s)$ is the scalar function which determines the instantaneous bending of the curve out of the tangent-normal plane.

For a general 3-D space curve, the explicit parameterization $\underline{x}(s)$ is specified by complete knowledge of three scalar functions. When the curve is expressed as the solution of a ordinary differential equation, we need to know three scalar functions *plus* the initial conditions: three scalar translation *values* and three scalar rotation *values*. The component *functions* of $\underline{x}(s)$ change under rotation and translation of CS whereas only the constant component *values* of $\underline{x}(0)$ and $\underline{t}(0),\underline{n}(0),\underline{b}(0)$ change. The speed, curvature, and torsion functions are invariant to these rotation and translation coordinate transformations.

There is a fundamental existence and uniqueness theorem for 3-D space curves which can be proven as a consequence of the fundamental theorem of ordinary differential equations.

(1) **Existence:** Let $\nu(s) > 0$, $\kappa(s) > 0$, and $\tau(s)$ be arbitrary continuous real functions on the interval $a \leq s \leq b$. Then there exists a unique space curve C, up to a translation and rotation, such that $\kappa(s)$ is the curvature function, $\tau(s)$ is the torsion function, and $\nu(s)$ is the speed function.

(2) **Uniqueness:** If two curves C and C^* possess curvature functions $\kappa(s)$ and $\kappa^*(s)$, torsion functions $\tau(s)$ and $\tau^*(s)$, and speed functions $\nu(s)$ and $\nu^*(s)$ respectively such that

$$\kappa(s) = \kappa^*(s) > 0 \quad \text{and} \quad \tau(s) = \tau^*(s) \quad \text{and} \quad \nu(s) = \nu^*(s) > 0,$$

then there exists an appropriate translation and rotation such that C and C^* coincide exactly.

This tells us that arbitrary 3-D smooth curve shape is captured by three scalar functions: *curvature, torsion, and speed.* Oftentimes, the constraint of unit speed is imposed on the original parametric function; this is equivalent to requiring the s parameter to be the arc length along the curve. In this case, curvature and torsion alone specify curve shape.

5.3. PLANE CURVES

Before we move on to surfaces, the case of a planar curve is examined due to the nature of its simplifications. Most of the non-moment-based two-dimensional shape description research makes use of some of the ideas which we present. In addition, this section enhances the contrast between the 2-D and 3-D shape recognition problems and 2-D and 3-D shape characteristics.

Planar curves have zero torsion at all points on the curve. When this is true, there is no reason to consider the binormal vector in the state equation. The tangent-normal coordinate frame state equation simplifies as follows:

$$\frac{d}{ds}\begin{bmatrix}\underline{t}\\ \underline{n}\end{bmatrix} = \begin{bmatrix}0 & \kappa(s)\nu(s)\\ -\kappa(s)\nu(s) & 0\end{bmatrix}\begin{bmatrix}\underline{t}\\ \underline{n}\end{bmatrix}$$

The general planar 2-D curve reconstruction formula can still be written the same as in the 3-D case. However, the 2-D case formula simplifies to the following special form for unit-speed curves:

$$x(s) = x(0) + \int_0^s \cos\left[\phi(0) + \int_0^\beta \kappa(\alpha)d\alpha\right]d\beta \qquad y(s) = y(0) + \int_0^s \sin\left[\phi(0) + \int_0^\beta \kappa(\alpha)d\alpha\right]d\beta$$

where $\phi(0)$ is the initial tangent angle and $(x(0),y(0))$ is the starting point of the curve. We see that arbitrary smooth planar curve shape is captured by two scalar functions: the *speed function* $v(s)$ and the *curvature* function $\kappa(s)$. We shall see that surface shape is captured by two almost exactly analogous 2x2 matrix functions: the *metric* and the *shape operator* respectively. Many 2-D shape recognition techniques use the curvature function, the integral of the curvature function (which is usually called the tangent angle function), or the curve itself as specified by the integrals for $(x(s),y(s))$. Much of the work done in *partial* 2-D shape recognition uses either the tangent angle function [51] or the curvature function [14].

5.4. SURFACES

We have seen that curvature, torsion, and speed uniquely determine the shape of curves. These characteristics are the ideal type of characteristic for a mathematical entity; they are invariant to coordinate transformations and they have a one-to-one relationship with curve shapes. We now discuss similar surface characteristics by generalizing the mathematical framework of curves to surfaces.

Surfaces are a natural geometric generalization of curves. We write down the explicit parametric form of a surface S with respect to some reference coordinate system:

$$S = \left\{ (x,y,z) : x = h(u,v),\ y = g(u,v),\ z = f(u,v),\ (u,v) \in D \subseteq \mathbf{R}^2 \right\}$$

We refer to this general parametric representation as $\underline{x}(u,v)$ where the x-component of the $\underline{x}$ function is $h(u,v)$, the y-component of $\underline{x}$ is $g(u,v)$, and the z-component is $f(u,v)$. In a later section, we use the Monge Patch surface form to describe depth map surface functions. In this case, $h = u$ and $g = v$, which are extremely simple functions. We consider only *smooth* surfaces where all three parametric functions possess continuous second partial derivatives.

There are two fundamental forms that are usually considered in the classical analysis of smooth surfaces [19][34][41], which are referred to as the first and second fundamental forms of a surface. We begin our treatment by defining what the fundamental forms of a surface are in terms of the general explicit parameterization.

The first fundamental form of a surface defined by $\underline{x}(u,v)$ is given by the following quadratic form:

$$I(du,dv) = d\underline{x} \cdot d\underline{x} = \begin{bmatrix} du & dv \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} = d\underline{u}^T \; [\, g \,] \; d\underline{u}$$

where the [g] matrix elements are defined to be

$$g_{11} = E = \underline{x}_u \cdot \underline{x}_u \qquad g_{22} = G = \underline{x}_v \cdot \underline{x}_v \qquad g_{12} = g_{21} = F = \underline{x}_u \cdot \underline{x}_v$$

where the subscripts denote partial differentiation

$$\underline{x}_u = \frac{\partial \underline{x}}{\partial u} \qquad \underline{x}_v = \frac{\partial \underline{x}}{\partial v} \quad .$$

$\underline{x}_u$ and $\underline{x}_v$ are referred to as the u-tangent vector and the v-tangent vector

respectively. We refer to the [g] matrix as the first fundamental form matrix or, more simply, as the *metric* of the surface. Since the vector dot product is commutative, this [g] matrix is symmetric and only has three independent components. We have used the E,F,G notation of Gauss along with the matrix element subscript notation since both are useful in different circumstances and both occur often in the literature of differential geometry. It turns out that the metric depends only upon the surface itself and does not depend on how the surface is embedded in three-dimensional space. It is therefore referred to as an *intrinsic* property of a surface. The *metric* 2x2 matrix function plays basically the same role as the *speed* function does for curves. The intrinsic geometry of a curve is one-dimensional whereas that of a surface is two-dimensional.

In contrast, the second fundamental form matrix of a surface is dependent upon how the surface is embedded in 3-D space and is therefore considered as an *extrinsic* property of the surface. This form determines the shape of surface. The second fundamental form is defined as

$$II\,(du, dv) = -\,d\underline{x} \cdot d\underline{n} = \begin{bmatrix} du & dv \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} = d\underline{u}^{\,T}\ [\ b\ ]\ d\underline{u}$$

where the [b] matrix elements can be defined to be

$$b_{11} = L = \underline{x}_{uu} \cdot \underline{n} \qquad b_{22} = N = \underline{x}_{vv} \cdot \underline{n} \qquad b_{12} = b_{21} = M = \underline{x}_{uv} \cdot \underline{n}$$

$$\underline{n} = \frac{\underline{x}_u \times \underline{x}_v}{\|\underline{x}_u \times \underline{x}_v\|} = \textit{Unit Normal Vector}$$

where the double subscript implies second partial derivatives

$$\underline{x}_{uu} = \frac{\partial^2 \underline{x}}{\partial u^2} \qquad \underline{x}_{vv} = \frac{\partial^2 \underline{x}}{\partial v^2} \qquad \underline{x}_{uv} = \frac{\partial^2 \underline{x}}{\partial u \partial v} = \underline{x}_{vu} \ .$$

The [b] matrix is the second fundamental form matrix and is also symmetric. The Gauss-like L,M,N notation is introduced again as above. Note the similarity in the definitions of these six fundamental form coefficients and in the definitions of curvature, torsion, and speed. Both involve dot products of vectors defined using derivatives. These definitions allow us to discuss the "state" equation for surfaces.

The Gauss-Weingarten equations for 3-D surfaces play the same role as the Frenet-Serret equations for 3-D curves. We write the Gauss-Weingarten equations as a matrix differential equation where the differential operator is a type of gradient operator which acts on the normal, u-tangent, v-tangent coordinate frame field:

$$\begin{bmatrix} \underline{x}_{uu} \\ \underline{x}_{uv} \\ \underline{x}_{vu} \\ \underline{x}_{vv} \\ \underline{n}_u \\ \underline{n}_v \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \end{bmatrix} \otimes \begin{bmatrix} \underline{x}_u(u,v) \\ \underline{x}_v(u,v) \\ \underline{n}(u,v) \end{bmatrix} = \begin{bmatrix} \Gamma^1_{11} & \Gamma^2_{11} & b_{11} \\ \Gamma^1_{12} & \Gamma^2_{12} & b_{12} \\ \Gamma^1_{21} & \Gamma^2_{21} & b_{21} \\ \Gamma^1_{22} & \Gamma^2_{22} & b_{22} \\ -\beta^1_1 & -\beta^2_1 & 0 \\ -\beta^1_2 & -\beta^2_2 & 0 \end{bmatrix} \begin{bmatrix} \underline{x}_u(u,v) \\ \underline{x}_v(u,v) \\ \underline{n}(u,v) \end{bmatrix} .$$

The "transition" matrix in this equation contains twelve coefficient functions

which we have not yet defined. One reason this matrix is more "messy" than the space curve matrix is because the u-tangent and v-tangent vectors of the frame field are not necessarily orthogonal to each other as are the tangent and binormal vectors of curves. The Christoffel Symbols of the Second Kind $\Gamma^k{}_{ij}$ depend only upon the metric $g_{ij}(u,v)$ and are defined as follows:

$$\Gamma^k{}_{ij}(u,v) = \frac{1}{2} \sum_{m=1}^{2} g^{km} \left[\frac{\partial g_{jm}}{\partial u^i} + \frac{\partial g_{mi}}{\partial u^j} - \frac{\partial g_{ij}}{\partial u^m} \right]$$

where g^{km} is the matrix inverse of g_{km}, which is the tensor notation for the metric [g] already defined, and where $u^1 = u$ and $u^2 = v$. Note that $\Gamma^k{}_{ij} = \Gamma^k{}_{ji}$ since the order of differentiation is immaterial for well-behaved smooth surfaces. Tensor notation is a must for maintaining fairly compact equations. The last two row equations of the matrix equation are often referred to as the Weingarten equations for 3-D surfaces. The Weingarten equations coefficients $\beta^i{}_j$ depend on both the first and second fundamental form matrices:

$$\beta^i{}_j = \sum_{k=1}^{2} b_{jk}\, g^{ki} \qquad or \qquad [\,\beta\,] = [\,g^{-1}\,][\,b\,]\,.$$

The $[\beta]$ matrix is sometimes referred to as the "shape operator" matrix [41] which defines the Weingarten mapping which maps tangent vectors to tangent vectors in the tangent plane associated with each point. One can view the $[\beta]$ matrix (or the $[b]$ matrix) as the entity which determines surface shape by relating the intrinsic geometry of the surface to the Euclidean geometry of

three-dimensional space. It is the generalization of the *curvature* of plane curves. In summary, we have now seen that all of the sixteen non-zero "state" matrix coefficient functions depend on only six scalar functions of two variables:

$$g_{11}(u,v) \quad g_{12}(u,v) \quad g_{22}(u,v)$$

$$b_{11}(u,v) \quad b_{12}(u,v) \quad b_{22}(u,v)$$

Assuming we can solve the first-order linear space-varying partial differential matrix equation for the $\underline{x}_u, \underline{x}_v, \underline{n}$ coordinate frame, we can also solve for the parametric surface function in the neighborhood of a point (u_0, v_0) using the following 3-D surface reconstruction formula:

$$\underline{x}(u,v) = \int_{u_0}^{u} \underline{x}_u(\alpha, v_0)\, d\alpha + \int_{v_0}^{v} \underline{x}_v(u,\beta)\, d\beta \qquad \underline{x} \in \mathbf{R}^3$$

This equation is the generalization of the space curve reconstruction formula. Note that it involves two integrals of the two tangent vectors.

As in the case for curves, there is a fundamental existence and uniqueness theorem for 3-D surfaces which is usually credited to O. Bonnet. It is more involved than the curve theorem because the state equation is a partial differential equation.

(1) **Existence:** Let $g_{11}(u,v)$, $g_{12}(u,v)$, $g_{22}(u,v)$ be continuous functions with continuous second partial derivatives. Let $b_{11}(u,v)$, $b_{12}(u,v)$, $b_{22}(u,v)$ be

continuous functions with continuous first partial derivatives. Assume all six functions are defined in an open set D containing the point (u_0, v_0). If all six functions satisfy the following set of compatibility equations and sign restrictions, then there exists a unique surface patch defined in the neighborhood of (u_0, v_0) such that g_{ij} and b_{ij} are the first and second fundamental form matrices respectively. Uniqueness is determined up to a translation and rotation. The sign restrictions are as follows:

$$g_{11} > 0 \qquad g_{22} > 0 \qquad (g_{11}g_{22} - (g_{12})^2) > 0$$

The compatibility equations are as follows:

$$(b_{11})_v - (b_{12})_u = b_{11}\Gamma^1_{12} + b_{12}(\Gamma^2_{12} - \Gamma^1_{11}) - b_{22}\Gamma^2_{11}$$

$$(b_{12})_v - (b_{22})_u = b_{11}\Gamma^1_{22} + b_{12}(\Gamma^2_{22} - \Gamma^1_{21}) - b_{22}\Gamma^2_{21}$$

$$(b_{11}b_{22} - (b_{12})^2) = g_{12}\big((\Gamma^2_{22})_u - (\Gamma^2_{12})_v + \Gamma^1_{22}\Gamma^2_{11} - \Gamma^1_{12}\Gamma^2_{12}\big) +$$

$$g_{11}\big((\Gamma^1_{22})_u - (\Gamma^1_{12})_v + \Gamma^1_{22}\Gamma^1_{11} + \Gamma^2_{22}\Gamma^1_{12} - \Gamma^1_{12}\Gamma^1_{12} - \Gamma^2_{12}\Gamma^1_{22}\big)$$

The first two compatibility equations are often referred to as the Mainardi-Codazzi equations. The third compatibility equation is a statement that the determinant of the second fundamental form matrix is a function only of the metric and is therefore an intrinsic property of the surface. This equation can be written in many different forms. One form

of it is referred to as the Gauss equation as it can be used to prove the *Theorema Egregium* of Gauss.

(2) **Uniqueness:** If two surfaces S and S^* possess fundamental form matrices g_{ij} and b_{ij} and g^*_{ij} and b^*_{ij} respectively such that the following matrix equalities hold

$$g_{ij} = g^*_{ij} \qquad b_{ij} = b^*_{ij} \ ,$$

then there exists an appropriate translation and rotation such that S and S^* coincide exactly.

This tells us that arbitrary smooth surface shape is captured by six scalar functions: g_{11}, g_{12}, g_{22}, b_{11}, b_{12}, b_{22}. We also refer to these as the E,F,G,L,M,N functions.

It is difficult to interpret what each of these functions are individually telling us about surface shape however. It would be advantageous if there were combinations of these functions which would give us easily interpretable surface shape characteristics. Fortunately, there are two curvature functions which combine the information in the six E,F,G,L,M,N functions in two different ways. These two curvature functions do not contain all the information of the six E,F,G,L,M,N functions, but they do contain a great deal of useful information which we describe subsequently. However, if we use only two functions as surface characteristics, we are using characteristics which do not uniquely determine a surface in general. However, we note that for compact *convex* surfaces with the Gaussian curvature $K(u,v) > 0$ (i.e., $LN > M^2$) at

every point, $K(u,v)$ *alone* uniquely specifies surface shape [9][19]. Hence, if we isolate simply connected bounded regions of positive Gaussian curvature, we at least know that we have uniquely determined surface shape within that region with only Gaussian curvature.

In conclusion, we have looked at smooth curves and surfaces and have identified functions which uniquely characterize the shape of these geometric entities. This mathematical survey was intended to motivate the following qualitative statement: *If one wants to characterize the shape of a geometric entity, such as a curve or surface, the characteristics which one chooses should at least have some dependence on those functions which uniquely determine shape according to the mathematics of differential geometry.* In what follows, we discuss in detail two specific functions, Gaussian curvature and mean curvature, which characterize surface shape in easily interpretable ways.

6. SURFACE CURVATURE

We have seen that surfaces can be uniquely characterized by six scalar functions which uniquely determine surface shape and intrinsic surface geometry. These functions are the independent elements of two 2 x 2 symmetric matrix functions of the surface. We now examine two curvature functions which combine the information from the six E,F,G,L,M,N functions.

We defined the shape operator (Weingarten map) matrix $[\beta]$ in the previous section as the matrix product $[g^{-1}][b]$. Hence, the $[\beta]$ matrix combines the first and second fundamental form matrices into one matrix. This matrix is a linear

operator which maps vectors in the tangent plane to other vectors in the tangent plane at a point on a surface. The metric $[g]$ is the generalization of the speed of a plane curve whereas the shape operator $[\beta]$ is a generalization of the curvature of a plane curve. The *Gaussian curvature* function of a surface can be defined from the first and second fundamental form matrices as the determinant of the shape operator matrix function as follows:

$$K = \det[\beta] = \det\left(\begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}^{-1}\right) \det\left(\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}\right) .$$

The *mean curvature* function of a surface can be defined similarly as the trace of the shape operator matrix function as follows:

$$H = tr[\beta] = \frac{1}{2}\, tr\left(\begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}^{-1} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}\right) .$$

Hence, we see that the two different surface curvature functions are obtained by mapping two fundamental form matrix functions into one scalar function. The surface curvature functions (H and K) are the natural algebraic invariants of the shape operator which is the generalization of curvature of a plane curve. Since a 2x2 matrix only has two natural algebraic invariants (the trace and determinant), we see that the surface curvature functions we have defined have a certain uniqueness about them. That is, any other scalar combinations of the E,F,G,L,M,N functions cannot be natural algebraic invariants of the shape operator.

We now elaborate on the mathematical properties of the Gaussian curvature K and the mean curvature H. These properties have motivated our selection of H and K as surface characteristics.

(1) Gaussian and mean curvature are invariant to arbitrary transformations of the u,v parameters of a surface as long as the Jacobian of the u,v transformation is always non-zero. In contrast, the six E,F,G,L,M,N functions all *vary* with u,v transformations. This means that the E,F,G,L,M,N functions are directly dependent upon the choice of the u,v coordinate system even though they uniquely characterize shape. Therefore, it is not desirable to use these six functions as shape characteristics due to their dependence on parameterization.

(2) Gaussian and mean curvature are invariant to arbitrary rotations and translations of a surface. This is due to the fact that E,F,G,L,M,N are also invariant to rotations and translations. This is clear from the definition of these functions and the properties of dot products and cross products. Rotational and translational invariance are extremely important properties for view-independent shape characteristics.

(3) Gaussian curvature is also an *isometric invariant* of a surface. (This statement is equivalent to the statement of the Theorema Egregium of Gauss which says that K depends only on E,F,G). An isometric invariant is a surface property which depends only on the E,F,G functions (and possibly their derivatives). Consider that any surface S with Gaussian curvature K may be mapped to any other surface S^* with Gaussian Curvature K^*. If

the mapping is a distance-preserving (isometric) bijection, then $K = K^*$ at corresponding points on the two surfaces. (An isometric mapping of surfaces is a mapping where corresponding arcs on the surfaces have the same length.) We can utilize this fact in a surface matching algorithm as follows: Two surfaces which can be made to coincide exactly via a rotation and a translation are said to be *congruent.* Congruence implies that an isometry exists between the two surfaces *and* that the shape operators of the two surfaces are equivalent. If the two surfaces are isometric, then there exists a matching between the Gaussian curvature values on the two surfaces which implies that that there exists a matching between regions of constant sign of the Gaussian curvature on the two surfaces. Therefore, *if there does not exist a matching between regions of constant sign of the Gaussian curvature of two surfaces, then the two surfaces are not congruent* and therefore cannot have the same 3-D shape. Similar reasoning can be used for the sign of the mean curvature.

(4) Isometric invariants are referred to as intrinsic surface properties. As stated above, Gaussian Curvature is an intrinsic surface quantity. Intrinsic properties have other important interpretations. For example, the Gaussian curvature function K of a surface does not "care" how the surface is embedded in a higher dimensional space. In contrast, the mean curvature function H does "care" about the embedding; it is an extrinsic surface quantity and is not an isometric invariant. The surface defined by a sheet of paper can be used to demonstrate these ideas: If the paper lies flat on a

desk top, we have $K = 0$ and $H = 0$ at each point on the sheet of paper. If we bend the paper so that no kinks occur, we still have $K = 0$ but now $H \neq 0$. When we bend the paper, we change how the surface is embedded in three-dimensional space, but we do not change the metric (intrinsic) properties of the surface. The within-surface distances between points on the paper remain the same, and the interior angles of a triangle still sum to π radians. In this example, Gaussian curvature is seen to be intrinsic whereas mean curvature is seen to be extrinsic. If the paper were deformed as if it were made of rubber, then Gaussian curvature would change as well as mean curvature.

(5) Another way of looking at intrinsic properties is that they do not change when the direction of the normal vector of the surface is reversed. Usually, we choose outward-pointing normals for surfaces of objects. If the surface is just an orientable surface patch floating in space, we could choose the normal to point in either direction. It turns out that Gaussian curvature maintains its sign when the direction of the normal vector is flipped whereas mean curvature flips its sign. This is due to the fact that the first fundamental form does not change sign while the second fundamental form does when the sign of the normal is flipped.

(6) Gaussian Curvature indicates *surface shape* at individual surface points. When $K(u,v) > 0$ at the surface point $\underline{x}(u,v)$, then the surface is *shaped like an ellipsoid* in the neighborhood of that point. When $K(u,v) < 0$, the surface is *locally saddle-shaped.* When $K(u,v) = 0$, the surface is locally flat,

cylindrical, or conical in shape. If $K = 0$ at every point on a surface, then that surface is referred to as a *developable* surface. Mean curvature also indicates surface shape at individual surface points when considered together with the Gaussian curvature at each point. If $H < 0$ and $K = 0$, the surface is locally ridge shaped. If $H > 0$ and $K = 0$, the surface is locally valley shaped. If $H = 0$ and $K = 0$, the surface is locally flat or planar. If $H < 0$ and $K > 0$, the surface is locally ellipsoidal and peaked (i.e., the surface bulges in the direction of the surface normal). If $H > 0$ and $K > 0$, the surface is locally ellipsoidal and cupped (i.e., the surface bulges in the direction opposite that of the surface normal). If $K > 0$, we can never have $H = 0$. When $K < 0$, $H \neq 0$ indicates whether the surface is predominantly valley shaped or ridge shaped. When $H = 0$ at every point on a surface, then that surface is referred to as a *minimal* surface. Minimal surfaces have some interesting mathematical properties.

(7) Gaussian and mean curvature are *local* surface properties. This allows surface curvature to be used in situations where *occlusion* is a problem because K and H do not depend on global properties of a surface.

(8) As a final note of comparison between H and K, a spherical surface of radius a has $H = \pm\frac{1}{a}$ at every point on the surface where the sign depends on the direction of the normal (inward or outward pointing) whereas $K = \frac{1}{a^2}$ at every point *independent* of the sign of the normal direction vector.

There are other interesting properties of Gaussian and mean curvature, but the above list highlights the important ones for our needs.

We now consider different ways in which Gaussian curvature and/or mean curvature can be defined and/or computed:

(1) **Parallel Transport Definition:** We start at some point P on a surface holding a vector which always points in the same direction (similar to a gyroscope). That direction is marked in a permanent fashion at the starting point. We go for a walk where we leave the point P and later return to it without crossing our path. Our path has enclosed some area we call ΔS. We compare the direction of our vector with the reference direction which was marked when we left and obtain some angle $\Delta\alpha$. We can define the Gaussian curvature of the surface at the point P to be

$$K = \lim_{\Delta S \to 0} \frac{\Delta\alpha}{\Delta S}$$

This definition could be used to derive all other formulas. Note that the *sign* of K is given correctly here. It is seen that there is a definite relationship between angles, area, and curvature.

(2) **Gauss Map Area Derivative Definition for Convex Surfaces:** In order to give this definition, we need first to describe the Gauss map which is shown pictorially in Figure 5. The Gauss mapping takes areas on surfaces to areas on the unit sphere. The unit surface normals at the surface points within the area ΔS on the surface are arranged in the unit sphere so

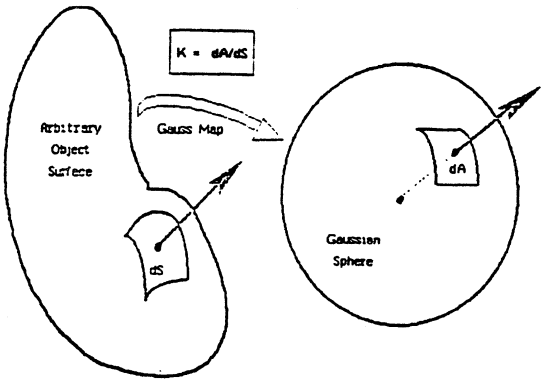

Figure 5. Gaussian Curvature = Gauss Mapping Derivative

that the tail of each normal vector is located at the sphere's center and the tip of the normal vector lies on the unit sphere's surface while preserving the direction of the normal vector. The surface area on the unit sphere (solid angle) subtended by these corresponding normal vectors is denoted ΔA. We can define the Gaussian curvature using the limit of the ratio of these two areas when the surface is convex:

$$K = \lim_{\Delta S \to 0} \frac{\Delta A}{\Delta S}$$

This popular definition seems to imply that K is a dimensionless quantity which is not true. The quantity ΔA should be measured in solid angle units such as steradians rather than area units. This definition can be extended to handle non-convex surfaces, but one needs to be careful about how the area on the unit sphere is computed.

(3) **Gauss Map Jacobian Definition:** This definition is closely related to the area derivative definition, but in this case we can define K in terms of the surface normal and u- and v-tangent vectors.

$$K = \frac{||\underline{n}_u \times \underline{n}_v||}{||\underline{x}_u \times \underline{x}_v||} \qquad where \qquad \underline{n} = \frac{\underline{x}_u \times \underline{x}_v}{||\underline{x}_u \times \underline{x}_v||}$$

(4) **Fundamental Form Matrix Coefficients Definitions:** These definitions of K and H are a restatement of the first matrix definitions given above. We define g = det [g] and b = det [b].

$$K = \frac{b}{g} = \frac{b_{11}b_{22} - (b_{12})^2}{g_{11}g_{22} - (g_{12})^2} = \frac{LM - N^2}{EG - F^2}$$

$$H = \frac{g_{11}b_{22} + g_{22}b_{11} - 2g_{12}b_{12}}{2(g_{11}g_{22} - (g_{12})^2)} = \frac{EN + GL - 2FM}{2(EG - F^2)}$$

(5) **Principal Curvatures Definitions:** At each point on a surface, there is a direction of maximum curvature and a direction of minimum curvature for all space curves which lie in the surface and pass through that point. If we let κ_1 be the curvature in the direction of maximum curvature and let κ_2 be the curvature in the direction of minimum curvature, then we can define the Gaussian and mean curvature in terms of the principal curvatures:

$$K = \kappa_1\kappa_2 \qquad H = \frac{(\kappa_1 + \kappa_2)}{2}$$

It is interesting to note that κ_1 and κ_2 are the two roots of the quadratic equation:

$$\kappa^2 - 2H\kappa + K = 0 .$$

Hence, if K and H are known at each point in a depth map, it is straightforward to determine the principal curvatures:

$$\kappa_{1,2} = H \pm \sqrt{H^2 - K}$$

If $H^2 = K$ at a surface point, the point is referred to as an *umbilic* point to denote that the principal curvatures are equal and every direction is a principal direction. A surface must be locally flat or spherical in the neighborhood of an umbilic point.

(6) **Partial Derivative Expressions:** We can express K and H directly in terms of partial derivatives of the parameterization if desired. We introduce the notation $[\,\underline{a}\ \underline{b}\ \underline{c}] \equiv \underline{a}\cdot(\underline{b}\times\underline{c})$ to simplify the expressions:

$$K = \frac{[\,\underline{x}_{uu}\ \underline{x}_u\ \underline{x}_v\,][\,\underline{x}_{vv}\ \underline{x}_u\ \underline{x}_v\,] - [\,\underline{x}_{uv}\ \underline{x}_u\ \underline{x}_v\,]^2}{\|\,\underline{x}_u \times \underline{x}_v\,\|^4}$$

$$H = \frac{(\,\underline{x}_v\cdot\underline{x}_v\,)[\,\underline{x}_{uu}\ \underline{x}_u\ \underline{x}_v\,] + (\,\underline{x}_u\cdot\underline{x}_u\,)[\,\underline{x}_{vv}\ \underline{x}_u\ \underline{x}_v\,] - 2(\,\underline{x}_u\cdot\underline{x}_v\,)[\,\underline{x}_{uv}\ \underline{x}_u\ \underline{x}_v\,]^2}{2\|\,\underline{x}_u \times \underline{x}_v\,\|^3}$$

In summary, we have seen a few of the many different ways of looking at the Gaussian and mean curvature of a surface. This list was intended to further

stress the properties of these functions as a shape descriptors and indicate how they can be computed given a general parameterization. Note that the two curvature functions are both *nonlinear* combinations of all six E,F,G,L,M,N functions.

The most important invariance properties of surface curvature for view-independent depth map object recognition are the following: (1) invariance under changes in u,v-parameterization and (2) invariance under 3-D translations and 3-D rotations. In addition, we see that mean curvature information significantly complements Gaussian curvature information and vice versa in determining surface shape because H is extrinsic while K is intrinsic. In fact, only *eight* basic local surface types are possible as discussed above, and these types can be determined solely from the signs of the mean and Gaussian curvature. Note also that when K and H are considered together, we have a good generalization of the curvature function of space curves. There exist other functions of the E,F,G,L,M,N functions which may also be useful, but we have attempted to show that the selection of K and H is at least very reasonable. Henceforth, we consider mean curvature and Gaussian curvature as the only surface curvature characteristics. Note that we can compute principal curvatures if needed if we already know mean and Gaussian curvature. We have not proved that K and H are the "optimal" surface characteristics in any sense, but we have given substantial justification for their use as surface shape characteristics. We discuss computation and matching next, but first we briefly review characterization

research done by others.

Surface (and curve) characterization and surface (and curve) matching have been addressed by others in the literature [2,12,16,18,31,33,36,37,39,44,46] and are surveyed in [4]. Some of this work has similarities to our own approach so we mention them here:

(1) Extended Gaussian Images (EGI's) have been studied in detail by Horn and Ikeuchi [17,18,21,22]. This orientation histogram approach provides unique rotationally-invariant shape description for convex objects [35,38], but does not maintain this property for non-convex objects. The properties of Gaussian curvature are important to the EGI concept. Faugeras et al. [12] utilize information which is essentially equivalent to EGI's to perform rotational matching.

(2) The topographic primal sketch proposed by Haralick et al. [16,30] labels each pixel of a surface with one of ten possible labels. Such a labeling is possible within the framework we present. Hessians and directional derivatives are discussed, but not in the context of differential geometry.

(3) Medioni and Nevatia [37] present their shape description ideas in the context of differential geometry, but limit themselves to the zero crossings of the Gaussian curvature and the maximum principal curvature and the maxima of the latter.

(4) Nackman [39] discusses the use of critical point configuration graphs for surface characterization. He isolates four canonical types of slope districts

of smooth surfaces. Our ideas are similar to his suggestions concerning curvature districts.

(5) Marimont [36] identifies patterns of curvature sign changes as view-independent properties of plane curves in 3-D space. Our emphasis on sign of mean and Gaussian curvature generalizes these ideas for surfaces.

7. DISCRETE SURFACE CURVATURE COMPUTATION

We have seen that surface shape is well characterized by two scalar functions, Gaussian curvature and mean curvature, which are independent of parameterization and are invariant to rotations and translations. Given a depth map with only discrete data, how can we compute surface curvature? To compute surface curvature, we only need estimates of the first and second partial derivatives of the depth map. In order to see this, we simplify the expressions for K and H for Monge patch surfaces since all depth maps are in the form of sampled Monge patch surfaces.

First, we recall that the parameterization for a Monge patch takes a very simple form: $\underline{x}(u,v) = [\ u \quad v \quad f(u,v)\ ]^T$. The T superscript indicates transpose so that $\underline{x}$ is column vector. This yields the following formulas for the surface partial derivatives and the surface normal:

$$\underline{x}_u = [\ 1 \quad 0 \quad f_u\ ]^T \qquad \underline{x}_v = [\ 0 \quad 1 \quad f_v\ ]^T$$

$$\underline{x}_{uu} = [\ 0 \quad 0 \quad f_{uu}\ ]^T \qquad \underline{x}_{vv} = [\ 0 \quad 0 \quad f_{vv}\ ]^T \qquad \underline{x}_{uv} = [\ 0 \quad 0 \quad f_{uv}\ ]^T$$

$$\underline{n} = \frac{1}{\sqrt{1 + f_u{}^2 + f_v{}^2}} \begin{bmatrix} -f_u & -f_v & 1 \end{bmatrix}^T$$

These vectors are combined using the dot product definitions given earlier to form the six fundamental form coefficients:

$$g_{11} = 1 + f_u{}^2 \qquad g_{22} = 1 + f_v{}^2 \qquad g_{12} = f_u f_v$$

$$b_{11} = \frac{f_{uu}}{\sqrt{1 + f_u{}^2 + f_v{}^2}} \qquad b_{12} = \frac{f_{uv}}{\sqrt{1 + f_u{}^2 + f_v{}^2}} \qquad b_{22} = \frac{f_{vv}}{\sqrt{1 + f_u{}^2 + f_v{}^2}}$$

Hence, we see that the *five* partial derivatives f_u, f_v, f_{uu}, f_{uv}, f_{vv} are all we need to compute the *six* fundamental form coefficient functions for a Monge patch surface.

Next, we recall that we can compute Gaussian curvature as the ratio of the determinants of the two fundamental form matrices. That expression can be written directly in terms of the depth map function (Monge patch) derivatives as follows:

$$K = \frac{f_{uu} f_{vv} - f_{uv}{}^2}{(1 + f_u{}^2 + f_v{}^2)^2} = \frac{\det(\nabla \nabla^T f)}{\|\nabla f\|^4}$$

∇ is the two-dimensional (u,v) gradient operator. $\nabla \nabla^T$ is then the Hessian matrix operator. Hence, if we are given a depth map function $f(u,v)$ which possesses first and second partial derivatives, we can compute the Gaussian curvature directly. We can view this equation as a second order hyperbolic non-

linear inhomogeneous partial differential equation. Since the analogous equation for H is quasilinear, we postpone our comments until the next paragraph.

We also recall that we can compute mean curvature as the trace of the shape operator. This expression can also be written directly in terms of the depth map function (Monge patch) derivatives as follows:

$$H = \frac{f_{uu} + f_{vv} + f_{uu} f_v{}^2 + f_{vv} f_u{}^2 - 2 f_u f_v f_{uv}}{2\,(\,1 + f_u{}^2 + f_v{}^2\,)^{\frac{3}{2}}} = \nabla\cdot\left(\frac{\nabla f}{\sqrt{1 + \|\nabla f\|^2}}\right)$$

($\nabla\cdot$) is the divergence operator of vector calculus. Again, if we are given a depth map function $f(u,v)$ which possesses first and second partial derivatives, we can compute the mean curvature directly. It is also interesting to note that the above equation (where H is known but f is not) is a second order elliptic quasi-linear inhomogeneous partial differential equation. If H is the mean curvature function of a surface, it should be possible to obtain a unique f from H. Plateau's problem of partial differential equations only addresses the existence of such functions. *We conjecture that $H(u,v)$ and $K(u,v)$ plus boundary conditions uniquely determine $f(u,v)$ for Monge patch surfaces. We have not yet found the proof.*

Since any sampled depth map function encountered in practice can be approximated arbitrarily well by a sufficiently smooth function (which possesses first and second partial derivatives), our next problem is how to compute estimates of these partial derivatives given the sampled data.

7.1. ESTIMATING PARTIAL DERIVATIVES OF SAMPLED SURFACES

We now address the problem of estimating the partial derivatives of a surface function based on samples of that function. In general, numerical differentiation is discouraged if a problem can be addressed using other means. It turns out that there is at least one way to compute Gaussian curvature without using explicit derivative estimates [33]. This technique originates in the Regge calculus of general relativity where geometry is analyzed without coordinates. One must first obtain a discrete triangularization of a surface to utilize this technique. If we look at a particular point $\underline{x}_k$ on the triangularized surface, it is the vertex for N different triangles. See Figure 6 for an example of the geometry. We assume that the lengths of the sides of the i–th triangle are a_i, b_i, c_i where c_i is the length of the side opposite the point of interest and where $a_{i+1} = b_i$. The angle deficit Δ_k at the point $\underline{x}_k$ is then given by

$$\Delta_k = 2\pi - \sum_{i=1}^{N} \theta_i \qquad where \qquad \theta_i = \cos^{-1}\left(\frac{a_i^{\,2} + b_i^{\,2} - c_i^{\,2}}{2 a_i\, b_i} \right) .$$

The Gaussian curvature at a point can be computed based on the angle deficit as follows:

$$K = \frac{2\Delta_k \cdot \delta(\underline{x} - \underline{x}_k)}{\left(\sum_{i=1}^{N} A_i \right)}$$

where

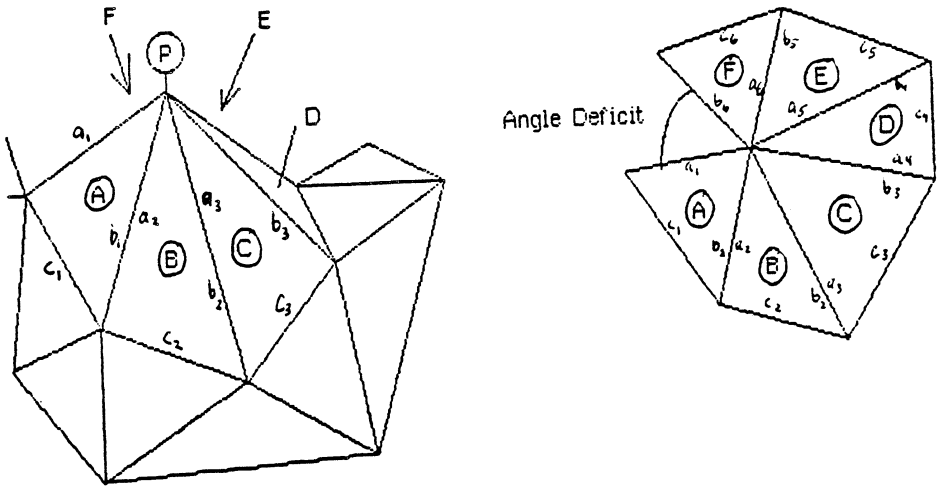

Figure 6. Discrete Gaussian Curvature at a Point

$$A_i = \sqrt{s\,(s - a_i)(s - b_i)(s - c_i)} \qquad s = \frac{a_i + b_i + c_i}{2}$$

and where $\delta(\cdot)$ is the Dirac delta function. We prefer our current method over this method, however, because our current method determines Gaussian curvature in addition to many other characteristics with comparable results and comparable noise sensitivity. It is impossible to compute mean curvature with a similar approach due to its extrinsic nature.

Our current method is now described. The basic approach is the following: (1) given discrete sample data, determine a continuous differentiable function which "best" fits that data in some sense, and (2) compute the derivatives of the continuous function analytically and evaluate them at the corresponding discrete points. Ideally, it would be nice to fit all data with one smooth surface. This problem is computationally intensive and should only be used as a

last resort if no simpler methods work. Instead, we compute only a local surface fit within each N x N window of discrete depth map surface data. Our experimental results show that this approach is adequate. This method, which uses the local quadratic facet model, is also discussed in [3] [6] [15].

It is much easier to describe the two-dimensional surface fitting problem if the one-dimensional curve fitting problem is treated first. We assume that we are given N data points where N is *odd*. Each data point is associated with a position u from the set

$$U = \left\{ \frac{-(N-1)}{2}, \cdots, -1, 0, 1, \cdots, \frac{(N-1)}{2} \right\}$$

$f(u)$ for each $u \in U$. We want to construct some function $\hat{f}(u)$ such that it closely approximates the given data from the original function $f(u)$. For lack of a better criterion, we use the least square error criterion: find the function $\hat{f}$ from some class of functions such that the error term

$$\epsilon = \sum_{u \in U} (f(u) - \hat{f}(u))^2$$

is minimized. Since we are interested in computing first and second derivatives, we really only want to retain the constant, linear, and quadratic terms from the Taylor series expansion for the function. That is, we should choose $\hat{f}$ from the set of all quadratic polynomials so that all "information" about the function is "concentrated" in the quantities we want to know. We could try to

find functions of the form

$$\hat{f}(u) = a_0 + a_1 u + a_2 u^2$$

and choose a_0, a_1, a_2 to minimize the error ϵ. This turns out to be not such a smart thing to do because we end up with $\hat{f}(0) = a_0 = f(0)$ which means that we require the function to pass through the central data point of the window but not any of the other points. A better approach is to look for functions of the form

$$\hat{f}(u) = a_0\phi_0(u) + a_1\phi_1(u) + a_2\phi_2(u)$$

where the $\phi_i(u)$ functions are discrete orthogonal polynomials. We use the following discrete orthogonal polynomials which provide a "dc" offset to the quadratic term which allows a_0 not to be fixed to the value $f(0)$:

$$\phi_0(u) = 1 \qquad \phi_1(u) = u \qquad \phi_2(u) = (u^2 - \frac{M(M+1)}{3})$$

where $M = \frac{(N-1)}{2}$ (N is odd). These discrete quadratic polynomials are easily shown to be orthogonal on U:

$$\sum_{u \in U} \phi_i(u)\phi_j(u) = 0 \quad \text{for } i \neq j$$

The least square error estimate of $f(u)$ is then given by

$$\hat{f}(u) = \sum_{i=0}^{2} a_i \phi_i(u)$$

where the coefficients a_i are given by the inner product sum

$$a_i = \sum_{u \in U} f(u)\, b_i(u)$$

where the $b_i(u)$ functions are normalized versions of the orthogonal polynomials

$$b_i = \frac{\phi_i(u)}{\sum_{u \in U} \phi_i^2(u)} .$$

The three functions b_0, b_1, b_2 can be computed explicitly:

$$b_0(u) = \frac{1}{N} \qquad b_1(u) = \frac{3}{M(M+1)(2M+1)}\, u$$

$$b_2(u) = \frac{1}{P(M)} \left(u^2 - \frac{M(M+1)}{3}\right)$$

where $P(M)$ is a fifth-order polynomial in M :

$$P(M) = \frac{8}{45} M^5 + \frac{4}{9} M^4 + \frac{2}{9} M^3 - \frac{1}{9} M^2 - \frac{1}{15} M .$$

Once $\hat{f}$ has been determined, the derivatives of $\hat{f}$ at the center of the wi
($u = 0$) are then particularly easy to compute:

$$\frac{d\hat{f}}{du}(0) = a_1 \qquad \frac{d^2\hat{f}}{du^2}(0) = 2\, a_2$$

The recipe for computing derivatives at a sample point using odd size data

windows is very simple since the $b_i(u)$ vectors can be precomputed and stored for any given window size: compute a_1 using the summation expression above to get the least squares estimate of the first derivative and compute 2 a_2 to get the second derivative estimate.

This technique is easily extended to two dimensions as follows. We are given surface sample data for the N x N window parameterized by $U \times U$, and we wish to estimate the first and second partial derivatives at the center of the window. We seek a function estimate $\hat{f}(u,v)$ of the form

$$\hat{f}(u,v) = a_{00}\phi_0(u)\phi_0(v) + a_{10}\phi_1(u)\phi_0(v) + a_{01}\phi_0(u)\phi_1(v) +$$

$$a_{11}\phi_1(u)\phi_1(v) + a_{20}\phi_2(u)\phi_0(v) + a_{02}\phi_0(u)\phi_2(v)$$

which minimizes the error term

$$\epsilon = \sum_{(u,v) \in U^2} (f(u,v) - \hat{f}(u,v))^2 \ .$$

The solution is a simple generalization of the 1-D case:

$$a_{ij} = \sum_{(u,v) \in U^2} f(u,v) b_i(u) b_j(v) \ .$$

The first and second partial derivatives are then given by

$$f_u = a_{10} \quad f_v = a_{01} \quad f_{uv} = a_{11} \quad f_{uu} = 2\, a_{20} \quad f_{vv} = 2\, a_{02} \ .$$

The fit error can be computed after the a_{ij} coefficients are determined:

$$\epsilon = \sum_{(u,v) \in U^2} f^2(u,v) - \sum_{i,j} a_{ij} \left(\sum_u \phi_i{}^2(u)\right)\left(\sum_v \phi_j{}^2(v)\right) .$$

Since it turns out that the discrete orthogonal quadratic polynomials over the 2-D window are separable in u and v as shown in the above equations, we can compute our partial derivative estimates for an entire depth map using a separable convolution operator. This is quite efficient. These derivative estimates can then be plugged into the equations for the Gaussian curvature and the mean curvature. This describes all the mathematical details necessary to compute curvature functions $K(u,v)$ and $H(u,v)$ given a samples from a continuous depth map function $f(u,v)$.

The disadvantages of this method are the following:

(1) A different quadratic surface is effectively fitted to the neighborhood of each point. No compatibility constraints are imposed on these surfaces so that the net continuous surface interpretation is meaningful. Unfortunately, we need to make important a priori assumptions about the surface in order to correct this, and making these kind of assumptions is contrary to our goal to use as few a priori assumptions as possible in our data-driven processing.

(2) It turns out that all columns (or rows) in a window operator are weighted equally which seems counter-intuitive. If one asks for a 9x9 window least squares estimate of the first derivative of a depth map at a particular pixel, the data which runs four pixels away has the same impact on the

final estimate as does the data which runs directly through the pixel at which we are estimating the derivative. This situation can be modified using weighted least squares techniques. The question then arises: What is the optimal assignment of weights? A particular triangular weight assignment was quickly tried on a few particular depth maps in experiments and was found to give different, but neither better nor worse results. One might argue that Gaussian weights should be used, but this is not likely to change the results much.

The use of surface critical points for surface characterization has been discussed by Nackman [39]. He notes that critical points surround slope districts in only four canonical ways. The critical points of a function $f(u,v)$ are those points (u,v) where $f_u(u,v) = 0$ *and* $f_v(u,v) = 0$. Since we have to compute f_u and f_v functions to compute K and H anyway, it is a trivial mathematical step to additionally determine the critical points of the given depth map. For surfaces, there are three kinds of *non-degenerate* critical points where $f_u = f_v = 0$ and $K \neq 0$:

(1) Peak Points: $K > 0$ and $H < 0$,

(2) Pit Points: $K > 0$ and $H > 0$,

(3) Pass (Saddle) Points: $K < 0$.

In addition, there are three kinds of *degenerate* critical points where $f_u = f_v = 0$ and $K = 0$:

(1) Ridge Points: $H < 0$.

(2) Valley Points: $H > 0$.

(3) Planar Points: $H = 0$.

Hence, if we compute the zero-crossings of the first partial derivatives in addition to Gaussian and mean curvature, then we will have a richer structural description of the depth map surface which can be used as a more detailed surface characterization.

The characterization which we have proposed is a generalization of the 1-D function characterization techniques. Computing critical points is a generalization of computing the zeros of the first derivative. Computing the sign of Gaussian and mean curvature is a generalization of computing the sign of the second derivative to see if the function is concave up or down.

It is also convenient to compute four other quantities which may be of interest in surface characterization and depth map segmentation. The first of these quantities is the square root of the determinant of the first fundamental form matrix:

$$\sqrt{g} = \sqrt{EG - F^2} = \sqrt{1 + f_u{}^2 + f_v{}^2}\,.$$

This *metric determinant* quantity can be summed over depth map regions to obtain the approximate surface area of the region. This summation corresponds to the continuous formulation

$$A = \int\int \sqrt{1 + f_u{}^2 + f_v{}^2}\; du dv\,.$$

It can also be considered as an edge magnitude map since it is approximately equal to the square root of the sum of the squares of the first partial derivatives. Hence, this output is similar to the output of many edge detectors. It can be thresholded using a minimum depth separation distance to create a simple binary edge image. In depth maps, these edges generally correspond to depth discontinuities in the real world which generally correspond to the occluding boundary of an object. Thus, the existence of a depth discontinuity and a surface region boundary along a curve reinforce the interpretation of that curve as an occluding object boundary for segmentation purposes.

A second quantity which is easy to compute given the derivative information already computed is the quadratic variation:

$$Q = f_{uu}{}^2 + 2f_{uv}{}^2 + f_{vv}{}^2 .$$

When this function is integrated (summed) over a depth map region, the integral (sum) is a measure of the *flatness* of that region. These two functions, or images, could be computed in parallel with the Gaussian and mean curvature using the computed derivative information and could be used to quickly provide surface area and flatness measures of the surface regions segmented later in the processing.

A third quantity is the coordinate angle function Θ which is defined as

$$\Theta = \cos^{-1}(F / \sqrt{EG}) = \cos^{-1}\left(\frac{f_u f_v}{1 + f_u{}^2 + f_v{}^2 + f_u{}^2 f_v{}^2} \right) .$$

This function measures the non-orthogonality of the u,v parameterization: $\Theta = \pi/2$ when the u- and v-tangent vectors are orthogonal and ranges between 0 and π when they are not orthogonal. Also $\cos\Theta = 0$ implies that at least one of the first partial derivatives is zero in the Monge patch formulation.

The last quantities which we mention are the principal directions of the surface at each point. The principal direction vectors of the surface along with H and K determine the shape operator of the surface. We can compute the principal direction angles in the u-v plane as follows:

$$\Phi_{1,2} = \tan^{-1}\left[\frac{-B \pm \sqrt{B^2 - AC}}{C}\right]$$

where $A = EM - FL \qquad 2B = EN - GL \qquad C = FN - GM$.

We do not propose to use these angles as surface characteristics because they seem to be very sensitive to noise. Nevertheless, we have included them in the discussion for completeness of the continuous surface description. We conjecture that the H,K,g,Θ,Φ_1,Φ_2 function description of a surface is equivalent to the E,F,G,L,M,N function description as related to the fundamental existence and uniqueness theorem of general surfaces. Our experimental results show a gray scale image of one principal direction at each depth map point for general interest.

We summarize the computational characterization process prescribed above in the continuous and discrete cases:

(1) **Continuous Case:**

Input: A function $f(u,v)$ defined on some subset of the plane.

Process:

(a) Compute f_u, f_v, f_{uv}, f_{uu}, f_{vv} using analytical techniques,

(b) Compute K, H, $\sqrt{g}$, $\cos\Theta$, and Q using the formulas given above,

(c) Compute zeros of f_u and f_v, K, H, and $\cos\Theta$.

Output:

(a) Two three-level functions $sgn(K)$ and $sgn(H)$ where $sgn(\ )$ is the signum function which yields 1 if the argument is positive, 0 if the argument is zero, and -1 if the argument is negative.

(b) Two non-negative functions $|K|$ and $|H|$ which describe the magnitude of the two surface curvatures.

(c) A binary function $c(u,v)$ which is 1 when (u,v) is critical and 0 when it is not. A result of Morse theory tells that the non-degenerate critical points of smooth surfaces are isolated. Ridge and valley points form curves. Planar points form areas. Each resulting point, curve, or area should be labeled with its appropriate classification.

(d) Two non-negative functions $\sqrt{g}$ and Q. These functions can be integrated in later processing to provide more features of segmented surface regions.

(e) The binary image functions denoting the zeros of K, H, and $\cos\Theta$.

(2) **Discrete Case:**

Input:

A matrix of values $\hat{f}(i,j)$ where $0 \leq i \leq (N_u - 1)$ and $0 \leq j \leq (N_v - 1)$ and $0 \leq \hat{f} \leq 2^{N_{bits}} - 1$ where N_{bits} is the number of bits used for sensor data quantization.

Process:

(a) Compute $\hat{f}_u$, $\hat{f}_v$, $\hat{f}_{uv}$, $\hat{f}_{uu}$, $\hat{f}_{vv}$ matrices using window convolution techniques described above,

(b) Compute K, H, $\sqrt{g}$, $\cos\Theta$, Q, and ϵ matrices using the analytical formulas given above,

(c) Compute the zeros of $\hat{f}_u$ and $\hat{f}_v$, K, H, and $\cos\Theta$.

Output:

(a) Two three-level images, or matrices, $sgn(K)$ and $sgn(H)$.

(b) Two non-negative images $|K|$ and $|H|$ which describe the magnitude of the two surface curvatures at each discrete point.

(c) A binary image matrix $c(i,j)$ which is 1 when (i,j) is critical and 0 when it is not. Each critical pixel is classified into one of the six categories listed above using the corresponding curvature values.

(d) Three non-negative images $\sqrt{g}$, Q, and ϵ which can be used for later feature computation.

(e) The binary images denoting the zeros of K, H, and $\cos\Theta$.

Note that we can "compress" a great deal of useful structural surface information about a N_{bits}-digitized depth map into eight levels (*only three bits*) if we use the signs of the Gaussian and mean curvature. This second order sign information substantially constrains the possibilities of visible surfaces. In

addition, we can create a classified list of critical points which describes the surface. This list usually only contains a small number of points compared to the total number of pixels in the range image and could be used as starting points for a matching algorithm. The other images provide additional information.

8. PROPOSED APPROACH FOR 3-D MATCHING ALGORITHM

A computational theory of depth map surface characterization has been formalized in the previous sections. Our surface matching theory has not been refined or implemented yet, but we wish to discuss our current ideas. We assume that we are given a set of real world objects which we would like to be able to recognize in range images and that we can model them by creating some 3-D object representation. We also assume that a depth map can be obtained from some arbitrary view using a laser rangefinder. (We have interactive solid modeling software and access to real range images.) We can compute the surface characteristics of the depth map as described above. *We must next establish a quantitative relationship between the modeled objects in the world model and the surface characteristic scene description domain.* There are many ways in which this might be done.

If we assume for a moment that we have an efficient way to hypothesize that a certain object exists at a particular location and orientation, we can consider how we might verify that the computed surface characterization is compatible with the hypothesized object. Since Gaussian and mean curvature are

invariant to rotations and translations of coordinate systems and object surface parameterization, we should be able to compute K and H at every object surface point in the object-centered coordinate system. We might therefore be able to obtain a view-independent 3-D surface curvature description as follows: (1) create three copies (A,B,C) of the surface geometry of each object, (2) compute the Gaussian and mean curvature at each point on the original object's surface, (3) "color" the surface of object copy A with the signs of the Gaussian and mean curvature at each point (eight "colors" are needed), (4) "shade" the surface of object copy B with the magnitude of the Gaussian curvature at each point, and (5) "shade" the surface of object copy C with the magnitude of the mean curvature at each point. The three object copies could be computed off-line prior to use and could automatically be projected as needed to the surface characteristic scene description domain for matching verification purposes using a depth-buffer algorithm on the given object model and then post-processing with the "coloring" and "shading" information of the three copies. Surface curvature maxima would then be easily available from the three-dimensional model. Window size information would need to be incorporated to determine what curvature values should be assigned to abrupt dihedral edges and vertices.

It would not be possible to compute a 3-D representation of all the possible depth-discontinuities and critical points for arbitrary objects because these quantities are inherently view-dependent. But this does not mean that they can't be useful. If the object is polyhedral, critical points of the depth map projections appear only at object vertices and depth discontinuities appear only at

facet edges except in degenerate viewing cases. For curved objects, other properties hold. For example, the surface curvature of a surface places constraints on the (curve) curvature of a depth-discontinuity curve generated at the occluding contour of that surface.

The ideas expressed above are an example of some of the ideas that we have been considering to establish quantitative relationships between the surface characteristic scene description domain and the world model domain. The main concepts of the model-based matching approach based on these relationships are shown in Figure 7. Using geometric object models and synthetic depth maps, we shall attempt to create a *matching representation* of 3-D objects which is

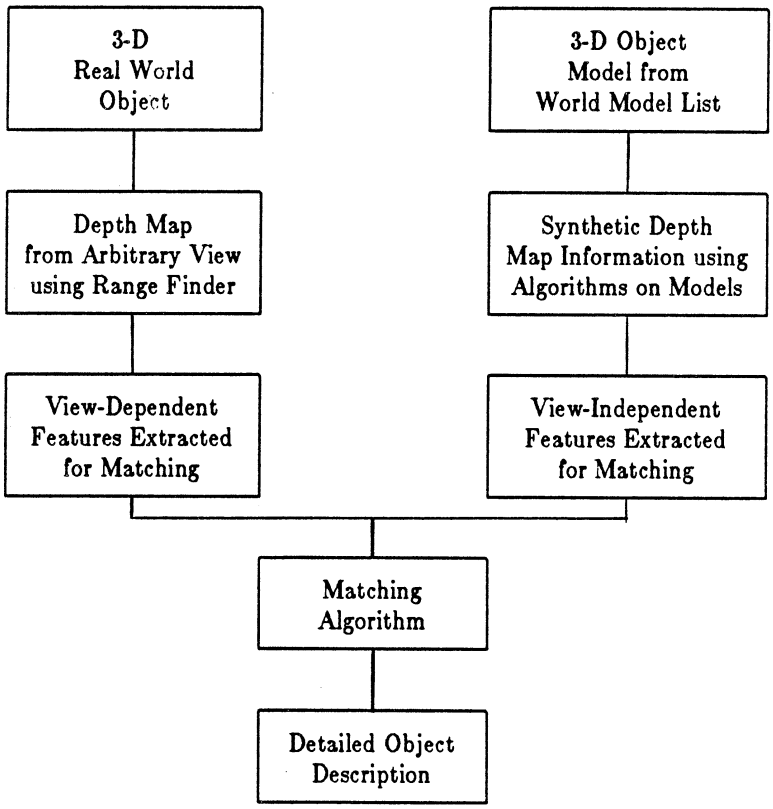

Figure 7. Model-Based Matching of Depth Map Features

independent of view. The matching algorithm must take the view-dependent features (the depth map surface characteristics) and the view-independent matching representation of each object and make comparisons. Segmentation will take place as surface regions of the depth map are matched to the possible objects which they correspond to. Possible matched objects will be tested for consistency with surrounding characterized surface regions in the depth map. The details of the matching representation of the object and the matching mechanism are not clear at this point in time. One idea is to use a condensed kind of characteristic view list or visual potential graph. The representation might be of the following form: IF object A is present, THEN set 1 of surface characteristics will be visible OR set 2 of surface characteristics will be visible OR ... OR set N_A of surface characteristics will be visible. N_A would be a measure of the object complexity. For example, if a sphere is present and visible, we expect to see a complete or partial 2-D circular region in the K-H sign map where $K > 0$ and $H < 0$ and where that region is surrounded by a depth-discontinuity or orientation-discontinuity. If the center of that circle is visible, we expect to see a critical point at the center of the circle with no other critical points present in the complete or partial circle. If the center of that circle is not visible, we should see no critical points whatsoever in the visible subset of the circle region. In addition, we could even look at the first derivative zero-crossing lines to see if pass through or near the circle center. Examples of this idea for spheres are shown in Figure 8. Of course, this is all special case logic being applied in this example. If we pursue this approach, one goal will be to

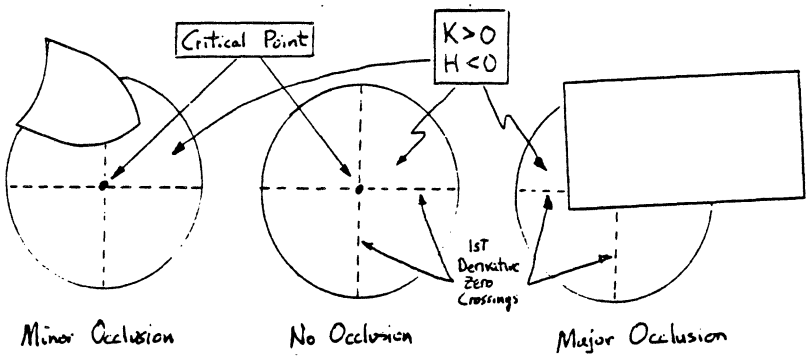

Figure 8. Examples of Depth Map Features for Spheres

devise an algorithm which produces lists of sets of surface characteristics associated with a given object automatically.

Regardless of the matching algorithm details, there are a few concepts about the necessary geometric data structures which will probably not change as research proceeds. Given a depth map, it is processed as described earlier. After this processing, we form a list of *surfaces* based on Gaussian and mean curvature, a list of *edges* based on the metric determinant and the magnitude of the mean curvature, and a list of *points* which will contain critical points and edge-to-edge junction points. Surfaces, edges, and points will have their own data structures to be described below. The depth map data structure is shown in Figure 9. We will want to pre-process each object in the object list of the world model to form its own intermediate data structure. This data structure will be situated between the solid modeling data structure and the matching

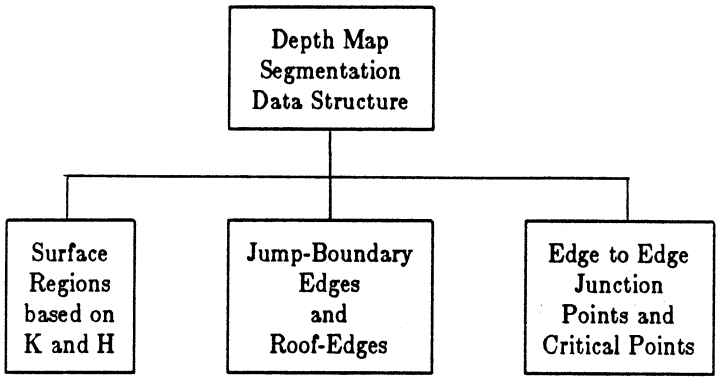

Figure 9. Depth Map Segmentation Data Structure

representation alluded to above. The intermediate object model data structure will *parallel* the depth map data structure. It will consist of a list of object surfaces partitioned based on mean and Gaussian curvature, a list of surface intersections or edges, and a list of edge intersections or points. Surfaces, edges, and points will have their own data structures. The object model data structure is shown in Figure 10.

For each surface, edge, and point in the depth map or object model data structures, we hope to employ common surface, edge, and point data structures respectively. The surface model data structure is shown in Figure 11. Each surface is segmented from the rest of the object or the rest of the depth map and classified using the sign of the surface curvature values. When a surface is planar, the unit normal vector is computed and stored with the surface. If the surface is a ridge or valley surface, the ridge or valley 3-D line segment is computed and stored. If the surface contains a peak or a pit or has negative Gaussian curvature(saddle), maximum curvature points will be computed and stored.

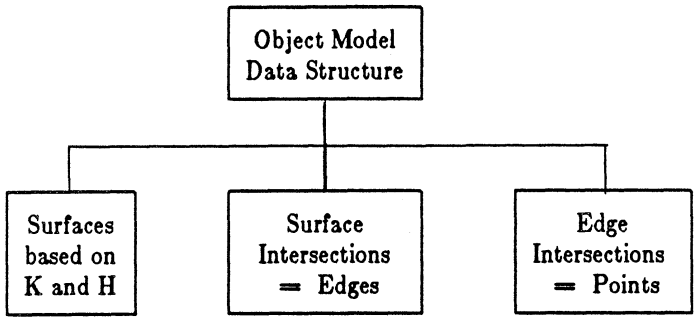

Figure 10. Object Model Data Structure

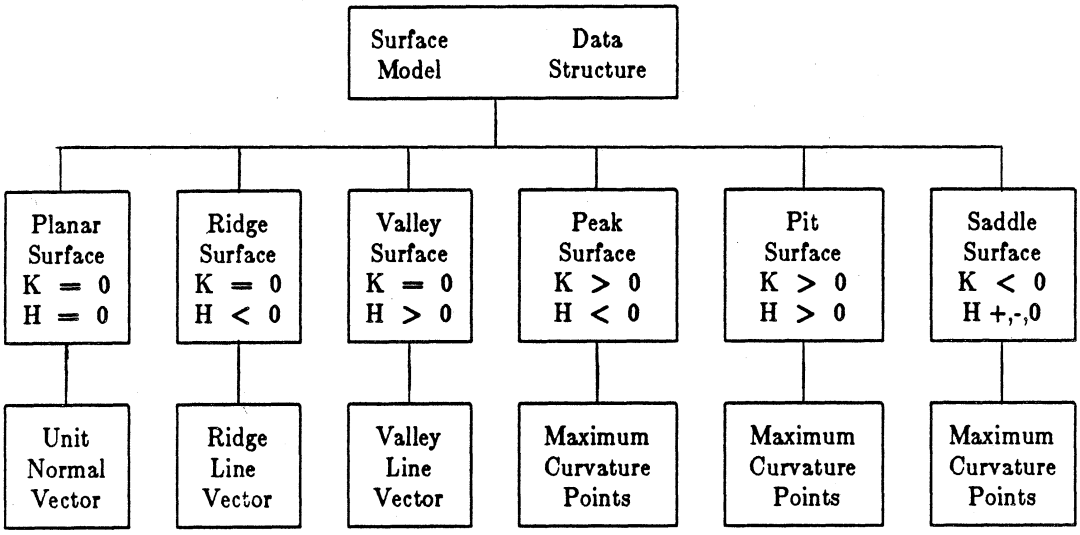

Figure 11. Surface Model Data Structure

It may also be necessary to compute the principal curvature directions at these curvature maxima. In addition to all this, we will also want to compute and store the following items with surfaces:

(1) The edge list which specifies the surface boundary

(2) The total surface area of the surface or the maximum viewable projected surface area.

(3) The length of all edges in the edge list: the boundary perimeter, or the occluding boundary perimeter of the maximum viewable surface area.

For peak, pit, and saddle surfaces, we will want to compute the average Gaussian curvature. For peak, pit, saddle, ridge, and valley surfaces, we will also want to compute the average mean curvature. For saddle surfaces, we will want to know the average absolute value of the mean curvature, too. We may want to split saddle surfaces into positive, negative, and zero mean curvature surfaces. Most of these things are easy to compute given the output of the surface characterization algorithm and will probably help to identify surfaces.

The edge model data structure is shown in Figure 12. Each edge will have a pointer to the one or two surfaces to which it is adjacent. We will probably

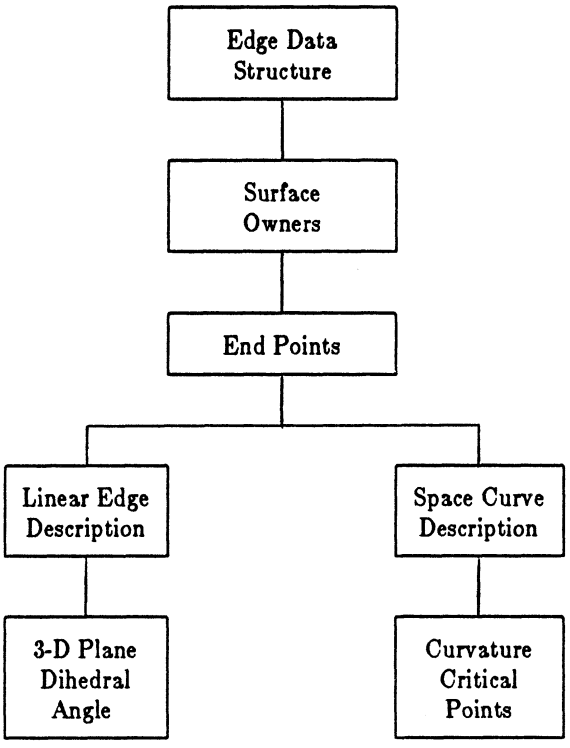

Figure 12. Edge Model Data Structure

want to decompose edges into intervals of constant curvature sign. The end-points of these edge intervals will be stored in the point list. If the edge is linear, we will want to compute and store the direction vector and the dihedral angle if it is known or the dihedral angle upper bound. If the edge is curved, we will want to store a space curve description and note the curve's maxima-of-curvature points as in Marimont [36]. We note at this point that the surface adjacency list can be computed for any surface by forming a list of the other surface owners of each edge which bounds that surface. We also note that depth-discontinuities generate a pair of 3-D edges: one at the occluding boundary of the front object and one at the occlusion edge on the back object.

The point model data structure is shown in Figure 13. We will probably want to come up with a normalized shape description of objects which constrains orientation, location, and size. Location could be normalized for object

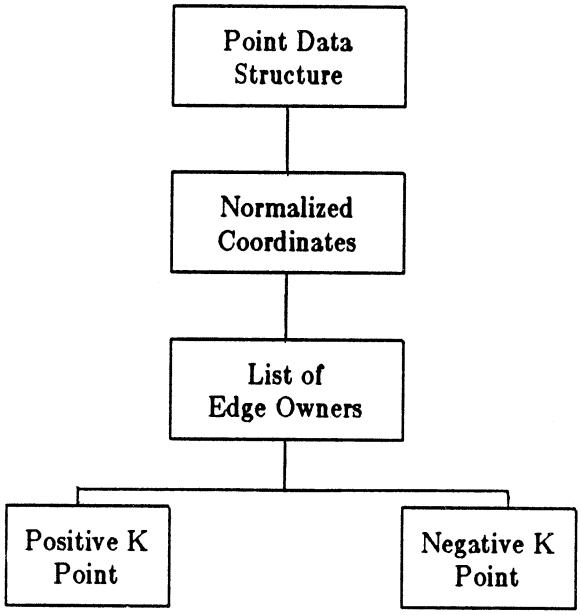

Figure 13. Point Data Structure

models by placing the object coordinate system origin at the center of mass of the object (assume uniform density). Orientation could be normalized by computing the principal axes and the principal moments of inertia for the object and aligning the largest moment with a specified axis (e.g., the z-axis), etc. Scale could be normalized by adjusting the size of the object so that it just fits inside the unit sphere with the unit sphere origin at the object origin (i.e., center of mass). In this normalized system we could give coordinates of the points of the object which are important to our characterization. For depth map points, we will leave the normalized coordinates slot blank. If a point has edge owners, we will store a list of pointers to the edges. Points can also be classified using the sign of K and H in the local neighborhood of the point. Some points, such as the tip of a cone, may connect no surface boundary edges but are definitely positive K or negative K. We feel that all of this data can be stored in a fairly compact form and many of the laborious computations can be done off-line before the recognition system is put to use.

We have discussed how matching hypotheses might be verified, but we have not discussed how they will be generated. An N x N image contains $O(N^2)$ degrees of freedom, one for each pixel. The number of non-degenerate critical points in a range image may vary from zero to approximately $N^2/4$, but is typically fairly small (i.e., less than 4N). We could collect all critical points in a list and rank them according to each point's distance to its nearest neighbor such that the maximum nearest neighbor distance comes first in the list and so on. That is, the first point in the sorted list is the critical point around which we

can draw the largest circle which does not include any other critical points. That point is either a positive K point or a negative K point. If it was a zero K point, it would not be an isolated critical point. If it is a negative K point, skip it and get the next point until a positive K point is found. If none are found in the interior of the depth map surface, the depth map exhibits a monotonic behavior that could be utilized in a special case surface matching algorithm for such range scenes. We like to work with positive K neighborhoods because positive Gaussian curvature constrains the surface shape to be convex. This point must lie on a smoothly curved surface, a smoothly curved edge, or at the vertex of a polyhedral-shaped corner. All these conditions can be detected from the sensor data. For each case, we might grow connected regions outward from that point cleaning up noise inconsistencies and/or sharpening edges until we have incorporated a fairly large set of range pixels. Next we take the "clean" surface sign characteristics in the neighborhood of the point and match this against surface regions in the view-independent matching representation. For each match, project the 3-D curvature sign map object back into the compatible surface curvature sign image and check these matches for the best possible match with respect to curvature sign. Then use surface curvature magnitude features to verify the compatibility of proposed surface curvature sign matches. Other surface characteristics might also be used to verify object matches. When an object is found, remove all parts of it from the sensor data and repeat the process. This kind of processing should dramatically decrease the magnitude of the search space for identifying objects. It should handle smoothly curved objects

and polyhedral objects or combinations of both without difficulty. Non-convex objects can be handled by using the convex subsurfaces of the object surface. A great deal of qualitative structural information is handled by utilizing only the sign bits of the surface curvature functions.

9. EXPERIMENTAL RESULTS

A range image processing program has been implemented to perform surface characterization computations. The potentially parallel computational structure of this program is shown in Figure 14. All five derivative images could be computed simultaneously after initial smoothing. Subsequently, all surface characteristics could be computed simultaneously after the derivative estimation stage. This program currently accepts an 8-bit 128x128 depth maps as input

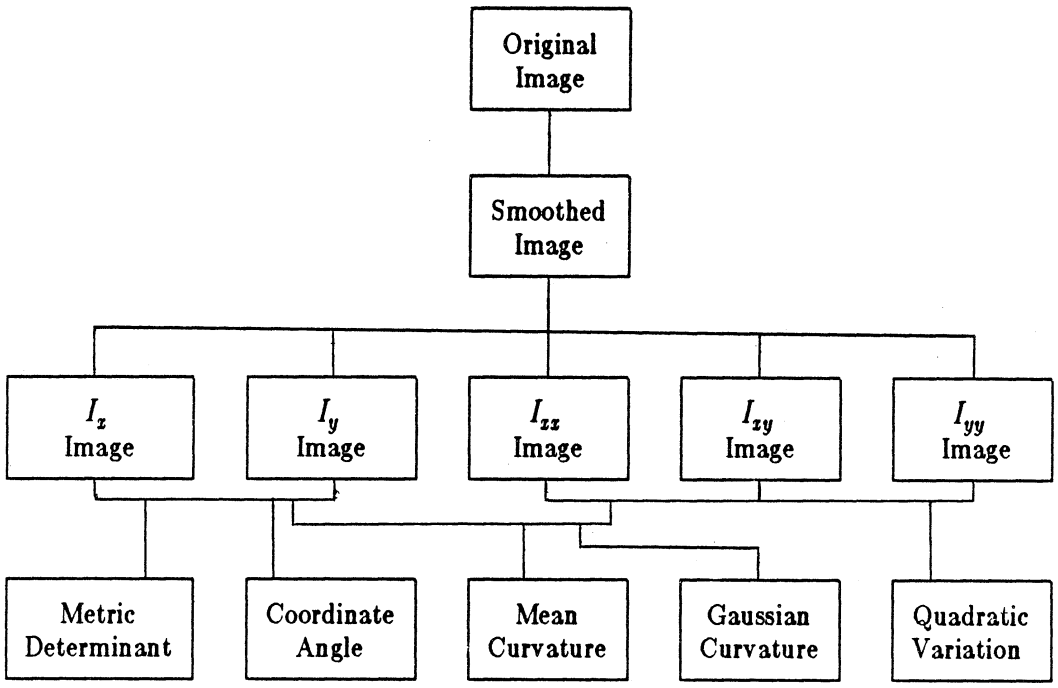

Figure 14. Range Image Surface Characterization Processing Structure

and generates the following images as output:

(1) Smoothed Range Image: f_{smooth}

(2) Square Root of Metric Determinant (Edge Magnitude) Image: $\sqrt{g}$

(3) Quadratic Variation (Flatness Measure) Image: Q

(4) Local Quadratic Surface Fit Error Image: ϵ

(5) Zeros of the Mean Curvature: $H = 0$

(6) Zeros of the Gaussian Curvature: $K = 0$

(7) Zeros of the Cosine of the Coordinate Angle Function: $\cos\Theta = 0$

(8) Cosine of the Coordinate Angle Function: $\cos\Theta$

(9) Sign Regions of Mean Curvature: $sgn(H)$

(10) Sign Regions of Gaussian Curvature: $sgn(K)$

(11) Magnitude of Principal Curvatures Difference: $\sqrt{H^2 - K}$

(12) Principal Direction Angle: Φ_1

(13) Magnitude of Gaussian Curvature: $|K|$

(14) Magnitude of Mean Curvature: $|H|$

(15) Non-degenerate Critical Points Image: $f_u = f_v = 0 \neq Q$

(16) Critical Points Image: $f_u = f_v = 0$

This output data characterizes the input depth map in a way which should be quite useful both for segmentation of sensor data and recognition of objects.

In the experimental results which follow, several computational steps were performed that have not been mentioned yet. The following points should be made:

(1) Our original depth map data is quantized to eight bits. Quantization noise alone caused many problems in our first tests on analytically computed surfaces. To fix this problem, the original image is smoothed using a window operator which is two pixels larger than the window operators used to perform the derivative estimation, and the results are stored in floating point form. This smoothing also compensates for random noise in the two or three least significant bits of range data. (No additional memory for arrays was required in our program because 16-bit integer image arrays had already proved to be too small to handle the results of large window convolutions.)

(2) The output curvature images are smoothed using the same smoothing operator which was used on the input to even out the variations in the curvature output.

(3) The curvature sign images are obtained using a threshold about zero. That is, $sgn(K) = 0$ if $|K| < \epsilon_K |K|_{\max}$. Also, $sgn(H) = 0$ if $|H| < \epsilon_H |H|_{\max}$. The two thresholds were set to 1% for the results shown in the figures. This seems like a reasonable number because the original 8-bit data can only measure zero to within 0.4% of the maximum value.

These assumptions are critical to the results displayed in this paper. Different smoothing schemes create different curvature results. Different thresholds create different curvature sign images. Some of our future research will be devoted to analyzing and experimenting with different smoothing and threshold setting alternatives.

Experimental results for different object depth maps are shown in Figures 16 through 26. Each depth map is briefly discussed individually below. The results are shown in the sixteen (16) subimage format. The contents of each subimage are noted in Figure 15. Zero Images are White when the quantity is zero and black otherwise. Surface Curvature Sign Images are coded as follows: *White for Positive, Gray for Zero, Black for Negative.* Other images are scaled so that the image *Minimum is Black* and the image *Maximum is White.* The exception to this rule is the depth map itself: *white* is used for pixels *closest* to the observer (depth is a minimum) whereas *black* is used for pixels *farthest* from the observer (depth is a maximum). This convention is reversed by some; our experience is that it is generally more difficult to interpret such reversed depth images. Some odd-looking convolutional edge effects have resulted in some of these 16 subimage figures because the objects were made as big as possible

(1) f_{smooth}	(2) $\sqrt{g}$	(3) Q	(4) ϵ
(5) zeros(H)	(6) zeros(K)	(7) zeros(cosΘ)	(8) cosΘ
(9) sgn(H)	(10) sgn(K)	(11) $\sqrt{H^2 - K}$	(12) Φ_1
(13) $\|H\|$	(14) $\|K\|$	(15) $f_x = f_y = 0 \neq Q$	(16) $f_x = f_y = 0$

Figure 15. Surface Characterization Results Format

within the depth map. Please ignore these artifacts. For each depth map, we also provide a surface plot of the depth map.

We obtained the depth maps used in this paper in two different ways. We can generate synthetic depth maps of arbitrary 3-D object models from arbitrary views using a combination of the SDRC/GEOMOD solid modeler [13] developed by Structural Dynamics Research Corporation which is used to create the object models and our own software which uses a depth-buffer algorithm to create the depth maps. We have also obtained real range images from the Environmental Institute of Michigan which were obtained using the ERIM laser rangefinder [49]. The depth map points in these images are obtained using equal-increment azimuth and elevation *angles*. This equal-angle-increment sampling causes flat surfaces in the real world to be mapped into slightly warped surfaces in depth maps.

We now consider the results for each object individually. Our first object is a coffee cup. Two gray scale images of two depth maps of this object are shown in Figure 16 along with a surface plot of one of them. The two depth maps were obtained from the ERIM laser rangefinder. The quality of these depth maps is almost comparable to what we obtain from our model-based synthetic depth map generation program. The surface curvature characterization of these depth maps are shown in Figure 17; a 7x7 derivative window operator was used. We find that the *zeros of the mean curvature* form a very good line drawing of the object shape. The square root of the metric determinant, the quadratic variation, and the quadratic surface fit error provide an interesting

sequence of edge detector-like images. Clusters of local maxima critical points are found at the two rims of the cup whereas local minima critical points are found on the inside of the cup. The magnitude of the principal curvature difference image shows that the principal curvatures on the surface differ the most on the rim of the cup.

The second depth map from the laser rangefinder is shown in Figure 18 along with a surface plot of the same data. A histogram of this image shows that all 8-bit depth values are confined to the 32 to 128 range with most values in the 96 to 128 range. This image represents a portion of a keyboard. Two S-curvature characterizations are shown in Figure 19. The top characterization was computed using a 3x3 derivative window operator and the bottom was computed using a 5x5 window. The reader can easily see the effect of an increase in window size. The concave shape of the top surface of the keyboard keys is detected by the small white regions in the mean curvature sign image. Again the zeros of the mean curvature yield a very good line drawing of the object surface in the image. There are a large number of critical points on this surface as one might expect. The critical points are fairly well clustered into groups for the 5x5 window operator case.

We now discuss two views of a road scene extracted from a range image sequence acquired by the laser rangefinder. Figure 20 shows the original range images (with phase wraparound lines at thirty-two feet and sixty-four feet), the processed range images with the first wraparound transition removed, and a surface plot of one of the processed images. The sixty-four foot line was not

removed because most of the data beyond that level is very noisy. Figure 21 shows the S-curvature characterization results for a 9x9 derivative window operator. The zeros of the mean curvature effectively isolate the ditches at the side of the road. The mean curvature sign image points out that the surface corresponding to the road itself is not flat in this image as expected from the angular sampling. The zeros of the cosine of the coordinate angle occur whenever either of the first partial derivatives is zero. Due to the equal angle increment sampling, the flat road samples are slightly warped yielding a line right up the center of the cosΘ zeros image. Because the noise beyond the sixty-four foot line was left in the image unwrapped, the maximum curvature points and the critical points all occur in this region but have no physical meaning.

Figure 22 is a surface plot of the depth map for a tilted torus. The S-curvature results for two different synthetic range views of the torus are shown in Figure 23. The other view is only tilted about five degrees. Note how well the surface critical points were detected. The structure of the lines in the zeros of the cosine of the coordinate angle function image gives important information concerning the surface structure. The irregularities in the curvature magnitude images are due to the fact the object model from the which the depth map was generated is a faceted polyhedral model. Note that the mean curvature sign image is almost exactly correct. The Gaussian curvature sign image shows that within the 1% threshold many parts of the surface are approximately flat.

A free-form undulating surface was created by "stretching a skin" over a series of curves using SDRC/GEOMOD. The depth map for this surface is

shown in Figure 24. The S-curvature results for two different views using 5x5 window derivative operator are shown in Figure 25. The critical points, which are also maximum Gaussian curvature points tend to line up along the joining curve in the center of the range image when we look straight down on the surface. These points move predictably in the second view. These depth maps have no substantial depth-discontinuities; therefore, detailed slope magnitude variations can be seen in the scaled edge map (square root of g) image. Note the slight changes in S-curvature sign between the two views.

As a final example of the capabilities of the S-curvature characterization algorithm, we show the results when it is applied to a block with three holes in it and a faceted cylinder generated by our software in Figure 26. Note how well "roof" edges show up in the S-curvature magnitude images as opposed to the edge detector-like images. Note also how easily planar surfaces can be identified as regions where $K = 0$ and $H = 0$.

10. FUTURE RESEARCH DIRECTIONS AND GOALS

An object recognition problem has been defined, surface characteristics for depth map segmentation and surface/object matching have been identified, and some preliminary ideas for the matching algorithm have been proposed. Our experimental results indicate the power of our surface characterization approach. We summarize some of the topics to be addressed in future research:

(1) *Noise:* Sensor noise and quantization noise are very important problems in range data processing. Fundmental tradeoff decisions between smoothing

and spatial localization of surface region edges must be addressed.

(2) *Matching:* There are many matching algorithm issues which need to be solved. We need to determine the details of the view-independent matching representation and to be able to compute that representation automatically given object models.

(3) *Feedback:* We mentioned verification feedback earlier. An interaction of data-driven and model-driven processes is required for robust recognition. We plan to analyze the use of feedback within our system.

(4) *Occlusion:* We hope to handle minor occlusion as part of the general purpose matching algorithm.

The applied goal of this research is to develop a system that will correctly recognize a single object from any view selected at random from an object list with ten to twenty objects. These objects should span a range of object complexities and include some objects which look very similar to test the robustness of the system.

11. ACKNOWLEDGEMENTS

We wish to thank the Environmental Institute of Michigan (ERIM) for allowing us to use range images obtained by their sensor. We also wish to thank Structural Dynamics Research Corporation (SDRC) and GE CAE International for allowing us to use the GEOMOD solid modeling software on our engineering workstations.

12. REFERENCES

(1) Altschuler, Martin D., Posdamer, Jeffrey L., Frieder, Gideon, Altschuler, Bruce R., and Taboada, John, "The Numerical Stereo Camera," *SPIE 3-D Machine Perception,* Vol. 283, pgs. 15-24, 1981

(2) Asada, H. and Brady, M., "The Curvature Primal Sketch," *IEEE Proc. Workshop on Computer Vision: Repres. and Control,* pgs. 8-17, May 1984

(3) Beaudet, Paul R., "Rotationally Invariant Image Operators," *Proc. 4th Int'l. Joint Conf. Pattern Recognition,* Kyoto, Japan, pgs. 579-583, November 1978

(4) Besl, Paul J. and Jain, Ramesh C., "An Overview of Three-Dimensional Object Recognition," RSD-TR-19-84, Center for Robotics and Integrated Manufacturing, Univ. of Mich., Ann Arbor, December 1984

(5) Bhanu, Bir, "Representation and Shape Matching of 3-D Objects," *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-6 No. 3, pgs. 340-350, May 1984

(6) Bolle, Ruud M. and Cooper, David B., ""Bayesian Recognition of Local 3-D Shape by Approximating Image Intensity Functions with Quadric Polynomials," *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-6 No. 4, pgs. 418-429, July 1984

(7) Bolles, Robert C.; Horaud, Patrice; Hannah, Marsha Jo, "3DPO: A Three-Dimensional Part Orientation System," *Proc. IJCAI-8,* pgs. 1116-1120, 1983

(8) Chakravarty, Indranil and Freeman, Herbert, ""Characteristic Views as a basis for three-dimensional object recognition", *SPIE Robot Vision,* Vol. 336, pgs. 37-45, May 1982

(9) Chern, S.S., "A Proof of the Uniqueness of Minkowski's Problem for Convex Surfaces," *Am. J. Math.,* Vol. 79, pgs. 949-950, 1957

(10) Coleman, E. North and Jain, Ramesh, "Obtaining Shape of Textured and Specular Surface using Four-Source Photometry," *Computer Graphics and Image Processing,* Vol. 18, No. 4, pgs. 309-328, 1982

(11) Faugeras, O.D., "New Steps Toward a Flexible 3-D Vision System for Robotics," *IJCPR,* pgs. 796-805, 1984

(12) Faugeras, O.D. and Hebert, M., "A 3-D Recognition and Positioning Algorithm using Geometrical Matching between Primitive Surfaces," *Proc. IJCAI-7,* pgs. 996-1002, 1983

(13) *GEOMOD 2.5 User Manual and Reference Manual,* Structural Dynamics Research Corporation, Milford, Ohio, 1984

(14) Grogan, Timothy A. and Mitchell, O. Robert, "Partial Shape Recognition using Fourier-Mellin Transform Methods," *Optical Society of America Winter '83 Topical Meeting on Signal Recovery and Synthesis with*

Incomplete Information and Partial Constraints, pgs. ThA19-1 - ThA19-4, January 1983

(15) Haralick, Robert M., "Digital Step Edges from Zero-Crossings of Second Directional Derivatives," *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-6 No. 1, pgs. 58-68, January 1984

(16) Haralick, R.M., Laffey, T.J., Watson, L.T., "The Topographic Primal Sketch," *The International Journal of Robotics Research,* Spring 1983

(17) Horn, Berthold K.P. and Ikeuchi, Katsushi, "The Mechanical Manipulation of Randomly Oriented Parts," *Scientific American,* pgs. 100-111, July 1984

(18) Horn, Berthold K.P., "Extended Gaussian Images," AI Memo 740, MIT AI Lab, July 1983

(19) Hsiung, Chuan-Chih, *A First Course in Differential Geometry,* New York: Wiley-Interscience, 1981

(20) Ikeuchi, Katsushi and Horn, Berthold K.P., "Numerical Shape from Shading and Occluding Boundaries," *Artificial Intelligence,* Vol. 17, pgs. 141-184, August 1981

(21) Ikeuchi, K., Horn, B.K.P., Nagata, S., Callahan, T., Feimgold, O., "Picking up an object from a pile of objects," AI Memo 726, MIT AI Lab, May 1983

(22) Ikeuchi, Katsushi, "Recognition of 3-D Objects using the Extended Gaussian Image," *Proc. IJCAI-7,* pgs. 595-600, 1981

(23) Inokuchi, Seiji; Sato, Kosuke; Matsuda, Fumio; "Range Imaging System for 3-D Object Recognition," *IEEE Int'l Conf. on Pattern Recognition,* pgs. 806-808, August 1984

(24) Jarvis, R.A., "A Perspective on Range Finding Techniques for Computer Vision", *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-5 No. 2, March 1983, pgs. 122-139

(25) Jarvis, R.A., "A Laser Time-of-Flight Range Scanner for Robotic Vision", *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-5 No. 5, September 1983, pgs. 505-512

(26) Koenderink, J. J. and Van Doorn, A.J., "Internal Representation of Solid Shape with Respect to Vision," *Biological Cybernetics,* Vol. 32, pgs. 211-216, 1979

(27) Koenderink, J. J. and Van Doorn, A.J., "The Singularities of the Visual Mapping," *Biological Cybernetics,* Vol. 24, pgs. 51-59, 1976

(28) Kim, H.S., Jain, R., Volz, R.A., "Multiple View Object Recognition," RSD Technical Report, Center for Robotics and Integrated Manufacturing, Univ. of Mich., Ann Arbor

(29) Kuan, Darwin T. and Drazovich, Robert J., "Model-Based Interpretation of

Range Imagery," *AAAI-84,* pgs. 210-215, 1984

(30) Laffey, Thomas J., Haralick, Robert M., Watson, Layne T., "Topographic Classification of Digital Image Intensity Surfaces," *IEEE Proc. Workshop on Computer Vision: Repres. and Control,* pgs. 171-177, August 1982

(31) Langridge, D.J., "Detection of Discontinuities in the First Derivatives of Surfaces," *Computer Vision, Graphics, and Image Processing,* Vol. 27, pgs. 291-308, October 1984

(32) Lewis, R.A. and Johnston, A.R., "A Scanning Laser Rangefinder for a Robotic Vehicle," *Proc. IJCAI-5,* pgs. 762-768, 1977

(33) Lin, C. and Perry, M.J., "Shape Description using Surface Triangularization," *IEEE Proc. Workshop on Computer Vision: Repres. and Control,* pgs. 38-43, August 1982

(34) Lipschutz, Martin M., *Differential Geometry,* New York: McGraw Hill, 1969

(35) Little, James J., "An Iterative Method for Reconstructing Convex Polyhedra from Extended Gaussian Images," *Proc. AAAI-83,* pgs. 247-250, 1983

(36) Marimont, David, H., "A Representation for Image Curves," *AAAI-84,* pgs. 237-242, 1984

(37) Medioni, G. and Nevatia, R., "Description of 3-D Surfaces Using Curvature Properties," *Proc. Image Understanding Workshop,* pgs. 291-299, October 1984

(38) Minkowski, H., "Allgemeine Lehrsatze uber die konvexen Polyeder," Nachrichten von der Koniglichen Gesellschaft der Wissenschaften, Mathematisch-Physikalische Klasse, Gottingen, pgs. 198-219, 1897

(39) Nackman, Leo R., "Two-dimensional Critical Point Configuration Graphs," *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-6 No. 4, pgs. 442-449, July 1984

(40) Nevatia, Ramakant and Binford, Thomas O., "Description and Recognition of Curved Objects," *Artificial Intelligence,* Vol. 8, pgs. 77-98, 1977

(41) O'Neill, Barrett, *Elementary Differential Geometry,* New York: Academic Press, 1966

(42) Oshima, Masaki and Shirai, Yoshiaki, "Object Recognition Using Three-Dimensional Information," *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-5 No. 4, pgs. 353-361, July 1983

(43) Parthasarathy, S. & Birk, J. & Dessimoz, J., "Laser Rangefinder for robot control and inspection", *Proc. SPIE,* Vol. 336, Robot Vision, pgs. 2-11, May 1982

(44) Potmesil, Michael, "Generating Models of Solid Objects by Matching 3D

Surface Segments," *Proc. IJCAI-8,* pgs. 1089-1093, 1983

(45) Sato, Y., Kitagawa, H., Fujita, H., "Shape Measurement of Curved Objects using Multiple Slit Ray Projections," *IEEE Trans. Pattern Anal. & Machine Intell.,* Vol. PAMI-4, pgs. 641-646, November 1982

(46) Sethi, I.K. and Jayaramamurthy, S.N., "Surface Classification using Characteristic Contours," *Proc. IJCPR,* pgs. 438-440, August 1984

(47) Smith, David R. and Kanade, Takeo, "Autonomous Scene Description with Range Imagery," *Proc. Image Understanding Workshop,* pgs. 282-290, October 1984

(48) Sugihara, Kokichi, "Range-Data Analysis Guided by Junction Dictionary," *Artificial Intelligence,* Vol. 12, pgs. 41-69, 1979

(49) Svetkoff, Donald J., Leonard, Patrick F., Sampson, Robert E., and Jain, Ramesh, "Techniques for Real-Time 3D Feature Extraction Using Range Information," *SPIE Robot Vision,* November 1984

(50) Tio, J.B.K., McPherson, C.A., Hall, E.L., "Curved Surface Measurement for Robot Vision," *Proc. PRIP,* pgs. 370-378, 1982

(51) Turney, Jerry L., Mudge, Trevor N., Volz, Richard A., "Recognizing Partially Occluded Parts," RSD-TR-20-83, Center for Robotics and Integrated Manufacturing, Univ. of Mich., Ann Arbor, December 1983

(52) Witkin, Andrew P., "Recovering Surface Shape and Orientation from Texture," *Artificial Intelligence,* Vol. 17, pgs. 17-45, August 1981

(53) Woodham, Robert J., "Analysing Images of Curved Surfaces," *Artificial Intelligence,* Vol. 17, pgs. 117-140, August 1981

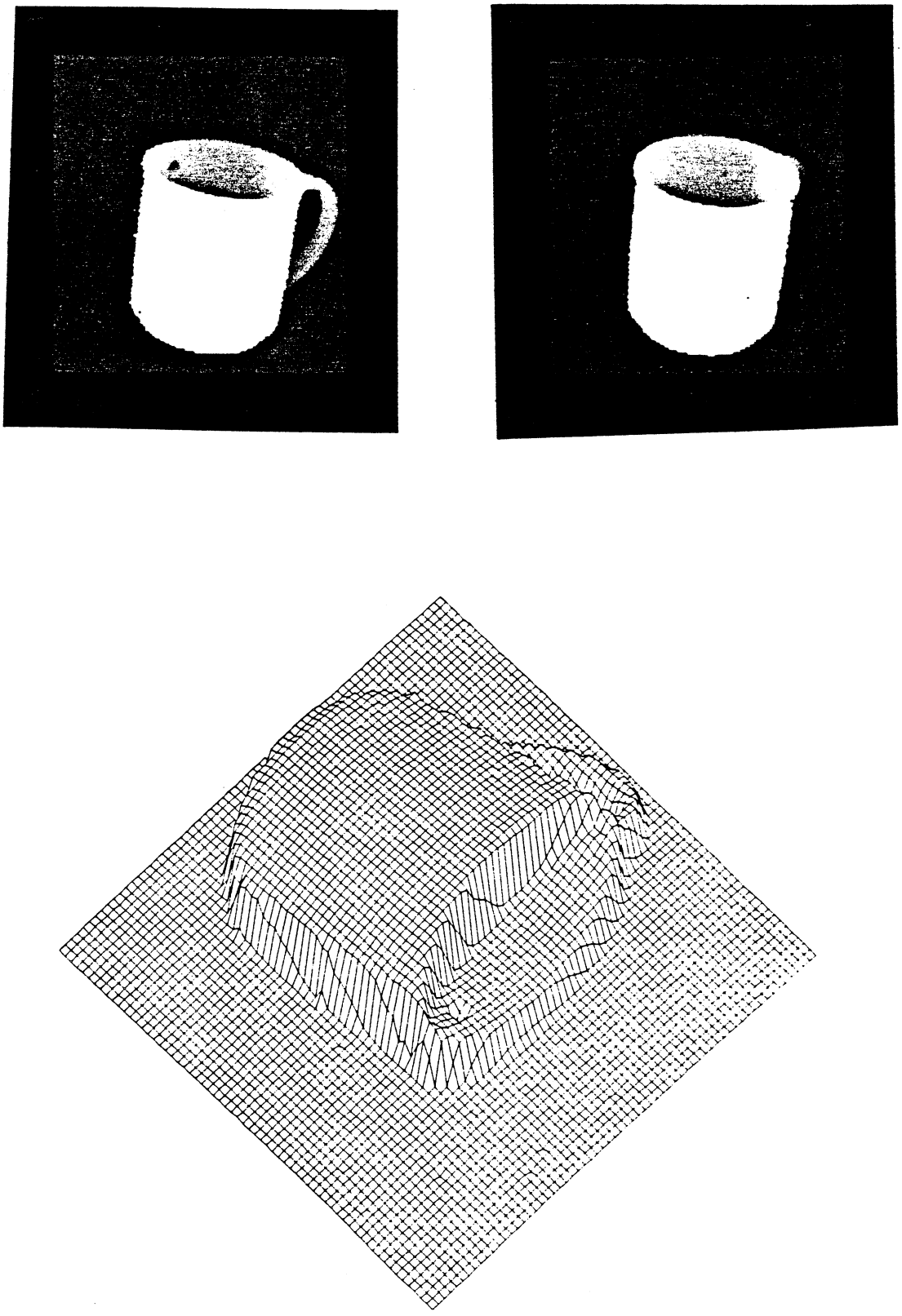

Figure 16. Coffee Cup Range Images and Surface Plot

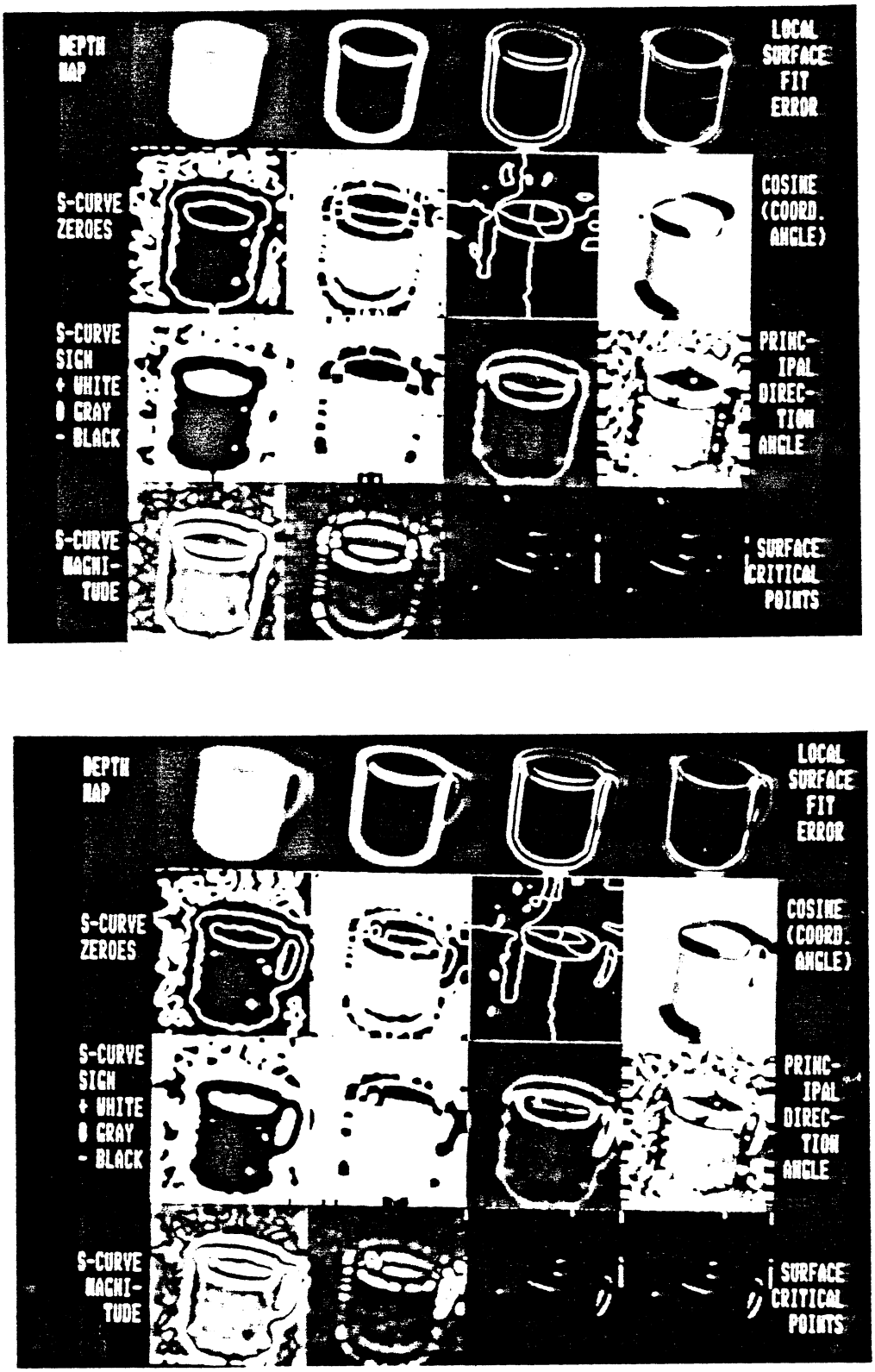

Figure 17. S-Curvature Characterizations of Two Views of Coffee Cup

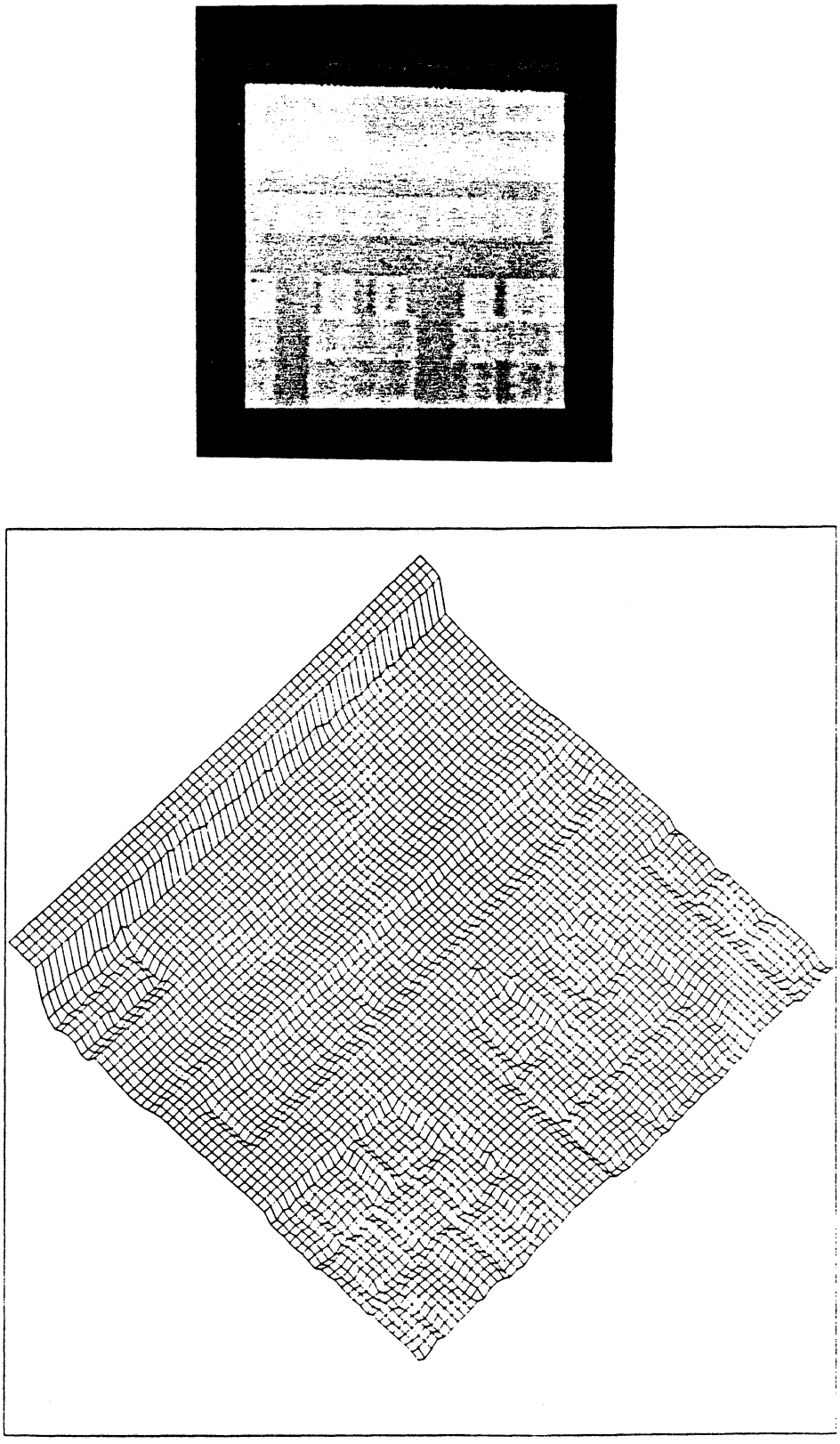

Figure 18. Keyboard Range Image and Surface Plot

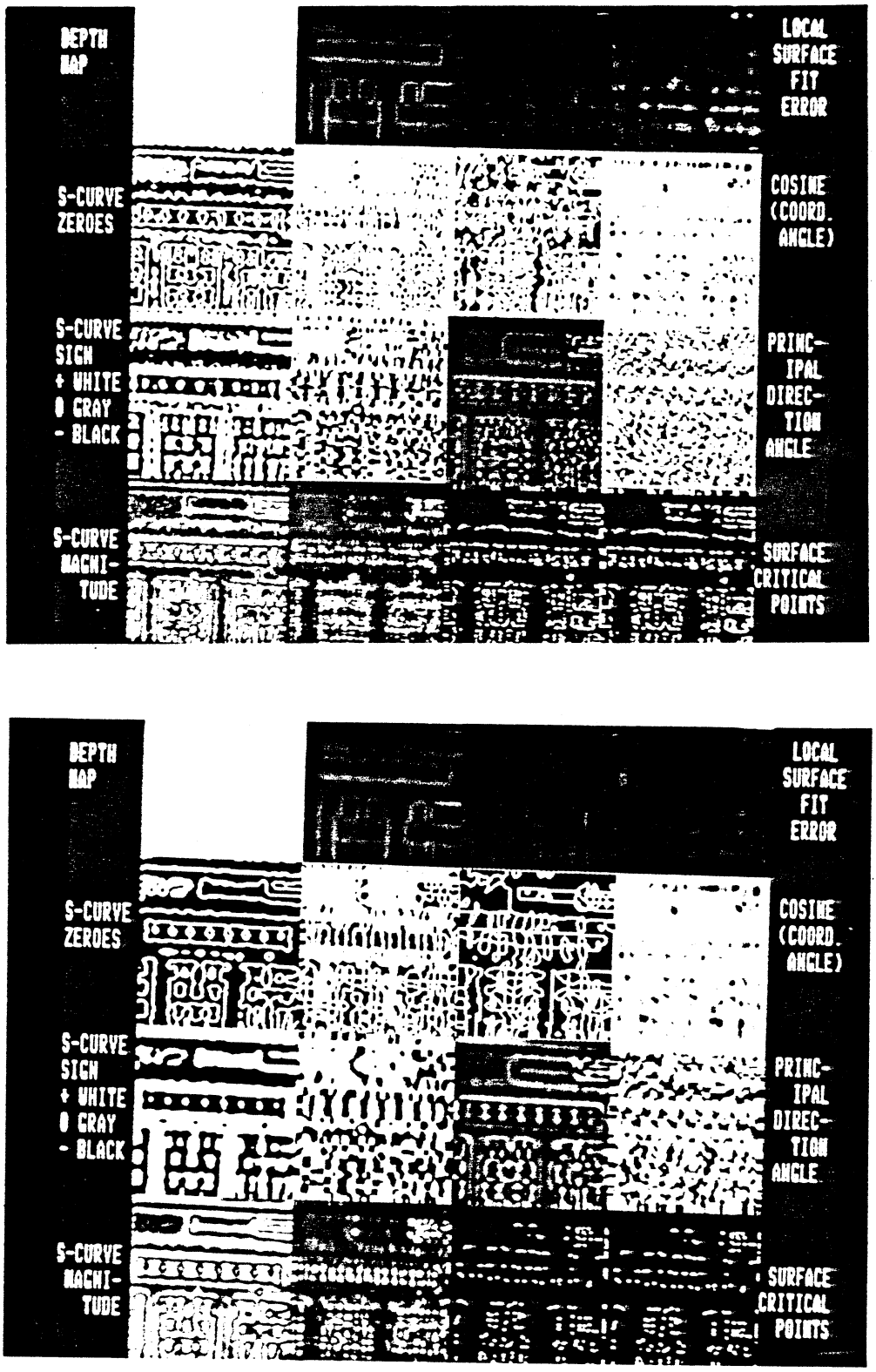

Figure 19. S-Curvature Characterizations of Keyboard
(3x3 window operator results - Top, 5x5 window results - Bottom)

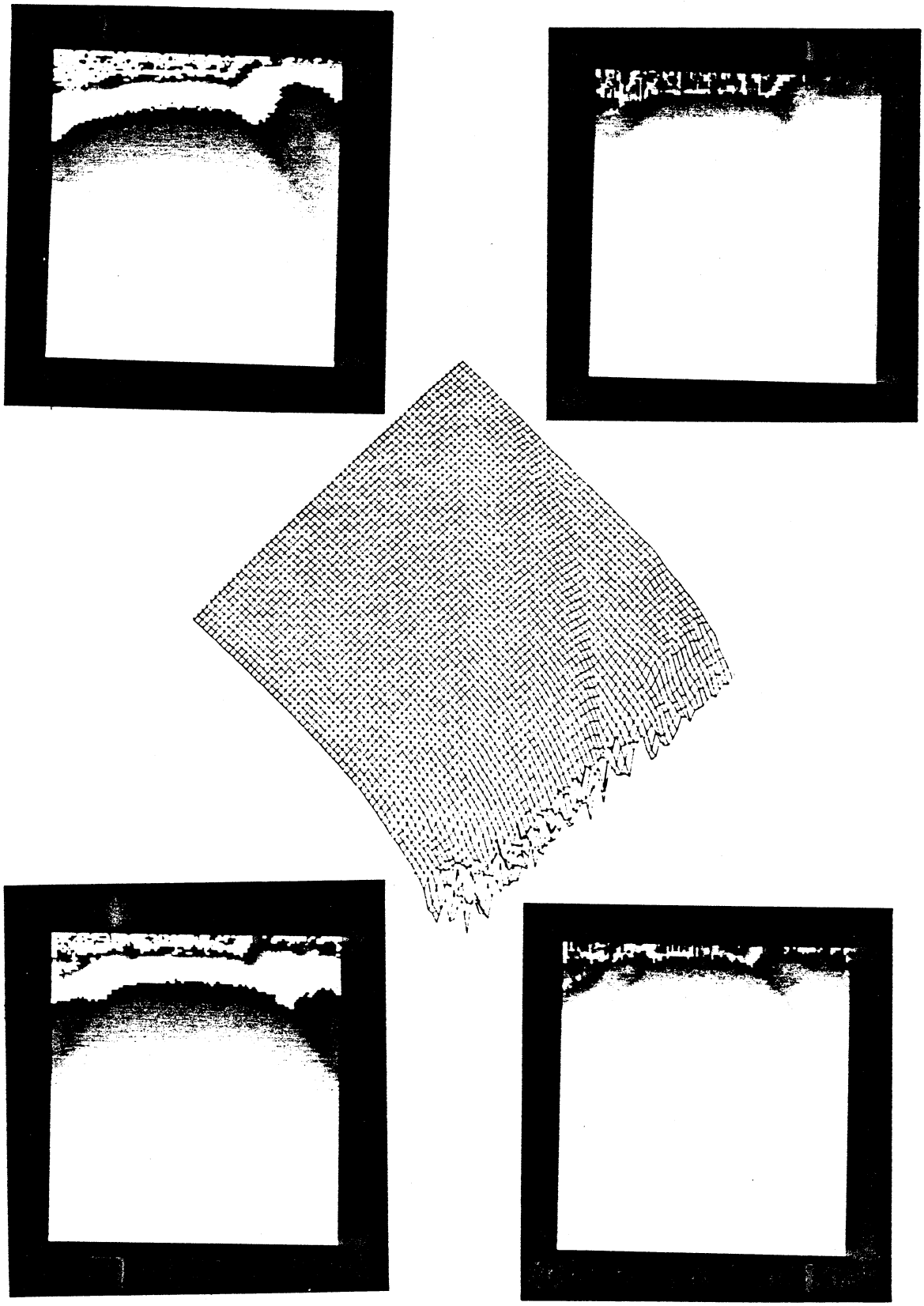

Figure 20. Original and Unwrapped Range Images of Road Scene

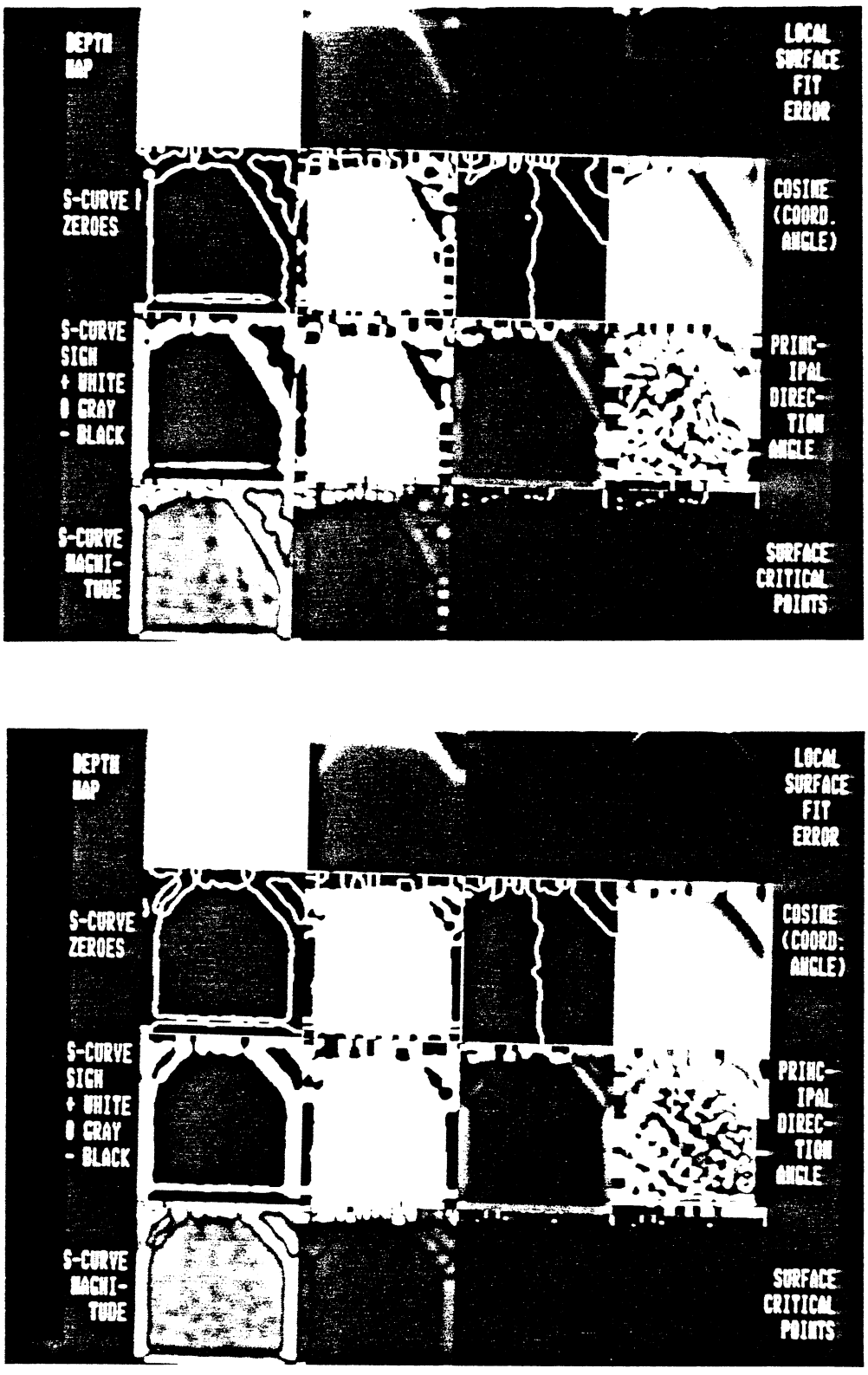

Figure 21. S-Curvature Characterizations of Road Scenes

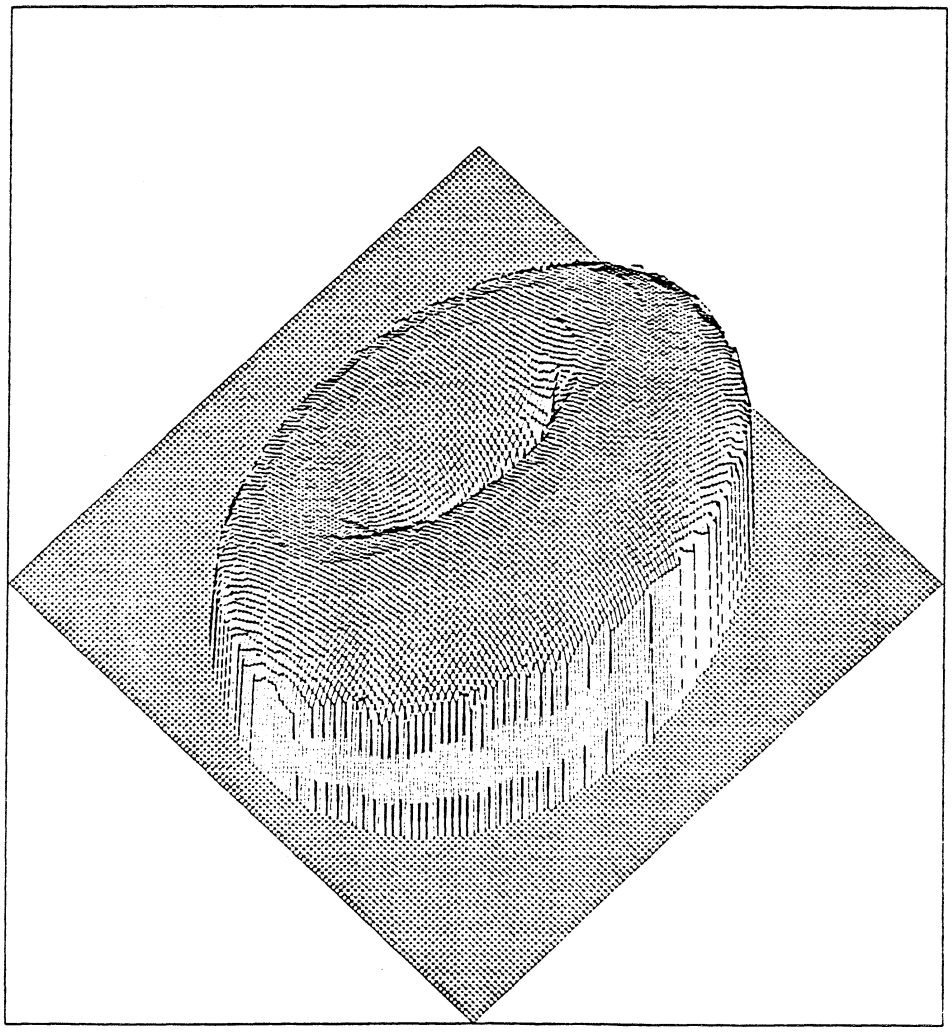

Figure 22. Surface Plot of Torus Depth Map

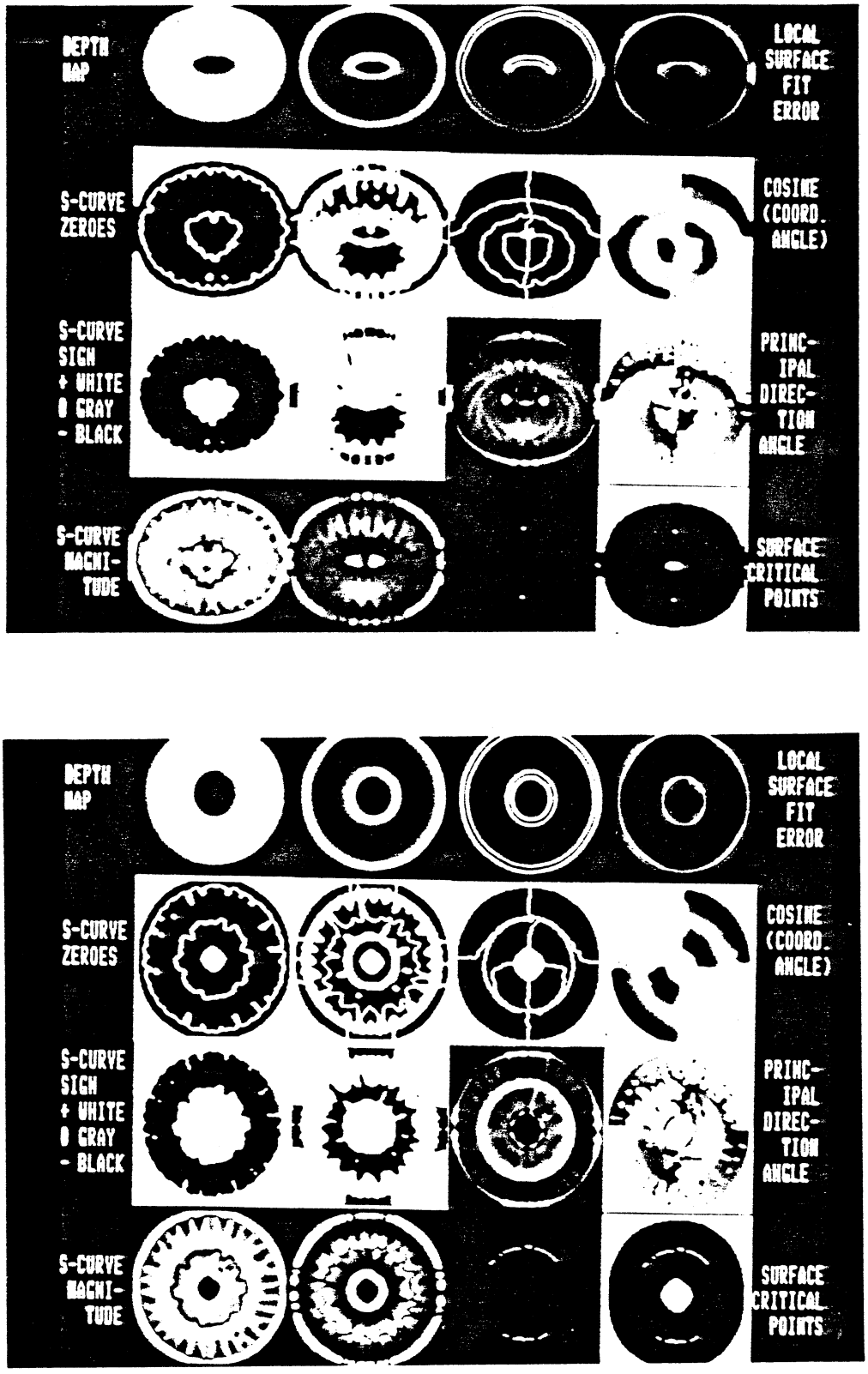

Figure 23. S-Curvature Characterizations of Two Views of Torus

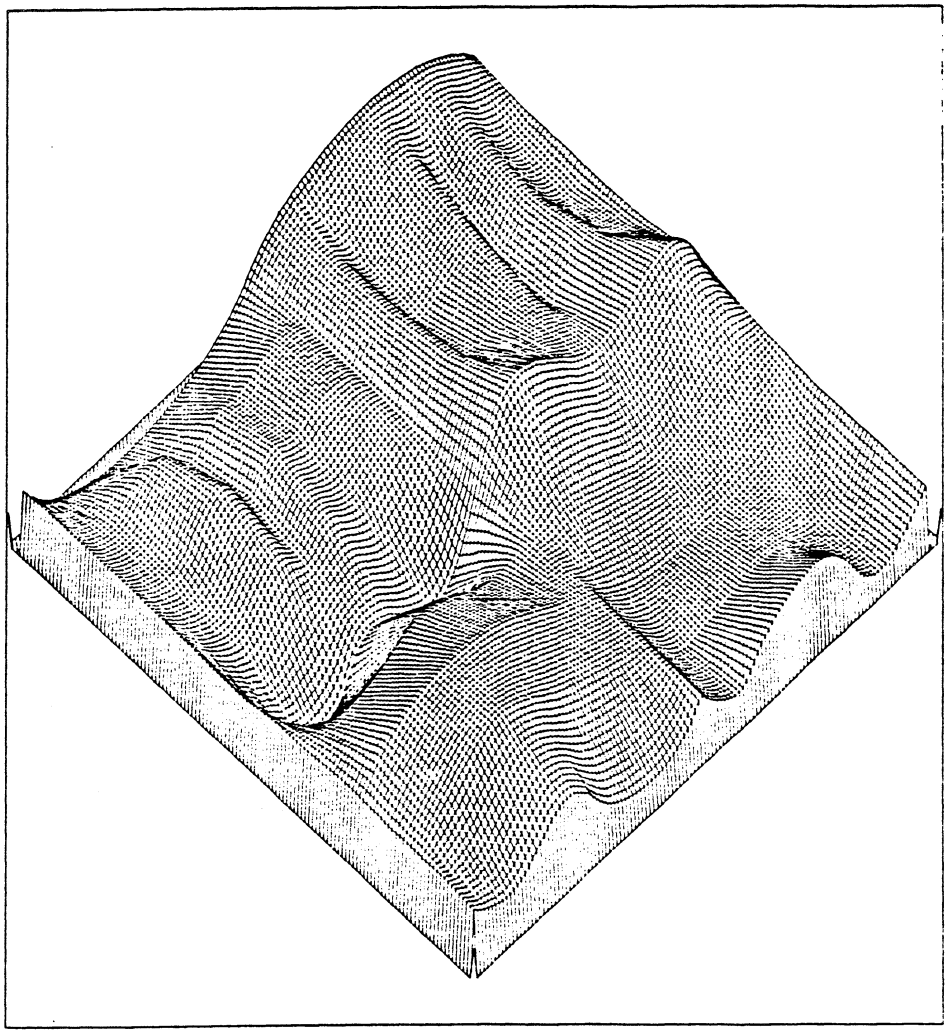

Figure 24. Surface Plot of Undulating Surface Depth Map

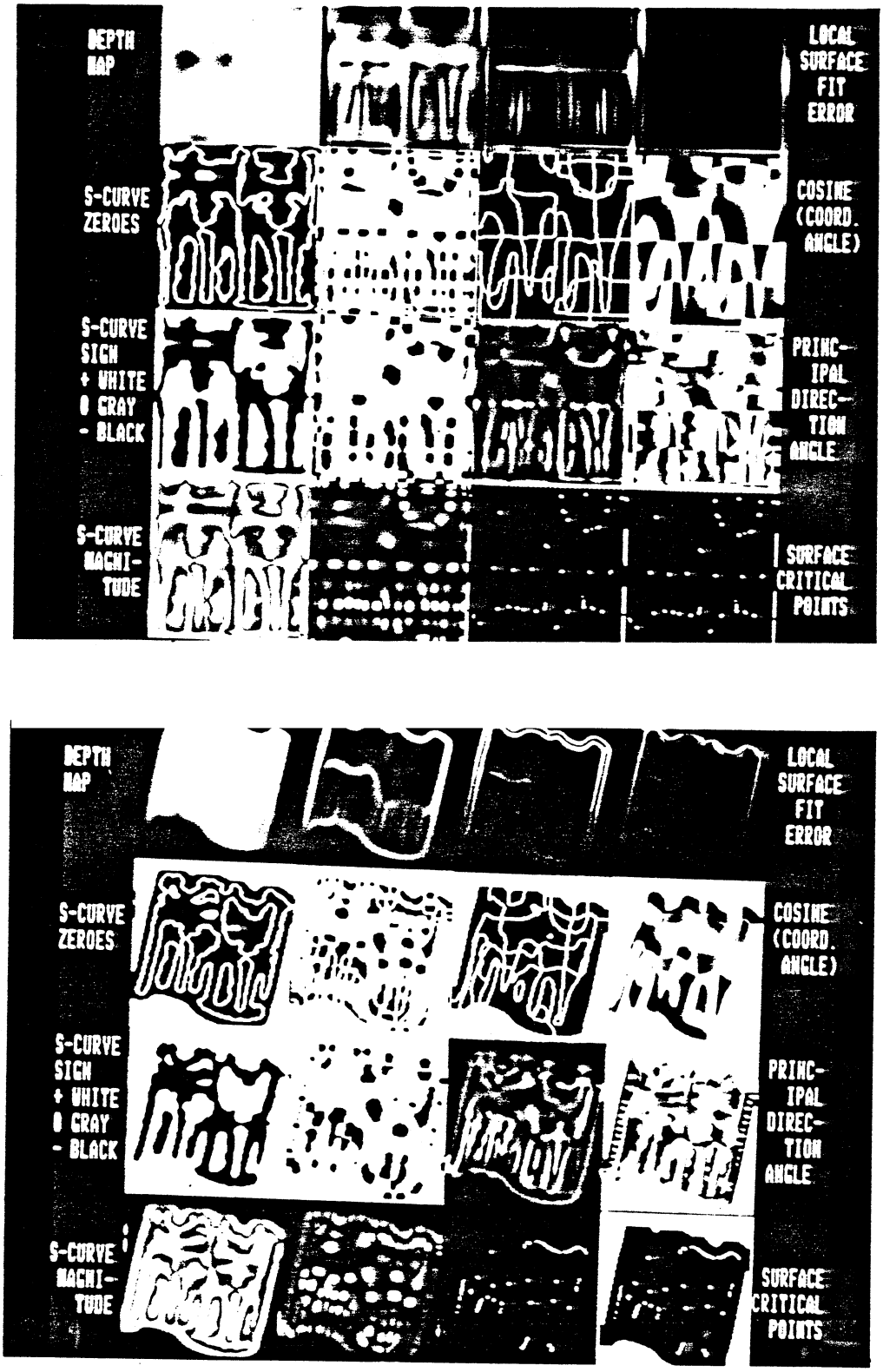

Figure 25. S-Curvature Characterizations of Two Views of Undulating Surfac

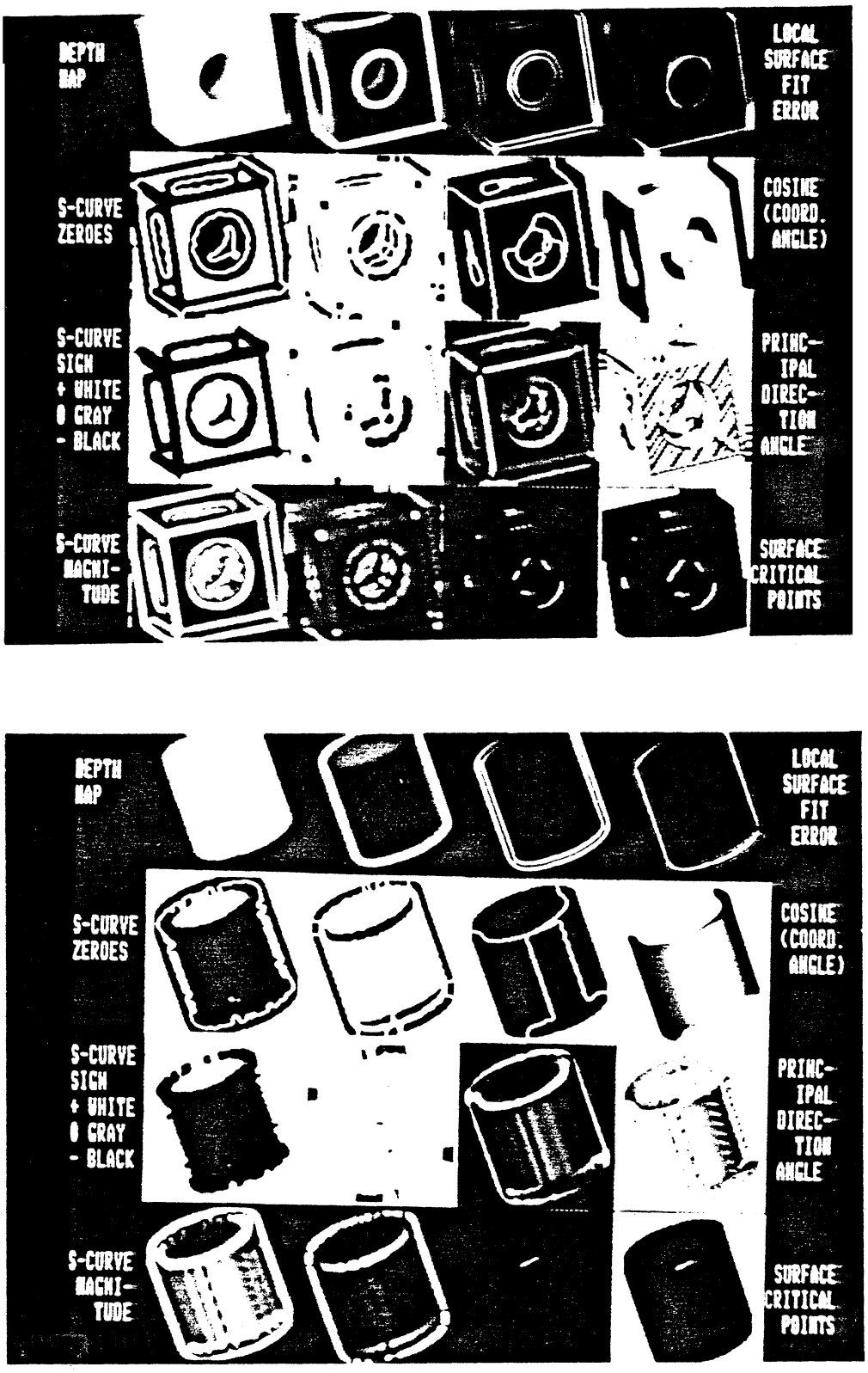

Figure 26. S-Curvature Characterizations of Block and Cylinder