SURFACES
IN
EARLY RANGE IMAGE UNDERSTANDING

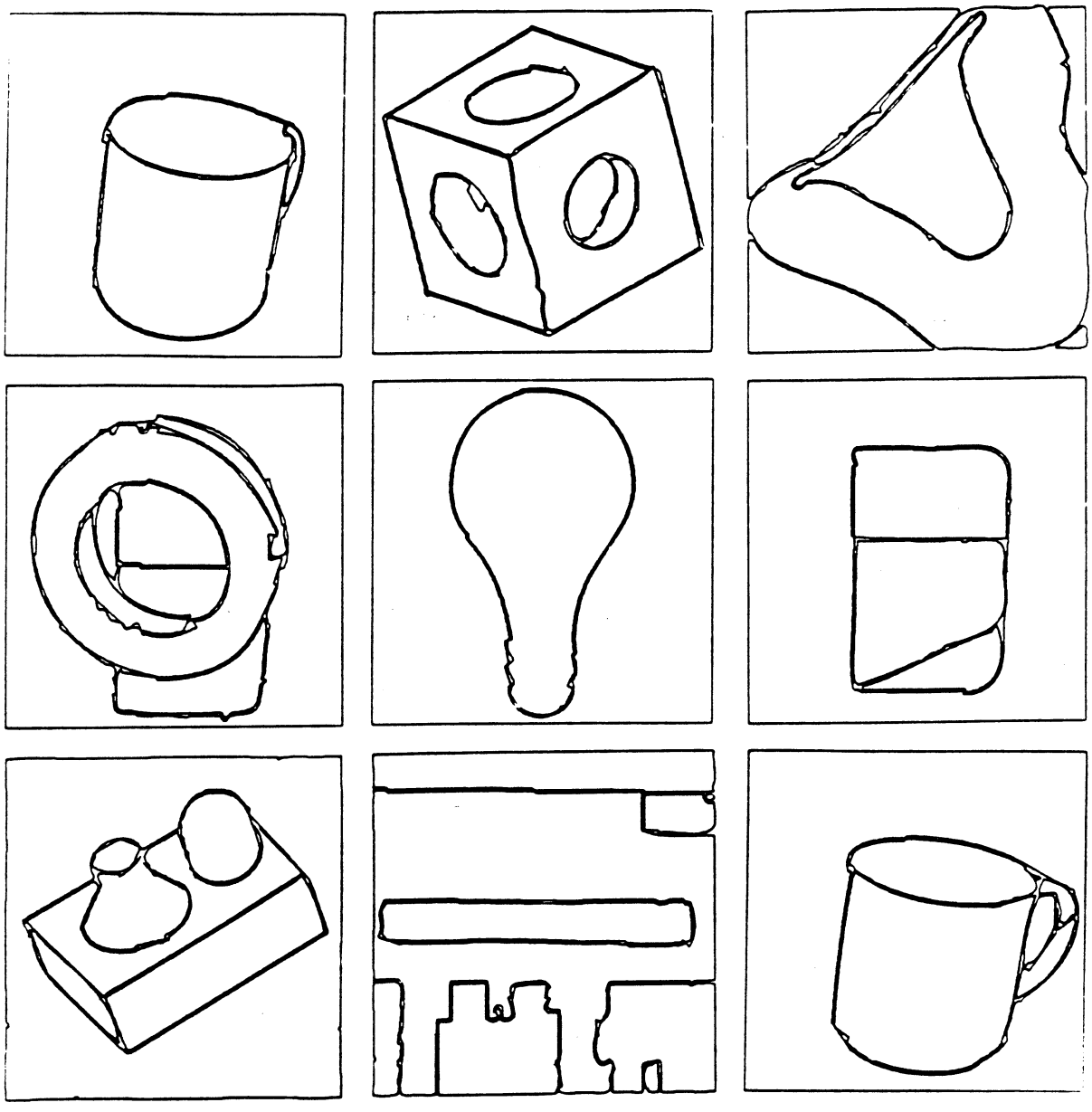
by

Paul Joseph Besl

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer, Information, and Control Engineering)
in The University of Michigan
1986

Doctoral Committee:

Professor Ramesh Jain, Chairman
Adjunct Professor Robert Haralick
Assistant Professor Laurence Maloney
Associate Professor Trevor Mudge
Assistant Professor Terry Weymouth



ABSTRACT

SURFACES IN EARLY RANGE IMAGE UNDERSTANDING

by
Paul Joseph Besl

Chairman: Ramesh C. Jain

Perception of surfaces plays a fundamental role in three-dimensional object recognition and image understanding. A range image explicitly represents the surfaces of objects in a given field of view as an array of depth values. Previous research in range image understanding has limited itself to extensions of edge-based intensity image analysis or to interpretations in terms of polyhedra, generalized cylinders, quadric primitives, or convex objects. Computer vision research has demonstrated the advantages of data-driven early processing of image data. If early processing algorithms are not committed to interpretation in terms of restrictive, domain-specific, high-level models, the same algorithms may be incorporated in different applications without substantial effort.

A general approach has been developed for processing range images to obtain a high-quality, rich (information-preserving), accurate, intermediate-level description consisting of graph surface primitives, the associated segmented regions, and their bounding edges. Only general knowledge about surfaces is used to compute a complete image segmentation; no object level information is involved. The early range image understanding algorithm consists primarily of a differential-geometric, visible-invariant pixel labeling method based on the sign of mean and Gaussian curvatures and an iterative region-growing method based on variable-order surface-fitting of the original image data. The high-level control logic of the current implementation is sequential, but all low-level

image processes can be executed on parallel architectures. This surface-based image analysis approach has successfully segmented a wide variety of real and synthetic range images and is also shown to have significant potential for intensity image analysis. It is interesting to note that the surface and edge description algorithms use the same basic "sign-of-curvature" paradigm in different dimensions.

ACKNOWLEDGEMENTS

The author thanks Professor Ramesh Jain for his extraordinary enthusiasm, optimism, ideas, insight, encouragement, and guidance during the course of this research. The efforts and contributions of the members of the doctoral committee are also gratefully acknowledged. Many conversations with others have helped to refine the ideas in this thesis. The author would particularly like to thank Layne Watson, Larry Maloney, Paul Eichel, Mario Micallef, and Edward Delp for their helpful comments on various aspects of this work. My appreciation is also expressed to Doug Thomas, Kent Gilbert, David Chen, Thawach Sriprad-isvarakul, and Joonhee Han. And I especially thank my wife, Betsy, for her continual understanding, patience, and support.

Several organizations have also helped to make this thesis possible. The author acknowledges the external support of the International Business Machines Corporation, the Structural Dynamics Research Corporation, the DeVlieg Machine Tool Company, the Environmental Research Institute of Michigan, and the Air Force Office of Scientific Research, and the internal support of the Robot Systems Division of the Center of Robotics and Integrated Manufacturing.

PREFACE

Perception of surfaces plays a fundamental role in three-dimensional object recognition and image understanding. A range image explicitly represents the surfaces of objects in a given field of view as an array of depth values. Previous research in range image understanding has limited itself to extensions of edge-based intensity image analysis or to interpretations in terms of polyhedra, generalized cylinders, quadric primitives, or convex objects. Computer vision research has demonstrated the advantages of data-driven early processing of image data. If early processing algorithms are not committed to interpretation in terms of restrictive, domain-specific, high-level models, the same algorithms may be incorporated in different applications without substantial effort.

A general approach has been developed for processing range images to obtain a high-quality, rich (information-preserving), accurate, intermediate-level description consisting of graph surface primitives, the associated segmented regions, and their bounding edges. Only general knowledge about surfaces is used to compute a complete image segmentation; no object level information is involved. The early range image understanding algorithm consists primarily of a differential-geometric, visible-invariant pixel labeling method based on the sign of mean and Gaussian curvatures and an iterative region-growing method based on variable-order surface-fitting of the original image data. The high-level control logic of the current implementation is sequential, but all low-level image processes can be executed on parallel architectures. This surface-based image analysis approach has successfully segmented a wide variety of real and synthetic range images and is also shown to have significant potential for intensity image analysis. It is interesting to note that the surface and edge description algorithms use the same basic "sign-of-curvature" paradigm in different dimensions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1. Perception, Surfaces, and Range Images

Machine perception requires the digitization of physically sensed signals, such as sound, light, depth, or pressure signals, before the information in such signals can be automatically interpreted. Researchers have attempted with limited success to use digitized sensor data to endow digital computers with the abilities to hear, see, and touch so that machines can intelligently interact with an uncertain, dynamic environment. Whereas automated speech recognition techniques attempt to understand one-dimensional (1-D) signals in terms of a language structure, computational vision methods attempt to use two-dimensional (2-D) sensor data to determine the spatial, geometric structure of corresponding three-dimensional (3-D) scenes. Each data point in 2-D sensor data is discrete in two spatial directions and in the level of the sensed quantity. This type of signal data is normally considered as a large rectangular array of numerical entries. Each array element, known as a *pixel*, has a different location and a value representing the level of the relevant physical quantity. It is useful to view a large array of this type as a *digital surface* because the sensed values at each pixel can be considered as noisy samples of an "analog surface" $z = g(x,y)$. Digital surfaces are more commonly referred to as *digital images*. Figure 1.1 shows a 20 x 20 array of depth values from an actual digital image and a corresponding digital surface view of the same values. The

usefulness and the final interpretation of such signals depends, among other things, on how the signals are processed. The type of processing depends on how the signal entity itself is viewed by the data analyst. Using Figure 1.1 as an example, a signal might be treated primarily as a statistical entity with a mean and standard deviation or as a geometric entity with surface shape.

For over twenty years, digital (light) intensity images (also called brightness, luminance, or irradiance images) have been studied by image processing and computational vision investigators and have received significantly more attention than any other type of digital surface for machine perception purposes. Nonetheless, many other types of digital images, such as tactile, radar, infrared, scanning electron microscope, eddy current, ultrasound, and X-ray have also been analyzed with the goal of *automated interpretation*. During the last ten years, *digital range images* (also known as *depth maps*) have become available from a variety of active and passive range sensors. The image quality and the speed of image acquisition for range sensors has been steadily improving to the point that high-resolution range images may be available at video frame rates in the near future. From a mathematical viewpoint, range imagery is *unique*

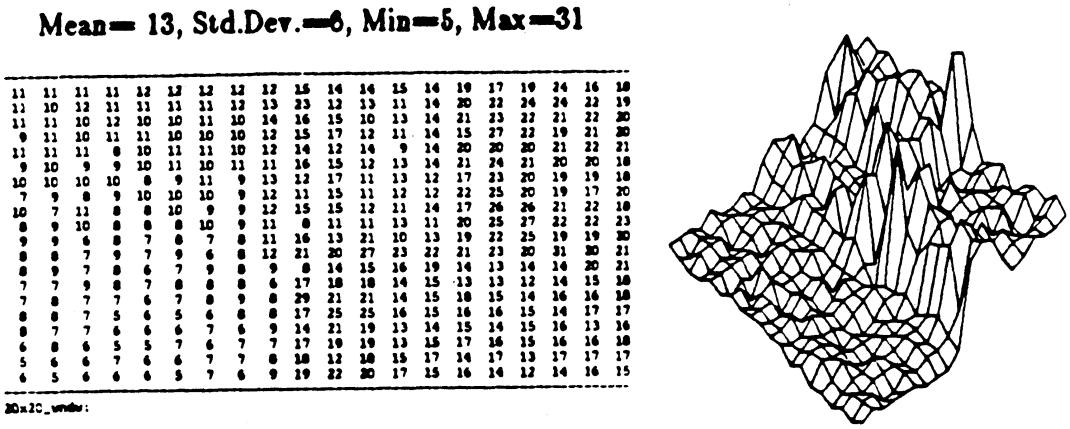


Figure 1.1. Sensor Data Viewed as Pixel Array and Digital Surface

among all other forms of 2-D sensor data for scene analysis in that this type of digital surface directly approximates the physical (analog) surfaces of a real-world 3-D scene. Quantitative scene surface geometry can be *directly* extracted from high-quality, high-resolution range images whereas, in other types of images, the surface structure of a scene must be inferred, or *indirectly* computationally extracted, from the sensor data regardless of the resolution or quality.

People are remarkably proficient at the qualitative visual inference and perceptual organization processes required for image interpretation regardless of image type and quality. The human visual system is able to discover continuity, smoothness, regularity, and spatial coherence in noisy image data and to organize (or group) pixels into important regions even when *a priori* knowledge about the meaning of the image is lacking. In contrast, it has been extremely difficult to program computers to perform similar visual tasks in a general purpose manner. When a machine vision system is successfully applied to a particular problem, one is fortunate if that same system can be applied to a significantly different problem without a major development effort. Why such a discrepancy in perceptual capability and flexibility when computers can easily outperform people in many other tasks that people find difficult or tedious? Part of the answer lies in the differences between the amount and the variety of *knowledge* that can be stored and accessed by each. Human beings can flexibly interpret an extremely wide variety of images using poorly understood, high-level visual mechanisms and a vast storehouse of quickly accessible knowledge. Current computer vision systems are limited to particular application domains where relevant knowledge can be organized in a form compatible with existing computer software and hardware. High-level mechanisms are still quite primitive in comparison to human capabilities and tend to get slower instead of faster as the amount of flexible knowledge is increased.

There are numerous examples of "high-level" knowledge at work in visual perception. Consider a simple image in which a shadow falls across a visible smooth surface. A person still perceives the smooth surface as a single entity. If a computer vision system is given this image as input, it would probably find a dividing line that separates the brighter part of the smooth surface from the darker part in the shadow. In order to correctly *segment* the image near the shadow boundary into image regions that correspond to physical scene surfaces, high-level knowledge is clearly needed to correctly infer scene surface structure from the digital (intensity) surface structure. High-resolution range images are extremely interesting and potentially very useful in that higher-level domain-specific knowledge is not required to segment a scene into its component physical surfaces because the component surfaces of the scene are explicitly represented by the *collection* of depth values. As this thesis will demonstrate, only generic information about surfaces is required to organize the depth values in range images into spatially coherent surface primitives. It is interesting to note that speech signals and tactile images are more similar to range images than intensity images because they contain explicit information about audio frequencies and skin pressure, quantities that are directly useful to interpretation tasks. In contrast, brightness and color contain surprisingly little explicit information about the 3-D world that we perceive. Of course, higher level knowledge is still required for any signal type in order to establish with certainty the meaning of the various parts of the signal by relating them to known structures in a domain model. For example, object level knowledge is required to decide which surfaces in a range image belong to which objects.

In the field of computational vision (where digital surfaces have been analyzed the longest), the 2.5-D sketch and the intrinsic-image vision paradigms proposed by Marr [1976,1982] and Barrow and Tennebaum [1978,1981] respectively seek a range image as

the primary intermediate output from early visual processes operating on an intensity image. In these paradigms dictating the overall organization of a vision system, the range image along with the intensity image and other computed quantities provide input to higher level, symbolic, goal-driven, domain-specific modules that perform segmentation and, ultimately, interpretation, object recognition, and image understanding. However, as vision research has continued, some investigators have begun to doubt the importance of range images in visual perception perhaps because the widely-accepted paradigms do not state exactly how range images should be used. For example, Witkin and Tennebaum [1983] have made the following statements:

> Even if a depth map could be reliably obtained, how would it be used? [pg. 493]

> Being just an array of numbers, it is difficult to think of tasks that a depth map directly supports. [pg. 493]

> ... recognizing objects requires that the depth data first be organized into larger structures corresponding, for example, to continuous visible surfaces. Although perhaps better defined, this process is a great deal like that of segmenting a brightness image. [pg. 493]

> However, all the fundamental difficulties due to noise and low contrast edges remain ... Range contributes incrementally, in much the same way as color; it adds a dimension that simplifies some decisions, but certainly does not make segmentation trivial. [pp. 493-495]

> Although depth maps can be helpful, it is not obvious that they are helpful enough to deserve their present stature as a central goal of vision. [pg. 495]

Clearly, range images in their full dimensionality cannot directly support high-level image interpretation processes without some type of perceptual organization, such as segmentation. And it certainly is true that range image segmentation is not trivial; many intensity-image segmentation difficulties are also encountered in range images. However, it would be premature to dismiss range image analysis as an unfruitful endeavor merely because depth maps are currently not helpful enough to image

understanding efforts. To the contrary, range provides a basic, fundamental (not incremental) contribution toward the goal of understanding 3-D shape, which is required for general-purpose object recognition and image understanding. Moreover, it appears that range image understanding research can provide a *fundamental understanding of the pixel grouping operations* that will assist intensity image understanding efforts. Furthermore, many common, high-level object recognition issues must be confronted by image understanding research regardless of whether range or intensity images are being used. If working with range image sensor data simplifies the framework for solving difficult object recognition problems as compared to intensity images, then insightful ideas for useful solutions might be more easily obtained. There are no reasons to believe that Marr is not correct about the essential role of the 2.5-D primal sketch (a range image) in visual perception.

The basic thrust of this thesis is that, although intensity image understanding is often regarded as a mathematically ill-posed problem, range image understanding is a mathematically well-posed problem that can be successfully solved if object shapes are known. This thesis addresses the data-driven surface-based processes involved in the early phases of range image understanding that are independent of the classes of objects that might be encountered. It is conjectured, based on experimental results, that the solution to the range image understanding problem, when realized, will provide key insights into intensity image understanding problems.

In summary, digital surfaces are used as input to machine perception systems, especially computer vision systems. To obtain useful information from high-dimensionality sensor input, digital surfaces must be processed. Range images contain explicit approximations of scene surfaces and do not require high-level, domain-specific knowledge to

obtain scene surface structure as is the case with other types of images. Thus, range images hold a unique place in the set of digital surfaces that are obtained from sensors for scene analysis. The aim of this thesis is to (1) develop a theoretical framework, (2) devise appropriate algorithms, and (3) demonstrate a software implementation of those algorithms that will confirm the importance, the usefulness, and the uniqueness of range images for machine perception.

1.2. Low-Level vs. High-Level Processing

Extracting useful information from digital surfaces is a significant signal-to-symbol problem. Computational processes that use the pixel values themselves or transformed pixel values are known as *low-level, early, or signal* processes. Processes that use properties of *groups of pixels* are known as *high-level, later, or symbolic* processes. These terms are, of course, relative to a given frame of reference. The level of two higher level processes may be compared in terms of the level of grouping and/or the "meaningfulness" of the groups. For example, a sentence is a higher level entity than a word even though both are groups of alphabetic symbols.

A series of low-level and high-level processing steps are typically required to extract useful information from digital surfaces. Low-level processes tend to be data-driven whereas the highest level processes tend to be model-driven. At some point, high-level (symbolic, model-driven) and low-level (signal, data-driven) processes must interact to provide complete system capabilities. It is still unclear exactly how or when this interaction of model-driven and data-driven processes should take place in computer vision. High-level knowledge has been used directly in low-level processes, but this tends to severely limit the scope of applicability of such approaches unless the process-to-knowledge-base interface is designed in a flexible, modular manner. Data-driven

perceptual processes should be developed to an appropriate level, which depends on the type of sensor data, before domain-specific high-level information is used. In range image understanding, it appears that data-driven processes can yield surface-level information about a scene before high-level, object model information is needed.

Closely associated to the question of interactions between low-level and high-level processes are the fundamental problems related to local and global digital surface properties. For example, a small neighborhood around a pixel may appear to be flat locally even though it is actually part of a curved surface. Mathematical reasoning provides good insights into the computation of local properties. However, as one begins to combine local properties into global ones, existing mathematics becomes much less clear about the methods that should be used for image understanding. It will be shown that local properties of digital surfaces are well described using concepts from the differential geometry of smooth surfaces whereas functional approximation ideas can be used to describe global properties of the data.

The functional approximation approach to describing global properties brings with it the fundamental grouping problems encountered when fitting models to noisy data from unknown statistical populations. When a collection of sample data points have been drawn from a known population, there are straightforward procedures for fitting models to the data that assume that measurement errors are responsible for all random (noisy) behavior. However, a collection of sample data points is often obtained where no *a priori* information is available concerning how these points are grouped or from which population the points are drawn. Attempts to fit models to this type of data should account for systematic grouping (or gross) errors as well as measurement errors. When this situation occurs, various clustering and segmentation algorithms can be applied

based on a uniformity criteria satisfied by the data points from a given population. The RANSAC technique [Bolles and Fischler 1981] is an example of an iterative algorithm that can be used for such purposes. Some basic concepts from that approach are used in the proposed surface-based range image segmentation algorithm.

Computer vision and machine perception are characterized by the need to address these basic dichotomies in the processing of digital surfaces: signals vs. symbols, data-driven vs. model-driven algorithms, low-level vs. high-level processes, local properties vs. global properties, and measurement errors vs. grouping errors. This thesis proposes a method for early range image understanding in which the above issues are clearly resolved.

1.3. Data-Driven Range Image Analysis

A range image is the unique, simplest type of digital surface for scene analysis, and a computer vision system equipped with a good range sensor can be a powerful machine perception system. As stated above, machine perception requires data-driven, low-level processes as well as model-driven, high-level processes. This thesis focuses on early, data-driven processes for range image understanding.

The problem of interest is the following: **Given** a large, rectangular array of numbers that represent noisy samples of the depth from a sensor focal plane to the real, physical surfaces of unknown objects in a real-world scene, **segment** (partition, organize) the array of numbers (i.e., digital surface) into high-level symbolic surface primitives that will be useful to higher-level application-specific algorithms without making any domain-dependent assumptions about specific objects, object classes, or applications.

The motivation for attempting to solve this isolated, component problem is given by way of two example problems. Consider an autonomous vehicle with a range sensor

that is attempting to navigate intelligently on its path through a largely unknown world. The vehicle will necessarily have relatively limited knowledge about the 3-D world. It is assumed that one cannot afford to pre-define and store all known object shapes for the vehicle's on-board computer. That is, it is assumed that all possible boulders, holes, road surfaces, trees, telephone poles, earth terrains, vehicles, and buildings cannot be digitized to create a detailed, deterministic, object-model database. It is also assumed that technology will remain limited enough during the next five to ten years that computers will not be capable of human-like abilities to understand, conceptualize, and move about in unfamiliar, changing surroundings. Yet, this vehicle must "understand" and navigate in its environment as well as possible using relatively qualitative, abstract notions about roads and obstacles. Range image pixel values may be directly useful for simple processing, but a higher-level description is necessary to make intelligent, automated decisions about road smoothness, obstacles, and other objects in the field of view of the vehicle's range sensor. This description should be useful for identifying all kinds of surfaces: horizontal, vertical, flat, curved, kinked, and even textured surfaces, regardless of the viewpoint from the vehicle. An algorithm that can provide a piecewise-smooth surface representation for arbitrary range images would almost certainly be useful for such a vehicle.

Next, consider a non-mobile factory robot equipped with a range sensor. It inspects component parts and/or assembles larger systems of parts, but is not given these parts in a rigidly constrained fashion: the parts may be in a jumbled pile in a bin, oriented randomly on a conveyor belt, or spread out on a table. A 3-D understanding of part shape is desired owing to the tendency of the parts to be scattered about in various orientations and to the need for inspecting all surfaces of the part before attaching the piece to the assembly. In this more limited domain, it is certainly feasible to build a

database of 3-D object models that completely describe and specify all allowable part shapes. This information is used to inspect the part and to determine location and orientation parameters of the part when it is in an unknown position in the robot's field of view. In this scenario, it is desirable to compute features of parts that are invariant to viewpoint transformations. The ability to compute a piecewise-smooth surface description of a part without being constrained to a particular set of objects would be useful for a general-purpose vision system.

What do these two applications of range imaging have in common? Both cases require high-level scene surface descriptions from range images that describe visible surface shape independent of viewpoint. The descriptions should be at a higher representational level than the original data in order to avoid working with tens of thousands (up to hundreds of thousands) of depth values directly. That is, some type of data compression is desired. For example, a list of fewer than a hundred surface regions or edges would be much easier to handle. The high-level scene surface structure description must be processed at another even higher level that depends on the application domain to derive information useful to that application.

It is argued then that the above two examples (and many, many more) require a technique for converting raw image data into a different, higher-level format that combines local data properties into more global data properties. If a range image is thought of as a signal (a 2-D signal as opposed to a more conventional 1-D voltage vs. time signal), and if higher-level descriptions of the image signal are viewed as symbols, then this problem is a *signal-to-symbol transformation* problem. Two commonplace signal-to-symbol processes are discussed to draw several important analogies.

When people listen to speech, a sound signal (air pressure as a function of time) is converted into an internal nervous system signal by the human ear. That internal signal is eventually grouped into sound primitives (phonemes), which are then grouped into words and sentences. The sounds associated with the letters 'r' and 'c' cannot be distinguished on the basis of instantaneous samples or even small groups of samples of either the internal or external signal. These signals must be observed over time to identify sound primitives (i.e., symbols). That is, signal samples must be *grouped* into symbolic primitives. Note that the sound primitives themselves have no intrinsic meaning (i.e., no higher level interpretation other than an 'r' or 'c' sound). Meaning is only assigned to groups of sound primitives (groups of symbols) within the context of a specific language.

Similarly, when people observe a scene, a light signal (photon count as a function of frequency and two spatial parameters) is converted into an internal nervous system signal by the human eye. This internal signal is eventually grouped into visual primitives (symbols), which are then grouped into objects (groups of symbols). The visual primitives associated with the cylindrical body and the handle of a coffee cup cannot be distinguished on the basis of individual samples or even small groups of samples of either the internal or external visual signals. Signals must be observed over space to identify visual primitives, or in other terms, pixels must be grouped into symbolic (visual) primitives. Again, the visual primitives themselves have no intrinsic meaning of their own. Meaning is only assigned to groups of visual primitives within the context of an almost universal "visual language" that involves an exceedingly complicated, cognitive world model and incorporates concepts of 3-D space and 3-D shape.

Range images can be interpreted visually by people when depth is encoded as brightness on a display monitor. Range image signals consist of (invariably) noisy estimates of scene depth as a function of two spatial parameters. This signal must be grouped into spatial primitives (symbols) first before higher level processes can accept it for interpretation. Once these spatial primitives have been identified, they can be grouped into meaningful objects in a manner depending on the given application domain. The autonomous vehicle might have a qualitative world model and may only be interested in planning paths through the environment that avoid obstacles. The factory robot might have a quantitative world model because it is interested in determining exact part position and verifying part integrity.

At an abstract level, biological systems exist that accept physical signals as input, transform signals to new signals, convert the new signals into symbolic primitives (symbols), and then group the symbolic primitives into even higher level symbols (groups of symbols) that permit the association of meaning within the context of a specific structured cognitive domain. In machine perception, the same sequence of operations applies. Let signal A be an ordered collection of N data samples (pixels). These data samples are transformed into signal B, another set of N data samples, which represents signal A. In a biological perception system, signal A could be the external signal and signal B, the internal signal. In a machine perception system, let the transformation of signal A to signal B represent all signal (image) processing operations that maintain the same dimensionality of the signal. Next, a ***grouping operation*** segments (or partitions) the set of N data samples (pixels) in signal A into a set of M disjoint groups (digital surface regions) where $M << N$. This set of M groups represents the early levels of symbolism, or perceptual organization of the data. The symbolic primitives (regions) at this level do not permit or require the attachment of meaning. Later in the perceptual process, meaning

is attached to L groups of symbols, which are also symbolic entities ($L < M << N$). This is represented as follows:

Signal A → Signal B → Symbol → Groups of Symbols ← Meaning

In the context of the proposed approach to range image understanding, signal A represents the set of N depth values in a range image. It is converted to a set of N local-surface-type labels, known as the surface curvature sign map (also known later as the HK-sign map). These local-surface-type labels enable the grouping of pixels into M surfaces, which are the basic symbols of this example. Meaningful groups of surfaces are later recognized as L objects by a higher-level process. The thesis focuses on the process of taking raw signal data (a range image) and converting it to early symbolic primitives (smooth surface regions) using only general principles.

Although one cannot prove that early vision should always be a data-driven process, there are many advantages to data-driven low-level processing for flexible vision systems. If one chooses to directly involve domain-specific, high-level information and world models in the early stages of the signal-to-symbol process, it may be difficult to use the same system for other purposes than that for which it was designed. Imagine needing three different eye-brain systems for reading, walking, and eating. If an early vision module of a vision system relies only on general principles in identifying symbolic primitives in the signal, that module should be applicable in any situation where those general principles apply. For example, 'r' and 'c' sounds can be recognized independent of the words or the language in which they are used. Consider the difficulty in learning a new language in which none of the normal sounds of alphabetic letters are used. For flexibility and adaptability, the raw data and its properties should be the driving factor in early vision processing algorithms, not high-level models.

Successful visual perception involves an interactive combination of model-driven and data-driven processing stages at some level in a vision system. It is critically important to the performance and flexibility of the system how and when these stages are combined, what type of information is used at each stage, and how it is used. It will be shown that it is appropriate to apply only data-driven processes to range images until a piecewise-smooth surface description of the image is obtained. This surface-based data description could then be matched against a set of surface-based object models. The autonomous vehicle might use a qualitative, surface-based object model in which roads are long, almost flat, roughly horizontal surfaces that are wider than the vehicle itself. The intelligent robot might use quantitative, surface-based object models to verify that a slanted machined surface on a part is flat to within a specified tolerance and has holes drilled at eight different locations. If the same data-driven range image segmentation algorithm could be used in these two applications, it would also serve as an excellent base for solving many other problems.

1.4. Qualitative Algorithm Description

The items above have attempted to explain the motivation for developing a domain-independent, data-driven algorithm to create a high-level, symbolic-primitive description of a range image. The surface-based early-vision signal-to-symbolic-primitive segmentation algorithm proposed herein is a three-stage algorithm involving an intermediate high-dimensionality signal domain and an intermediate low-dimensionality symbol domain. The input to the algorithm is any range image that exhibits the digital surface coherence property, which is discussed in Chapters 2 and 4. The concept of digital surface coherence may be briefly expressed by noting that a range image of a piecewise-smooth surface with no added noise is completely coherent (if the smallest smooth sur-

face piece is represented by at least thirty pixels) whereas as an image of random noise is completely non-coherent. The accuracy and usefulness of the algorithm's output will be directly related to the spatial coherence of the input. Moreover, a method is included so that the input is checked automatically to see how well the surface coherence assumption is satisfied. The final output of the range image segmentation algorithm is (1) the smooth surface symbolic-primitive list, which contains 2-D region data, 3-D graph surface equations, 2-D region boundary equations, and fit errors, (2) the segmentation plot showing the various surface primitives' regions, and (3) the reconstructed image using information from the smooth surface primitive list, which allows the human observer to visually evaluate the approximation quality of the symbolic primitives. This final output contains much more information than just the 2-D region descriptions available from most segmentation algorithms. It is a rich (information-preserving) description that allows one to reconstruct an image representation of the coherent surfaces in the range image.

Figure 1.2 provides a reasonably detailed overview of the three-stage surface-based segmentation algorithm. The non-iterative first stage of the algorithm (a) smooths the range image retaining the fractional part of depth values at each pixel, (b) estimates the partial derivatives of the composite, digital surface, and (c) computes the surface-curvature-sign pixel-label image where every pixel is labeled according to its fundamental surface type based on the depth values at nearby pixels. The iterative second stage of the algorithm (a) isolates connected regions of pixels of the same surface type, (b) finds small, highly reliable seed regions where the connected region pixels' values are noisy samples from the same spatially coherent, underlying surface of a given fundamental surface type, and (c) performs an iterative, region growing procedure on each seed region based on variable-order surface fitting to obtain simple, spatially coherent, graph surface

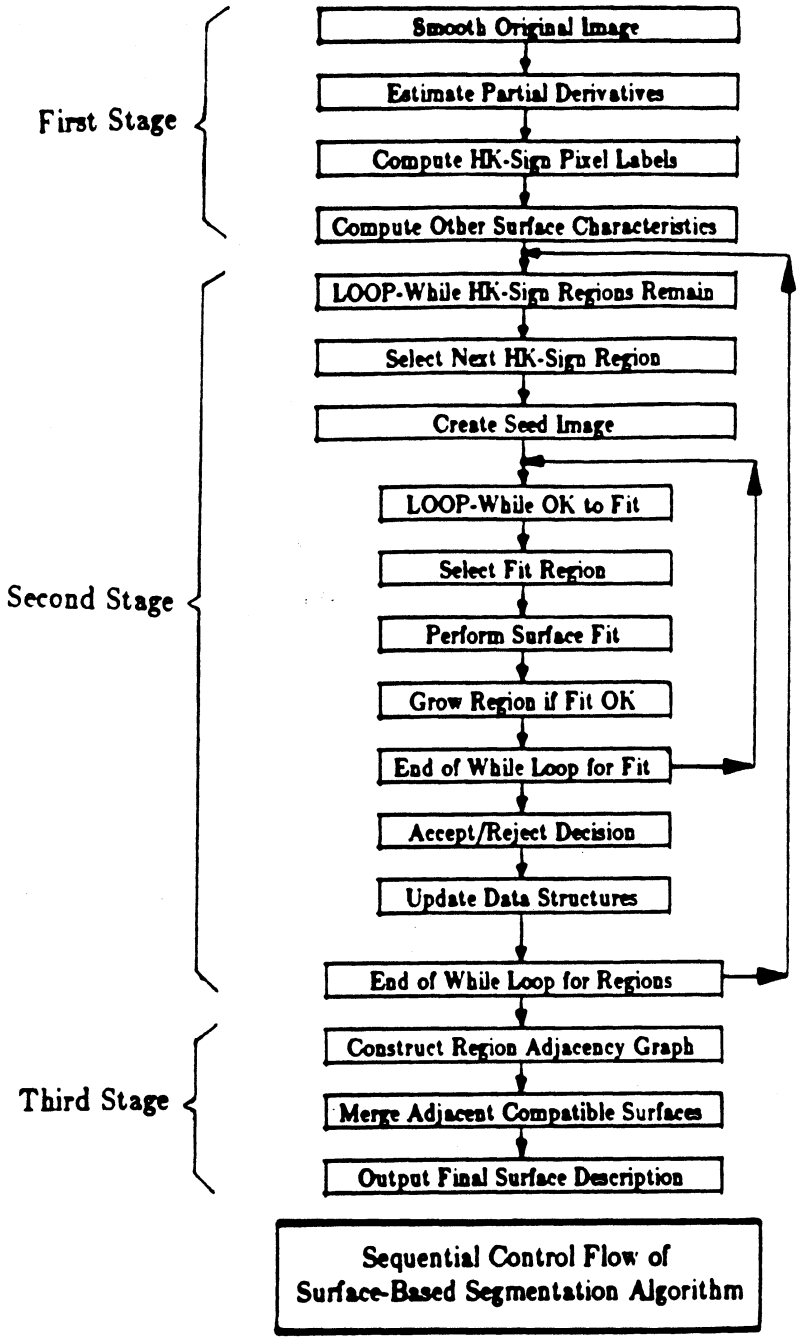


Figure 1.2.

primitives and three complementary 2-D region descriptions. The non-iterative third stage of the algorithm (a) combines the three 2-D region description to obtain a unified region description for each surface, (b) computes a region adjacency graph, and (c) merges compatible, smoothly joining, adjacent surface primitives into larger smooth

surface regions yielding a final segmentation of the range image. The output from this algorithm should be useful for a wide variety of applications. The proposed algorithm is a mixture of high-level sequential and low-level parallel elements that could be executed on appropriate hardware in a matter of seconds. Moreover, the algorithm can be applied to other types of digital surfaces, including intensity images, to obtain useful signal-level segmentation results.

The primary strengths of this range image analysis algorithm are listed below.

(1) A theoretical approach for grouping pixels in range images based on the surface coherence predicate is provided. Arbitrary curved surfaces are isolated and represented as being curved, and flat surfaces are isolated and represented as being flat in the context of the same algorithm as determined *by the data*, not by restrictive higher level models.

(2) Depth discontinuities and orientation discontinuities between surface regions in a scene become well defined in the algorithm's output representation by means of the surface approximation mechanisms used by the algorithm. The occluded and occluding surfaces of depth discontinuities are isolated, and convex and concave orientation discontinuities are also distinguished for arbitrary smooth surfaces.

(3) Region boundaries conform to whatever arbitrary (four-connected) region structure is present in the data. There are no problems with the blocky boundaries of window-based split-and-merge segmentation techniques. Convex regions are not required.

(4) A parallel region growing technique allows non-connected, non-adjacent regions of the same surface to be grouped together based on shape similarity.

(5) A noise-free piecewise-smooth image reconstruction is produced by the algorithm indicating its noise cleaning abilities.

(6) The iterative region growing process exhibits quick convergence properties. The average number of iterations is approximately seven and almost all regions converge in under fifteen iterations.

(7) It is possible to determine a fixed set of input thresholds that yield excellent performance across a wide variety of range and intensity images.

(8) Although the coefficients of the graph surface primitives are necessarily dependent on viewpoint transformations of the underlying scene surfaces, viewpoint independent surface characteristics are used and symbolic-surface-primitive shape properties are relatively insensitive to viewpoint transformations.

(9) The exact same sign-of-curvature paradigm for surface shape description is also applicable to edge shape description. The method is applied to region boundaries to identify linear and curved portions and to provide a parametric edge description.

This thesis attempts to substantiate this list of claims by means of theoretical arguments, algorithm descriptions, and experimental results on real data. The algorithm is not claimed to be a foolproof, hands-off segmentation technique, and it is not claimed that the image segmentation problem has been completely solved. Rather, an interesting set of ideas and an encouraging set of results point out what appears to be an extremely fruitful direction for early image understanding research.

1.5. Thesis Overview

This thesis addresses several different issues in varying amounts of detail. Figure 1.3 shows the hierarchy of the subjects of interest and lists the relevant chapter numbers

of each. The main emphasis is on surface characterization and range image segmentation.

Chapter 1 has been dedicated to the task of introducing the reader to the data-driven range image analysis (or early range image understanding) problem and to the task of describing the overall solution approach used by the surface-based segmentation algorithm. A literature review of previous surface characterization and image segmentation research is presented in the next section to conclude this chapter. In Chapter 2, a systems-level approach to range image analysis is taken. The object recognition problem and the digital surface segmentation problem are defined and a solution approach for object recognition based on surface characterization, surface segmentation, and surface matching is proposed. Surface characterization and range image segmentation are then subsequently addressed in a framework that requires no domain-specific knowledge, only knowledge of surfaces. In Chapter 3, the requirement that surface characterization should be based on visible-invariant quantities leads us to differential geometry. Relevant concepts for curves and surfaces are reviewed, and several theorems are stated in support of the choice of mean and Gaussian curvature as basic visible-invariant

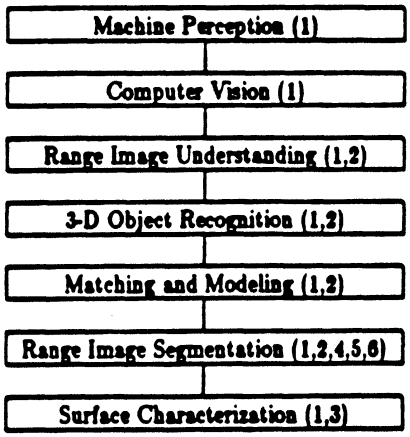


Figure 1.3. Hierarchy of Subjects of Interest by Chapters

features. Methods for estimating the necessary partial derivatives of the digital surface are also discussed. Experimental surface characterization results are presented. In Chapter 4, surface curvature sign characteristics are used in an iterative region-growing method based on variable-order surface fitting. This method computes the surface equation and three different, but complementary, region descriptions of the lowest level symbolic surface primitives as a segmentation of the digital surface. In Chapter 5, methods for combining the three separate region descriptions into a single description and for merging adjacent surface primitives that join smoothly along their separating boundary are presented. This step is necessary, in general, to group surface primitives that jointly represent meaningful smooth surfaces of complicated shape. (For scenes consisting of only basic surface types, this processing step is not required.) In Chapter 6, it is shown that the surface curvature sign concepts for surface description reduce to a sign of curvature method for edge description that is useful for describing region boundaries. In this way, surface primitives are specified by graph surface equations and by the equations of curves that bound the support region of the graph surface. In Chapter 7, the experimental results of the entire range image analysis algorithm are displayed and critiqued. The algorithm is also applied to several intensity images indicating that the digital surface approach is general enough to be used for other types of images. In Chapter 8, a complete thesis summary is given, and future research directions are outlined. Several appendices are included to stress technical points mentioned only briefly in the text.

1.6. Review of Previous Research

This thesis addresses the question "How should high-dimensionality range images be processed to provide a complete low-dimensionality image description that preserves the visual information in the image and that is useful to high-level 3-D object recogni-

tion and image understanding algorithms?" The two central topics encountered in answering this question are surface characterization and image segmentation. The goal of this section then is to briefly review previous research in those areas with an emphasis on material related to this thesis. The 3-D object recognition literature is reviewed in [Besl and Jain 1985] whereas computational approaches to image understanding are reviewed in [Brady 1982].

1.6.1. Surface Characterization Review

The term *range image processing* algorithm refers to any algorithm that processes range images to obtain some type of useful information, such as edge maps or planar region descriptions. The term *surface characterization* algorithm refers to any algorithm that is capable of describing several different types of surfaces. Both terms are applicable to many algorithms, such as the method proposed in this thesis. Therefore, range image processing algorithms are reviewed along with methods for surface characterization to give a better summary of previous early vision research with range images. Several related papers are also reviewed.

Duda et al. [1979] discusses the use of registered range and reflectance images to find planar surfaces in 3-D scenes. A *sequential* planar region extraction procedure is utilized on range and reflectance images obtained from a modulated-continuous-wave laser rangefinder [Nitzan et al. 1977]. Reflectance images are slightly different than intensity images because shadows cannot occur. *A priori* scene assumptions concerning man-made horizontal and vertical surfaces motivate the procedure. First, horizontal surface regions of significant size are segmented and removed from the images using a filtered range (z) histogram. Second, major vertical surfaces are extracted from the remaining scene data using a Hough transform method. Third, arbitrary planar surfaces

are identified with the help of reflectance histogram data. All planar surfaces are thus segmented and labeled. All unlabeled regions correspond to depth discontinuities, non-planar regions, or very small planar regions not found in the three main processing steps. The overall technique appears to work well on three test scenes, but many domain-dependent assumptions have been used.

Milgram and Bjorklund [1980] also discuss planar surface extraction in range-images created by a laser rangefinder. The spherical coordinate-transformation converts slant range, azimuth angle, and elevation angle sensor data into standard Cartesian data before image processing (see Appendix D). For each Cartesian-coordinate range-pixel, a plane is fitted to the surrounding 5 x 5 window, and the two normal-vector orientation angles, the position variable, and the planar-fit-error are computed. This data forms *connected components* of pixels that satisfy planarity constraints, which then are extracted to form planes. After subsequent region growing is complete, a "sensed plane list" is built. This plane list, the symbolic scene description of the system, is compared with a reference plane list to determine sensor position with respect to a stored scene model. Experimental results are discussed for four real world range-images and two synthetic range-images displaying different viewpoints of the same building site. Their method appears to be a much better, more straightforward approach to planar surface extraction than that of Duda et al. [1979]. This system was planned for vehicle navigation.

Henderson [1984] has developed a method for finding planar faces in range data. First, a list of 3-D object points are assumed given by a rangefinder. To handle multiple depth-maps, points are transformed into one object-centered coordinate system using transformation data recorded during range-image formation [Henderson and Bhanu

1982]. These points are first stored randomly in a list with no topological connectivity information. The points are then organized into a 3-D binary tree, which is done in O(NlogN) time (where N is the number of points). Second, each point's neighbors are determined with the aid of the 3-D tree, and the results are stored in a 3-D spatial proximity (nearest neighbor) graph. Third, a spiraling, sequential planar-region-growing algorithm, known as the three-point seed method [Henderson and Bhanu 1982], creates convex planar faces using the spatial proximity graph as input. The union of these faces form the polyhedral object representation as extracted from the range data. Several processing steps mentioned above are required because of the (x,y,z) list format of the input data. Neighbors are given explicitly in standard range-image formats, but range images can only represent a scene from a single view. This method can be used for either range data segmentation or object reconstruction. It can also work on dense range data or a sparse collection of points. Curved surfaces are approximated by many polygons.

Wong and Hayrapetian [1982] suggest the use of range-image histograms to segment corresponding registered intensity-images. All pixels in the intensity-image that correspond to pixels in the range-image with depth values not in a certain range are set to zero, segmenting all objects in that particular range. This segmentation trick is useful in specific applications, but it hardly begins to take advantage of explicit range image information. Also, it cannot work on long objects that span the dynamic range of the range-image.

Reeves et al. [1985] have extended their previous work in moment-based 3-D shape analysis to include range moments of range images. Single-object, pre-segmented, synthetic range images of aircraft were classified using silhouette moments, range moments, and a combination of both types of moments. It was shown that this set of images was

best characterized by the combination of range and silhouette moments. Although good results were obtained, this method relies on global object characteristics and cannot be expected to provide reasonable results if objects are partially occluded.

Gil et al. [1983] have demonstrated the usefulness of combining intensity and range edges from registered range and intensity images to obtain more reliable edge information.

Langridge [1984] reports on an investigation into the problem of detecting and locating discontinuities in the first derivatives of surfaces determined by arbitrarily spaced data. Neighbor computations, smoothing, quadratic variation, and the biharmonic equation are discussed. The techniques are useful for detecting roof-edges in range images.

Inokuchi et al. [1982] present an edge-region segmentation ring operator for depth-maps. The ring operator extracts a one-dimensional (1-D) periodic function of depth values that surround a given pixel. This function is transformed to the frequency domain using an FFT algorithm for either 8 or 16 values. Planar-region, step-edge, convex-roof-edge, and concave-roof-edge pixels (see Figure 1.4) are distinguished by examining the 0th, 1st, 2nd, and 3rd frequency components of the ring surrounding that pixel. These pixel types are grouped together, and the resulting regions and edges are labeled. Experimental results are shown for one synthetic block-world-scene range-image. The ring-operator method appears to compute roof-edges fairly well at the boundaries of planar surfaces. But it is not stated how range-images with curved surfaces are handled. Two years earlier, Inokuchi and Nevatia [1980] discussed another roof-edge detector that applied a radial line operator at step-edge corners and followed roof-edges inward.

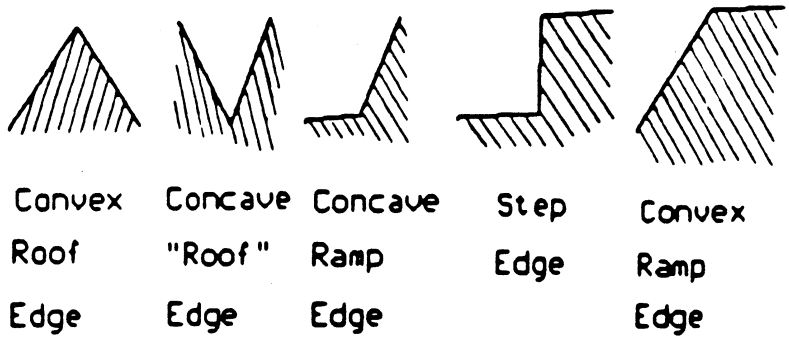


Figure 1.4. Edge Types of Interest in Range Images

Mitiche and Aggarwal [1983] have developed an edge detector that is insensitive to noise due to use of a probabilistic model that attempts to account for range measurement errors. The computational procedure is as follows: (1) Step-edges are extracted first from a depth-map. (2) For each direction (usually four) in the image at each pixel, a roof-edge is hypothesized. For each hypothetical roof-edge, two planes are fitted to the immediate neighborhood of the pixel, and the dihedral angles between these planes are recorded. They call this the "computation of partitions." (3) Pixels are discarded if all dihedral angles are less than a threshold. For every remaining pixel, *a Bayesian likelihood ratio* is computed, and the most likely partition (edge-direction) is chosen. If the angle for the chosen direction is less than another threshold, the pixel is also discarded. This is called the "dismissal of flat surfaces." (4) All remaining pixels are passed through a non-maxima suppression algorithm that theoretically leaves only the desired edge pixels. The method can handle a large amount of added noise because the system is constrained by its internal model to look for horizontal and vertical edges (a domain-specific constraint).

Lynch [1981] presents a range-image enhancement technique for range data acquired by a 94 GHz (3.2mm) radar. In the special case of systems with a shallow

(nearly horizontal) line of sight, a strong depth gradient always exists in a range-image. This gradient makes it very difficult for people to interpret such a range-image using a typical 256-gray level display device. Two *1-D high-pass* filters (a normalized filter and a differenced filter) are derived, discussed, and applied to an example scene to create a feature image that is more easily interpreted by a human observer than the original range-image. Lynch's approach distorts the shape information in range data to achieve high local contrast.

Sugihara [1979] proposes a range-image feature-extraction technique for edge-junctions similar to the junction features used by Waltz [1972] and others for intensity-image understanding. A junction dictionary of possible 3-D edge junctions is implemented as a directed-graph data structure and is useful in aiding 3-D scene understanding. Unlike intensity-image edges, range-image edges are classified as *convex, concave, obscuring, or obscured* without additional higher-level information from surrounding image regions; junction knowledge is not necessary for line categorization. This categorization is therefore used to help predict missing edges. Since several junctions are only possible when two or more objects are in a scene, junction information can then be used to segment the range-image into different objects. It is noted that junction points are *local curvature maxima* points in the depth-map surface. A system is described that uses depth-discontinuity contours (step-edges) and junctions for complete scene segmentation. It is limited by the constraint that every vertex is connected to at most three faces.

Several papers have discussed range-image processing techniques for the detection of cylinders in range data [Agin and Binford 1973] [Nevatia and Binford 1973] [Popplestone et al. 1975] [Bolles and Fischler 1981]. Only the last and most recent of these is reviewed.

Bolles and Fischler [1981] present the Random Sample Consensus (RANSAC) technique for fitting models to noisy data containing a large percentage (20% or more) of gross errors. Gross errors occur, for instance, when fitting a plane to a set of points where most points belong to the plane, but some are from other nearby surfaces. Linear least-square techniques are effective for filtering out normally-distributed measurement-errors, but cannot remove gross errors. They propose a two-step filtering process where (1) initial estimates of model parameters are computed to eliminate gross errors, and (2) an improved fit is computed by applying standard smoothing techniques to the pre-filtered data. For example, the RANSAC approach to circle-fitting is to select only three points at random, compute the fitting circle, and count the number of other compatible points in the data that are within the expected measurement error threshold. If there are enough compatible points, then least-squares smoothing is applied to the three initial points and all other compatible points. If not, another set of three points is selected and the process is repeated. If the number of trials exceeds a preset threshold, then the process is stopped. The RANSAC technique is applied to finding ellipses, and then cylinders, in light-stripe range data. Although it is slower than standard fitting methods, model parameters are insensitive to gross errors. The approach to surface fitting advocated later in this thesis has several similarities to the RANSAC approach, but, for example, the seed points for surface fitting are not selected at random.

Dreschler and Nagel [1981] proposed the use of Gaussian curvature to find corner points in intensity images. Such a technique is equally valid for range image corner points. They compute Gaussian curvature using the 5x5 window operators of Beaudet [1978] to estimate the necessary partial derivatives. Gaussian curvature for surface characterization is computed using a similar approach in Chapter 3.

Most research in the range image processing category is limited to searching for particular structures in the data, such as planes, cylinders, edges, and vertices. More general techniques are discussed now that allow flexibility in the shapes of surfaces.

Extended Gaussian Images (EGI's) have been studied in detail by Horn [1984] and Ikeuchi et al. [1983]. An EGI of a range image is a 2-D histogram of the distribution of normal vectors on the digital range image surface. This orientation histogram approach provides unique rotationally-invariant shape description for convex objects [Little 1983][Minkowski 1897], but does not maintain this property for non-convex objects (see Figure 1.5). The EGI concept has been extended by Ikeuchi et al. [1983] so that each discrete view of a non-convex object has its own orientation histogram. EGI's from two different surfaces may be compared using a template matching approach. The properties of Gaussian curvature are important to the EGI concept because the EGI is a discrete approximation of the Gaussian curvature function over all latitudes and longitudes on the unit sphere. Since the EGI approach requires only surface orientation information, photometric stereo may be used to obtain sensor data for input to the EGI algorithm. Object surface regions are assumed to be pre-segmented by another process in

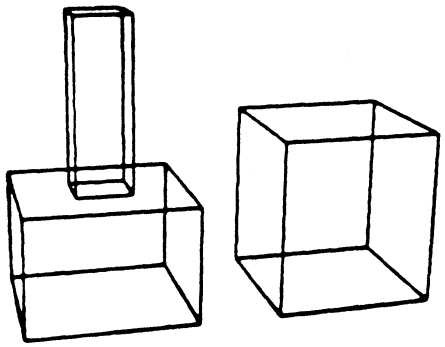


Figure 1.5. Two Objects with Same Basic EGI Representation

most EGI approaches.

Faugeras et al. [1985] at INRIA use an approach that is similar to the EGI approach in its dependence on surface normal information and its assumption that object surface regions have been pre-segmented by another process. Planar patches are isolated in range data using a region growing approach. Rotational matching on planar components of an object surface is performed using a *quaternion-based* algorithm that finds the best-fit rotation parameters to align a set of planar normal vectors. Quaternions provide a simple derivation of a method for converting a three-variable non-linear least squares problem into a 4x4 matrix eigenvalue problem. Translational matching of the planar patches is a standard least-squares problem. Mathematics for quadric surface set matching is also described. A working object recognition system has been developed based on this approach. The EGI and INRIA approaches are both able to characterize any piecewise planar surface in a way that is amenable to matching algorithms.

Early researchers [Shirai and Suwa 1971] [Popplestone et al. 1975] have processed range data by analyzing small surface regions and then linking regions with compatible characteristics into surfaces. This technique is discussed as it appears in Oshima and Shirai [1983].

Oshima and Shirai [1983] address object recognition via the processing of 3-D information from a light-stripe range finder. The range data is processed as follows: (1) points (range pixels) are grouped into small planar surface elements, (2) surface elements are merged into elementary regions that are classified as either planar or curved (spherical, cylindrical, conical), (3) curved elementary regions are merged into consistent global regions that are fitted with quadric surfaces, and (4) a complete scene description is generated from global region properties and relationships between these regions. The ele-

mentary region properties are based on the best-fit planar region and its boundary, and include the following quantities: perimeter; area; peround; minimum, maximum, and mean region radii around the region centroid; and the standard deviation of the radii of the boundary. The region relationships are characterized by the distance between region centroids, the dihedral angle between best-fit planes, and the type of intersection curve between the regions. This surface characterization is used as input to a higher-level view-dependent matching algorithm that recognizes objects.

Medioni and Nevatia [1984] have suggested a *curvature*-based description of 3-D range data. They propose the following features for shape description: (1) zero-crossings of the Gaussian curvature, (2) zero-crossings of the maximum principal curvature, and (3) maxima of the maximum principal curvature. This set of differential geometric shape descriptors is a subset of the descriptors that are computed in Chapter 3. These features are computed by smoothing a depth-map with a large window and by using relatively primitive 1-D windows to compute directional derivatives. This 1-D derivative approach will more sensitive to noise than 2-D window derivative operators. Experimental results for the vase shape shown in the paper agree with this statement.

Ittner and Jain [1985] have taken quite a different approach to surface characterization based on clustering and local surface curvature measures. Range data points are first clustered (segmented) into groups based on 3-D coordinates and surface normal vectors using a mean square error criterion. The groups of range data points are then characterized by six separate surface curvature estimates, including mean and Gaussian curvature, the principal curvatures, average normal curvature, and the principal curvature ratio. Their methods for estimating surface curvature are quite different than the 1-D and 2-D window methods encountered in other research. Although good results are

obtained with this method, the segmentation is currently limited to identifying small surface patches because point coordinates and mean square error are used in the clustering algorithm.

Brady et al. [1985] use differential geometric features to describe surfaces, but they concentrate only on lines of curvature, asymptotes, bounding contours, surface intersections, and planar and spherical (umbilic) surface patches. By relying on surface curves to constrain (describe) surface shapes, the curvature primal sketch work of Asada and Brady [1986] can then be used for *planar* curve shape description of surface curves. One problem with this approach, as pointed out in [Ponce and Brady 1985], is that lines of curvature are not necessarily planar. Another problem is that, because every surface curve of each surface is processed individually by the Curvature Primal Sketch algorithm, the method is computationally intensive; one-hour running-times for 128x128 range-images are mentioned for a dedicated Lisp machine. Also, while lines are necessary for 3-D surface plots, the implicit premise that lines are necessary for a rich 3-D surface description does not necessarily follow. They compute principal curvatures and principal directions, but rely on an *ad hoc* scheme using a breadth-first search to link principal directions at each point into lines of curvature. A Surface Primal Sketch is proposed that combines information on significant surface discontinuities. This approach is domain-independent and information-preserving, but the final description appears to include an excessive amount of information. Experimental results for a light bulb, a telephone receiver, a coffee mug, and an oil bottle look quite good.

Nackman [1984] discusses surface description using critical-point configuration graphs (CPCG). All critical-points are isolated from each other except in degenerate circumstances. Non-degenerate critical-points of surfaces are local maxima, local minima,

or saddle-points. If critical-points of a surface are identified as the nodes of a graph, and the connecting ridge and course (valley) lines (zero crossings of the first partial derivatives) are considered the arcs of a graph, a surface is characterized by a critical-point configuration graph. Slope-districts are the regions bounded by graph cycles. Two important theorems relating to these graphs are discussed: (1) only eight types of critical-points are possible: peaks (local maxima), pits (local minima), and six types of passes (saddle points) (this is discussed in Chapter 3); (2) only four non-equivalent types of slope-districts are possible. All surfaces have a well-defined characterization as the union of slope-district regions where each region belongs to one of four basic slope-district types. This characterization approach is a generalization of techniques for describing 1-D functions f(x). In the 1-D case, only two types of non-degenerate critical points exist: local maxima and local minima. Between these critical points are intervals of constant sign of the first derivative. Slope-districts are generalizations of these intervals. Nackman also mentions curvature districts determined by the *sign* of the mean and Gaussian curvature of a surface, but does not explore them. This work resulted from research to generalize the symmetric axis transform for 3-D objects [Nackman 1982].

Haralick et al. [1983] discuss topographic classification of digital surfaces. They review seven earlier papers on the subject by various authors, and their ten topographic labels are a superset of all labels used previously: peak, pit, ridge, ravine (valley), saddle, flat (planar), slope, convex hill, concave hill, and saddle hill. At each pixel in an image, a local *facet-model* bicubic-polynomial surface is fitted to estimate the first, second, and third partial derivatives of the surface at that pixel. (Other types of functions have been tested in Watson et al. [1985].) Once the derivatives have been estimated, the magnitude of the gradient vector, the eigenvalues of the 2x2 Hessian matrix, and the directional

derivatives in the direction of the Hessian matrix eigenvectors are computed. These five scalar values are the input to a function table that produces the pixel classification. Pixel-by-pixel classification forms groups of pixels of a particular type. The topographic primal sketch is proposed for use with intensity-image surfaces because of the invariance of the pixel-labels to monotonic transformations of gray levels, such as changes in brightness and contrast. It will also be useful for range-images, but more than half of the pixel-labels are not invariant (in general) to changes in viewpoint. This approach to surface characterization is similar to the approach discussed in Chapter 3, but differs significantly in the treatment of viewpoint invariance properties. A detailed comparison of these two methods is found in [Besl and Jain 1986].

Lin and Perry [1982] have investigated surface shape description using surface triangularization. Differential-geometry-based shape measures are useful if they are reliably computed from sensor data. When a surface is decomposed into a network of triangles, many features are easily computed. Discrete coordinate-free formulas for surface area, Gaussian curvature, aspect ratio, volume, and the Euler-Poincare characteristic are given. The formula for Gaussian curvature is interesting because estimates of first and second partial derivatives are not needed and the formula is coordinate-system independent, reflecting the isometric invariance properties of the Gaussian curvature (Appendix B). Integral Gaussian curvature, integral mean curvature, surface area, volume, surface area to volume ratio, integral curvature to the n-th power, and genus (or handle number) are given as scalar values characterizing the shape of a surface. No experimental results are recorded in the paper, but Besl et al. [1985] have used several of the described features, such integral mean and Gaussian curvature, for classifying intensity-image surfaces of solder joint images for industrial inspection.

Peet and Sahota [1985] have used surface curvature measures of image texture for discriminating biological cell nuclei in gray-scale intensity images. Given an image or subimage, they compute seventeen (17) different surface curvature averages based on local quadratic surface fit using a 3x3 window: principal curvature difference, maximum absolute value principal curvature, minimum principal curvature, maximum principal curvature, principal curvature absolute values, mean curvature, Gaussian curvature, minimum absolute value principal curvature, sum and difference of absolute value principal curvature, and the number of elliptic points, parabolic points, saddle points, peak points, pit points, and flat points. The results for discrimination tests were not impressive, but the first two surface curvature averages did appear to improve the performance of an existing system. Based on this author's experience in using similar features to classify solder joint quality, 3x3 windows on 8-bit data produce extremely poor surface curvature results if any noise is present in the image. Thus, it is not surprising that only two such curvature measures proved to be worthwhile.

Hebert and Ponce [1982] propose a method of segmenting depth-maps into plane, cylindrical, and conical primitives. Surface normals are computed at each depth pixel using the best-fit plane in 3x3 windows. These normals are mapped to the *Gaussian sphere* where planar regions become very small clusters, cylinders become unit radius semicircles, and cones become smaller radius semicircles. This orientation histogram is referred to as the extended Gaussian image, or EGI. The Hough transform detects these circles and clusters. Regions are then refined into labeled, connected components. Although somewhat restricted, this technique handles at least several types of curved surfaces.

Sethi and Jayaramamurthy [1984] have investigated surface classification using characteristic contours. The input is a needle map of surface normals. A characteristic contour is defined as the set of points in the needle map where surface normals are at a constant inclination to a reference vector. The following observations are made concerning these contours: (1) The characteristic contours of spherical/ellipsoidal surfaces are concentric circles/ellipses, (2) The characteristic contours of cylindrical surfaces are parallel lines, and (3) The characteristic contours of conical surfaces are intersecting lines. These contours are computed for all normals using a 12x12 scanning window. The identity of the underlying surface for each window is computed using the Hough transform on the contours. A consistency criterion is used to fight noise effects and the effects of multiple surface types within a given window. This approach is similar to that of Hebert and Ponce [1982]. Correct classification results are discussed for synthetic 40x40 needle maps of adjacent cones and cylinders. It may be difficult to generalize the characteristic contour method to arbitrary surfaces because the characteristic contours of a general surface may not exhibit any usable regularities.

Tomita and Kanade [1984] generate range image scene descriptions using an edge detection approach. Gap (step) edges and corner (roof) edges are detected and linked together. Surfaces are defined by the closed boundary of edges. All edges are assumed to be linear or circular, and all surfaces are assumed to be planar or cylindrical (conic). Smith and Kanade [1984] and Hebert and Kanade [1985] use a similar approach for describing range images.

Bolles et al. [1983] and Horaud and Bolles [1984] also rely completely on roof edges and step edges for characterizing the properties of range images with their 3DPO system. The system groups edges into coplanar clusters and searches for circular arcs in these

clusters. The main emphasis of this work is on special feature detection, such as dihedral edges and corners. Excellent part position determination results have been achieved for range images of a parts bin containing one type of industrial part.

Herman [1985] extracts a detailed polyhedral object description of range images also using an edge detection approach. Edges, which may be occluding, convex, or concave, are computed to create a line drawing using the Hough transform. These line segments are refined to eliminate gaps so that 3-D planar faces may be formed.

Potmesil [1983] developed an algorithm for generating 3-D surface models of solid objects from multiple range image projections. Although this work emphasizes the integration of range information obtained from known multiple views for object reconstruction, it is of interest to range image understanding because it treats the basic problems of surface matching and surface representations for vision. Potmesil fits a sheet of parametric bicubic patches to a range image to describe its shape. These rectangular surface patches are recursively merged into a hierarchical quadtree structure where a surface patch at each level approximates the shape of four finer resolution surface patches. Evaluation points at surface control points and points of maximum surface curvature are used to manipulate the surface patches for matching purposes, and therefore, constitute a surface description. Excellent experimental results for a turbine blade and a car model were obtained by the object reconstruction algorithm.

Marimont [1984] presents a representation for image curves and an algorithm for its computation. His work is mentioned because similar concepts are also useful for surfaces. The curve representation "is designed to facilitate the matching of image curves with model *plane curves* and the estimation of their orientation in space despite the presence of noise, variable resolution, or partial occlusion." This representation is based

on the curvature function computed at a pre-determined list of scales, or smoothing-filter window-sizes. For each scale, the points, or knots, which are the zeros and the extrema of curvature, are stored in a knot list with a tangent direction and a curvature value for each knot. These knots have the following properties:

(1) The zeros of the curvature of a 3-D plane curve almost always project to the zeros of the curvature of the corresponding (projected) 2-D image curve.

(2) The sign of the curvature value at each point does not change within an entire hemisphere of viewing solid angle. The *pattern of curvature sign changes* along a curve is invariant under projection except in the degenerate case when the viewing point lies in the plane of the curve.

(3) Curvature is a local property, which makes it much more suitable than global curve properties for handling occlusion.

(4) Points of maximum curvature of 3-D plane curves project to points that are very close to points of maximum curvature of the projected 2-D image curves. The relationship between these points is stable and predictable depending upon viewpoint. Moreover, the relative invariance of these points increases as the curvature increases. Ideal 3-D corners almost always project to ideal 2-D corners.

The algorithm is outlined as follows: (1) The image curve data is smoothed at multiple scales by Gaussian filters and fitted at each scale with a continuous curve parameterization in the form of composite monotone-curvature splines (arcs). (2) Curvature extrema (critical-points) are extracted at each scale and stored in a list. (3) Dynamic programming procedures are used to construct a list of critical-points that is consistent across the range of scales. (4) The integrated critical-point information from the image curve is matched against the computed critical-point information for the plane curve model. No

experimental results for curve matching are quoted in the paper. Related work on multiple scale curve matching has been done by Mokhtarian and Mackworth [1986].

1.6.2. Image Segmentation Review

The term *image segmentation* refers to the process of partitioning the pixels in a 2-D image into regions such that there is a meaningful correspondence between the regions in the image and the surfaces of objects in the 3-D scene represented by the image. It is clear from this definition of segmentation that an image cannot be segmented without knowledge of possible surfaces and objects. It is also generally recognized that, in the absence of special purpose object features, it is difficult to recognize objects in an image unless the image is already segmented. Given the difficulty of the segmentation problem, it is not surprising that many solutions have been constrained to limited domains. As researchers approached the problem from many different angles, the volume of the vision literature on image segmentation grew proportionately large. Therefore, the segmentation literature itself will not be reviewed here. Instead, a chronological review of image segmentation survey papers is presented where references to individual exemplary articles are included in each discussion to represent the categorization advocated in that survey. This is followed by a review of two approaches that are the most similar to the approach to range image segmentation presented by this thesis. Finally, two papers that directly address segmentation in range data are briefly discussed.

Since Zucker [1976] wrote the first segmentation survey paper, which concentrated on region growing techniques, the literature has seen about one new segmentation survey paper per year for the last ten years. It is still an excellent reference on region growing indicating the lack of significant developments since then. The general mathematical definition of segmentation given by Horowitz and Pavlidis [1974] is used in this survey

paper and also in Chapter 2 of this thesis. Zucker classifies region growing schemes into the following categories:

(1) Regional Neighbor Search Schemes: Images are divided into small cells. Neighboring cells are joined if statistically compatible to form labeled image segments until no joinable neighbors are left [Muerle and Allen 1968].

(2) Multiregional Heuristic Schemes: Connected-component regions of nearly constant intensity are formed. The phagocyte and weakness heuristics are applied sequentially to remove weak boundaries between adjacent regions [Brice and Fennema 1970].

(3) Functional Approximation and Merging Schemes: Image strips are approximated by piecewise linear functions. Compatible pieces of image strips are merged to form image segments [Pavlidis 1972].

(4) Split and Merge Schemes: A pyramid (or quadtree) data structure is used. Image quadrants are recursively divided into smaller quadrants until sufficiently well approximated (constant) quadrants are obtained. Compatible, adjacent smaller quadrants are merged. Similar squares are grouped to form irregularly shaped regions [Horowitz and Pavlidis 1974].

(5) Semantic-based Schemes: Connected-component regions of nearly constant intensity are formed. *A priori* domain-dependent knowledge about an image is used to maximize the probability that the image is correctly segmented. The probability of all correct region interpretations and all correct region boundary interpretations is maximized [Feldman and Yakimovsky 1974].

Zucker also presents good discussions about the critical issues of threshold selection, order dependence, partitioning, and global initialization.

Riseman and Arbib [1977] view the goal of the initial stages of processing in visual systems as segmentation, which is "a *transformation* of the data into a partitioned image with the parts in a representation which is more amenable to *semantic* processing." They divide all segmentation approaches into two broad categories: (1) boundary (edge) formation approaches, which focus on differences in image data, and (2) region formation approaches, which focus on similarities. They investigate the boundary extraction tools of (a) spatial differentiation [Hueckel 1973] (see survey by Davis [1975]), (b) nonmaxima suppression [Rosenfeld and Thurston 1971], (c) relaxation [Rosenfeld et al. 1976], and (d) edge linking. Region extraction approaches are divided into (a) region growing under local spatial guidance [Brice and Fennema 1970], (b) semantic-guided region merging [Feldman and Yakimovsky 1974], (b) histograms for global feature analysis [Ohlander 1975], and (c) spatial analysis of feature activity [Hanson et al. 1975]. This lengthy survey includes detailed accounts of many individual approaches.

Rosenfeld and Davis [1979] recognized the critical importance of *underlying image model assumptions* to the performance and limitations of image segmentation techniques. Statistical image models, such as first-order gray level probability density models, random field models, and time series models, are reviewed in addition to several different spatial (structural) image models. They informally examine the use of image models to predict performance of given segmentation methods and to help design new segmentation methods. The discussion is limited to 2-D scenes so as to avoid three-dimensional issues. They do not categorize segmentation techniques since Rosenfeld and Kak [1976] already presented discussions of (a) thresholding, (b) edge detection, (c) region growing, (d) border following, (e) spectral signature classification, and (f) template matching techniques for segmentation in the first edition of their text.

Kanade [1980] presents a model of image understanding that is jointly model-driven and data-driven, and points out that region segmentation methods may have three different goals depending on the levels of knowledge used by the method:

(1) **Signal**-level segmentation methods use cues from the image domain only, and are therefore limited in what they can achieve.

(2) **Physical**-level segmentation methods use cues from the scene domain by using physical knowledge about how scene domain cues transform into image domain cues.

(3) **Semantic**-level segmentation methods use semantic, physical, and signal level knowledge to compute an entire image interpretation in terms of an instantiated model.

Kanade categorizes existing signal-level schemes as (a) region growing by spatial criteria [Muerle and Allen 1968] [Brice and Fennema 1970][Pavlidis 1972], (b) region segmentation by global spectral distribution [Prewitt and Mendelsohn 1966][Tsuji and Tomita 1973][Ohlander 1975], and (c) combined usage of spatial and spectral information [Milgram 1977][Rosenfeld 1978]. Semantic region segmentation schemes are discussed but not categorized [Barrow and Popplestone 1971][Feldman and Yakimovsky 1974] [Tennebaum and Barrow 1976]. Physical knowledge processing schemes are also reviewed [Horn 1977] [Woodham 1977] [Barrow and Tennebaum 1978] [Kanade 1981]. Note that *image domain cues are scene domain cues* in range image segmentation.

Fu and Mui [1981] categorize signal level image segmentation techniques into (1) characteristic feature thresholding or clustering, (2) edge detection, and (3) region extraction. The emphasis of the survey is on techniques in categories (1) and (2); Zucker [1976] is referenced followed by a brief survey of region techniques, which are divided into

merging, splitting, and split and merge approaches. The Horowitz and Pavlidis [1974] definition of segmentation is also used as in Zucker [1976] (see Chapter 2). It is mentioned that despite all the research into segmentation, very little is known about how to compare algorithms and how to measure segmentation error besides the percentage of misclassified pixels criterion [Yasnoff et al. 1977]. Thresholding algorithms are divided into statistical and structural methods. The statistical thresholding methods [Weszka 1978] are divided into global (e.g. [Prewitt and Mendelsohn 1966]), local (e.g. [Ohlander 1975]), and dynamic algorithms. Structural thresholding techniques, such as [Tsuji and Tomita 1973], are not divided further. Many, many pattern-recognition-type clustering algorithms for image segmentation are reviewed. Edge detection techniques are categorized into parallel and sequential as in [Davis 1975]. Parallel edge element extraction techniques are subdivided into spatial frequency filters, gradient operators, and functional approximation techniques. Edge element combination techniques (edge linking) are subdivided into heuristic search, dynamic programming, relaxation, and line and curve fitting.

DiZenzo [1983] has written a comprehensive review of image segmentation at the signal level. The main contribution of this survey is that it is more up-to-date than previous surveys. Although written in an introductory style, many current topics are discussed, such as random-field image models, the sloped-facet model, surface fitting, quadtrees, and relaxation methods. The main subject divisions used by this survey article are image models, the labeling of collections, pixel labeling, edge detection, and region-based split-and-merge methods using quadtrees.

Mitiche and Aggarwal [1985] present a review of conventional image segmentation techniques, which stress region extraction and boundary placement approaches, to pro-

vide a base for a discussion of novel *information-integrating segmentation* techniques. They recognize the importance of *integrating sensor data over time from many different sensors*, such as intensity, thermal, and range images. Thus, motion cues can contribute to segmentation along with individual cues from each sensor. By combining several sources of information, segmentation results can be achieved that are better than those possible from any single source.

Haralick and Shapiro [1985] divide signal-level image segmentation techniques, which are viewed as clustering processes, into the following list of categories:

(1) Single-linkage region-growing schemes: Individual pixel values used for similarity measurement.

(2) Hybrid-linkage region-growing schemes: Neighborhood pixel values used for similarity measurement.

(3) Centroid-linkage region-growing schemes: Individual pixel values are compared to mean of growing region to determine similarity measure.

(4) Measurement space clustering schemes: Histogram mode-seeking guides spatial clustering via thresholding.

(5) Spatial clustering schemes: Histograms guide region growing techniques.

(6) Split and merge schemes: Pyramid data structures and homogeneity measures are used to split inhomogeneous image quadrants into four smaller quadrants. Small regions are merged based on similarities.

This survey is unique in that it applies several different segmentation techniques to the same image, an F-15 bulkhead image, to compare the performance of different algorithms. They note how similarity tests in the various algorithms can be made more powerful by using rigorous statistical tests (e.g. F-test) at a given level of significance.

In summary, the fundamental issues in image segmentation are centered around regions vs. edges, uniformity vs. contrast, signal vs. semantics, and statistical vs. structural controversies. In the terminology of these surveys, the segmentation approach proposed by this thesis is a *functional approximation and merging* method although it is completely different than any existing approaches in that category. It may also be considered as a *hybrid-linkage/centroid-linkage region growing* scheme, but again it is not very similar to any existing techniques. It has no connections whatsoever to any segmentation techniques based on histograms, edge detection, or split-and-merge algorithms. Two other functional approximation methods and two range image segmentation papers are now reviewed.

Pavlidis [1972] posed the image segmentation problem as an optimization problem of functional analysis. He sought a set of approximating functions defined over regions that partition the image. However, he chose to divide the image into N thin strips. Each image strip is then approximated by one-dimensional piecewise-linear functions over the J segments of each strip. Adjacent segments in different strips are merged if the slopes of the segments are sufficiently close to one another. The merging continues until no more merges are possible. This work appears to have evolved directly from previous work in waveform segmentation using piecewise linear functions. As in the surface-based approach of this thesis, it is possible to reconstruct a representation of the original image from the approximations used to segment the image. The problem with the suboptimal approach is that it is attempting to solve an inherently 2-D problem by decomposing it into two 1-D problems that can be solved separately, one after the other. It is certainly possible to obtain good results on a certain set of images with such an approach, but it does not provide a general-purpose framework for addressing the problem.

Haralick and Watson [1981] developed a facet model for image data that provides a unified framework for addressing edge detection and region growing. A *facet* is a connected region of image pixels with a local surface fitted to the pixel values in that region. Thus, a flat-facet model is a piecewise constant representation of an image, and a sloped-facet model is a piecewise planar representation. The facet iteration algorithm is a provably convergent relaxation algorithm that partitions an arbitrary image into facet regions and creates an ideal, noise-cleaned version of the image. It was not designed specifically as an approach to image segmentation, but does segment an image into homogeneous facet regions. The basic concepts of the algorithm are summarized here.

Each pixel in an N x N image that is more than K/2 pixels from the image boundary lies in K^2 different K x K windows. Each of these K^2 windows may be fitted with an approximating polynomial surface of order M. One of these windows will have a minimum fit error for the surface fit. The output value of that pixel is set to the value of the minimum-fit-error surface at that point. The iterative process will eventually converge leaving a noise-cleaned image of facet regions. The borders of the facet regions form a segmentation of the image.

Results are shown in Haralick and Watson [1981] for 3x3 windows applied to a house scene from the University of Massachusetts. It is concluded that the sloped-facet model is a good model for the interior of regions, but not so good for edges. The strengths of such an approach are the following: (1) Contrast between regions is improved in the output image, (2) Textured areas become smoother in the output image, and (3) The algorithm is *parallel*, and hence there is *no order dependence* to the way the regions are defined. Several weaknesses of the approach are listed: (1) Features smaller or thinner than 3x3 windows are degraded, (2) Slanted, high contrast edges become

blocky in the output image, (3) Many, many facet regions are formed, and (4) One must select the window size K and the surface function order M depending on how one wants the algorithm to perform. As stated above, the surface-based facet iteration algorithm was not designed specifically for image segmentation. A later paper by Pong et al. [1984] discusses segmentation results using a facet model region grower based on property vectors. Good results were obtained on a complex aerial scene. Many of the same ideas above are involved in this approach.

Faugeras et al. [1983] have developed a region-growing algorithm for segmenting range data into planar and quadric patches. They state the segmentation problem as follows: Given a set of 3-D points, determine the minimum number of regions of the data points such that each region is approximated by a polynomial of order *at most two* to within some fit error threshold. The region growing algorithm is the following: Find the pair of adjacent regions with the smallest fit error (less than some fit error threshold) for the merged pair. Merge. Repeat until no further merges are possible. Selecting breakpoints for curves and breakcurves for surface is a major problem in shape segmentation. Region merging is prevented across lines of high curvature that are computed first using a separate approach. Attention is also given to the problem that least-squares quadric surface fitting does not yield a fit error that corresponds to the RMS distance of the data points from the fitted surface. Their algorithm is similar to that presented by Dane [1982]. Results are presented for the Renault auto part.

Snyder and Bilbro [1985] discuss the difficulties in processing and segmenting range images. Their segmentation approach is as follows: Determine points of high surface curvature first (e.g. the edges and vertices of polyhedra). Perform connected-component analysis on the remaining points to determine regions where the surface curvature

changes smoothly. Group high-curvature (boundary) points with associated regions. They make the following important qualitative points: (1) Local geometric features of range images are sensitive to quantization noise. (2) Surface normals and surface curvature (mean and Gaussian curvature) are useful for range image segmentation. (3) The chain rule should be used to evaluate spatial derivatives whenever the range image projection is not orthographic. No experimental results were included.

CHAPTER 2

OBJECT RECOGNITION AND SEGMENTATION

Both the ability to organize signal samples into symbolic primitives and the ability to recognize groups of symbolic primitives are necessary for machine perception. Just as a person must identify phonemes to form words to understand a sentence, a computer vision system must identify 3-D surfaces to form 3-D objects to understand a 3-D scene. Hence, surface segmentation and object recognition are useful, fundamental tasks of image understanding systems. Although the immediate goal is to devise a data-driven algorithm that extracts surface primitives from range images without knowledge of higher level objects, a systems approach is taken, and the entire problem is analyzed first before attempting to solve a part of it.

2.1. Three-Dimensional Object Recognition

Three-dimensional (3-D) object recognition is a rather nebulous term. A brief survey of the literature on this subject demonstrates this point [Roberts 1965] [Guzman 1968] [Shirai and Suwa 1971] [Sadjadi and Hall 1980] [Wallace and Wintz 1980] [Ikeuchi 1981] [Douglass 1981] [Brooks 1982] [Casasent et al. 1982] [Fang et al. 1982] [Oshima and Shirai 1983] [Sato and Honda 1983] (see [Besl and Jain 1985]). The different research efforts represented by this list have relatively little in common. Several schemes handle only single, pre-segmented objects while others can interpret multiple object scenes. However, some of these other schemes are really performing two-dimensional (2-D) processing using 3-D information. There are systems that require objects to be placed on a

turntable during the recognition process. A few published methods even require that intermediate data be provided by the person operating the system. Many techniques assume that idealized data will be available from sensors *and* intermediate processors. Others require high-contrast or backlit scenes. Most efforts have limited the class of recognizable objects to polyhedra, spheres, cylinders, cones, generalized cones, or a combination of these. Many papers fail to mention how well the proposed method can recognize objects from a large set of objects that are not customized to individual algorithms (e.g., at least twenty objects). Therefore, a reasonably precise definition of the object recognition problem is given. Human visual capabilities are discussed first and then related to computer vision.

The real world that we see and touch is primarily composed of solid *objects*. When people 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 storing that information is known as *model formation*. Once we are familiar with many objects, they can normally be identified from an *arbitrary viewpoint* without further investigation.

People are also able to *identify, locate, and qualitatively describe the orientation* of objects in black-and-white photographs. This basic capability is significant to computer vision research because it involves the spatial variation of only a *single* parameter within a framed rectangular region corresponding to a fixed, single view of the real world. Human color vision is more difficult to analyze and is typically treated as a *three*-parameter color variation within a large, almost hemispherical solid angle that corresponds to a continually changing viewpoint. For an automatic, computerized recognition process, sensor input data must be compatible with digital computers. The term

digitized sensor data refers to an input matrix of numerical values (which can represent intensity, range, or any other scalar parameter) and associated auxiliary information concerning how that matrix of values was obtained.

The above motivates the following definition of the autonomous single-arbitrary-view 3-D object recognition problem:

(1) Given any collection of labeled solid objects:

(a) Each object may be examined as long as the object is not deformed.

(b) Labeled models may be created using information from this examination.

(2) 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;

Given any data stored previously during the model formation process; and

Given the list of distinguishable objects; the following issues must be addressed for each object using the capabilities of a single autonomous processing unit:

(a) Does the object appear in the digitized sensor data?

(b) If so, how many times does it occur?

(c) For each occurrence:

(i) Determine the location in the sensor data,

(ii) Determine the 3-D location (or translation parameters) of that object with respect to a known coordinate system (if possible with the given sensor), and

(iii) Determine the 3-D orientation (or rotation parameters) with respect to known coordinate system (if possible with the given sensor).

(3) (Optional) If there are regions in the sensor data that do not correspond to any objects in the list, characterize these regions in a way that will make them recognizable should they occur in future images. Presumably, an object is present that is

not known to the system. Ideally, the system should attempt to learn whatever it can about the unknown object from the given view.

The problem of successfully completing these tasks using real world sensor data while obeying the given constraints is referred to as the 3-D object recognition problem. This problem is not successfully addressed in the object recognition systems discussed in the literature; more constrained problems, which are limited to particular surface types or applications, are normally addressed. If the stated 3-D object recognition problem could be solved successfully by a vision 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 stated so that it may be feasible to use computers to solve the problem, and it is also clearly solvable by human beings.

Item (3) above can be interpreted as a partial model-building task to be performed on data that cannot be explained in terms of known objects. It cannot be interpreted as recognition because something is present in the sensor data that the system "knows" nothing about. The vision system is being asked to "learn from experience" in a flexible way.

How does one decide if a given approach solves the problem, and how can different approaches be compared to see if one is better than another? The performance of object recognition systems could be measured using the number and type of errors made by a system in performing the assigned problem tasks on standardized sets of digitized sensor data that challenge the capabilities mentioned in the problem definition. The following list enumerates some of the possible types of errors that are made by such systems:

(1) Miss error: An object's presence is not detected.

(2) False alarm error: The presence of an object is indicated even though there is no evidence of this in the sensor data.

(3) Location error: An object occurrence is correctly identified, but the location of the object is wrong. Location error is a vector quantity with magnitude and direction.

(4) Orientation error: The object occurrence and position are determined correctly, but the orientation is wrong. Orientation error is also a vector quantity with magnitude and direction.

In the comparison of different object recognition systems, the term "successful" can be made quantitative by establishing a performance index that quantitatively combines the number, the type, and the magnitude of the various errors. If a system consists of many different components, its high-level performance depends upon the performance of each component. If each component of the system can achieve low error rates in the tasks they perform, the entire system can achieve good performance. In particular, if excellent segmentation can be achieved using only general information about surfaces, then the high-level matching algorithms of object recognition systems would be much easier to analyze, develop, and test.

2.2. Mathematical Recognition Problem Formulation

It is often beneficial to define a problem in a stricter mathematical form to eliminate possible problem ambiguities. For example, how should a system respond if several distinct objects appear to be identical from a given viewpoint? Therefore, *range-image object recognition* is now redefined in precise mathematical terms as a *generalized inverse set mapping*. Intensity-image object recognition is then briefly treated in the same formalism.

First, world modeling issues must be considered. Let the world be approximated as a set of N_{tot} objects that can exist in different positions and orientations. The number of distinguishable objects is N_{obj}. Hence, $N_{obj} \leq N_{tot}$. (Two objects are not distinguishable if a person cannot tell them apart using only shape cues.) The *i-th* distinguishable object is denoted as A_i. The number of occurrences, or instances, of that object is denoted as N_i. This means that, in general, N_{tot} is the sum of the N_i's for all N_{obj} objects. People can recognize an enormous number of 3-D objects depending on personal experience ($N_{obj} > 50{,}000$ is probably a conservative estimate). The number of objects to be recognized by an object recognition system depends upon the application and system training.

It is sometimes difficult to decide what is an object and what is an assembly of objects. To resolve this difficulty, each object could possess its own coordinate system and list of sub-objects. However, in this discussion, only simple objects with no sub-parts and with only one occurrence (or instance) are considered. Therefore, $N_{obj} = N_{tot}$. The general case of multiple instances of objects with sub-parts is not conceptually different, but it does present notation problems and important implementation difficulties for higher level recognition processing. The origin of the object coordinate system is defined 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 precisely determined for any given solid object or solid object model.

Each object occupies space, and at most one object can occupy any given point in space. It is necessary 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. For reference purposes, a world coordinate system is placed at some con-

venient location. Objects are positioned in space relative to this coordinate system using translation and rotation parameters. The translation parameters of an object are denoted as the vector $\vec{\alpha}$, and the rotation parameters are denoted as the vector $\vec{\theta}$. The number of parameters for each vector depends on the *dimension* of the range-image recognition problem. For example, the 2-D problem requires a total of three parameters. For the *3-D* case, we write the necessary six parameters as follows:

$$\vec{\alpha} = (\alpha,\beta,\gamma) \qquad \text{and} \qquad \vec{\theta} = (\theta,\phi,\psi) \ . \tag{2.1}$$

See Figure 2.1 for the graphical meaning of these parameters. The *world model* W is defined as a set of ordered triples (object, translation, rotation):

$$W = \left\{ (A_i,\vec{\alpha}_i,\vec{\theta}_i) \right\}_{i=0}^{N_{obj}} \ . \tag{2.2}$$

The object A_0 is considered to be the sensor object with position $\vec{\alpha}_0$ and orientation $\vec{\theta}_0$. If a time-varying world model is required, all objects and their parameters can be functions of time. For the current purposes of single-view object recognition, only static

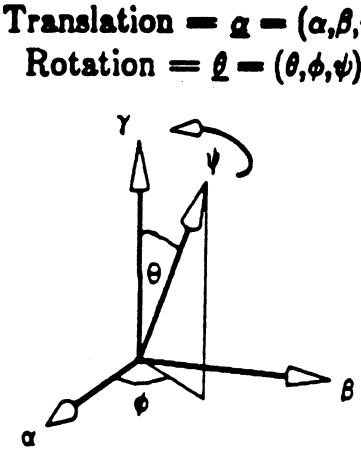


Figure 2.1. Rigid Objects in 3-D Space have 6 Degrees of Freedom

parameter values are of interest. The set of all objects, the *object list*, is denoted as $\mathbf{O} = \{ A_i \}$. The set of all translations is denoted $\mathbf{R}^t$, and the set of all rotations is denoted $\mathbf{R}^r$. In the 3-D object recognition problem, $t=3$ and $r=3$. (In 2-D, $t=2$ and $r=1$.) $\mathbf{R}$ is the set of all real numbers.

A depth sensor (or rangefinder) obtains a depth-map projection of a scene. This projection is modeled as a mathematical operator P that maps elements in the set $\Omega_{t,r} = \mathbf{O} \times \mathbf{R}^t \times \mathbf{R}^r$ into elements in the set of all scalar functions of $t-1$ variables, which is denoted as Ψ_{t-1}:

$$P : \Omega_{t,r} \rightarrow \Psi_{t-1} \tag{2.3}$$

These real-valued functions are called *depth-map functions*. This projection operator might be *orthographic* or *perspective* [Newmann and Sproull 1979] [Foley and van Dam 1982]. For range image projections, different types of perspective transformations are possible (see Appendix D). Any given projection operator may be written as

$$f(\vec{x}) = h_{A,\vec{\alpha},\vec{\theta}}(\vec{x}) = P(A,\vec{\alpha},\vec{\theta}) \tag{2.4}$$

where $\vec{x}$ is the vector of $t-1$ spatial variables of the focal plane of the sensor. The spatial parameters of the sensor object (the location $\vec{\alpha}_0$ and the orientation $\vec{\theta}_0$) are *implicitly* assumed arguments of the projection operator P. Because only one sensor is involved in this formalism, there is no need to be more explicit. The depth-map function is denoted as $f(\cdot)$ when the identity of the object and its parameters are unknown. The symbol $h(\cdot)$ with subscripts refers to the depth-map function of a known object at a known location and orientation. This notation clearly indicates that the set of depth-map functions associated with a single object is an *infinite family of functions*. Two of the rotation parameters in the $\vec{\theta}$ vector have a particularly profound effect on this family

of functions: the 3-D 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 negligible 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 occupy all space, a convention is needed for the value of the depth-map function for values of the spatial vector $\vec{x}$ that do not correspond to object surface points. If the point $(\vec{x}, f(\vec{x}))$ cannot lie on an object surface, the value of $-\infty$ is assigned to $f(\vec{x})$. It is assumed that the value of the depth map function increases as objects move toward the sensor. Hence, the projection of a set of M objects is written as

$$f(\vec{x}) = \max_{1 \le i \le M} h_{A_i, \vec{\alpha}_i, \vec{\theta}_i}(\vec{x}) \tag{2.5}$$

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

These ideas can be written more precisely using inverse set mappings. For each *single object* depth-map function, there is a corresponding inverse set mapping to yield all single objects that could have created the given object depth-map function. The inverse set mapping of P is denoted as the set P^{-1} where

$$P^{-1}\Big(f(\vec{x})\Big) = \Big\{(A,\vec{\alpha},\vec{\theta}) \in \Omega_{t,r} \mid P(A,\vec{\alpha},\vec{\theta}) = f(\vec{x})\Big\} \tag{2.6}$$

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

$$P^{-1} : 2^{\Psi_{t-1}} \rightarrow 2^{\Omega_{t,r}} \tag{2.7}$$

For recognition purposes, the input sets in $2^{\Psi_{t-1}}$ may be restricted to be singletons (i.e., single depth-map functions or range images) with no loss of generality. Therefore, $2^{\Psi_{t-1}}$ is replaced with Ψ_{t-1}. For multiple object depth-map functions, P^{-1} must be generalized due to possible combinations of objects as shown in Figure 2.2. Hence, given $f(\vec{x}) \in \Psi_{t-1}$, a generalized inverse set mapping $\mathbf{P}^{-1}$ is sought such that

$$\mathbf{P}^{-1}\Big(f(\vec{x})\Big) = \Big\{\omega_J \subseteq 2^{\Omega_{t,r}} \mid \max_{j \in J} P(A_j,\vec{\alpha}_j,\vec{\theta}_j) = f(\vec{x})\Big\}. \tag{2.8}$$

Thus, the desired mapping takes elements of the depth-map function space into the power set of the power set of $\Omega_{t,r}$:

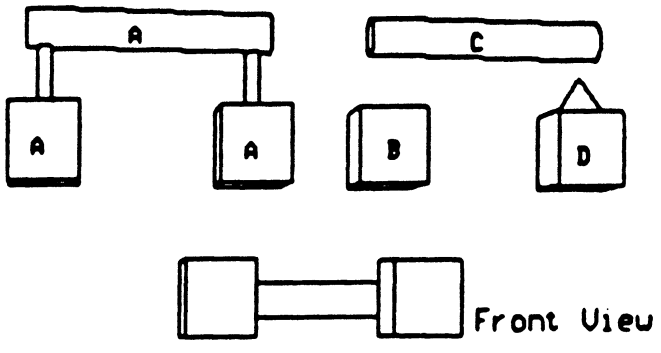


Figure 2.2. Different, Valid Interpretations of a Simple Scene

$$\mathbf{P}^{-1} : \Psi_{t-1} \rightarrow 2^{2^{\Omega_{t,r}}} \quad . \tag{2.9}$$

The range-image 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 $f(\vec{x})$, compute the inverse projection set mapping $\mathbf{P}^{-1}(f(\vec{x}))$ to obtain all possible explanations (subsets of $\Omega_{t,r}$) of the function $f(\vec{x})$ in terms of the world model. In many cases, there is only one valid scene interpretation. Nonetheless, a general-purpose vision system must generate the list of all valid scene interpretations whenever ambiguous single-view situations are encountered. The problem of determining the next best view from which sensor data should be acquired so as to eliminate ambiguity is a separate issue and has been addressed by Kim et al. [1985] and Connolly [1985].

Given the above formalism, the recognition error for a given implementation of the recognition mapping, a given image, and a given interpretation of that image could be expressed as follows. Let $e(\omega_J)$ denote the interpretation error for the interpretation ω_J where J is interpretation index set. Let $\vec{\alpha}_j^{\,a}$ be the actual position of the object in a scene, and let $\vec{\alpha}_j^{\,e}$ be the estimated position. Similarly, let $\vec{\theta}_j^{\,a}$ be the actual orientation of the object in a scene, and let $\vec{\theta}_j^{\,e}$ be the estimated orientation. Let $m(A_j)$ be the number of pixels (or the area) occupied by the object A_j in the image. An *example* of a possible recognition error function might then be

$$e(\omega_J) = \sum_{\substack{j \in J \\ A_j \text{ is} \\ \text{in image}}} m(A_j)(w_t \mid \vec{\alpha}_j^{\,a} - \vec{\alpha}_j^{\,e} \mid_{\mathbf{R}^t} + w_r \mid \vec{\theta}_j^{\,a} - \vec{\theta}_j^{\,e} \mid_{\mathbf{R}^t}) +$$

$$\sum_{\substack{j \in J \\ A_j \text{ not} \\ \text{in image}}} w_f \; m(A_j) + \sum_{\substack{j \notin J \\ A_j \text{ is} \\ \text{in image}}} w_m \; m(A_j)$$

where w_t , w_r , w_f , w_m are the weighting factors for translation errors, rotation errors, false alarm errors, and miss errors respectively. The rotation and translation norms in the expression are not intended to be the same.

Since there is no general theory regarding the computation of the recognition mapping, researchers are free to choose their own methods. It is proposed that the generalized inverse set mapping can best be computed by recognizing the *individual surface regions* associated with the $h_{A,\vec{\alpha},\vec{\theta}}(\vec{x})$ *depth-map surface function families* of the individual objects. That is, it is proposed that the *object recognition* problem be solved by decomposing it into a *surface characterization* and *surface segmentation* problem combined with a *surface matching* algorithm constrained by known object structures. The pixel data in a range image is first characterized to find properties that are useful for grouping or segmentation. Once the image data has been segmented into individual surface regions, these surface descriptions can be used to match the data with surface-based object model descriptions. Although previous work has been done in each of these areas and in complete systems research, more research is required to combine these concepts into a general purpose working solution.

This formalism is now augmented to state the *object recognition problem for intensity-images*. By adding an illumination-reflectance operator λ to the depth-map function f , an intensity-image $\iota(\vec{x})$ is obtained, which is given by

$$\iota(\vec{x}) = \lambda\,(\vec{x},\ \max_i\ P\ (A_i,\vec{\alpha}_i,\vec{\theta}_i\ ))\ . \tag{2.10}$$

The function $\lambda(\cdot)$ requires the spatial vector $\vec{x}$ and the depth at that point $f(\vec{x})$ to determine the surface, the object, and the light sources that are involved in reflecting light back toward the image sensor. The intensity-image object recognition problem is generally much more difficult due to the additional inversion of the λ operator. One

needs *a priori* knowledge of all surface reflectances and all illumination sources of a scene (in addition to object shapes) to be able to invert this λ operator. Note that the *max* computation is the multiple object occlusion operator, and the vector $\vec{x}$ is required to be given to the reflectance and illumination operations that are part of the λ operator. To expand the world model for understanding intensity-images, it is necessary to add objects that generate light in addition to the single sensor object that receives light. Shape from shading [Ikeuchi and Horn 1981], shape from photometric stereo [Woodham 1981][Coleman and Jain 1982], and shape from texture [Witkin 1981] techniques attempt to uniquely invert the λ operator producing the depth-map function f . Note that any type of image that is formed on a point-by-point basis and is dependent on the geometry of the scene may be considered by this formalism.

It is interesting to observe that people have the capacity for understanding images even when shading operators other than the natural illumination-reflectance operator are used to shade visible surfaces. For example, people can correctly interpret photographic negatives, radar images, or pseudo-colored images where the usual color and/or light-dark relationships are completely distorted. Other examples include the range images in this thesis where depth is encoded as brightness.

In summary, a unified model has been developed in which range images, intensity images, and other images can be discussed. When $\lambda(\cdot)$ is the *identity operator* (i.e. $f(\vec{x}) = \lambda(\vec{x}, f(\vec{x}))$, the simplest non-null operator, the model describes the range image understanding problem. When $\lambda(\cdot)$ is more complicated, some other image understanding problem is described. When $\lambda(\cdot)$ is the illumination-reflectance operator, the model yields a formulation for the intensity image understanding problem in terms of scene surfaces. From this point of view, it is argued that the general range image under-

standing problem is the unique, simplest problem in the set of digital surface interpretation problems for physical scenes. Thus, range images are a natural starting point for an analysis of any image understanding problem.

2.3. Recognition System Components

The specific tasks to be performed by an object recognition system are given in the two previous sections. It has also been suggested that one can measure how well these tasks are performed. But how can these tasks be accomplished?

Recognition implies awareness of something already known. How can one recognize something unless one knows what one is looking for? Even though model-formation is not specifically required in the problem definition, it is demanded by the circumstances of the problem. Many different kinds of qualitative and quantitative models, both view-independent and view-dependent, have been used for modeling real world objects for recognition purposes. For now, let us assume that the necessity of model formation has been established and that a representation scheme is required to store object model data. That is, a *world model* is a necessary object recognition system component.

Once one knows what object to look for, how can one find it in the digitized sensor data? In order to determine how recognition will take place, a method for matching the model data to the sensor data must be considered. A straightforward blind-search approach would entail transforming all possible combinations of all possible known object models in all possible distinguishable orientations and all possible distinguishable locations into the digitized sensor data format and computing a matching error quantity to be minimized. The minimum matching error configuration of object models would correspond to the recognized scene. All the tasks mentioned in the statement of the object recognition problem would be accomplished except the characterization of

unknown regions not corresponding to known objects. Of course, this would take an enormous amount of processing time even for the simplest scenes. A better algorithm is needed. In particular, if the pixels in an image can be effectively organized into a small number of pixel groups that will be meaningful to higher-level algorithms, it would be much easier to manipulate this set of groups rather than doing pixel-by-pixel comparisons at every stage in a matching algorithm.

Since object models usually contain more object information than the sensor data, one is prohibited from transforming sensor data into *complete* model data and matching in the model data format. However, this does not prevent matching with *partial* model data. Given the incompleteness of sensor derived models and the natural desire to reduce the large dimensionality of the input sensor data, it is advantageous to work with an intermediate domain that is computable from both the sensor data and the model data. This domain is referred to as the *symbolic scene description domain.* A matching procedure can then be carried out on the quantities in this intermediate domain, which are often referred to as *features.* The best matching results occur when a hypothetical object model configuration accurately represents the real world scene represented in the sensor data. It is proposed that a matching procedure and intermediate symbolic scene description mechanisms are necessary object recognition system components.

Interactions between the individual components of a recognition system are diagrammed in Figure 2.3. The real world, the digitized sensor data domain, the world modeling domain, and the intermediate symbolic scene description are the fundamental domains of the system. Mappings between these domains are listed: The image-formation process (I) creates intensity or range data (or other quantities) based purely on physical principles. The description process (D) acts on the sensor data and extracts

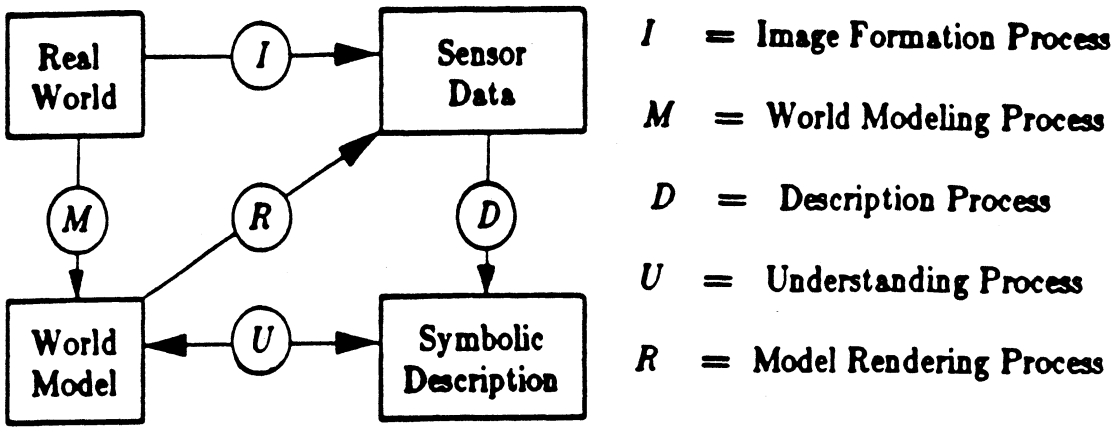


Figure 2.3. General Object Recognition System Structure

relevant, application-independent features. This process should be completely data-driven and incorporate only the knowledge of the image formation process and rudimentary facts about the real world. No *a priori* assumptions about real world object shapes should be incorporated for maximum flexibility. The description process (D) and the symbolic scene description domain are the central topics of this thesis.

The modeling process (M) provides object models for real world objects. Object reconstruction from sensor data is one method for building models automatically. This method is preferred for a fully automated system, but is not required by the problem definition, which addresses visual recognition, not modeling *per se*. Hence, object models could also be constructed manually using geometric solid modeling programs. The understanding, or recognition, process (U) involves an algorithm to perform matching between models and data descriptions. This process might include interacting data-driven and model-driven sub-processes where segmented sensor-data regions seek explanations in terms of models and hypothesized models seek verification from the data description. The rendering process (R) produces synthetic sensor data from object models. A vision system should be able to communicate its understanding of a scene

verbally *and visually* to any person querying the system. Rendering also provides an important feedback link because it allows an autonomous system to check on its own understanding of the sensor data by comparing synthetic images to the sensed images. Synthetically generated (rendered) range images are so similar to real range data that simple image differencing operations may suffice for feedback computations. Even though this thesis is concerned with the data description process, the concepts of rendering and feedback from the original data will play important roles in Chapter 4.

It is proposed that any object recognition system can be discussed within the framework of this system model. This description is basically in agreement with the ideas brought forth by Brooks [1982] except for the addition of the rendering process.

2.4. Capabilities of an Ideal Recognition System

What are the capabilities of an ideal system that handles the object recognition problem as defined? Some that might be realized by object recognition systems in the near future are summarized below.

(1) It must be able to handle sensor data from arbitrary viewing directions without preference to horizontal, vertical, or other directions. This requires a view-independent modeling scheme that is compatible with recognition processing requirements.

(2) It must handle arbitrarily complicated real world objects without preference to curved or planar surfaces or to convex or non-convex objects.

(3) It must handle arbitrary combinations of a large number of objects in arbitrary orientations and locations without being sensitive to superfluous occlusions (i.e., if occlusion does not affect human understanding of a scene, then it should not affect the ideal automated object recognition system either).

(4) It must be able to handle a certain amount of noise in the sensor data without a significant degradation in system performance.

(5) It should be able to analyze scenes quickly and correctly.

(6) It should not be difficult to modify the world model data to handle new objects and new situations.

(7) It is desirable for the system to be able to express its confidence in its own understanding of the sensor data.

It is certainly not implied that any existing computer vision systems have all of these capabilities, nor is it implied that the research necessary to build such a complete system has been completed. The motivation is to describe a general-purpose object recognition system in detail to provide direction for the design of the various components of the system, specifically the data-driven description process.

2.5. Segmentation into Surface Primitives

The focus now shifts from general system-level concerns to the details of the data-driven description process. The input to data-driven description process is a high-dimensionality image (digital surface): a large set of pixels where each pixel has been assigned a value. The physical meaning of those values depends on the type of input image. The output description should satisfy several requirements:

(1) Pixels should be grouped together into a relatively small set of symbolic primitives. The number of primitives should be *much smaller* than the number of pixels, but of course this depends on the complexity of the input image.

(2) The set of symbolic primitives should contain all pixels in the image, but all pixels should belong to one and only one symbolic primitive.

(3) All pixels within a pixel group defined by a symbolic primitive should be consistent with each other with respect to some defining statement about that symbolic primitive.

The proposed data-description approach requires several internal, intermediate levels, where each level uses basic knowledge about surfaces since input images are digital surfaces.

The first step is to characterize the input image (digital surface) at every pixel using a small set of surface types. In order to do this, the local neighborhood of every pixel is examined and characterized by *local differences* in the form of partial derivative estimates and curvature functions. This first step reduces the dimensionality in the value assigned at each pixel: in the range image case, N-bit depth values are transformed into 3-bit surface type values. Chapter 3 goes into complete detail about the relationship between the fundamental shape descriptors of differential geometry and the surface curvature signs used to define surface type. Other valuable surface characteristics can be easily computed during the surface type computation and stored for later use. The connected regions of identical surface type pixels provides the first level of grouping, a rough initial segmentation. It should be noted that increasing the number of accurate bits of depth resolution in images yields better initial segmentations.

The second step takes the original image and the initial surface type segmentation and creates a refined segmentation based on *global similarities* of the pixel values. The shapes present in the digital surface data drive this second stage by selecting surface type shapes from a small but flexible set of surface functions. Chapter 4 describes this process in detail. The refined symbolic surface primitives that are created by the second step have grouped the pixels very effectively, but are constrained by the nature of the

basic surface types isolated by the first step process and by the limitations of the small set of surface functions.

In order to group pixels together from arbitrary, smooth surface primitives, a third step is required to merge adjacent surfaces that appear to belong to the same smooth surface. Chapter 5 describes an algorithm to achieve the required merging. The final output is a segmentation in terms of smooth surface primitives, where complicated smooth surfaces are described as composite patches of simple smooth surfaces.

The implicit assumption here is that most object volumes are bound by relatively smooth surfaces. Although many object surfaces are visibly textured, which may cause extremely large variations in intensity, the actual variations in the relative depth of many textured surfaces is quite small. By isolating the smooth surfaces in a range image of a scene, the scene surfaces are in a symbolic form that is suitable for matching to the object surface data available from the world model. The texture of a real surface is essentially treated as a random noise process superimposed on an underlying smooth surface. At this point, the understanding process (U) takes over to produce the list of object interpretations corresponding to the generalized inverse set mapping formulation discussed earlier.

2.6. Mathematical Segmentation Problem Formulation

The image segmentation problem has received a great deal of attention during the last fifteen years as discussed in Chapter 1. This problem admits a general analytical formulation in which a logical predicate is left unspecified. The logical predicate is a defining statement about the consistency properties of pixels within a region of pixels (a symbolic primitive). In this section, the qualitative description of the segmentation approach (discussed above) is formulated in quantitative terms. For the proposed

approach to digital surface segmentation, the logical predicate can be precisely defined in terms of a spatial consistency predicate known as the *surface coherence predicate.*

Separate spatial regions in an image correspond to different physical entities in a scene. When a set of different objects in a scene undergo a depth-map projection, the bounding surface primitives of each object become partial or complete surface regions in a range image. Image understanding requires the isolation of these surface regions within the range image before the relationships between the regions are established. This surface region isolation problem can be cast in the form of the general image segmentation problem using the surface coherence predicate.

The general segmentation problem is usually stated as follows [Horowitz and Pavlidis 1974][Zucker 1976]. Given the set of image pixels I, find a partition (segmentation) S of the image I in terms of a set of regions R_i. Let N_R be the number of regions in the set, and let $| R_i |$ be the number of pixels in the region R_i.

$$R_i \subseteq I \quad \text{for } \textit{each } i \tag{2.11a}$$

$$\bigcup_{i=1}^{N_R} R_i = I \tag{2.11b}$$

$$R_i \bigcap R_j = \phi \quad \text{for } \textit{all } i \neq j \tag{2.11c}$$

$$R_i \quad \textit{is a 4-connected set of pixels} \tag{2.11d}$$

$$\textit{Predicate } P(R_i) = \textit{TRUE} \quad \text{for } \textit{all } i \tag{2.11e}$$

$$R_i \textit{ adjacent\_to } R_j \Rightarrow P(R_i \bigcup R_j) = \textit{FALSE}. \tag{2.11f}$$

The result of the segmentation process is the list of regions

$$S = \left\{ R_i \right\}_{i=1}^{N_R} \tag{2.12}$$

accompanied by the list of adjacent regions, which can be written as

$$S_A = \left\{ (i_k, j_k) \right\}_{k=1}^{N_A} \tag{2.13}$$

where N_A is the number of adjacency relations and where the pair (i,j) indicates the adjacency of the i th region and the j th region:

$$(i,j) \in S_A \Leftrightarrow R_i \ adjacent\_to \ R_j \ . \tag{2.14}$$

An important parameter of the segmentation output is the smallest region size, denoted $A_{\min}$:

$$A_{\min} \equiv \min_{1 \le i \le N_R} | R_i | \ .$$

The parameter $A_{\min}$ should be specified to avoid degenerate situations where every pixel is its own region.

Although years of research have been focussed on the segmentation problem, no ideal programs exist that work correctly for large classes of images. This has not been an insurmountable problem for many applications though because they have been able to constrain environments such that machine segmentation is achieved quickly and accurately. The general-purpose image segmentation problem is so difficult because high-level semantic knowledge about a scene is usually required in order to obtain accurate segmentation into meaningful image regions.

A specialization of the segmentation problem appropriate for range images is presented below. The predicate $P(\cdot)$, left unspecified in the above problem statement, is given a general mathematical form, known as the surface coherence predicate, which is useful for range image segmentation and may be useful for other types of images. This surface coherence predicate is most appropriate for range images because sampled scene surfaces are directly represented in the digital surface of the range image. Since many

scene surfaces are relatively smooth, surfaces segmented using the surface coherence predicate are likely to be meaningful surfaces. In intensity images, it is quite common for scene surfaces that are relatively smooth to be textured or shadowed in such a way that pixel intensity values corresponding to these surfaces do not yield a coherent digital surface in the intensity image. Rather, the gray levels of an intensity image may fluctuate wildly even though the image represents relatively smooth surfaces. Consider wood grain, brick, concrete, gravel, cork, burlap, carpet, or a matte surface with a complex shadow pattern. It is conjectured that the surface coherence predicate is appropriate for a limited subset of intensity images in which texture and shadows are *not* strongly represented.

Assume that a piecewise-smooth image surface function $g(x,y)$ defined on an image domain I is given. It is hypothesized that this image surface function results from a transformation $\lambda(\cdot)$ of the physical depth-map surface function $f(x,y)$ and is corrupted by additive measurement noise $\mathbf{n}(x,y)$:

$$g(x,y) = \lambda(x, y, f(x,y)) + \mathbf{n}(x,y) \tag{2.15}$$

where as before (Eqn. 2.5)

$$f(x,y) = \max_i h_{A_i,\alpha_i,\theta_i}(x,y) \tag{2.16}$$

If it is assumed that each object volume is bounded by a set of relatively smooth surfaces, then $f(x,y)$ will consist of the depth map projections of these surfaces subject to multiple object occlusion operation. Hence, $f(x,y)$ is piecewise smooth. When surfaces are not exactly smooth, let the texture of the surface be incorporated in the noise function $\mathbf{n}(x,y)$.

The actual pixel values in the digital image are obtained from the hypothesized underlying piecewise smooth surface by the sampling operation $\mathbf{S}_I$ on a grid of image pixels I and then quantizing $\mathbf{Q}$ the function $g(x,y)$ to a certain number of bits to get the digital image $\mathbf{Q}(\mathbf{S}_I(g(x,y)))$. The $\mathbf{S}_I$ operator depends on sensor parameters and the spatial size of the $n_X \times n_Y$ frame buffer (rectangular array) to be filled with image data. The $\mathbf{Q}$ operator depends only on the number of bits N_{bits} that are allotted for each pixel in the buffer. $\mathbf{Q}$ is a representation for the analog-to-digital converter used by the sensor whereas $\mathbf{S}_I$ is a representation of the scanning and sampling mechanisms.

In order to express the necessary concepts mathematically, the characteristic function of an image region is introduced:

$$\chi_{R_i}(x,y) = \begin{cases} 1 & \text{if } (x,y) \in R_i \subseteq I \\ 0 & \textit{otherwise} \end{cases} \tag{2.17}$$

The function $f(x,y)$ can then be expressed as a piecewise smooth surface function

$$f(x,y) = \sum_{i=1}^{N_R} f_i(x,y)\,\chi_{R_i}(x,y) \tag{2.18}$$

where each $f_i(x,y)$ function describes a completely smooth surface with no depth or orientation discontinuities. The properties of the characteristic function and the assumption that the function $\lambda(\cdot)$ depends only upon each point in the image allows us to make the following statement:

$$\begin{aligned} \lambda(x, y, f(x,y)) &= \lambda\left[x, y, \sum_{i=1}^{N_R} f_i(x,y)\,\chi_{R_i}(x,y)\right] \\ &= \sum_{i=1}^{N_R} \lambda(x, y, f_i(x,y))\,\chi_{R_i}(x,y)\,. \end{aligned} \tag{2.19}$$

This yields a modified statement of the original equation (2.15):

$$g(x,y) = \sum_{i=1}^{N_R} \lambda(x, y, f_i(x,y))\, \chi_{R_i}(x,y) + \mathbf{n}(x,y) \qquad (2.20)$$

If all the composite functions $\lambda(x,y,f_i(x,y))$ are not piecewise smooth due to the action of λ, then problems can arise in this formalism. But if λ is sufficiently well-behaved (e.g., if only simple lighting and shadowing of matte surfaces is involved), then the partition $S = \left\{ R_i \right\}$ can be refined so that every composite function $g_i(x,y) = \lambda(x,y,f_i())$ is smooth over the corresponding region.

As will be discussed in Chapter 3, every smooth surface can be segmented into constant surface-curvature-sign surface primitives using the sign of the mean and gaussian curvature. As will be argued in Chapter 4, these surface primitives are approximated very well by low-order polynomial surfaces. For current purposes, it is only necessary to assume that (1) every smooth surface region in an image can be partitioned into a set of surface primitives, where each primitive belongs to one of a small finite set of fundamental surface primitive types, and (2) each fundamental surface primitive type can be approximated very closely by a surface function from a small finite set of approximating functions. Hence, if S' is a segmentation of a piecewise smooth surface function into smooth surface functions, then there exists a refinement S of S' such that the functions $g_i()$ over $R_i \in S$ are well represented by the approximating functions $\hat{g}_i()$.

The digital surface segmentation problem is then stated as follows for images satisfying the surface coherence hypothesis:

- Given only quantized samples of a surface function $g(x,y)$ defined over the image region I that is hypothesized to be of the form

$$g(x,y) = \sum_{i=1}^{N_R} g_i(x,y)\, \chi_{R_i}(x,y) + \mathbf{n}(x,y) \tag{2.21}$$

for some unknown set of smooth surfaces $G = \left\{ g_i \right\}$ and some unknown set of regions $S = \left\{ R_i \right\}$ that satisfy (a) the segmentation conditions (2.11) and (b) the condition that $\min_i | R_i | = A_{\min} > T_R$ = the minimum region size threshold, find a set of approximate regions $\hat{S} = \left\{ \hat{R}_i \right\}$ and a set of approximating functions $\hat{G} = \left\{ \hat{g}_i(x,y) \right\}$ defined over the corresponding regions $\hat{R}_i$ such that the estimated number of regions is small and the approximation error ϵ is small for the function $\hat{g}$ as specified below. The function $\mathbf{n}(x,y)$ is assumed to be a random process (or random field) that provides the measurement errors for an otherwise ideal sensor. It may also provide surface texture for otherwise smooth surfaces. The error metric is given by

$$\epsilon = | \; g - \hat{g} \; |_1 \tag{2.22}$$

where $\hat{g}$ takes the form

$$\hat{g}(x,y) = \sum_{i=1}^{\hat{N}_R} \hat{g}_i(x,y)\, \chi_{\hat{R}_i}(x,y)\,. \tag{2.23}$$

Note that if only the approximation error were required to be small, an algorithm could generate one degenerate surface and one degenerate region for every pixel in an image to yield zero error. Similarly, if only the number of regions were required to be small, an algorithm could fit a single surface to the entire image. An ideal algorithm might segment an image more the way a person would do it by balancing the opposing tendencies to minimize approximation error and region count. It has

been difficult to formulate an exact objective function for minimization that would be effective and allow us to solve the problem analytically. The proposed segmentation approach does not use an objective function of this type, but does provide a function $\hat{g}$ for which both the approximation error and the number of regions are small.

- Note that the surface coherence predicate is represented implicitly by the fact that all pixels p in region R_i have values $z(p)$ that lie sufficiently close to the smooth surface function values $g_i(x(p),y(p))$. Moreover, if R_i and R_j are adjacent regions and if $p \in R_i$, then

$$| z(p) - g_i(x(p),y(p)) | < | z(p) - g_j(x(p),y(p)) | \tag{2.24}$$

for almost every pixel in R_i. The term "almost every" is used because it is possible for g_i to intersect g_j as shown in the example in Figure 2.4. The intersection set should always be a set of measure zero (i.e., a set with no area).

The key problem in computing this decomposition is that *both the region set and the approximating function set* are unknown. Nonetheless, such a decomposition, if realized, would be useful for image understanding purposes. Not only is the image segmented according to the predicate that the pixels in a region belong to a particular smooth surface, an entire image reconstruction is generated that looks quite like the original image, and the 3-D shape of the individual surface regions are explicitly approximated. This 3-D shape extraction will be extremely useful to a wide variety of higher-level systems. Systems for segmenting digital surfaces following this general approach have already been built based on planar approximating functions [Faugeras 1984][Bhanu 1984][Duda et al. 1979][Herman 1985] [Haralick and Watson 1981][Milgram and Bjorklund 1980] and quadric (often specifically conic) approximating surfaces of volumes [Nevatia and Binford

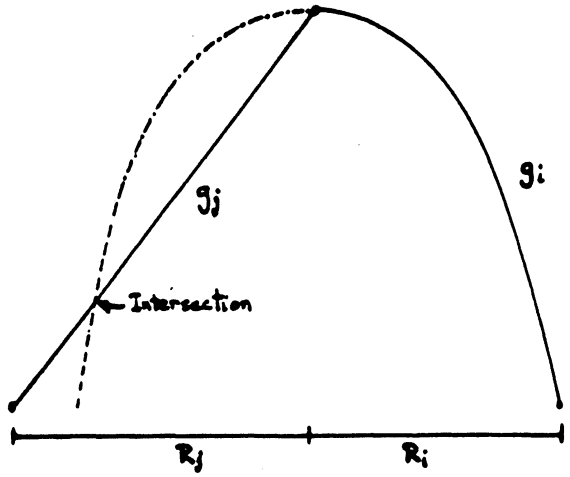


Figure 2.4. Possible Intersection of Adjacent Surface Regions

1977][Kuan and Drazovich 1984] [Smith and Kanade 1984][Tomita and Kanade 1984][Gennery 1979]. However, none of the systems discussed in the literature appear to be able to handle arbitrary combinations of flat and curved surfaces on non-convex objects in a theoretically unified manner. The goal of this surface-based segmentation algorithm is to compute meaningful, accurate segmentations and noise-free reconstructions of arbitrary digital surfaces without imposing limited models of planar or particular smooth curved surfaces.

Contrast the decomposition mentioned above with 2-D Fourier analysis where the set of orthogonal basis functions defined over the *entire* 2-D plane is known prior to analysis and the decomposition problem is reduced to computing an integral (or sum) to obtain the basis function coefficients. In Fourier techniques, the spatial surface function is converted to a spatial-frequency surface function for frequency domain analysis. This conversion can be performed regardless of the spatial coherence of the original surface function. A useful, region-based decomposition above can be obtained only under the

coherent surface assumption. But rather than infinite support spatial-frequency basis functions, compact-support geometric shape functions are advocated. All the capabilities of frequency-based signal processing techniques on arbitrary signals are acknowledged, but when the information content of a signal is explicit geometry, it appears that a geometry-based signal processing approach may be very useful. Frequency-based analysis allows simple compression of data if certain frequency bands can be ignored. However, an accurate description of surfaces in a scene requires lower spatial frequencies for surface interiors and higher spatial frequencies for surface boundaries, which limits the potential of this type of data compression. Moreover, spatial frequency information is useful for interpretation only in constrained circumstances. In geometry-based signal processing, coherent surface data is compressed into functional parameters and region descriptions, which can be directly useful for interpretation purposes.

Note that I is a discrete set of pixel locations. Using the Euclidean error metric, equation (2.22) takes the form:

$$\epsilon = \frac{1}{|I|} \sum_{I} \Big(\mathbf{Q}(\mathbf{S}_I(g(x,y))) - \mathbf{S}_I(\hat{g}(x,y)) \Big)^2. \tag{2.25}$$

where $|I|$ is the number of pixels in the set I and $\mathbf{S}_I$ is the operator that samples the surface function at the individual pixels of the image pixel set I. If the regions can be estimated correctly so that there are no region segmentation errors ($R_i = \hat{R}_i$ for all i from 1 to $\hat{N}_R = N_R$), then the Euclidean error metric would become the weighted sum of errors for each region:

$$\epsilon = \frac{1}{|I|} \sum_{i=1}^{N_R} \epsilon_i \, |R_i| \qquad (if \;\; R_i = \hat{R}_i) \tag{2.26}$$

where

$$\epsilon_i = \frac{1}{|R_i|} \sum_{R_i} \Big(\mathbf{Q}(\mathbf{S}_I(g_i(x,y))) - \mathbf{S}_I(\hat{g}_i(x,y)) \Big)^2 \qquad (2.27)$$

This *special case* decomposition of the error metric is only allowed if the estimated regions and the actual regions are identical (i.e., if the region segmentation errors are zero). In general, the image error ϵ will consist of contributions from *region segmentation errors and approximation errors*. Since the proposed method allows for arbitrary four-connected regions, but allows only a finite set of approximating functions, it is possible for the image error to consist entirely of approximation errors as in equation (2.26).

The error metric formulation may seem slightly asymmetric in terms of the actual vs. approximate functions. It is certainly permissible to add an additional quantization operator to the samples of the approximating functions. However, least-squares and most other approximation techniques do not minimize the error metric with the quantization operator $\mathbf{Q}$ applied to the approximating functions. Therefore, the above error metric is correctly stated.

2.7. Range Image Segmentation for Recognition

When range images are used, the λ operator in the formalism above can be dropped because it is the identity operator. Hence, this implies that the smooth surface functions $g_i(x,y)$ are exactly equal to the smooth depth-map primitives $f_i(x,y)$ that result from the decomposition of the depth map $f(x,y)$ as mentioned above in Equation (2.18).

The proposed approach takes the function $g(x,y)$ and creates an intermediate image $sgn_{HK}(x,y)$, which is called the HK-sign map. Each pixel in this intermediate image contains a surface type label. The HK-sign map is then used to find seed-region subsets $\hat{R}_i^{(0)}$ of the actual regions R_i. These seed regions are then combined with a

small set of approximating functions and an iterative, surface-fitting, region-growing method to create the refined region descriptions $\hat{R}_i$ and the approximate surface primitives $\hat{g}_i$. Since a smooth surface may consist of several visible-invariant, HK-sign surface primitives as discussed in Chapter 3, it is necessary to do a final merging operation on the HK-Sign surface primitives to create a final set of composite smooth surface regions that correspond to the natural smooth surface primitives in the scene.

This digital surface segmentation, which is the symbolic scene description created by the description process, is useful to the range image object recognition problem. An example of how this information might be used is given below.

Each smooth surface primitive region can be associated with at least one surface on at least one object in the object list or else it is a new surface shape that has never been seen before. If the surface g_i over the region R_i can be associated with the object A_k, then the object model is placed in a set of compatible objects $O_i = \left\{ A_k \right\} \subseteq \mathbf{O}$, which are said to be potential owners of the surface primitive from the data. Hence, there exists an ownership function $O(\cdot)$ that maps data surface regions to sets of object models based on the compatibility of model surfaces:

$$O_i = \left\{ A_k \right\} = O(R_i) \,. \tag{2.28}$$

For example, a sphere model can never be the owner of a flat data surface, and a simple polyhedral model can never be the owner of a spherical data surface. Thus, curved-to-flat and flat-to-curved ownership hypotheses could be quickly dismissed for surfaces with significant curvature. Assuming scale factors are known, a small flat model surface bounded by a hexagon could never be the owner of a large flat data surface bounded by

a quadrilateral. The converse is not true because a large flat model surface might own a small flat data surface owing to occlusion. Nevertheless, flat-to-flat ownership hypotheses might also be handled rather quickly since 3-D planar figures can be aligned rotationally very easily and matched with subsequent 2-D computations.

A more difficult problem is the curved-to-curved ownership hypothesis evaluation problem. As discussed in Chapter 3, let $\vec{x}_k(\eta,\xi)$ be a parametric representation of an arbitrary curved surface bounding the object A_k. Let $g_i(u,v)$ be the parametric representation of the surface region R_i produced by the proposed segmentation algorithm. Let $d_{ki}(\vec{\alpha},\vec{\theta})$ be the matching decision parameter for the surface $\vec{x}_k$ in the position $\vec{\alpha}$ and the orientation $\vec{\theta}$ and the given i-th data surface region. Let $\mathbf{U}_{\vec{\theta}}$ be the 3x3 rotation matrix associated with the rotation vector $\vec{\theta}$. The curved-model-surface to curved-data-surface matching decision parameter could be expressed as the integral over the segmented region of the squared difference of the *rotated, translated, reparameterized, projected* model surface and the data surface provided by the segmentation algorithm:

$$d_{ki}(\vec{\alpha},\vec{\theta}) = \frac{1}{|R_i|}\int_{R_i}(P(\mathbf{U}_{\vec{\theta}}\,\vec{x}_k(\eta(u,v),\xi(u,v)) + \vec{\alpha}) - g_i(u,v))^2 du dv \quad (2.29)$$

where $P(\cdot)$ is the depth map projection operator that extracts the z coordinate from the transformed model surface parameterization. This matching decision parameter must be minimized over all possible rotations, translations, and reparameterizations. These computations have not yet been implemented. If the minimized matching decision parameter is small enough, then the model surface is a possible owner of the data surface. This is a general statement of the model-to-data surface ownership evaluation process for arbitrary surfaces.

Surface ownership computations are just one part of the general matching computations that will be needed. The adjacency relationships of model surfaces and data surfaces will also play an important role in a general-purpose object recognition algorithm. The proposed segmentation algorithm (as well as many other approaches) can provide useful surface boundary labels: occluding, occluded, convex, and concave. Other special features, such as model surface critical points (local extrema) and data surface critical points, may also play an important role in surface matching and object recognition.

The previously mentioned understanding process might then be viewed as a hypothesis testing algorithm where the hypotheses take the form of possible ownership functions, possible adjacency relationships, and possible special feature instances. If the surface matching algorithm is robust enough, surfaces will be matched by their shape, *not only by their boundaries.* If this is possible, there is real hope of successfully handling occlusion in range images.

If there are N_R regions in an image segmentation and N_S possible object surfaces in the world model, this could potentially require testing $N_R\,N_S$ ownership hypotheses. It appears that feature-indexed methods can be developed that do not require so many hypothesis tests (see [Knoll and Jain 1985]). Objects might be identified by intersecting different ownership lists from different regions when the ownership lists contain similar objects. The different regions would normally be adjacent to each other, but this would not be required for occlusion handling purposes. The above speculations are intended as an indication of how the results of the segmentation might be used for recognition tasks.

In the previous sections of this chapter, it was mentioned how object recognition performance might be measured in terms of several different types of *recognition errors.* Just as object recognition systems can commit high-level recognition errors, segmenta-

tion processes can commit several different types of low-level *segmentation errors*. In a complete system evaluation, it will be important to separate recognition errors into those errors caused by high-level matching difficulties and those caused by low-level segmentation difficulties. Several range image segmentation errors are listed below:

(1) Region Merge Error: Two regions are merged even though a depth or orientation discontinuity exists between the regions. A new region boundary must be inserted to correct the error.

(2) Failure-to-Merge Error: Two regions that should have been merged are not merged and are separated by a meaningless boundary. A region boundary must be deleted to correct the error.

(3) Boundary Error: Most of a region is correctly segmented, but the boundary of the region is not correctly positioned. A region boundary must be substituted to correct the error.

(4) Same-Region-Label Error: Two non-adjacent regions are given the same label because of their compatibility even though they should not be grouped together. A new region label is needed.

(5) Different-Region-Label Error: Two non-adjacent regions are given different labels because of their non-compatibility even though they should be grouped together. One label must be substituted for another to correct the error.

These errors require high-level knowledge in order to evaluate and correct them because, if they could be recognized using low-level mechanisms, those mechanisms could internally correct the segmentation results. The number of errors, the length of region boundaries, the area of the involved regions, and the size of the differences in the regions involved in those errors will be useful in establishing a segmentation error metric that

could be used to compare different segmentation techniques. It seems possible that a generalization of the Levenshtein metric for two bit strings might be useful for a segmentation error metric: "What is the minimum number of region relabelings and insertions, deletions, and substitutions of region boundaries needed to derive one segmentation from the other?" Region boundary modifications would need to be weighted by the length of the region boundary for insertions and deletions and the magnitude of the boundary modification for substitutions. The concept of distance between two regions and the norm of a region can be reasonably defined as in Appendix E and may be useful to segmentation error metrics. Levine and Nazif [1985] have developed a general-purpose low-level performance measurement scheme for image segmentation algorithms based on uniformity and contrast measures of regions and lines. Whatever methods are used, a good analysis of segmentation and matching errors will be important to quantitative vision system evaluation methods.

2.8. Chapter Summary

By examining object recognition requirements first, a system-level approach has been used for specifying the requirements of a particular vision system component, the data-driven description process, which accepts a digital surface, such as a range image, as input and produces a segmentation of that surface, which may be considered a symbolic scene representation. Such a process is critical to object recognition efforts and should be considered within the appropriate context.

The term *digital surface segmentation* as used in this thesis implies that the following items have been computed: (1) a list of 2-D image regions where each region is described explicitly, (2) a list of approximating surface functions, one for each region, and (3) the approximation error(s) for each region as compared to the original digital

surface data. This fit error information should be included as an indication of the confidence in the surface relative to the mean image noise. For human interpretation of segmentation results, the following items should also be computed: (4) the reconstructed digital surface showing the algorithm's approximation of the image, and (5) a segmentation plot or region label image showing clearly the partitioned regions. This set of outputs is more than what is required from most segmentation algorithms.

CHAPTER 3

SURFACE CURVATURE CHARACTERISTICS

Computing surface characteristics for pixel grouping purposes is the first step toward the final goals of object recognition and range image understanding. The range image object recognition problem is being addressed with the aim of developing a *general purpose* approach that handles arbitrary surface shapes and arbitrary viewing directions. To handle the problem of arbitrary viewing directions, view-invariant surface characteristics are needed that are robust enough to describe both polyhedra and objects with arbitrary curved surfaces. Note that this statement assumes the validity of *invariant features hypothesis* [Palmer 1983], which states that "shape perception is mediated by the detection of those geometrical properties of a figure that do not change when the figure is transformed in different ways." This notion of invariance to viewpoint changes is extremely important and is often glossed over in the 3-D vision literature. Therefore, this topic is discussed in more detail.

The term "invariant" refers to a quantity that does not change under a specified group of transformations. For example, the length of a 3-D line segment does not change under the group of pure rotation transformations, and is therefore said to be rotationally invariant, but note that the length of a 3-D line segment as projected in a 2-D image plane does change under rotation and is not invariant. Opaque, rigid, physical objects do not, in general, possess explicit surface or edge features that are visible

from any viewing angle. There are almost always degenerate viewing angles in which visible object features are radically different. Consider, for example, an object as simple as a cylinder. A flat planar surface with a circular boundary is visible when looking down the axis of a cylinder. On the other hand, a curved surface with a rectangular projected boundary is visible when looking perpendicular to the axis direction. See Figure 3.1 for the two views under consideration. There are *no explicit* invariant features even in this simple case. (For example, the minimum projected silhouette area is not considered as an explicit feature.) The *roundness* of the cylinder manifests itself in both views, but in different ways. In the first view, an explicit *circular arc* depth-discontinuity boundary surrounding a flat region is visible while, in the second view, a constant-negative-mean-curvature, zero-Gaussian-curvature surface bounded by a projected rectangular depth-discontinuity boundary is visible. (The curvature definitions are given later in this chapter.) The computer vision literature often uses the term "invariant" to mean "invariant if visible." The term "visible-invariant" is used here to be more specific. A *visible-invariant surface characteristic* is a quantitative feature of a visible surface region that does not change under the group of viewpoint transformations

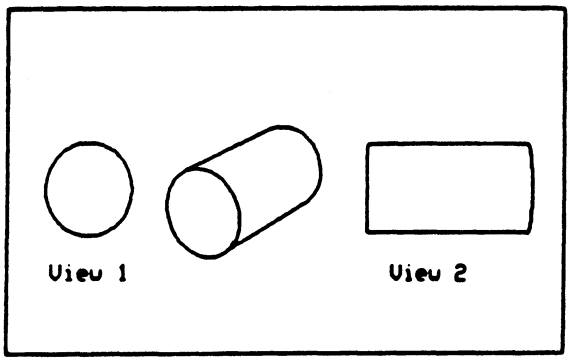


Figure 3.1. Two Views of a Cylinder with No Common Features

that do not affect the visibility of that region. Hence, the subset of the group of transformations that leaves all portions of a particular surface region visible will depend on the shape of that surface region. Since it would not be worthwhile to specify separate sets of transformations for each surface region of interest, surface characteristics that are invariant to all rotation and translation transformations (the Euclidean rigid motion group) are sought. It is equally as important that surface characteristics are invariant to changes in the parameterization of a surface. When a visible, smooth, curved surface is sampled on a rectangular image grid from two different viewpoints, the effective parameterizations of the surface in the corresponding range images are completely different. Hence, numerical visible-invariant surface quantities must also be invariant to changes in surface parameterization. The key point of these statements is that a general-purpose vision system should use visible-invariant features, but must also be aware that salient object features may not be visible even when an object is present and visible in the field of view. Systems based on generalized cylinders, for example, seem to assume that the roughly parallel curves or lines delineating the outline of a generalized cylinder will *always* be visible [Brooks 1981]. General systems should be able to handle degenerate alignment cases.

The visible-invariant surface characteristics chosen for this work are the *mean curvature (H)* and the *Gaussian curvature (K)*, which are referred to collectively as *surface curvature*. When a surface region is visible, its surface curvature is invariant not only to *translations and rotations* of the object surfaces, but also to *changes in surface parameterization* (see Appendix A) under the orthographic depth-map projection. In addition, mean curvature is an *extrinsic* surface property whereas Gaussian curvature is *intrinsic*. These terms are discussed later, but are mentioned now to indicate their complementary nature. Differential geometry emphasizes that these are quite reasonable surface features

to consider.

Since one seldom obtains perfect sensor data from the real world, it is desirable to compute a rich characterization of the surface that preserves the surface structure information and is insensitive to noise. Noise insensitivity might be achieved by computing redundant, or at least overlapping, information about a surface. In order to have a very rich geometric low-level representation, surface curvature characteristics may be combined with surface critical points (local maxima, minima and saddle points), large metric determinant points (depth-discontinuities), and other miscellaneous surface properties to characterize a digital surface in more detail. These provide useful complementary information and can be computed for a small additional cost once surface curvature is already being computed. The main thrust of this thesis is that, given the surface characterization of a range image, it is possible to segment that range image into meaning surface primitives based on the computed surface characteristics and the original range image. Range images are so explicit that it is not necessary to incorporate domain-specific high-level information, such as a particular world model, in order to analyze and segment range images into meaningful surfaces.

3.1. Differential Geometry Review

In the previous chapter, it was discussed how object recognition in range images might be decomposed into a surface recognition problem, which is in turn decomposed into a surface characterization problem and a surface segmentation problem based on those characteristics. And it is established that visible-invariant surface characteristics should be invariant to rotations and translations and surface parameterization changes so that they will be useful for 3-D object recognition.

But the term "characteristic" should be defined first before proceeding. A *characteristic* of a mathematical entity, such as a surface function, is defined to be any well-defined feature that can be used to distinguish between different mathematical entities of the same type. One may consider ideal characteristics that *uniquely* determine a corresponding entity, or characteristics that are many-to-one (although the former are more desirable). A simple example of a characteristic that uniquely determines a function is a detailed description of the function itself. Another simple example of a many-to-one characteristic is roundness. A circle and an ellipse are both *round* closed curves. This round characteristic distinguishes them from rectangles, triangles, and other polygons, but it does not distinguish between circles and ellipses. In this chapter, the aim is to find a good mathematical characterization of depth-map function surfaces. Differential geometry is used to motivate and justify the final choice of surface characteristics, and for this reason, the basic concepts of differential geometry are reviewed, beginning with curves.

3.1.1. Space Curves

Before surfaces are treated, the corresponding theory for curves is reviewed to help provide a qualitative understanding of and a conceptual foundation for the more complicated surface concepts. It is well known that *curvature, torsion, and speed* uniquely determine the shape of 3-D space curves [Faux and Pratt 1979][Hsiung 1981][Lipschutz 1969][O'Neill 1966]. General curves can be represented parametrically as a function of an interval of the real line. A curve C is written as follows:

$$C = \left\{ \vec{x}(s) : a \leq s \leq b \right\}. \tag{3.1}$$

where $\vec{x}(s)$ is a vector function of a scalar argument s which is *not* assumed to be arc

length. Only *smooth* curves are considered where the components of the $\vec{x}$ vector have continuous second derivatives. The tangent unit-vector function $\vec{t}(s)$ is defined as

$$\vec{t}(s) = \frac{d\vec{x}(s)/ds}{|\, d\vec{x}(s)/ds \,|} \quad . \tag{3.2}$$

The normal unit-vector function $\vec{n}(s)$ is defined as

$$\vec{n}(s) = \frac{d\vec{t}(s)/ds}{|\, d\vec{t}(s)/ds \,|} \quad . \tag{3.3}$$

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

$$\vec{b}(s) = \vec{t}(s) \times \vec{n}(s) \, . \tag{3.4}$$

This coordinate system is shown in Figure 3.2. The *speed* of a curve C is defined as

$$\nu(s) = |\, d\vec{x}(s)/ds \,| \, . \tag{3.5}$$

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

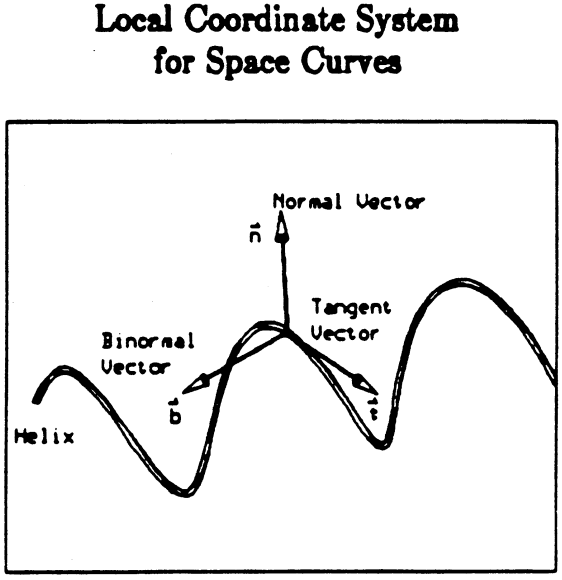


Figure 3.2. Tangent-Normal-Binormal Coordinate System on a Curve

$$\kappa(s) = \frac{\vec{n}(s) \cdot d\vec{t}(s)/ds}{\nu(s)} \quad . \tag{3.6}$$

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

$$\tau(s) = -\frac{\vec{n}(s) \cdot d\vec{b}(s)/ds}{\nu(s)} \quad . \tag{3.7}$$

All of the main concepts associated with space curves have now been defined.

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

Of course, all these questions have been answered in classical mathematics. From the definition information above, it is possible to formulate the Frenet-Serret equations. These equations are written here as a single differential matrix equation:

$$\frac{d}{ds}\begin{bmatrix} \vec{t}(s) \\ \vec{n}(s) \\ \vec{b}(s) \end{bmatrix} = \begin{bmatrix} 0 & \kappa(s)\nu(s) & 0 \\ -\kappa(s)\nu(s) & 0 & \tau(s)\nu(s) \\ 0 & -\tau(s)\nu(s) & 0 \end{bmatrix} \begin{bmatrix} \vec{t}(s) \\ \vec{n}(s) \\ \vec{b}(s) \end{bmatrix} \quad . \tag{3.8}$$

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 this first-order linear space-varying matrix ordinary differential equation were integrated over the entire length of the curve, one would only obtain what the coordinate frame does along the curve, not the curve itself. It is necessary 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

$$\vec{x}(s) = \int_0^s \vec{t}(s)\nu(s)\, ds + \vec{x}(0) \qquad \vec{x}(s) \in \mathbf{R}^3 \tag{3.9}$$

Thus, the curve C can be decomposed into different functional components with different invariance properties: $\vec{x}(0)$ is the starting point vector of three numbers that determine the starting point of the curve. $\vec{t}(0)$ is the initial tangent vector, which can be completely determined by two numbers (it is a unit vector). $\vec{n}(0)$ is the initial normal vector, which can be completely determined by one additional number. $\vec{b}(0)$ is the initial binormal vector, which is completely determined by the specification of the initial tangent and normal vectors. $\nu(s)$ is a normalization function that accounts for the 1-D intrinsic geometry of the given curve parameterization. In fact, if s is arc length along the curve, the speed is always one and thus may be dropped from all equations. $\kappa(s)$ is the scalar function that determines the instantaneous radius of curvature at each point on the curve. $\tau(s)$ is the scalar function that determines the instantaneous bending of the curve out of the tangent-normal plane.

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

Moreover, there is a fundamental existence and uniqueness theorem for 3-D space curves that 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

$$\kappa(s) = \kappa^*(s) > 0, \quad \tau(s) = \tau^*(s), \quad \nu(s) = \nu^*(s) > 0, \tag{3.10}$$

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 completely captured by three scalar functions: *curvature, torsion, and speed.* The constraint of unit speed is often 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.

3.1.2. Plane Curve

Before moving on to surfaces, we

case of planar curves (1-D manifolds in 2-D space) is examined due to the nature of its simplifications as compared to general space curves (1-D manifolds in 3-D space). Not surprisingly, most non-moment-based 2-D shape description research makes use of the

ideas covered here, which helps to motivate the proposed method of 3-D shape description. In addition, the contrast between the 2-D and 3-D shape recognition problems and 2-D and 3-D shape characteristics will be enhanced. The sign-of-curvature paradigm, developed in Chapter 4, is applicable to planar curves, surfaces, and higher dimensions where n-D entities are embedded in (n+1)-D space. In Chapter 6, the curvature of 2-D region boundaries (edges), which are planar curves, is used to characterize planar curve shape so that the region boundary can be segmented into linear and curved intervals.

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:

$$\frac{d}{ds}\begin{bmatrix}\vec{t}\\ \vec{n}\end{bmatrix} = \begin{bmatrix}0 & \kappa(s)\nu(s)\\ -\kappa(s)\nu(s) & 0\end{bmatrix}\begin{bmatrix}\vec{t}\\ \vec{n}\end{bmatrix}. \tag{3.11}$$

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:

$$x(s) = x(0) + \int_0^s \cos\left[\phi(0) + \int_0^\beta \kappa(\alpha)d\alpha\right] d\beta \tag{3.12}$$

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

where $\phi(0)$ is the initial tangent angle and $(x(0),y(0))$ is the starting point

Note that, for unit-speed curves,

$$\kappa(s) = \frac{d}{ds}\tan^{-1}\left[\frac{dy/ds}{dx/ds}\right]. \tag{3.13}$$

It is seen that arbitrary smooth planar curve shape is captured by two scalar functions: the *speed function* $v(s)$ and the *curvature* function $\kappa(s)$. Surface shape is captured by two almost exactly analogous 2x2 matrix functions: the *metric* and the *shape operator*, which generalize speed and curvature 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 $(x(s),y(s))$. Previous research in *partial* 2-D shape recognition has used either the tangent angle function (e.g. [Turney et al. 1985]) or the curvature function (e.g. [Grogan 1983]).

3.1.3. Surfaces

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. Surface characteristics with similar properties are now discussed.

The explicit parametric form of a general surface S with respect to a known coordinate system may be written as follows:

$$S = \left\{ \begin{bmatrix} x \\ y \\ z \end{bmatrix} : \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} d(u,v) \\ e(u,v) \\ f(u,v) \end{bmatrix}, (u,v) \in D \subseteq \mathbf{R}^2 \right\} \tag{3.14}$$

This general parametric representation is referred to as $\vec{x}(u,v)$ where the x-component of the $\vec{x}$ function is $d(u,v)$, the y-component of $\vec{x}$ is $e(u,v)$, and the z-component is $f(u,v)$. In a later section, the graph surface (Monge Patch surface) form is used to describe depth-map surface functions represented in range images. In the graph surface case, $d(u,v) = u$ and $e(u,v) = v$, which are extremely simple functions. Only *smooth* surfaces where all three parametric functions possess continuous second partial

derivatives are considered in this analysis.

There are two basic mathematical entities that are considered in the classical analysis of smooth surfaces. They are referred to as the first and second fundamental forms of a surface [Hsiung 1981][Lipschutz 1969]. Modern mathematics favors an equivalent formulation of this knowledge in terms of the metric tensor and the Weingarten mapping (the "shape" operator) [O'Neill 1966]. It is pointed out subsequently how complete knowledge of these forms uniquely characterizes and quantifies general smooth surface shape. The surface review begins by defining the fundamental forms of a surface in terms of the general explicit surface parameterization $\vec{x}(u,v)$.

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

$$\begin{aligned} I(u,v,du,dv) &= d\vec{x} \cdot d\vec{x} = Edu^2 + 2Fdudv + Gdv^2 \\ &= \begin{bmatrix} du & dv \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} \\ &= d\vec{u}^T \, [\mathbf{g}] \, d\vec{u} \end{aligned} \tag{3.15}$$

where the [g] matrix elements are defined to be

$$\begin{aligned} g_{11} = E = \vec{x}_u \cdot \vec{x}_u \qquad g_{22} = G = \vec{x}_v \cdot \vec{x}_v \\ g_{12} = g_{21} = F = \vec{x}_u \cdot \vec{x}_v \end{aligned} \tag{3.16}$$

where the subscripts denote partial differentiation

$$\vec{x}_u(u,v) = \frac{\partial \vec{x}}{\partial u} \qquad \vec{x}_v(u,v) = \frac{\partial \vec{x}}{\partial v} \quad . \tag{3.17}$$

$\vec{x}_u$ and $\vec{x}_v$ are referred to as the *u-tangent vector* and the *v-tangent vector* functions respectively, and they may or may not be orthogonal to each other. These two tangent vectors are shown in Figure 3.3 and are said to lie in (and form a basis for) the tangent

plane $T(u,v)$ of the surface at the point $\vec{x}(u,v)$. The [**g**] matrix is referred to as the first fundamental form matrix or as the *metric* (or metric tensor) of the surface. Since the vector dot product is commutative, this [**g**] matrix is symmetric and only has three independent components. The E,F,G notation of Gauss is used along with the matrix element subscript notation because both are useful in different circumstances, and both occur often in the literature of differential geometry.

The first fundamental form $I(u,v,du,dv)$ measures the small amount of movement $|\ d\vec{x}\ |^2$ on the surface at a point (u,v) for a given small movement in the parameter plane (du,dv) as shown in Figure 3.3. This function is invariant to surface

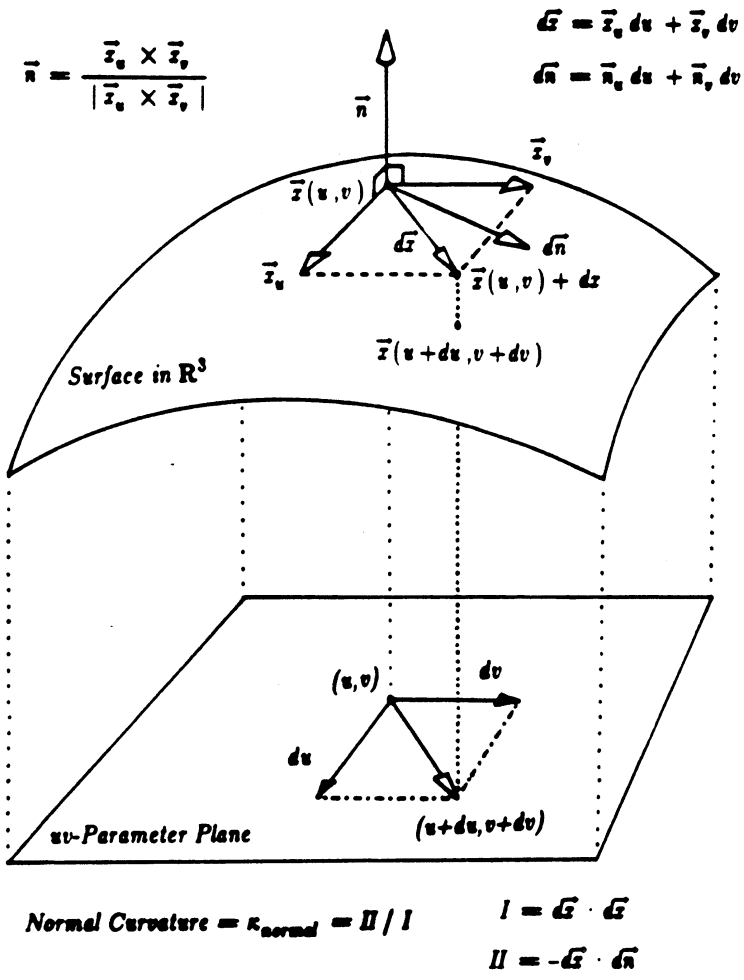


Figure 3.3 Local Coordinate Frame at Surface Point

parameterization changes and to translations and rotations of the surface. The first fundamental form (or the metric) depends only on the surface itself, and not on how the surface is embedded in 3-D space. Such properties are therefore referred to as *intrinsic* properties of a surface. In fact, the metric functions E,F,G determine all intrinsic properties of a surface. The *metric* 2x2 matrix function plays the same role as the scalar *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 of a surface is dependent on the embedding of the surface in 3-D space and is therefore considered as an *extrinsic* property of the surface. The second fundamental form II is given by

$$\begin{aligned} II\,(u,v,du,dv) &= -\,d\vec{x}\cdot d\vec{n} = Ldu^2 + 2Mdudv + Ndv^2 \\ &= \begin{bmatrix} du & dv \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} \\ &= d\vec{u}^{\,T}\,[\mathbf{b}]\,d\vec{u} \end{aligned} \tag{3.18}$$

where the [b] matrix elements are defined to be

$$\begin{aligned} b_{11} &= L = \vec{x}_{uu}\cdot\vec{n} \qquad b_{22} = N = \vec{x}_{vv}\cdot\vec{n} \\ b_{12} &= b_{21} = M = \vec{x}_{uv}\cdot\vec{n} \end{aligned} \tag{3.19}$$

where

$$\vec{n}(u,v) = \frac{\vec{x}_u \times \vec{x}_v}{|\,\vec{x}_u \times \vec{x}_v\,|} = \textit{Unit Normal Vector} \tag{3.20}$$

and where the double subscript implies second partial derivatives

$$\begin{aligned} \vec{x}_{uu}(u,v) &= \frac{\partial^2\vec{x}}{\partial u^2} \qquad \vec{x}_{vv}(u,v) = \frac{\partial^2\vec{x}}{\partial v^2} \\ \vec{x}_{uv}(u,v) &= \frac{\partial^2\vec{x}}{\partial u\,\partial v} = \vec{x}_{vu}(u,v)\,. \end{aligned} \tag{3.21}$$

The [b] matrix is the second fundamental form matrix and is also symmetric if the

surface is well-behaved (i.e., if the mixed partial derivatives are equal). The Gauss-like L,M,N notation is introduced again as above. These definitions allow us to discuss the "state" equation for surfaces.

The second fundamental form measures the correlation between the change in the normal vector $d\vec{n}$ and the change in the surface position $d\vec{x}$ at a surface point (u,v) as a function of a small movement (du,dv) in the parameter space. This is also indicated in Figure 3.3. The differential normal vector $d\vec{n}$ always lies in the tangent plane $T(u,v)$. The ratio of $II\ (u,v,du,dv)\ /\ I\ (u,v,du,dv)$ is known as the the normal curvature function κ_{normal}. Normal curvature at a surface point varies only as a function of the *direction* of the differential vector (du,dv) in the parameter space. If $d\vec{n}$ and $d\vec{x}$ are aligned for a particular direction of (du,dv), that direction is called a *principal direction* of the surface at that surface point. The extrema of the normal curvature function at a given point occur in these directions and are referred to as *principal curvatures.*

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

$$\begin{bmatrix} \vec{x}_{uu} \\ \vec{x}_{uv} \\ \vec{x}_{vu} \\ \vec{x}_{vv} \\ \vec{n}_u \\ \vec{n}_v \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \end{bmatrix} \begin{bmatrix} \vec{x}_u(u,v) \\ \vec{x}_v(u,v) \\ \vec{n}(u,v) \end{bmatrix} = \begin{bmatrix} \Gamma^1_{11} & \Gamma^2_{11} & b_{11} \\ \Gamma^1_{12} & \Gamma^2_{12} & b_{12} \\ \Gamma^1_{21} & \Gamma^2_{21} & b_{21} \\ \Gamma^1_{22} & \Gamma^2_{22} & b_{22} \\ -\beta^1_1 & -\beta^2_1 & 0 \\ -\beta^1_2 & -\beta^2_2 & 0 \end{bmatrix} \begin{bmatrix} \vec{x}_u(u,v) \\ \vec{x}_v(u,v) \\ \vec{n}(u,v) \end{bmatrix} . \quad (3.22)$$

The "transition" matrix in this state equation contains twelve coefficient functions not yet defined. The Christoffel Symbols of the Second Kind $\Gamma^k{}_{ij}$ (connection coefficients) depend only on the metric functions $g_{ij}(u,v)$ and are defined as follows:

$$\Gamma^k{}_{ij}(u,v) = \frac{1}{2}\sum_{m=1}^{2} g^{km}\left[\frac{\partial g_{jm}}{\partial u^i} + \frac{\partial g_{mi}}{\partial u^j} - \frac{\partial g_{ij}}{\partial u^m}\right] \tag{3.23}$$

where $u^1 = u$ and $u^2 = v$ and where g^{km} is the matrix inverse of g_{km}, which is the tensor notation for the metric [**g**] already defined. Note that $\Gamma^k{}_{ij} = \Gamma^k{}_{ji}$ because the metric is symmetric.

The last two row equations of this matrix equation are referred to as the Weingarten equations for 3-D surfaces whereas the top three row equations are know as Gauss' equation. The Weingarten equations' coefficients β^i_j depend on both the first and second fundamental form matrices:

$$\beta^i_j = \sum_{k=1}^{2} b_{jk}\, g^{ki} \qquad or \qquad [\,\beta\,] = [\,\mathbf{g}^{-1}\,][\,\mathbf{b}\,]\,. \tag{3.24}$$

The $[\beta]$ matrix is referred to as the shape operator matrix [O'Neill 1966] or the *Weingarten mapping* matrix. The Weingarten mapping maps tangent vectors to other tangent vectors in the tangent plane $T(u,v)$ associated with each point $\vec{x}(u,v)$. For example, $\vec{n}_u(u,v)$ is specified as a linear combination of the u- and v-tangent vectors. One can view the $[\beta]$ matrix as the entity that determines surface shape by relating the intrinsic geometry of the surface to the Euclidean geometry of 3-D space. It is the generalization of the *curvature* of plane curves.

In summary, it has now been shown that all of the sixteen non-zero state matrix coefficient functions depend on only six scalar functions of two variables:

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

For brevity, these function are also referred to as the E,F,G,L,M,N functions. Assuming the first-order linear homogeneous space-varying partial differential matrix equation can be solved for the $\vec{x}_u, \vec{x}_v, \vec{n}$ coordinate frame, one 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:

$$\vec{x}(u,v) = \int_{u_0}^{u} \vec{x}_u(\alpha,v_0)\, d\alpha + \int_{v_0}^{v} \vec{x}_v(u,\beta)\, d\beta \qquad \vec{x} \in \mathbf{R}^3 \tag{3.25}$$

Just as for curves, there is a *fundamental existence and uniqueness theorem* for 3-D surfaces (which is credited to O. Bonnet):

(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 (3.27,3.28,3.29) and sign restrictions (3.26), 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:

$$g_{11} > 0 \qquad g_{22} > 0 \qquad \det[\mathbf{g}] = (g_{11}g_{22} - (g_{12})^2) > 0 \tag{3.26}$$

The compatibility equations are as follows:

$$(b_{11})_v - (b_{12})_u = b_{11}\Gamma^1_{12} + b_{12}(\Gamma^2_{12} - \Gamma^1_{11}) - b_{22}\Gamma^2_{11} \tag{3.27}$$

$$(b_{12})_v - (b_{22})_u = b_{11}\Gamma^1_{22} + b_{12}(\Gamma^2_{22} - \Gamma^1_{21}) - b_{22}\Gamma^2_{21} \tag{3.28}$$

$$(b_{11}b_{22} - (b_{12})^2) = g_{12}\left((\Gamma^2_{22})_u - (\Gamma^2_{12})_v + \Gamma^1_{22}\Gamma^2_{11} - \Gamma^1_{12}\Gamma^2_{12}\right) + \quad (3.29)$$
$$g_{11}\left((\Gamma^1_{22})_u - (\Gamma^1_{12})_v + \Gamma^1_{22}\Gamma^1_{11} + \Gamma^2_{22}\Gamma^1_{12} - \Gamma^1_{12}\Gamma^1_{12} - \Gamma^2_{12}\Gamma^1_{22}\right)$$

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 of only the metric and is therefore an intrinsic property of the surface. This equation may be written in several different forms. It is referred to as the Gauss equation because it proves the *Theorema Egregium* of Gauss, which states that Gaussian curvature is a function of only E,F,G and their derivatives.

(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

$$g_{ij} = g^*_{ij} \qquad b_{ij} = b^*_{ij} \ , \qquad (3.30)$$

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} (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 that would yield more easily interpretable surface shape characteristics. Fortunately, there are two curvature functions that combine the information in the six E,F,G,L,M,N functions in two different ways. These two curvature functions do not, in general, contain all the "3-D shape information" contained in the six E,F,G,L,M,N functions, but they do contain a substantial amount of useful information, which is described

subsequently. In addition, the two curvature functions do contain essentially all 3-D shape information under certain sets of constraints.

For compact *convex* surfaces (where $LN > M^2$ at every point), there is a *single scalar function* (the Gaussian curvature function $K(u,v)$) that uniquely specifies surface shape [Chern 1957][Hsiung 1981][Minkowski 1897]. This is the Gaussian Curvature Uniqueness Theorem for convex surfaces. It can be shown that if simply-connected bounded regions of positive Gaussian curvature are isolated, then surface shape is uniquely determined within those regions if the Gaussian curvature function of the surface is known. Conditions are discussed later where the mean curvature function uniquely determines graph surface shape. It is called the Mean Curvature Uniqueness Theorem for graph surfaces.

In conclusion, general smooth 3-D surfaces have been examined, and six functions that uniquely characterize the shape of these geometric entities have been identified. This review was intended to motivate the following simple qualitative statement on which this work is based: *If the shape of a geometric entity, such as a curve or surface, is to be characterized, the chosen characteristics should have a well-defined relationship to those functions that uniquely determine shape according to the mathematics of differential geometry.*

3.2. Surface Curvature

It is established that general 3-D smooth surface:
scalar functions that completely determine surface shape and intrinsic surface geometry. These six functions are the independent elements of two 2 x 2 symmetric matrix functions of the surface. Two curvature functions are now examined that combine the information from the six E,F,G,L,M,N functions.

The shape operator (Weingarten mapping) matrix $[\beta]$ was defined in the previous section as the matrix product $[\mathbf{g}^{-1}][\mathbf{b}]$. Hence, the $[\beta]$ matrix combines the first and second fundamental form matrices into one matrix. This matrix is a linear operator that maps vectors in the tangent plane to other vectors in the tangent plane at each point on a surface. The metric $[\mathbf{g}]$ is the generalization of the speed of a planar curve whereas the shape operator $[\beta]$ is a generalization of the curvature of a planar curve. The *Gaussian curvature* function K 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:

$$K = \det[\beta] = \det\left(\begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}^{-1} \right) \det\left(\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \right) . \tag{3.31}$$

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

$$H = \frac{1}{2} tr\,[\beta] = \frac{1}{2} tr\left(\begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}^{-1} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \right) . \tag{3.32}$$

Hence, these two surface curvature functions are obtained by mapping the two fundamental form matrix functions into a single 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 planar curve. (The "natural" algebraic invariants of a matrix are the coefficients of the characteristic polynomial of the matrix.) Since a 2x2 matrix only has two natural algebraic invariants (the trace and determinant), these two surface curvature functions arise naturally in the analysis of surface curvature.

There are other ways of looking at surface curvature based on the curves that lie in the surface. At each point on a surface, there is a direction of maximum normal curva-

ture and a direction of minimum normal curvature for all space curves that (1) lie in the surface, (2) pass through that point, and (3) have (curve) normals that align with the surface normal at that point. If κ_1 denotes the maximum principal curvature (the maximum of the normal curvature function) and κ_2 denotes the minimum principal curvature (the minimum of the normal curvature function), then one can compute the Gaussian and mean curvature in terms of these principal curvatures:

$$K = \kappa_1 \kappa_2 \qquad H = \frac{(\kappa_1 + \kappa_2)}{2} \tag{3.33}$$

Note that κ_1 and κ_2 are the two roots of the quadratic equation:

$$\kappa^2 - 2H\kappa + K = 0 \,. \tag{3.34}$$

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

$$\kappa_{1,2} = H \pm \sqrt{H^2 - K} \tag{3.35}$$

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 (i.e., the normal curvature function at an umbilic point is constant). A surface must be either flat or spherical in the neighborhood of an umbilic point.

The principal curvatures κ_1 and κ_2 are a perfectly valid pair of surface curvature descriptors, which are analytically equivalent to the mean and Gaussian curvature pair. The principal curvatures are the two eigenvalues of the 2x2 matrix shape operator and the extrema of the normal curvature function. They specify the curvature of surface curves in the directions of maximal and minimal normal curvature at each point. The two surface curvatures $\{ H, K \}$ are now compared to the principal surface curvatures $\{ \kappa_1, \kappa_2 \}$.

(1) Principal curvatures values are often associated with the principal directions for meaningful interpretation whereas mean and Gaussian curvature are direction-free quantities. Hence, extra storage overhead is needed for principal curvatures.

(2) If only the signs of the principal curvatures are used to determine basic surface types, six surface types result: peak, pit, ridge, valley, flat, and saddle as shown in Figure 3.4(a). The signs of mean and Gaussian curvature yield *eight* basic surface types, as shown in Figure 3.4(b), because saddle surfaces can be resolved into saddle ridge, saddle valley, and minimal surfaces. Figure 3.5 shows the shapes of the eight surfaces. Note that K cannot be strictly positive when H is zero.

(3) Gaussian curvature exhibits isometric invariance properties that are not exhibited by either of the principal curvatures. That is, Gaussian curvature is an intrinsic property of a surface. Both principal curvatures and the mean curvature are extrinsic properties of a surface. Isometric invariance and intrinsic and extrinsic properties are discussed later in more detail.

(a) Table of Surface Shapes from Principal Curvature Signs

	$\kappa 2$		
$\kappa 1$	−	0	+
−	peak	ridge	saddle
0	ridge	flat	valley
+	saddle	valley	pit

(b) Table of Surface Shapes from Gaussian (K) and Mean (H) Curvature Signs

	K		
H	+	0	−
−	peak	ridge	saddle ridge
0	*(none)*	flat	minimal surface
+	pit	valley	saddle valley

Figure 3.4. Surface Types Determined By Surface Curvature Signs

(4) The mean curvature is the average of the principal curvatures. Therefore, it is slightly less sensitive to noise in numerical computations than the principal curvatures. Gaussian curvature is more sensitive to noise.

(5) According to the Gaussian Curvature Uniqueness Theorem, the Gaussian curvature function of a convex surface uniquely determines the surface. A single principal curvature function does not permit a comparable theorem because of its directional nature.

(6) As discussed in the Mean Curvature Uniqueness Theorem section later in this chapter, the mean curvature function of a graph surface taken together with the boundary curve of a graph surface uniquely determines the graph (Monge patch) surface from which it was computed. Range images are sampled graph surfaces. A single principal curvature function does not permit a comparable theorem because of its directional nature.

(7) A few more numerical computations are required to compute principal curvatures as compared to mean and Gaussian curvature. Moreover, it is extremely simple to compute the sign of the mean and Gaussian curvature whereas principal curvature sign computations are more involved when given only partial derivatives.

To summarize, the pair $\{ \kappa_1, \kappa_2 \}$ contain the exact same surface curvature information as the pair $\{ H, K \}$. Brady et al. [1985] favor the principal curvatures and principal directions whereas mean and Gaussian curvature are preferred here. There are slight advantages or disadvantages working with either pair depending on the application. Researchers have also worked with other pairs of surface curvatures, such as $\{ \kappa_1, K \}$ [Medioni and Nevatia 1984]. The same surface curvature information is maintained. For visible-invariant pixel labeling purposes, the sign of the mean and Gaussian curva-

tures can be computed most easily yielding the best classification of surface types in a range image.

The mathematical properties of the Gaussian curvature K and the mean curvature H are now discussed in more detail to stress the importance of these quantities to surface characterization and to give a better, more complete understanding of their qualities.

(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 (see theorem and proof in Appendix A). 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 depend directly on the choice of the u,v coordinate system even though they uniquely characterize the 3-D shape of the surface. Therefore, it is not desirable to use these six functions as shape characteristics because of their dependence on parameterization. Different views of the same surface yield different image parameterizations.

(2) Gaussian and mean curvature are invariant to arbitrary rotations and translations of a surface because of the invariance of the E,F,G,L,M,N functions to rotations and translations (see theorem and proof in Appendix A). 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 an *isometric invariant* of a surface. An isometric invariant is a surface property that 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 continuous mapping where corresponding arcs on the surfaces have the same length.

) Isometric invariants are also referred to as intrinsic surface properties. Therefore, Gaussian Curvature is an intrinsic surface quantity. Intrinsic properties have 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 is readily used to demonstrate these ideas: If the paper lies flat on a desk top, $K = 0$ and $H = 0$ at each point on the sheet of paper. If the paper is bent making sure that no kinks occur, it is still true that $K = 0$, but now $H \neq 0$. When the paper bends, the way in which the surface is embedded in 3-D space changes, but the metric (intrinsic) properties of the surface do not change. 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. It should be clear that surface area is also an intrinsic surface property (can only depend on E,F,G).

(5) Another way of looking at intrinsic properties is that they do not change sign when the direction of the normal vector of the surface is reversed. Outward-pointing normals are usually chosen for surfaces of objects. If the surface is just an orient-

able surface patch floating in space, the surface normal could be chosen to point in either direction. It should be clear that Gaussian curvature maintains its sign when the direction of the normal vector is flipped whereas mean curvature flips its sign. This is because the first fundamental form does not change sign while the second fundamental form does when the sign of the normal is flipped (see definitions above).

(6) Gaussian Curvature indicates *intrinsic surface shape* at individual surface points. When $K(u,v) > 0$ at the surface point $\vec{x}(u,v)$, then the surface is *locally 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, cone-shaped, ridge-shaped, or valley-shaped. 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. Figure 3.5 shows drawings of the eight basic surface shapes. If $H < 0$ and $K = 0$, the surface is locally ridge shaped. If $H > 0$ and

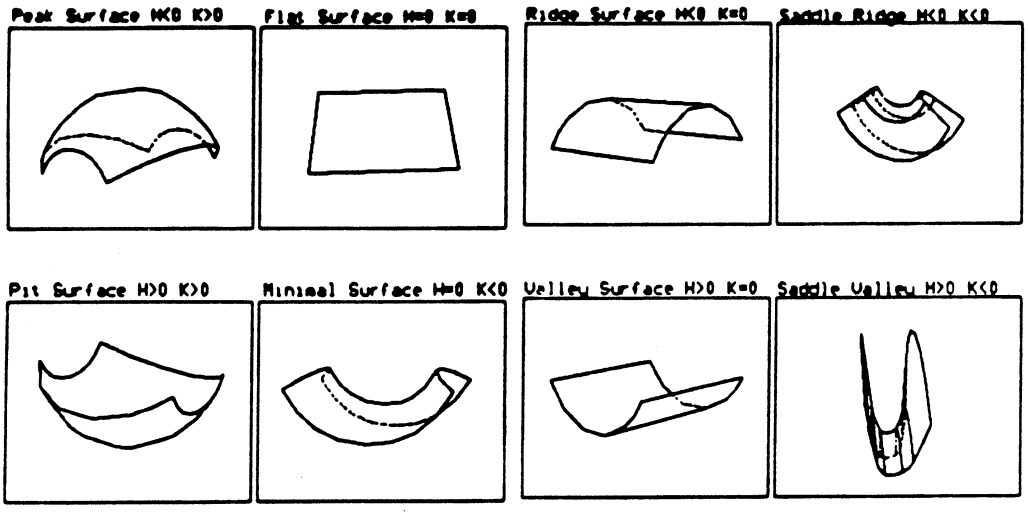


Figure 3.5. The Eight Visible-Invariant HK-Sign Surface Types

$K = 0$, 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 $K > 0$, one can never have $H = 0$. When $K < 0$, $H \neq 0$ indicates if the saddle surface is predominantly valley shaped ($H > 0$) or ridge shaped ($H < 0$). When $H = 0$ at every point on a surface, then that surface is referred to as a *minimal* surface. Minimal surfaces have interesting mathematical properties and are often studied in texts on partial differential equations. The eight surface types above are the only possible local surface types for smooth surfaces.

Note that each of the fundamental visible-invariant surface types is a simply shaped surface that could be well approximated by low-order, even-order polynomial surface functions. The key property of the sign of surface curvature characteristic is that arbitrary surfaces can be partitioned into a disjoint union of relatively simple surfaces no matter how complicated they might appear. It is proposed that these basic HK-sign surface types form a set of symbolic primitives upon which a theory of segmentation can be built. Meaningful smooth surfaces of finite area in a range scene may consist either of a single HK-sign surface primitive or a whole set of HK-sign surface primitives. This implies that smoothly joining HK-sign surface primitives may need to be merged together to yield the smooth surfaces of a scene. Hence, there is a segmentation hierarchy for range image interpretation formed by the HK-sign surface primitives at the lowest level, the smooth surface primitives consisting of HK-sign surface primitives, the objects bounded by smooth surfaces, and the scene consisting of those objects at the highest level. This

hierarchy of geometric entities is shown in Figure 3.6.

(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 $H = \pm 1 / a$ at every point on the surface where the sign depends on the direction of the normal (outward or inward pointing) whereas $K = 1 / a^2$ at every point *independent* of the direction of the normal vector. This also points out the dimensions of the curvature quantities and indicates how these quantities will change under 3-D scale transformations.

There are many other interesting properties of Gaussian and mean curvature, but the above list highlights most of the relevant ones.

If the sign of the surface curvature is reliably and accurately computed from high-resolution digital surfaces, these facts could be used directly by a surface matching algorithm as follows. Two surfaces that 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

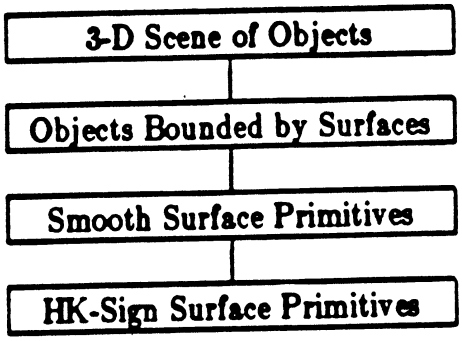


Figure 3.6. Segmentation Hierarchy of Geometric Entities

equivalent. If the two surfaces are congruent, then there exists a matching between the mean and Gaussian curvature values on the two surfaces. This implies that there is a matching between regions of constant sign of the mean and Gaussian curvatures on the two surfaces. Therefore, if there does not exist a matching between regions of constant sign of the mean and Gaussian curvatures of the two surfaces, then the two surfaces cannot have the same 3-D shape and therefore cannot be congruent. Since a combined mean and Gaussian curvature sign image has only eight levels, it would be possible to easily discard surfaces that do not have similar shape.

Gaussian curvature and/or mean curvature can be defined and/or computed in several different ways:

(1) **Gauss Map Area Derivative Definition for Convex Surfaces:** In order to give this definition, it is first necessary to describe the Gauss map, which is shown pictorially in Figure 3.7. The Gauss mapping takes areas on surfaces to areas on the unit sphere. The unit surface normals at the surface points within the area

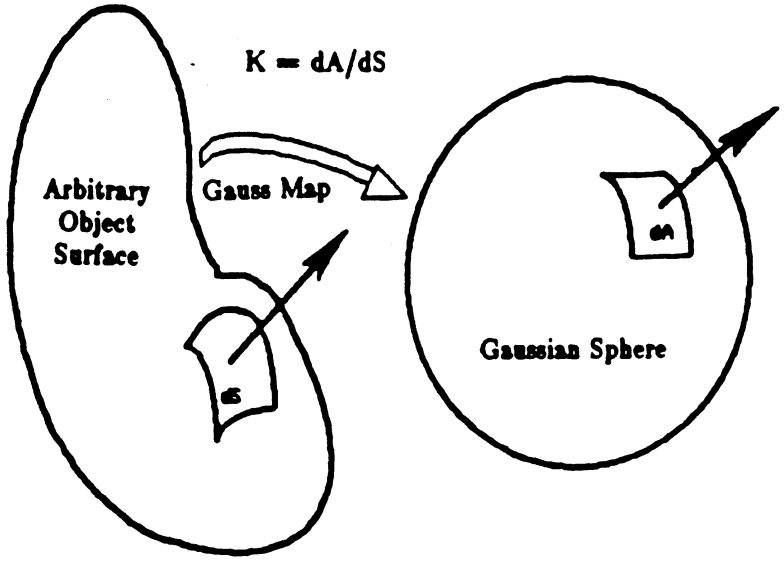


Figure 3.7. Gaussian Curvature is the Gauss Mapping Derivative

ΔS on the surface are arranged in the unit sphere so 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. The Gaussian curvature can be defined using the limit of the ratio of these two areas when the surface is convex:

$$K = \lim_{\Delta S \to 0} \frac{\Delta A}{\Delta S} \tag{3.35}$$

This popular definition seems to imply that K is a dimensionless quantity, which is not true of course. 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 solid angle on the unit sphere is computed.

(2) **Parallel Transport Definition** [Misner et al. 1973][Abelson and diSessa 1980]: Suppose you start at a point P on a surface holding a vector that always points in the same direction in space (similar to a gyroscope). That direction is marked in a permanent fashion at the starting point. You now go for a walk by leaving the point P and later returning to it without crossing your path. The path has now enclosed an area called ΔS. When arriving back at P, you compare the direction of your current vector with the reference direction, which was marked when you left the point, to obtain the angle $\Delta\alpha$. The Gaussian curvature of the surface at the point P is then defined to be

$$K = \lim_{\Delta S \to 0} \frac{\Delta \alpha}{\Delta S} \tag{3.36}$$

Note that the *sign* of K is given correctly here. There is a definite, intimate

relationship between angles, area, and curvature on a surface.

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

$$K = \frac{|\vec{n}_u \times \vec{n}_v|}{|\vec{x}_u \times \vec{x}_v|} \quad \textit{where} \quad \vec{n} = \frac{\vec{x}_u \times \vec{x}_v}{|\vec{x}_u \times \vec{x}_v|} \tag{3.37}$$

(4) **Fundamental Form Matrix Coefficients Definitions:** These definitions of K and H are a restatement of the first matrix definitions given above, but may appear more familiar. Define g = det [**g**] and b = det [b].

$$K = \frac{b}{g} = \frac{b_{11}b_{22} - (b_{12})^2}{g_{11}g_{22} - (g_{12})^2} = \frac{LM - N^2}{EG - F^2} \tag{3.38}$$

$$H = \frac{g_{11}b_{22} + g_{22}b_{11} - 2g_{12}b_{12}}{2(g_{11}g_{22} - (g_{12})^2)} = \frac{EN + GL - 2FM}{2(EG - F^2)} \tag{3.39}$$

(5) **Partial Derivative Expressions:** K and H can be expressed directly in terms of partial derivatives of the parameterization if desired. The triple-vector product notation is introduced to simplify the expression: $[\vec{a}\ \vec{b}\ \vec{c}] \equiv \vec{a}\cdot(\vec{b} \times \vec{c})$ to simplify the expressions:

$$K = \frac{[\vec{x}_{uu}\ \vec{x}_u\ \vec{x}_v][\vec{x}_{vv}\ \vec{x}_u\ \vec{x}_v] - [\vec{x}_{uv}\ \vec{x}_u\ \vec{x}_v]^2}{|\vec{x}_u \times \vec{x}_v|^4} \tag{3.40}$$

$$H = \frac{(\vec{x}_v \cdot \vec{x}_v)[\vec{x}_{uu}\ \vec{x}_u\ \vec{x}_v] + (\vec{x}_u \cdot \vec{x}_u)[\vec{x}_{vv}\ \vec{x}_u\ \vec{x}_v] - 2(\vec{x}_u \cdot \vec{x}_v)[\vec{x}_{uv}\ \vec{x}_u\ \vec{x}_v]^2}{2|\vec{x}_u \times \vec{x}_v|^3} \tag{3.41}$$

A few of the many different ways of looking at the mean and Gaussian curvature of a surface have been summarized. This list was intended to further stress the properties of

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

The most important invariance properties of surface curvature for view-independent range image object recognition are the following (see Appendix A): (1) invariance under changes in (u,v)-parameterization and (2) invariance under 3-D translations and 3-D rotations. In addition, mean curvature information significantly complements Gaussian curvature information and vice versa in determining surface shape because H is extrinsic whereas K is intrinsic. Only *eight* basic local surface types are possible as discussed above, and these types are determined solely from the signs of the mean and Gaussian curvature. Note also that K and H, considered together, provide a good generalization of the curvature function of space curves. There are other functions of the E,F,G,L,M,N functions that are also useful, but it has been shown that the emphasis on K and H is very reasonable. Henceforth, only mean curvature and Gaussian curvature are considered as surface curvature characteristics. Remember that one can compute principal curvatures, if needed, if mean and Gaussian curvature are already known. It has not been proved that K and H are the optimal surface characteristics with respect to any criterion, but substantial justification has been given for their use as surface characteristics.

3.3. Graph Surface Curvature from Partial Derivatives

Arbitrary 3-D surface shape is well characterized by two scalar functions, Gaussian curvature and mean curvature, which are independent of parameterization and invariant to rotations and translations. Given a range image with only discretely sampled, quantized data, how can one compute surface curvature? To compute surface curvature of

range images, only estimates of the first and second partial derivatives of the depth map are needed. In order to see this, the expressions for K and H are simplified for graph (Monge patch) surfaces because all range images and intensity images are sampled graph surfaces.

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

$$\vec{x}_u = [\, 1 \quad 0 \quad f_u \,]^T \tag{3.42}$$

$$\vec{x}_v = [\, 0 \quad 1 \quad f_v \,]^T \tag{3.43}$$

$$\vec{x}_{uu} = [\, 0 \quad 0 \quad f_{uu} \,]^T \tag{3.44}$$

$$\vec{x}_{vv} = [\, 0 \quad 0 \quad f_{vv} \,]^T \tag{3.45}$$

$$\vec{x}_{uv} = [\, 0 \quad 0 \quad f_{uv} \,]^T \tag{3.46}$$

$$\vec{n} = \frac{1}{\sqrt{1 + f_u^{\,2} + f_v^{\,2}}} [\, -f_u \quad -f_v \quad 1 \,]^T \,. \tag{3.47}$$

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

$$g_{11} = 1 + f_u^{\,2} \qquad g_{22} = 1 + f_v^{\,2} \qquad g_{12} = f_u f_v \tag{3.48}$$

$$b_{11} = \frac{f_{uu}}{\sqrt{1 + f_u^{\,2} + f_v^{\,2}}} \tag{3.49}$$

$$b_{12} = \frac{f_{uv}}{\sqrt{1 + f_u^{\,2} + f_v^{\,2}}}$$

$$b_{22} = \frac{f_{vv}}{\sqrt{1 + f_u^{\,2} + f_v^{\,2}}} \,.$$

Hence, *five* partial derivatives f_u, f_v, f_{uu}, f_{uv}, f_{vv} are all that is needed to

compute the *six* fundamental form coefficient functions for a graph surface.

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

$$K = \frac{f_{uu} f_{vv} - f_{uv}^2}{(1 + f_u^2 + f_v^2)^2} = \frac{\det(\nabla\nabla^T f)}{|\nabla f|^4} \tag{3.50}$$

∇ is the 2-D (u,v) gradient operator, and $\nabla\nabla^T$ is the Hessian matrix operator. Hence, if given a depth map function $f(u,v)$ that possesses first and second partial derivatives, the Gaussian curvature can be computed directly.

Also recall that mean curvature can be computed as half the trace of the shape operator. This expression can also be written directly in terms of the depth map function derivatives as follows:

$$\begin{aligned} 2H &= \frac{f_{uu} + f_{vv} + f_{uu} f_v^2 + f_{vv} f_u^2 - 2 f_u f_v f_{uv}}{(1 + f_u^2 + f_v^2)^{\frac{3}{2}}} \\ &= \nabla \cdot \left(\frac{\nabla f}{\sqrt{1 + |\nabla f|^2}} \right). \end{aligned} \tag{3.51}$$

$(\nabla\cdot)$ is the divergence operator of vector calculus. Again, if given a depth map function $f(u,v)$ that possesses first and second partial derivatives, the mean curvature can be computed directly.

3.3.1. Mean Curvature Uniqueness Theorem

It is important to note that the above differential equation (3.51) (where H is known but f is not) is a non-homogeneous second-order elliptic quasilinear partial differential equation. If D is a subset of $\mathbf{R}^2$ and H is an arbitrary function of two vari-

ables with continuous first partial derivatives defined over D, it is not possible to say whether or not a solution to the differential equation above even exists. By imposing certain restrictions, it is sometimes possible to prove the existence and the uniqueness of solutions. For example, Guisti [1978] has proven that under certain extremal conditions, H alone *without Dirichlet boundary conditions* can *uniquely* determine f up to an additive constant (translations in depth). Also, Gilbarg and Trudinger [1983] show that, under a set of certain other conditions (which include the restriction that the *boundary curve's* curvature must be greater than or equal to the absolute value of the sum of the principal curvatures of the surface at the boundary of the domain), there exists a unique solution f to the Dirichlet boundary value problem defined by H plus the function f restricted to the boundary of the region D. However, *there is a separate uniqueness theorem* [Gilbarg and Trudinger 1983] (that does not address existence) that states that if (1) H is continuously differentiable, (2) f_1 and f_2 are both solutions to the partial differential equation above in D, and (3) $f_1 = f_2$ on the boundary of the domain D, then $f_1 = f_2$ throughout that domain. In this sense, a smooth surface function $f(u,v)$ defined over a compact domain D with a simple closed contour boundary ∂D is essentially *equivalent* to that surface's mean curvature function H taken together with the boundary curve of the surface f restricted to ∂D. Hence, H plus f on ∂D constitute an ideal type of graph surface characteristic; all "information" present in the original smooth depth map function is maintained in the characteristic data. Given $f(u,v)$, one can compute $H(u,v)$, and, in theory, the Dirichlet problem can be solved to reproduce $f(u,v)$. The mean curvature surface characteristic is very valuable to an object recognition algorithm because of its invariance and uniqueness properties. It must be stressed that this uniqueness property of the mean curvature function is only valid for graph surfaces. But since all range images (and all intensity images) are

sampled graph surfaces, this is an extremely important concept for digital surface characterization. A single principal curvature function does not have a similar uniqueness theorem because of its directional nature.

Since any sampled depth map function encountered in practice may be approximated arbitrarily well by a sufficiently smooth function that possesses first and second partial derivatives, the next problem is computing estimates of these partial derivatives given the sampled data.

3.4. Estimating Partial Derivatives of Digital Surfaces

In general, direct numerical differentiation is discouraged if a problem can be addressed using other means. The basic approach in the currently implemented method is the following: (1) given discrete sample data, determine a continuous differentiable function that "best" fits the data, and (2) compute the derivatives of the continuous function analytically and evaluate them at the corresponding discrete points. Ideally, it might be desirable to fit all data with one smooth surface. This problem is computationally intensive and should only be used as a last resort if simpler methods do not work. Instead, a local surface fit is computed within each N x N window of the digital surface. The experimental results show that this approach is adequate. This method uses a local least-squares surface model that is discussed in many papers [Anderson and Houseman 1942][Prewitt 1970][Beaudet 1978] [Haralick and Watson 1981][Bolle and Cooper 1984][Haralick 1984]. For this reason, only the final results of the analysis that are required to implement the local quadratic surface approach are stated here.

Each data point in a given N x N window is associated with a position (u, v) from the set $U \times U$ where N is odd:

$$U = \left\{ \frac{-(N-1)}{2}, \cdots, -1, 0, 1, \cdots, \frac{(N-1)}{2} \right\} . \tag{3.52}$$

The following discrete orthogonal polynomials provide quadratic surface fitting capability:

$$\phi_0(u) = 1 \qquad \phi_1(u) = u \qquad \phi_2(u) = (u^2 - \frac{M(M+1)}{3}) \tag{3.53}$$

where $M = \frac{(N-1)}{2}$. The $b_i(u)$ functions are normalized versions of the orthogonal polynomials:

$$b_0(u) = \frac{1}{N} \qquad b_1(u) = \frac{3}{M(M+1)(2M+1)}\, u \tag{3.54}$$

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

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

$$P(M) = \frac{8}{45} M^5 + \frac{4}{9} M^4 + \frac{2}{9} M^3 - \frac{1}{9} M^2 - \frac{1}{15} M . \tag{3.55}$$

The recipe for computing derivatives at a sample point using odd size data windows is simple since the $b_i(u)$ vectors are precomputed and stored for any given window size. A surface function estimate $\hat{f}(u,v)$ is obtained of the form

$$\hat{f}(u,v) = \sum_{i,j=0}^{2} a_{ij} \phi_i(u) \phi_j(v) \tag{3.56}$$

that minimizes the total square error term

$$\epsilon = \sum_{(u,v) \in U^2} (f(u,v) - \hat{f}(u,v))^2 . \tag{3.57}$$

The solution for the unknown coefficients is given by

$$a_{ij} = \sum_{(u,v) \in U^2} f(u,v) b_i(u) b_j(v) \quad . \tag{3.58}$$

The first and second partial derivative estimates are then given by

$$f_u = a_{10} \quad f_v = a_{01} \quad f_{uv} = a_{11} \quad f_{uu} = 2\,a_{20} \quad f_{vv} = 2\,a_{02} \quad . \tag{3.59}$$

The total fit error is computed after the a_{ij} coefficients are determined:

$$\epsilon = \sum_{(u,v) \in U^2} f^2(u,v) - \sum_{i,j} a_{ij} \left(\sum_u \phi_i^2(u)\right)\left(\sum_v \phi_j^2(v)\right) \quad . \tag{3.60}$$

Since the discrete orthogonal quadratic polynomials over the 2-D window are separable in u and v as shown in the above equations, partial derivative estimates can be computed for an entire depth map using a separable convolution operator. This is quite efficient compared to non-separable convolution operations. 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 samples from a continuous depth map function $f(u,v)$.

The disadvantages of this local surface fit method are the following:

(1) A different quadratic surface is 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, one may need to make *a priori* assumptions about the surface in order to correct this. But making these kind of assumptions is contrary to the data-driven goal of using as few *a priori* assumptions as possible.

(2) It seems counter-intuitive that all columns (or rows) in a least squares derivative window operator are weighted equally. If one asks for a 9x9 window least squares estimate of the first derivative of a depth map at a particular pixel, the data that

runs four pixels away has the same impact on the final estimate as does the data that runs directly through the pixel where the derivative is being estimated. This situation can be modified using weighted least squares techniques. The question then arises: What is the best assignment of weights? A triangular weight assignment was tried on a few 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 results substantially.

There are other approaches to computing intrinsic differential geometric properties, such as Gaussian curvature. One very interesting method that does not require partial derivative estimation is discussed in Appendix B. Young [1985] suggests that partial derivative estimates should be obtained using a difference of offset Gaussians (DOOG) approach and shows evidence that such computations are done biologically. However, despite the disadvantages listed above and the existence of other methods, excellent results have been obtained here and elsewhere using the local surface fit approach. No other methods have been tested and proven to give better results.

3.5. Other Surface Characteristics

Many other surface characteristics besides surface curvature can be easily computed given the partial derivative estimates at each pixel of a digital surface. The use of surface critical points for surface characterization has been discussed by Nackman [1984]. He notes that critical points and ridge and course (valley) lines 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 one must estimate 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 by detecting the zero-crossings of the first partial derivatives. For surfaces, there are seven kinds of *non-degenerate* critical points where $f_u = f_v = 0$ and $K \neq 0 \neq H$:

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

(2) Ridge Critical Points: $H < 0$ and $K = 0$,

(3) Saddle Ridge Critical Points: $H < 0$ and $K < 0$,

(4) Minimal Critical Points: $H = 0$ and $K < 0$,

(5) Saddle Valley Critical Points: $H > 0$ and $K < 0$,

(6) Valley Critical Points: $H > 0$ and $K = 0$,

(7) Pit Critical Points: $H > 0$ and $K > 0$.

In addition, there is one kind of *degenerate* critical point where $f_u = f_v = 0$ and $K = 0$ and $H = 0$: Planar Critical Points. Hence, if the zero-crossings of the first partial derivatives are computed in addition to Gaussian and mean curvature, then a richer structural description of the range image surface is obtained. It should be noted, however, that critical points are clearly view-dependent quantities.

The proposed critical point characterization is a generalization of the 1-D function characterization techniques. Computing critical points is the generalization of computing the zeros of the first derivative of a function of one variable. 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 that may be of interest in surface characterization and range image segmentation. The first of these quantities is the square root of the determinant of the first fundamental form matrix:

$$\sqrt{g} = \sqrt{EG - F^2} = \sqrt{1 + f_u{}^2 + f_v{}^2}\,. \tag{3.61}$$

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

$$Surface\ Area = \int\int \sqrt{1 + f_u{}^2 + f_v{}^2}\, du dv\,. \tag{3.62}$$

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. This type of output image is very similar to the output of edge detection algorithms. It can be thresholded using a minimum depth separation distance to create a simple binary edge image. Local non-maxima suppression can be included to create desirable thin edges. In depth maps, these edges generally correspond to depth-discontinuities, which generally correspond to the occluding boundary contour of an object. Thus, the existence of a depth discontinuity and a surface region boundary along a curve in an image will reinforce the interpretation of that curve as an occluding object boundary for segmentation purposes.

A second extrinsic quantity that is easy to compute pointwise given the derivatives already computed is the so-called *quadratic variation*:

$$Q = f_{uu}{}^2 + 2f_{uv}{}^2 + f_{vv}{}^2\,. \tag{3.63}$$

When this function is integrated (summed) over a depth map region, the integral (sum) is a measure of the *flatness* of that region. This image function and the metric determinant image 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 (intrinsic) quantity is the *coordinate angle function* Θ, which is defined as

$$\Theta = \cos^{-1}(F \;/\; \sqrt{EG}\;) = \cos^{-1}\left(\frac{f_u f_v}{\sqrt{1 + f_u^{\,2} + f_v^{\,2} + f_u^{\,2} f_v^{\,2}}} \right) . \tag{3.64}$$

This function measures the non-orthogonality of the u,v parameterization at each point: $\Theta = \pi/2$ when the u- and v-tangent vectors are orthogonal, and Θ 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 graph surface formulation. It is not clear how this intrinsic surface feature can contribute in general to segmentation. However, the zeros of this function form ridge and course (valley) lines that are useful in critical point configuration graphs [Nackman 1984].

The last quantities discussed in this section are the *principal directions* of the surface at each point. The principal direction vectors of the surface, along with either H and K or the principal curvatures, completely determine the shape operator of the surface. The principal direction angles in the u-v plane are analytically computed as follows:

$$\Phi_{1,2} = \tan^{-1}\left(\frac{-B \pm \sqrt{B^2 - AC}}{C} \right) \tag{3.65}$$

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

Note that these directions are in general *not orthogonal in the (u,v) parameter plane* even though the 3-D principal direction vectors in the tangent planes are orthogonal. These angles are not currently used as surface descriptors, as in [Brady et al. 1985], because it is not clear that they are useful unless the lines of curvature approach is taken. Nevertheless, they are included in this discussion for completeness of the continuous sur-

face description. It is conjectured 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. The main difficulty in proving this conjecture lies in the complexity of the differential equation for K in terms of E,F,G and their derivatives. If this conjecture is true, it would provide an interesting split between angular and non-angular functions that describe a surface.

3.6. Chapter Summary

The proposed digital surface characterization process is summarized below. This process is critical to the segmentation process described in the next chapter, and it may be directly useful for some applications.

Input:

A digital surface 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 local quadratic-surface model window-convolution techniques described above,

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

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

Output:

(a) Two three-level images, or matrices, $sgn(K)$ and $sgn(H)$, where $sgn(\cdot)$ is the signum function that yields 1 if the argument is positive, 0 if the argument is zero, and -1 if the argument is negative. These two functions can be combined into one eight-level

function, known as the HK-sign map, via a linear combination of the signum functions (e.g., $sgn_{HK}(H,K) = 3*(sgn(H)+1) + (sgn(K)+1)$). In the example case, sgn_{HK} can never be 5.

(b) Three non-negative images $|H|$, $|K|$, and $\sqrt{H^2-K}$ that describe the magnitude of the two surface curvatures and the scaled magnitude of the difference of the principal curvatures respectively ($(\kappa_1-\kappa_2)^2/4 = H^2 - K$.)

(c) A binary image matrix $c(i,j)$ that is 1 when (i,j) is a critical point and 0 when it is not. Peak, pit, saddle ridge, saddle valley, and minimal critical points of smooth surfaces are always isolated critical points. Ridge and valley critical points can form planar curves. Planar critical points can form planar areas. Each resulting critical point, curve, or area is labeled with its appropriate classification.

(d) Three non-negative images $\sqrt{g}$, Q, and ϵ are useful for edge detection and for computing region features, such as surface area and flatness. Fit error indicates the reliability of the partial derivative estimates, and therefore it also indicates the reliability of HK-sign and all other characteristics.

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

Note that a large amount of useful surface structure information about a N_{bits}-digitized depth map can be "compressed" into eight levels (*only three bits*) if the signs of the Gaussian and mean curvature are used to create the HK-Sign map. This second-order sign information *substantially constrains* the possibilities of visible surfaces and possesses the right type of invariance properties. Also, the sign of a second-order quantity computed from digital sensor data is more reliable than the magnitude because second derivatives are so difficult to estimate accurately from noisy digital data. In addition, a classified list of critical points, that also substantially constrains the surface, can be com-

puted. This list normally contains a very small number of points compared to the total number of pixels in the range image and is useful for view-dependent critical point configuration graph surface descriptions. The other images provide additional, useful, overlapping information about a digital surface.

Depth discontinuities (step edges) and orientation discontinuities (roof edges) have been computed by applying a non-directional non-maxima suppression algorithm to the images that may be interpreted as edge magnitude images: $\sqrt{g}$, $|H|$, $\sqrt{H^2 - K}$. More robust edge detection/linking algorithms, such as the Eichel edge linker/detector [Eichel 1985], have also been used with success to extract edges from these types of images. Because the main thrust of the thesis is surface-based methods, more details are not presented. It is only noted that such images are useful for edge detection purposes.

3.7. Experimental Surface Characterisation Results

A range image processing program was written to do the surface characterization computations in the C programming language on a VAX/UNIX system. This program is an integral part of the surface-based segmentation algorithm presented in Chapter 4. The potentially parallel computational structure of this program is shown in Figure 3.8. All five derivative images could be computed simultaneously after initial smoothing on a parallel architecture. Subsequently, all surface characteristics could be computed simultaneously after the derivative estimation stage. On a sequential machine, note that *all five* derivative images can be computed for the *cost of four* derivative images if the separable derivative operations are arranged correctly and separate memory is used for each output image and a scratchpad image. This is a 20% cost savings in the derivative computation stage. The software currently accepts a square range image (with 8-bits of depth) as input and generates the following images as output:

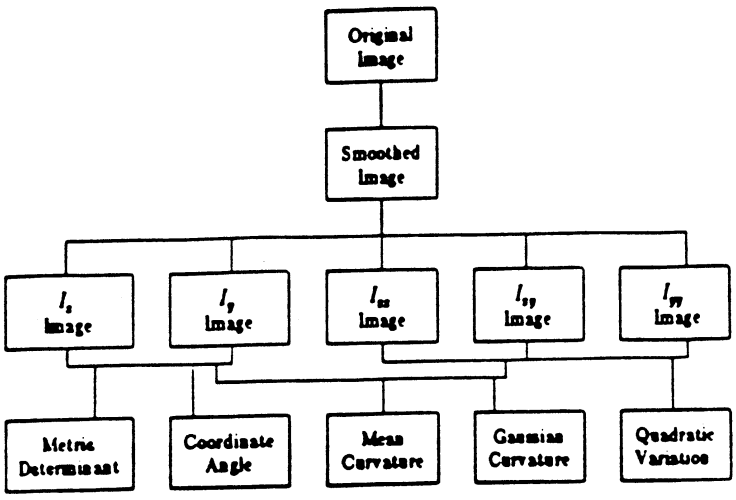


Figure 3.8. Surface Characterization Algorithm Structure

(1) Smoothed Range Image: $f_{smooth}(u,v)$

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

(3) Quadratic Variation (Flatness Measure) Image: $Q(u,v)$

(4) Local Quadratic Surface Fit Error Image: $\epsilon(u,v)$

(5) Zeros of the Mean Curvature: (u,v) *such that* $H(u,v)=0$

(6) Zeros of the Gaussian Curvature: (u,v) *such that* $K(u,v)=0$

(7) Zeros of the Cosine-of-the-Coordinate-Angle Function:

(u,v) *such that* $\cos\Theta(u,v)=0$

(8) Cosine-of-the-Coordinate-Angle Function: $\cos\Theta(u,v)$

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

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

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

(12) Principal Direction Angle: $\Phi_1(u,v)$

(13) Magnitude of Gaussian Curvature: $|K(u,v)|$

(14) Magnitude of Mean Curvature: $|H(u,v)|$

(15) Non-degenerate Critical Points Image:

$$(u,v) \ \textit{such that} \ f_u(u,v) = f_v(u,v) = 0 \neq Q(u,v)$$

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

This output data characterizes the input depth map in a way that is useful both for segmentation of sensor data and recognition of objects. A range image can be decomposed into the basic eight types of surface regions by using a combination of the $sgn(H)$ and $sgn(K)$ images, known as the HK-Sign map. Step edges and surface area are obtained from the $\sqrt{g}$ image. It is possible to detect roof edges and ramp edges using the $|H|$ and $\sqrt{H^2 - K}$ images. Critical point configurations describe surfaces as in Nackman [1984]. Data-driven processing can yield rich, interrelated surface, edge, and point information.

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

(1) All original range images are quantized to eight bits of depth. Quantization noise alone caused many problems in the first computational tests on analytically computed surfaces, such as spheres. This appears to be due to the fact that quantization noise is not independent and identically distributed, but rather it is correlated in a spatially dependent manner. The least-squares techniques used to compute the window operators are not optimized to handle such noise. In an attempt to fix this problem, the original image is smoothed using a Gaussian smoothing window operator that is two pixels larger than the window operators used to do the derivative estimation, and the results are stored in floating point form to maintain the

fractional part of the smoothed value. This smoothing also tends to compensate for random noise in the two or three least significant bits of range data.

(2) The output curvature images are also smoothed using the same smoothing operator that was used on the input to even out the variations in the surface curvature values. However, a smaller window size is used.

(3) The surface curvature sign images are obtained using a threshold around zero. That is, $sgn(K) = 0$ if $|K| < \epsilon_K$ Also, $sgn(H) = 0$ if $|H| < \epsilon_H$. The two thresholds were set to $\epsilon_K = 0.01 K_{max}$ and $\epsilon_H = 0.01 H_{max}$ for synthetic range images with no noise where K_{max} and H_{max} are the maximum absolute values attained by the surface curvatures in the image. Noisy images required larger thresholds to obtain good HK-sign images. For the experimental results shown in Chapter 7, the two zero curvature thresholds were fixed to $\epsilon_H = 0.015$ and $\epsilon_K = 0.06$.

These items are critical to the results displayed in this paper. Different smoothing schemes will create different surface curvature results. Different thresholds create different surface curvature sign images. Experiments with a smoothing algorithm that inhibits smoothing operations over depth discontinuities were also performed, and several different threshold setting alternatives were examined. No methods were found to be consistently better than those described above. If more than eight bits of depth resolution become available from newer sensors, the presmoothing step will be proportionately less important than what it is for the current 8-bit data.

Experimental results for different object depth maps are shown in Figures 3.10 through 3.25. Each depth map is discussed briefly below. The results are shown in a sixteen (16) subimage format. The contents of each subimage are noted in Figure 3.9.

f_{smooth}	$\sqrt{g}$	Q	ϵ
zeros(H)	zeros(K)	zeros(cosΘ)	cosΘ
sgn(H)	sgn(K)	$\sqrt{H^2 - K}$	Φ_1
$\lvert H \rvert$	$\lvert K \rvert$	$Q \neq 0$ $\nabla f = 0$	$\nabla f = 0$

f = surface function
H = mean curvature
K = Gaussian curvature
g = metric determinant
Q = quadratic variation
ϵ = local surface fit error
Θ = coordinate angle
Φ_1 = principal direction angle

Figure 3.9. Surface Characterization Results Format

Zeros 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 sometimes reversed by other authors. My experience is that, although possible, it is generally more difficult to interpret such reversed range images. It is equivalent to the difference between black and white photographs and negatives. Convolutional window effects near the edges of the sixteen subimages are not relevant to the image. For several range images, a surface plot of the range data is also provided.

The range images shown here were obtained in two different ways. Synthetic range images of arbitrary 3-D object models from arbitrary views are generated using a combination of the SDRC/GEOMOD solid modeler [GEOMOD 1983] developed by Structural Dynamics Research Corporation, which creates object models; and the author's depth-

buffer graphics software, which creates the range images. Real range images have been obtained from the Environmental Research Institute of Michigan (ERIM), which were acquired using the ERIM laser rangefinder [Svetkoff et al. 1984]. The range image points in these images are obtained using equal angle-increment sampling. This equal-angle-increment sampling causes flat surfaces in the real world to be mapped into slightly warped surfaces in range images (see Appendix D).

The results for each object are now considered individually. The first object is a coffee cup. Two gray scale images of two depth maps of this object are shown in Figure 3.10 along with a surface plot of the one on the right. The two depth maps were obtained from the ERIM laser rangefinder. The quality of these depth maps is almost comparable to that of synthetic depth maps. The surface characterizations of these depth maps are shown in Figure 3.11. A 7x7 derivative window operator was used. It has been found that the *zeros of the mean curvature* form a good line drawing of the object shape, which is similar to a Laplacian zero-crossings image. This is quite reasonable because the Laplacian, typically used to obtain edges [Marr 1982], is a second-order linear elliptic differential operator, whereas mean curvature is a second-order quasi-linear elliptic differential operator. The square root of the metric determinant, the quadratic variation, and the local quadratic surface fit error provide an interesting sequence of edge-detector-like images. Clusters of local maxima and saddle ridge critical points are found at the closer and farther rims of the cup respectively whereas local minima critical points are found on the inside of the cup and inside the handle. Despite the noise present in this real image, very few spurious critical points are found. The magnitude of the principal curvature difference image shows that the surface's principal curvatures differ most on the cup's rim.

A second depth map from the laser rangefinder is shown in Figure 3.12 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 surface characterizations are shown in Figure 3.13. 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 the 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 image yields a very good line drawing of the keyboard. 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.

Two views of a road scene are now discussed that were selected from a range image sequence acquired by the ERIM laser rangefinder. Figure 3.14 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 the processed image on the right. A special purpose program written by the author removes the wraparound transition automatically. This was easy in this case because of the smooth, almost flat shape of the road. The sixty-four foot line was not removed because most of the data beyond that level is excessively noisy. Figure 3.15 shows the surface 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. Because of the equal-angle-increment sampling, the flat road samples are 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 3.16 is a surface plot of the range image of a tilted torus. The surface characterization results for two different synthetic range views of the torus are shown in Figure 3.17. In the other view, the torus is tilted only five degrees. Note how well the surface critical points were detected. The structure of the ridge and course lines in the zeros of the cosine-of-the-coordinate-angle image gives important view-dependent information about the surface in terms of slope districts. The irregularities in the curvature magnitude images occur because the object model from which the range image 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, to within the specified 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 3.18. The surface characterization results for two different views using 5x5 window derivative operator are shown in Figure 3.19. The critical points that are also maximum Gaussian curvature points tend to line up along the joining curve in the center of the range image when looking 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 are seen in the scaled edge map (square root of g) image. Note the slight changes in the surface curvature sign images between the two views.

To give an idea of how window size and noise level affect the results of the surface characterization algorithm, a synthetic image of a cube with three holes in it is used.

Pseudo-random pseudo-Gaussian noise (rounded to the nearest integer) was added to the original image to create four different synthetic noisy images as shown in Figure 3.20. The surface characterization results for these four images are shown in Figures 3.21 through 3.25, which correspond respectively to additive Gaussian noise standard deviations (σ 's) of 2.3, 9.2, 16.0, and 22.9 gray levels (depth levels) added to an original image with a dynamic range of 256 levels. The resulting noisy images were rescaled to fit into the 8-bit depth range. These images were then processed with 5x5, 7x7, 9x9, 11x11, and 13x13 derivative window operators. The following five figures have been selected to demonstrate the noise performance:

- Figure 3.21 - 5x5 operator applied to the $\sigma = 2.3$ noisy image.
- Figure 3.22 - 7x7 operator applied to the $\sigma = 9.2$ noisy image.
- Figure 3.23 - 9x9 operator applied to the $\sigma = 16.0$ noisy image.
- Figure 3.24 - 11x11 operator applied to the $\sigma = 22.9$ noisy image.
- Figure 3.25 - 13x13 operator applied to the $\sigma = 22.9$ noisy image.

Note how well the sign of the mean curvature represents the important surface variations of the cube even in the presence of significant noise. The mean curvature images are surprisingly consistent and qualitatively meaningful in all five figures even though second derivatives of very noisy data are involved in its computation. The Gaussian curvature images seem to be more susceptible to noise as expected, but the closest vertex of the cube is consistently marked as a high curvature spot. The degradation in the zeros of the cosine-of-the-coordinate-angle is interesting to observe. The critical point images are consistent despite the noise and demonstrate the necessity of a large window size to suppress spurious critical points in the presence of noise. Note that fewer spurious critical points result in the 11x11 operator, $\sigma = 22.9$ results shown in Figure 3.24,

than in the 5x5 operator, $\sigma = 2.3$ results shown in Figure 3.21. The conclusion is that even though second derivative information is being used, the surface descriptors are still useful in the presence of noise. Moreover, the surface characteristics appear to degrade slowly as the noise level increases. In practical applications, it is unlikely that the noise would ever exceed the $\sigma = 9.2$ level.

In summary, the experimental results indicate the efficacy of this differential-geometry-based surface characterization approach for digital surfaces in the presence of noise. However, these results are still in a low-level form and are not directly suitable for use by higher level processes. This is remedied by the pixel grouping and refined image segmentation processes addressed in the next chapter. The sgn(H) and sgn(K) images shown in this chapter can be combined as described above to create the HK-sign map, which does provide a rough initial segmentation of a digital surface. (Combined HK-sign map images (or visible-invariant pixel labelings) were not shown in this chapter because they are shown in Chapter 7 along with final segmentation results.) This HK-sign map segmentation is rather noisy for the 8-bit images shown in this thesis, but as higher depth-resolution sensors become available, the initial HK-sign map segmentation will also become better .

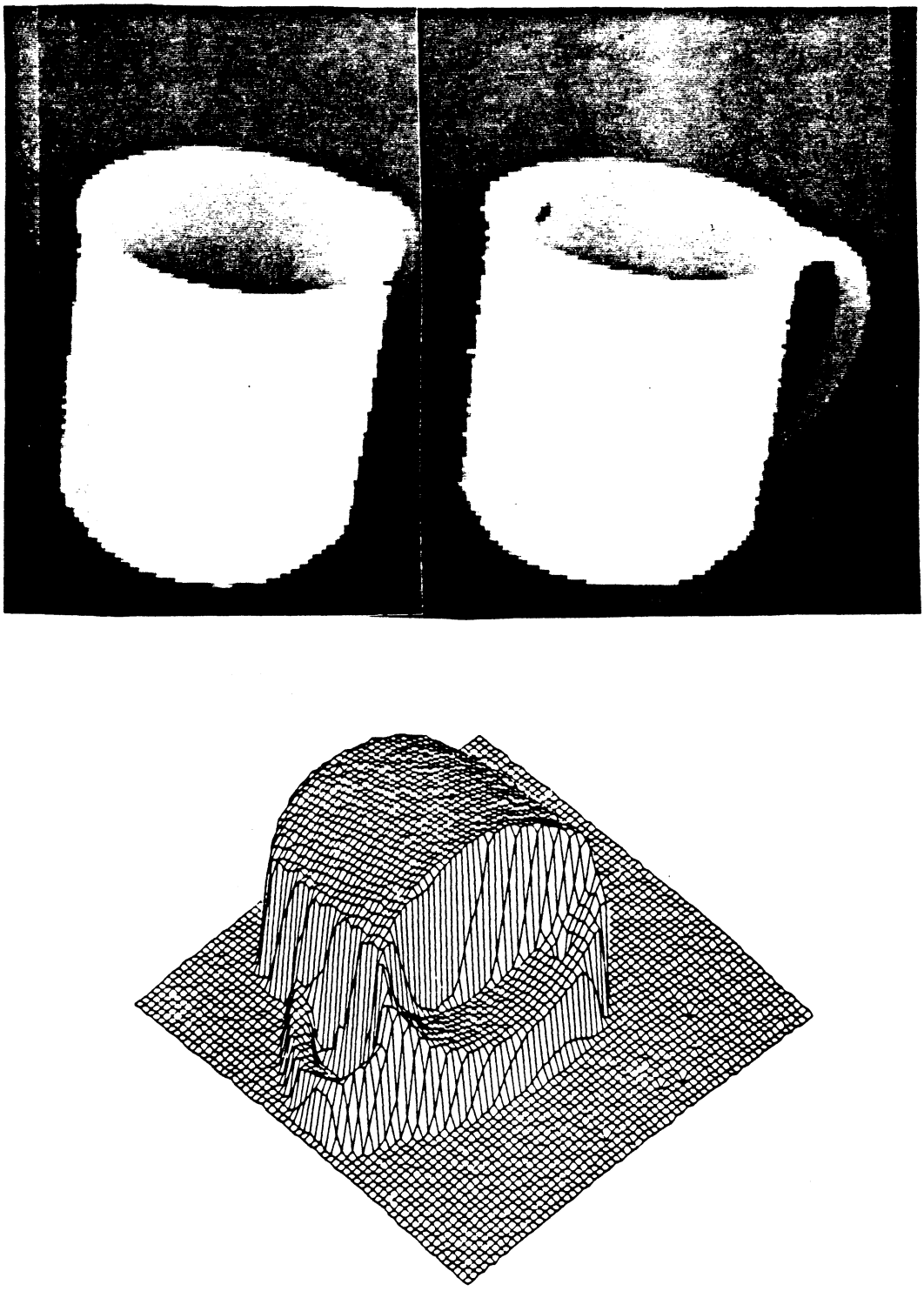


Figure 3.10. Coffee Cup Range Images and Surface Plot
(128x128 ERIM Range Images)

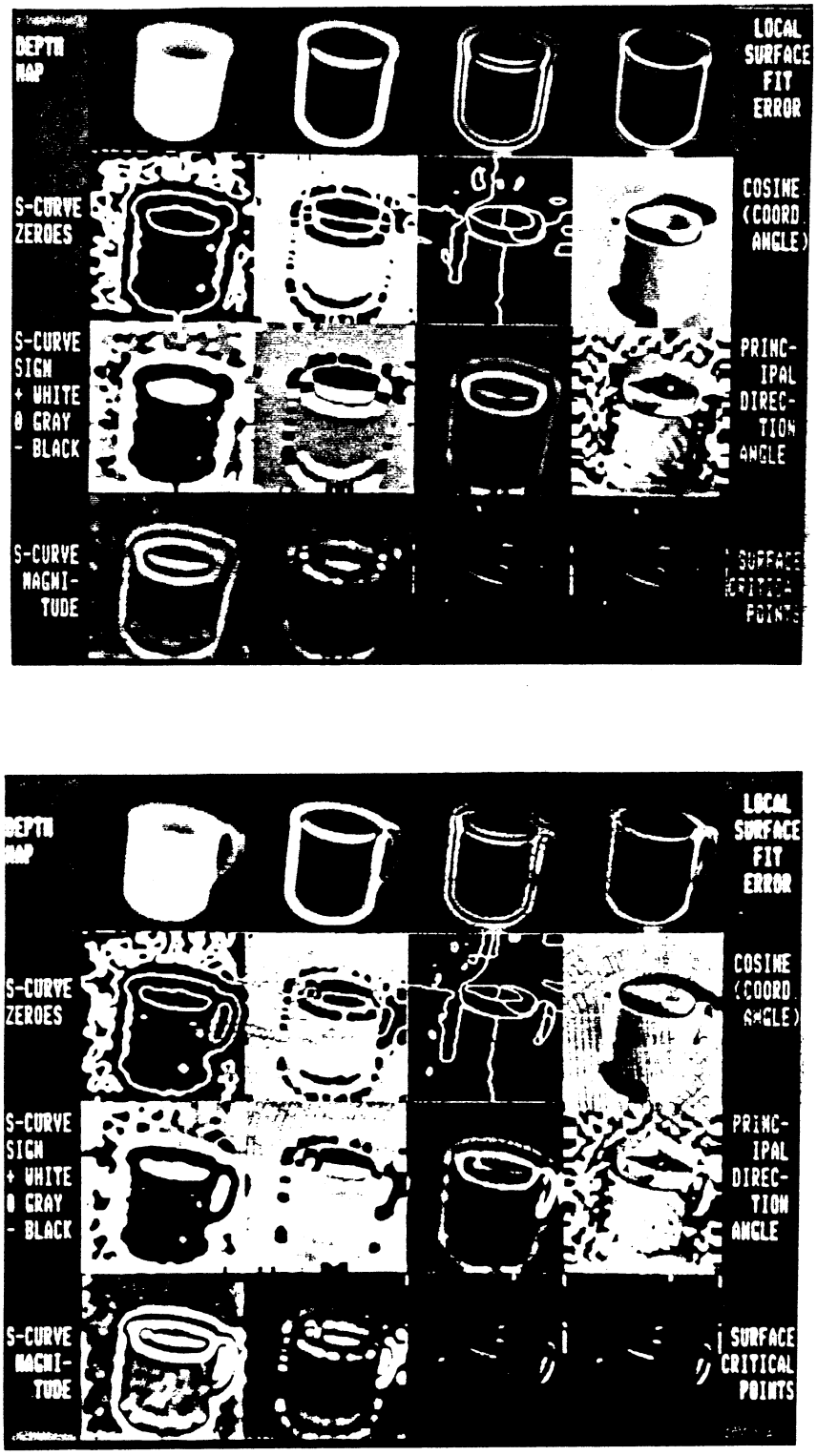


Figure 3.11. Surface Characterizations of Two Views of Coffee Cup
(7x7 Derivative Window Operator, Zero Threshold=4%)

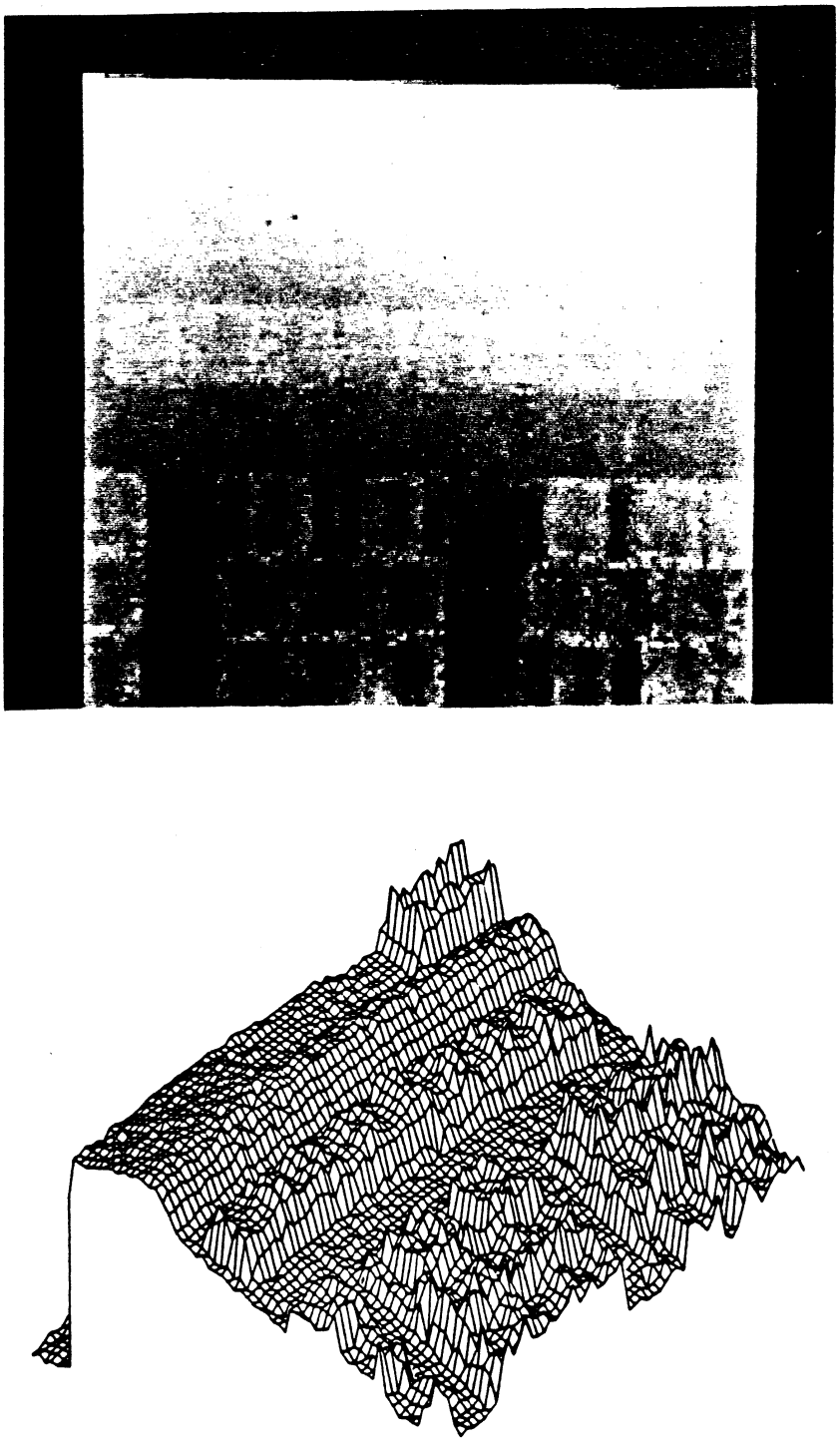


Figure 3.12. Keyboard Range Image and Surface Plot
(128x128 ERIM Range Image)

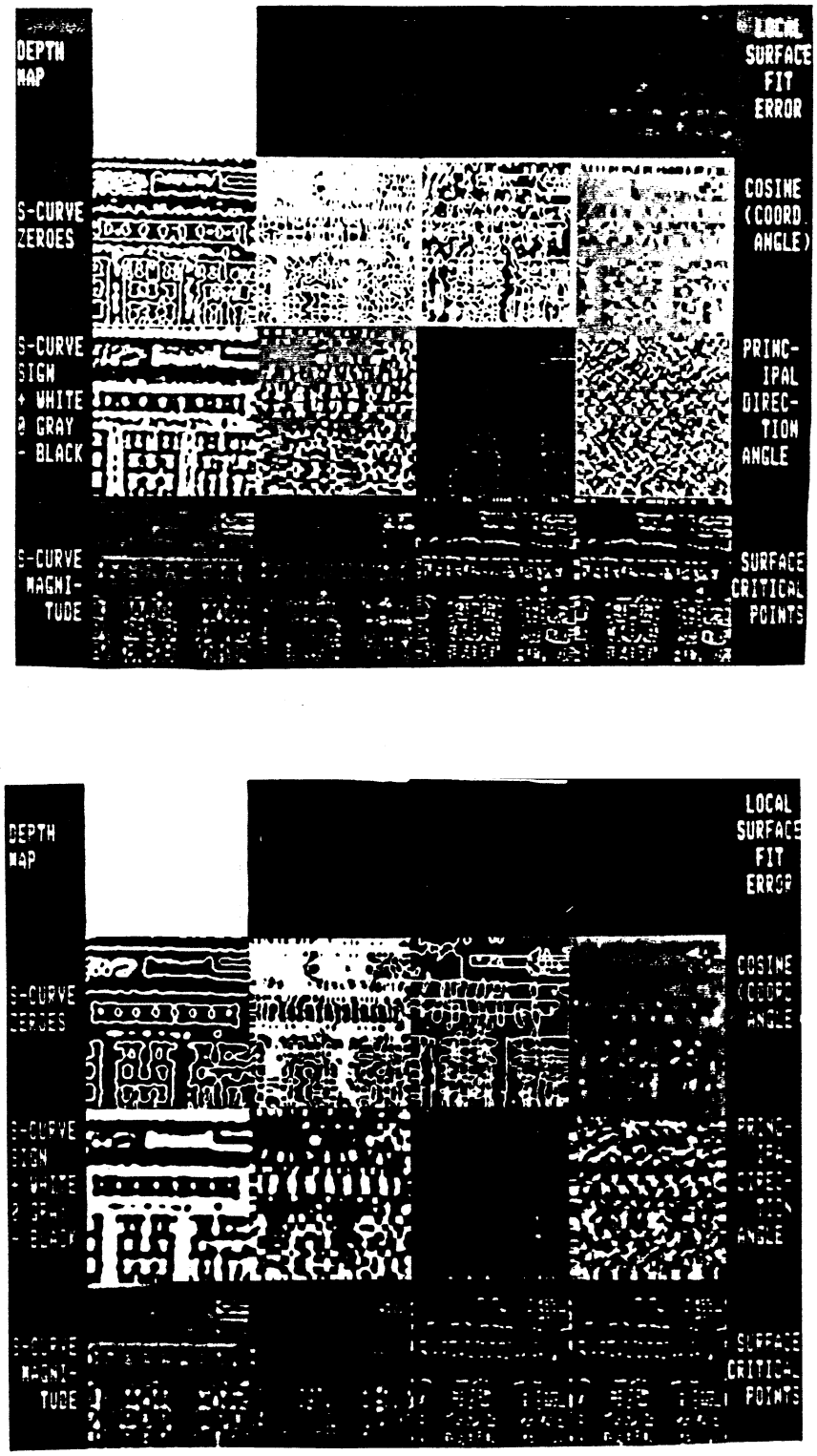


Figure 3.13. Surface Characterizations of Keyboard
(3x3 Window Operator Results - Top, 5x5 Window Results - Bottom)

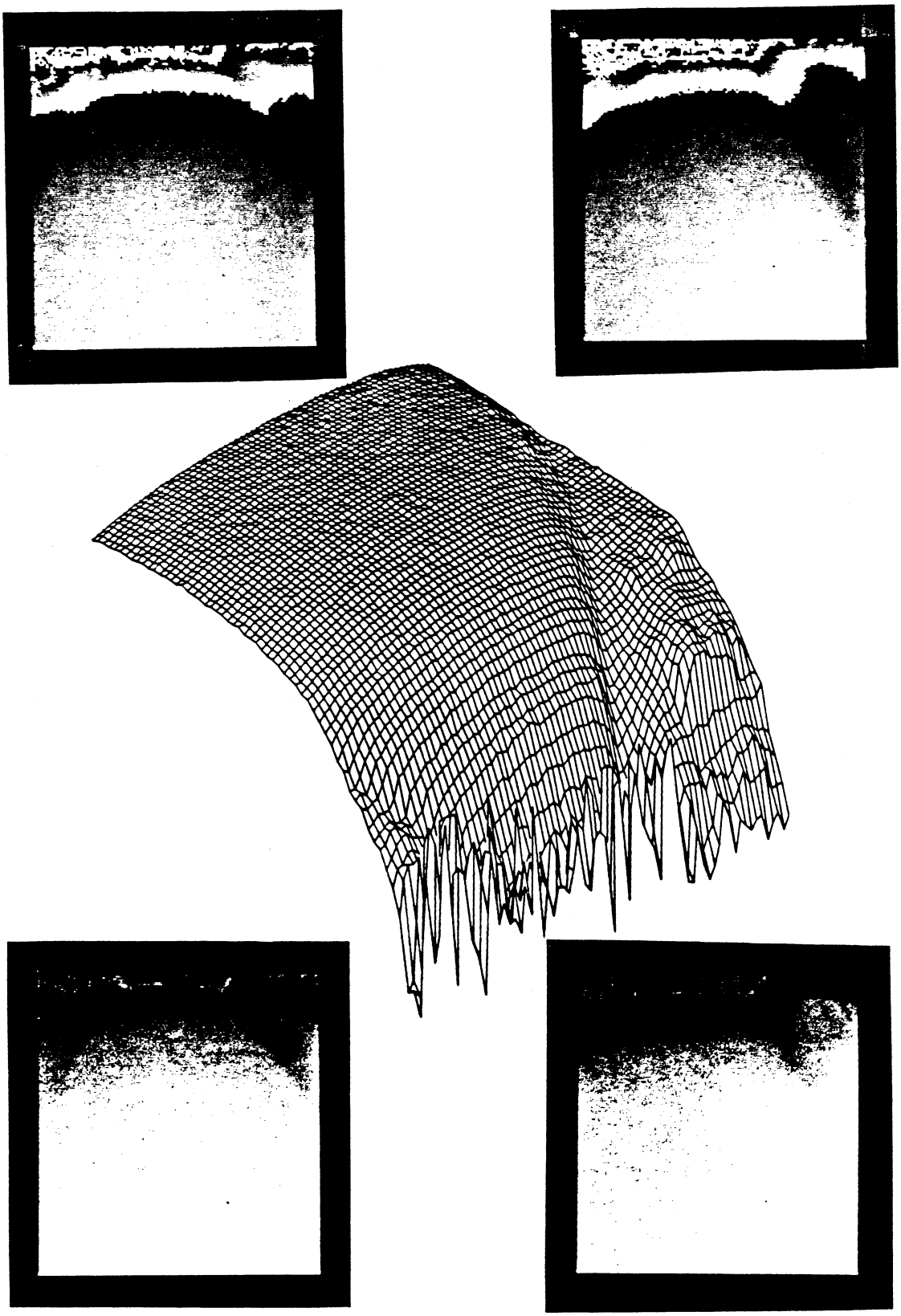


Figure 3.14. Original and Unwrapped Range Images of Road Scenes
(128x128 ERIM Range Images)

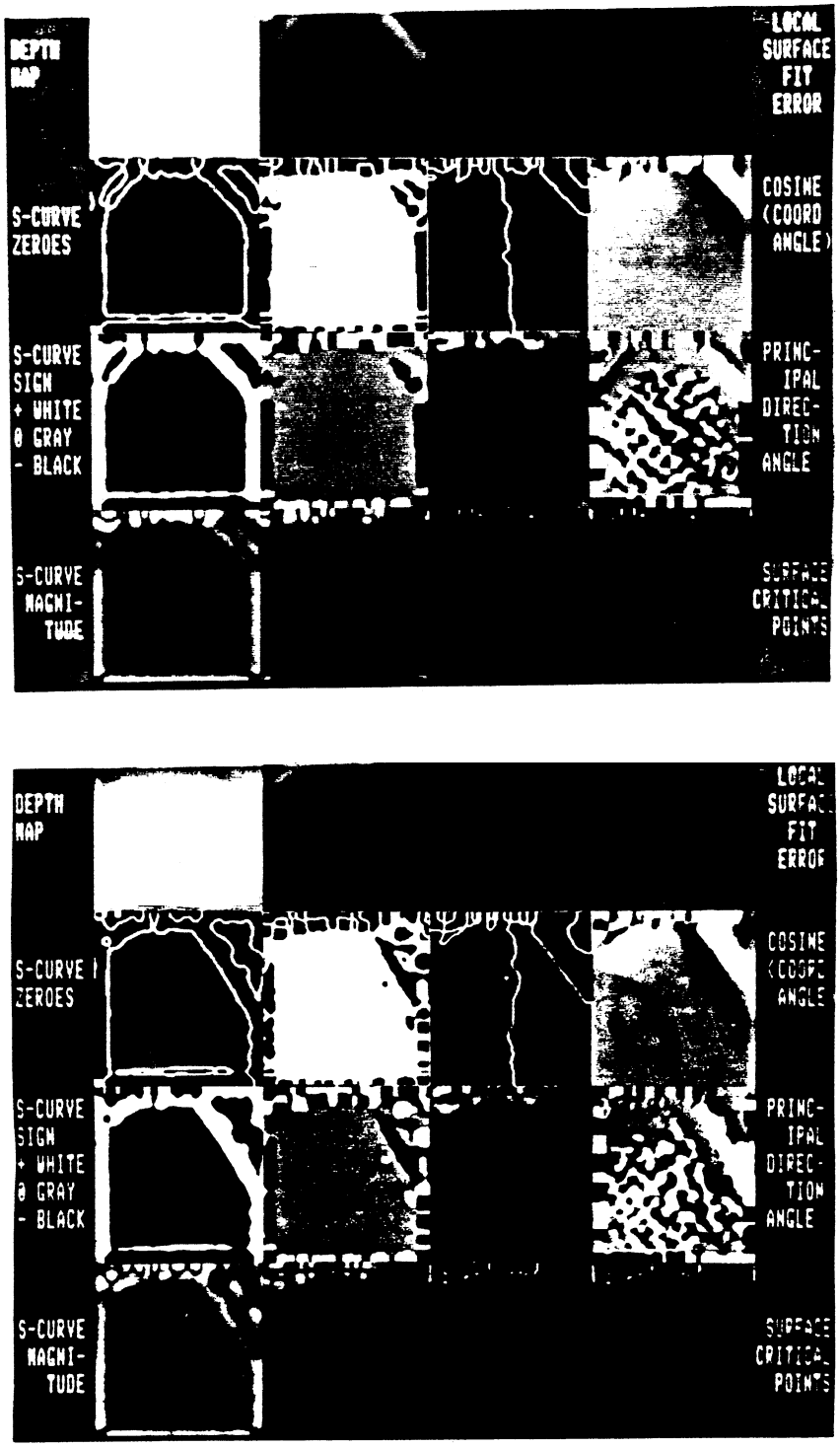


Figure 3.15. Surface Characterizations of Road Scenes
(9x9 Derivative Window Operator, Zero Threshold=2%)

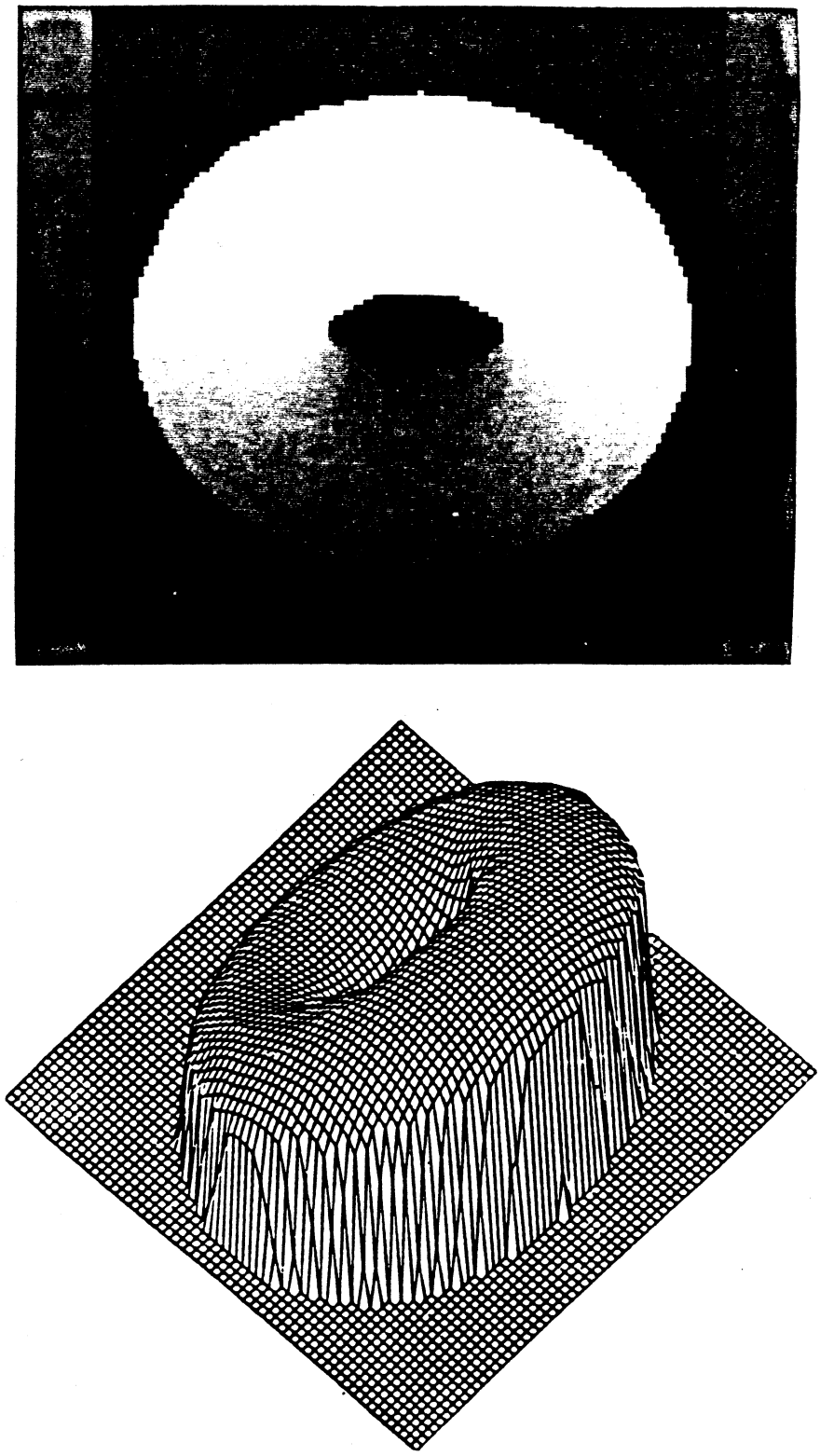


Figure 3.16. Range Image and Surface Plot of Tilted Torus
(128x128 Synthetic Depth Map)

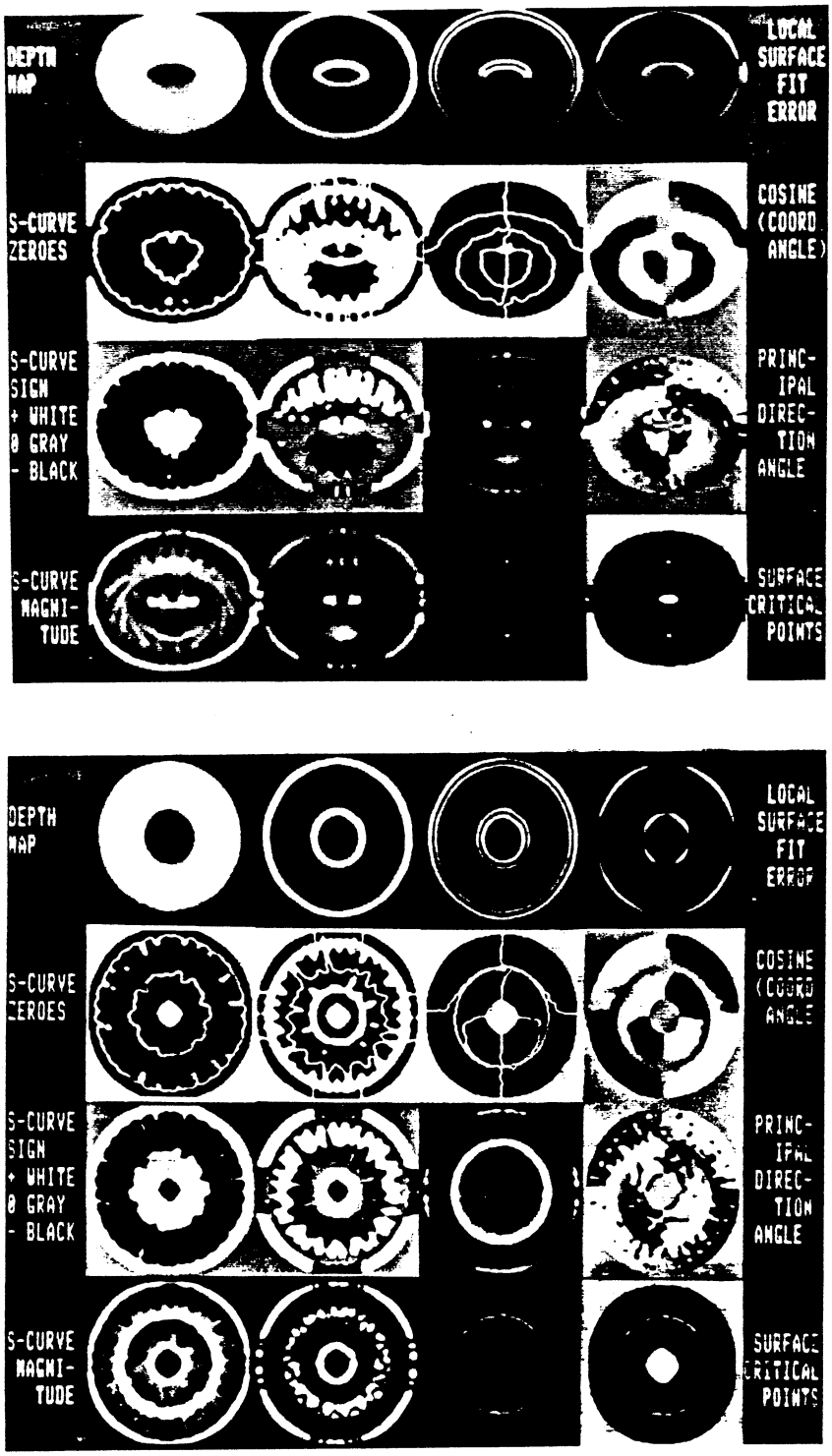


Figure 3.17. Surface Characterizations of Two Views of Torus
(5x5 Derivative Window Operator, Zero Threshold=1%)

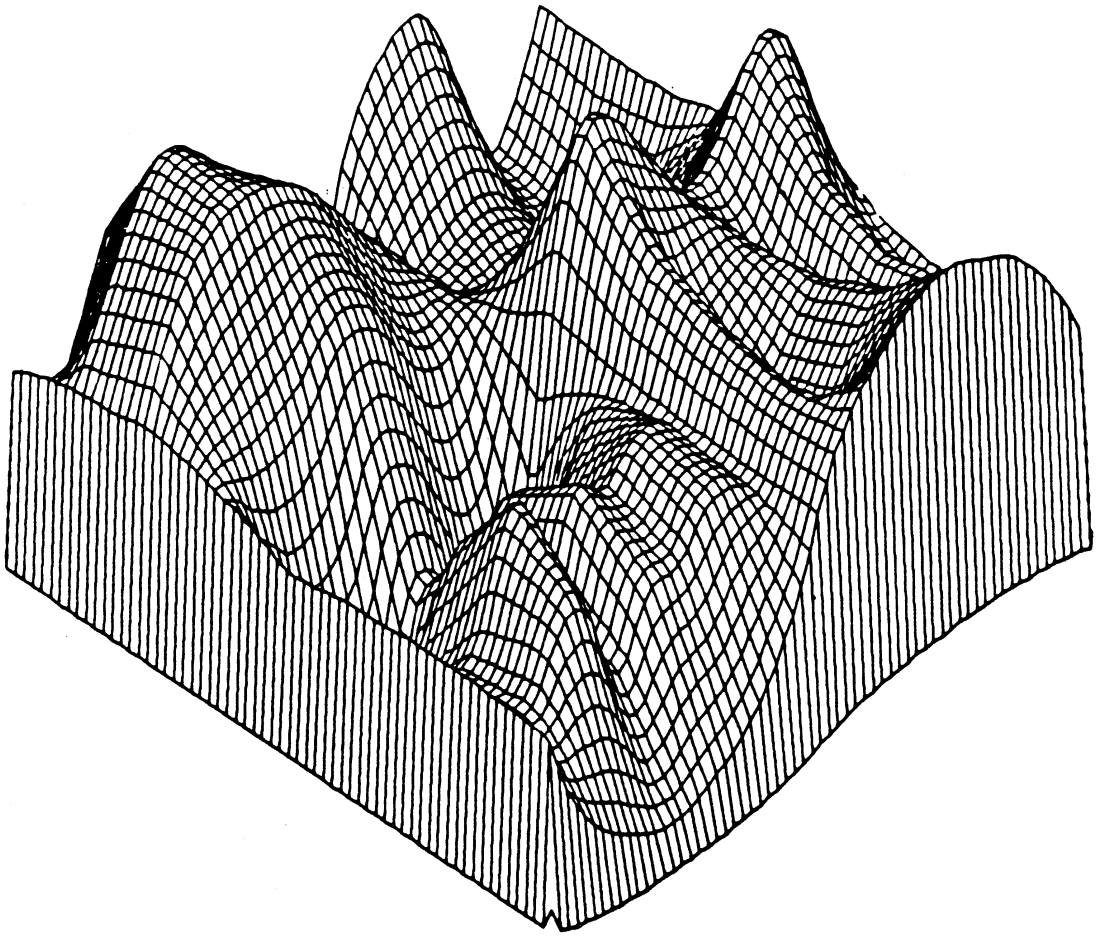


Figure 3.18. Surface Plot of Undulating Surface Range Image
(128x128 Synthetic Depth Map)

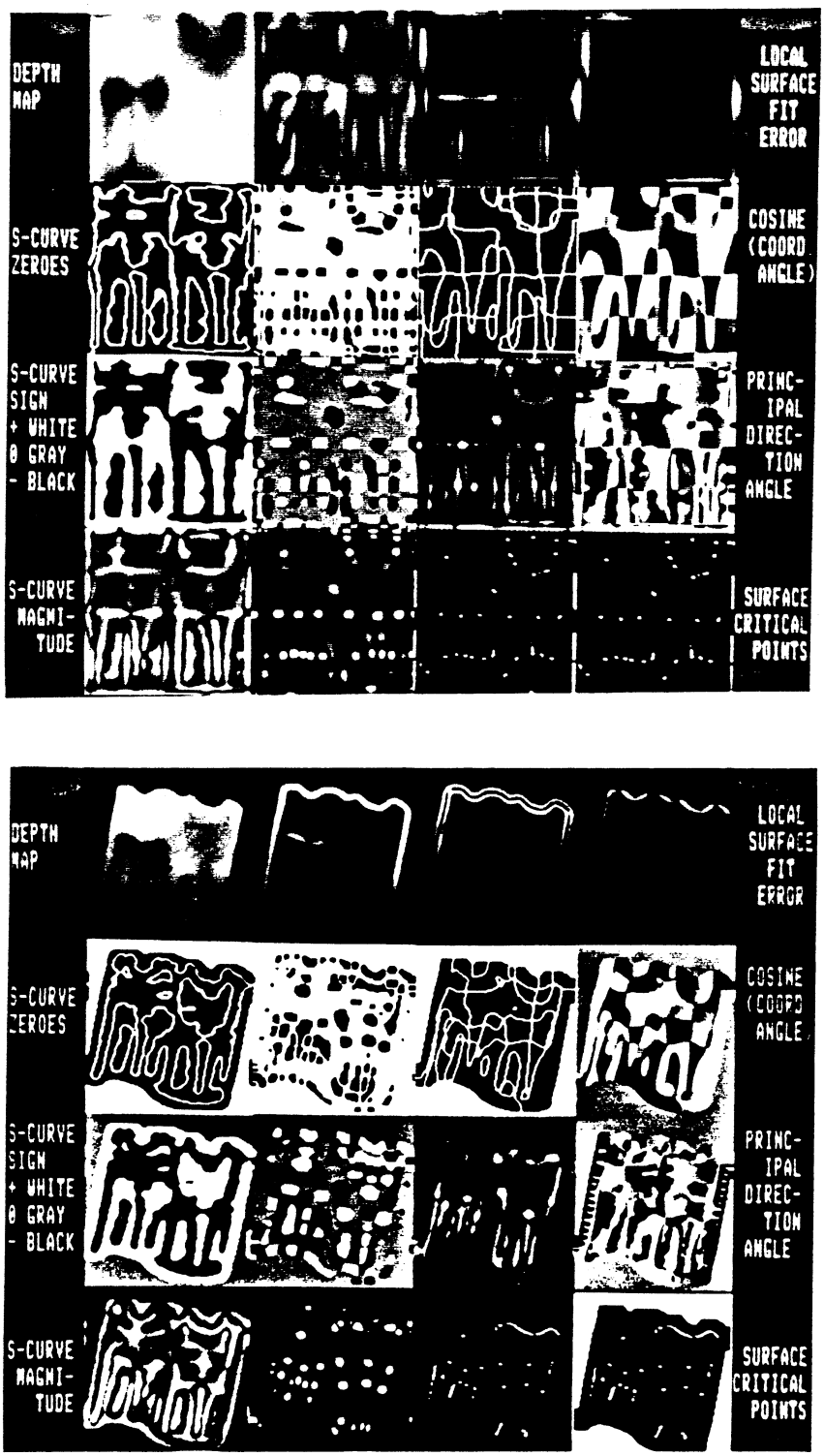


Figure 3.19. Characterizations of Two Views of Undulating Surface
(5x5 Derivative Window Operator, Zero Threshold=1%)

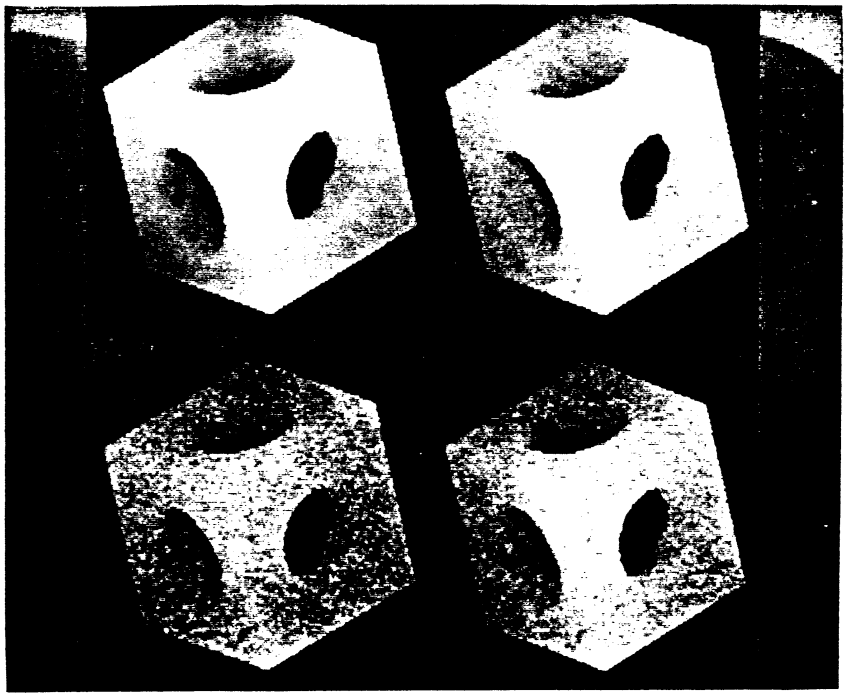


Figure 3.20. Block with Different Noise Levels (2.3,9.2,16,22.9)
(128x128 Synthetic Depth Maps)

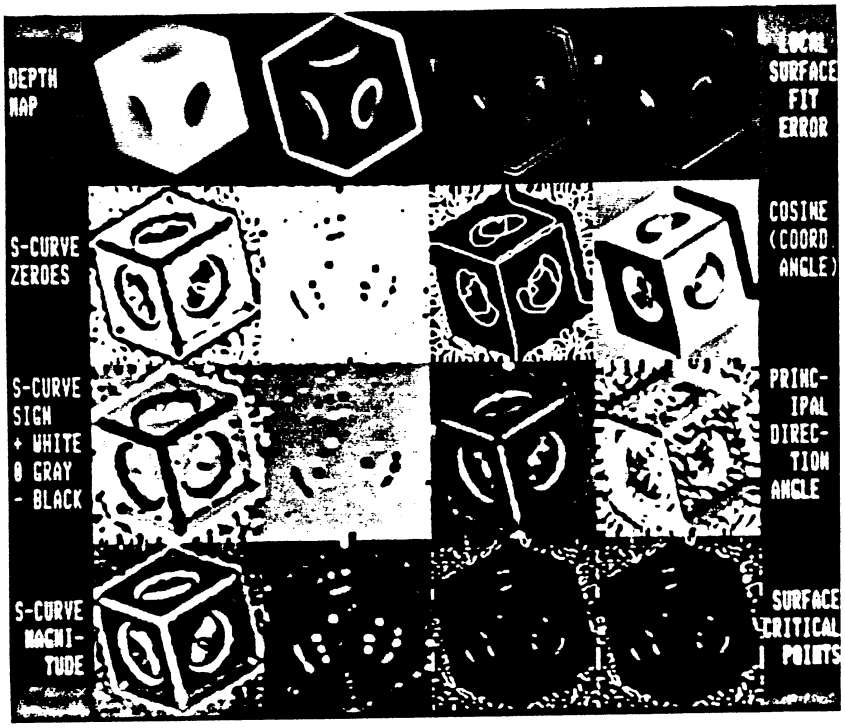


Figure 3.21. Results for 5x5 Operator with Sigma=2.3
(Zero Threshold=6%)

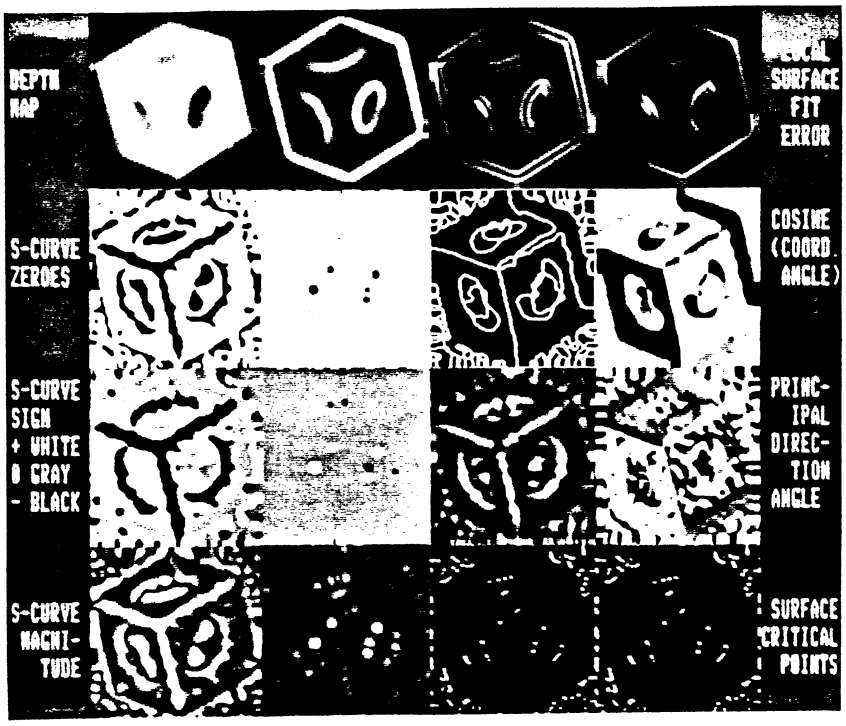


Figure 3.22. Results for 7x7 Operator with Sigma=9.2
(Zero Threshold=12%)

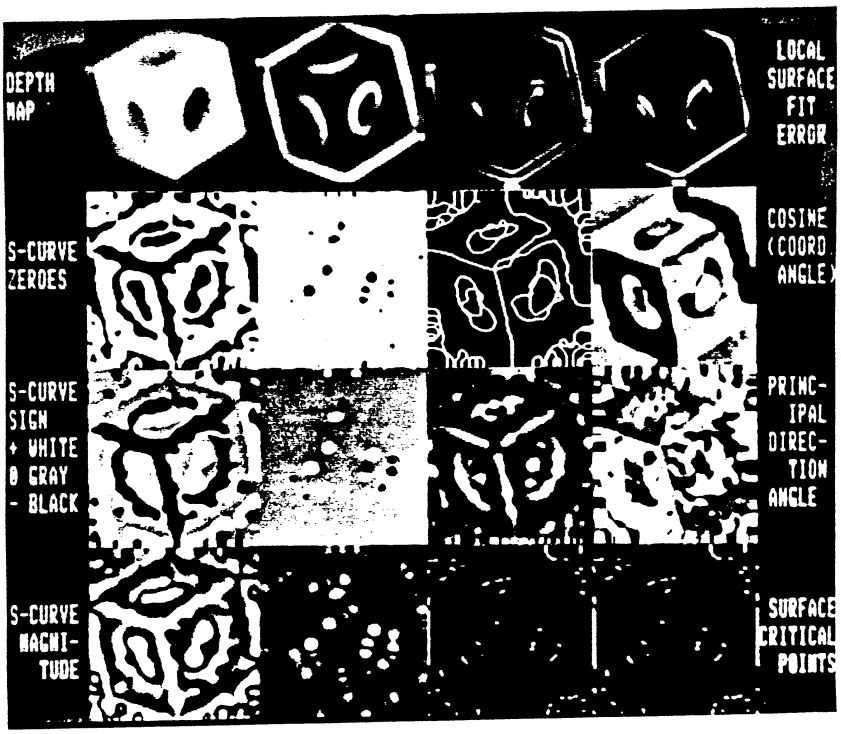


Figure 3.23. Results for 9x9 Operator with Sigma=16.0
(Zero Threshold=14%)

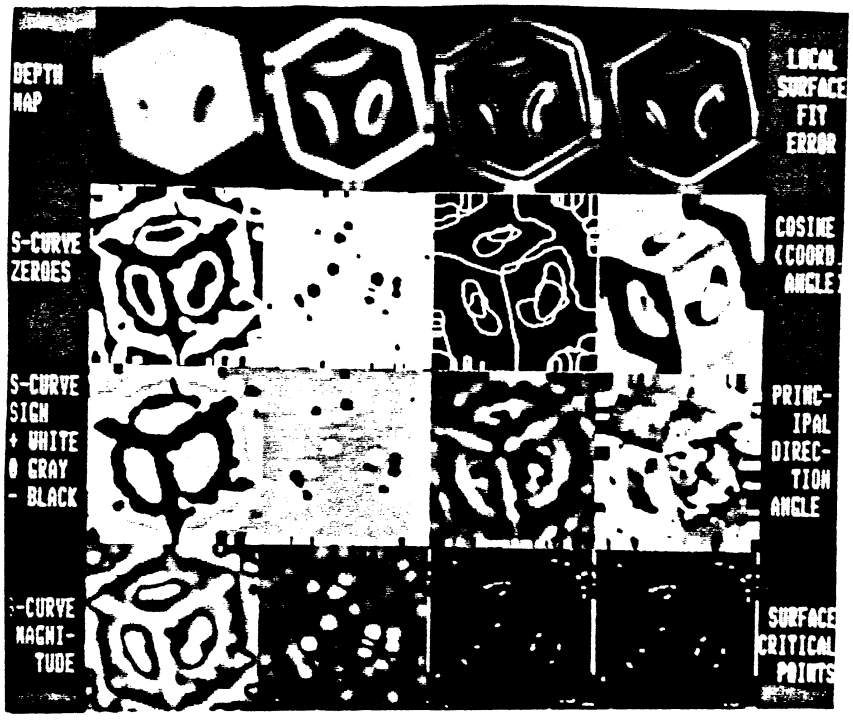


Figure 3.24. Results for 11x11 Operator with Sigma=22.9
(Zero Threshold=14%)

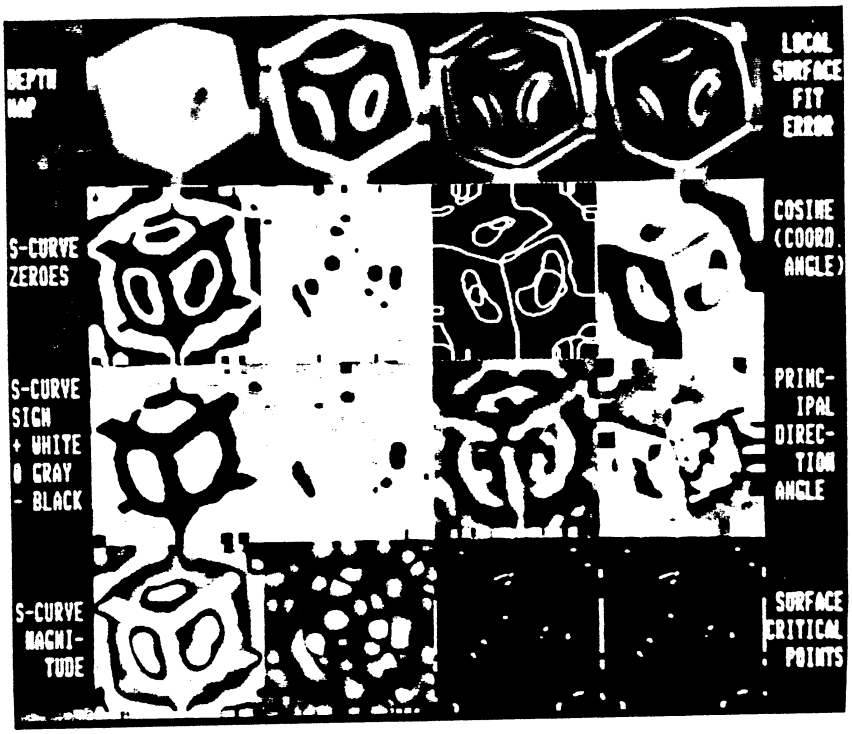


Figure 3.25. Results for 13x13 Operator with Sigma=22.9
(Zero Threshold=14%)

CHAPTER 4

FROM SURFACE LABELS TO SURFACE PRIMITIVES

An analytical framework for object recognition and image segmentation was formulated in Chapter 2. It was proposed that objects can be recognized in range images through surface characterization, surface segmentation, and surface matching. Although surface matching is inherently model-dependent, it was proposed that surface characterization and surface segmentation do not require knowledge of objects, only basic knowledge of surfaces. It was shown in Chapter 3 that differential geometric concepts for visible-invariant descriptions of continuous surfaces are also applicable to digital surfaces. That is, each point on a surface (continuous or digital) can be characterized by the spatial properties of other points on the surface in small neighborhoods surrounding the given point. One key difference is that, in the continuous surface case, the neighborhood of a point consists of an uncountably infinite number of points whereas a small finite number of points form the neighborhood of a digital surface point.

As the name implies, the *differential* geometry of surfaces analyzes the *local differences* of surface points. Although *global similarities* in surface structure are also analyzed within the context of differential geometry, most existing theorems address only global topological similarities, such as the one-hole equivalence of a doughnut and a coffee cup with a handle. However, there are global shape similarity theorems for the surfaces of convex objects. The appropriate mathematics has already been successfully

incorporated in Extended Gaussian Image (EGI) shape matching schemes (e.g. [Horn 1984]). Classical mathematics gives little guidance for computational matching methods if local geometric descriptors are used to identify the shape of arbitrary *non-convex* objects from arbitrary-viewpoint range-image projections. Although special feature recognition approaches offer important advantages for applied computer vision systems (e.g. [Horaud and Bolles 1984]), a successful surface-matching approach based on arbitrary surfaces (e.g. [Potmesil 1982]) will provide considerably more general object recognition capabilities. The aim is to use local difference information to help describe global similarities in surface points (range image pixels) for arbitrary surfaces without making domain-specific assumptions.

If the pixels of a digital surface can be correctly grouped into smooth surface regions that directly correspond to the surfaces of objects in a scene, this grouping process would provide a fundamental service to higher level matching (recognition) processes. This is the basic tenet of the proposed range image segmentation approach. In this chapter, a process for converting the local difference information in the HK-sign map into global similarity information is presented. The specific form of the desired global similarity information was set forth in Chapter 2. The goal is to convert the original digital surface into a set of 2.5-D graph surface approximants $\{\hat{g}_i(x,y)\}$ and an associated set of 2-D region descriptions $\{\hat{R}_i\}$ as guided by the surface characterization results.

4.1. Problems with HK-Sign Maps

It has been established that mean and Gaussian curvature possess many desirable properties for characterizing the 3-D shape of smooth surfaces, especially graph surfaces, and methods for computing mean and Gaussian curvature from digital surfaces through

least-squares derivative estimates have been examined. Experimental surface characterization results have been presented showing that meaningful differential geometric quantities can be computed even in the presence of noise by using an appropriate pre-smoothing operator on range images. However, it is still unclear at this point that a more precise data-driven range image segmentation can be achieved using this computed information unless higher resolution range sensors are used. Three important observations are made about the results obtained thus far using the surface curvature characteristics approach:

(1) **SMOOTHING:** Smoothing is required to even out the local fluctuations in order to obtain reasonable differential-geometric quantities from digital surface data. As a result of standard low-pass filter smoothing, sharp discontinuities in depth and orientation are blurred. The HK-sign surface labels, as currently computed, reflect only the geometry of the smoothed surface and not the original surface data. The final data-driven description, however, should be as precise as possible about the original image and should not contain misleading information about the original digital surface shape. Hence, the raw visible-invariant pixel labeling results obtained via smoothing, derivative estimation, and surface curvature computation must be refined into a more precise form to be useful. It is not clear how this should be done, but it is a fundamental problem that must be addressed. Research by several investigators [Terzopoulos 1983,1985] [Grimson and Pavlidis 1985] has explored the idea that the unwanted effects of smoothing can be eliminated, or at least significantly attenuated, by using adaptive smoothing and derivative estimation operators. That is, window operators may change shape and size depending upon local data variations. Figure 4.1 shows an example of a simple adaptive smoothing filter (developed by the author, but not discussed here) that maintains

the sharpness of the underlying step function by incorporating an estimate of the standard deviation of the measurement noise. Note that the data has a different step shape in the fixed-window-size smoothing cases, but it is not distorted near the discontinuity by the adaptive smoother. Such operators are successful when the measurement noise is not large [Grimson and Pavlidis 1985]. In this thesis, it will be shown that sharp depth and orientation discontinuities can be preserved (and even enhanced in the presence of noise) in a high-level data-driven range image description even though a fixed-window-size, fixed-window-shape smoothing operator is used to compute sign-of-curvature features. This is accomplished by isolating smooth-surface-primitive seed regions on both sides of a discontinuity and subsequently growing the seed regions outward. Pixels on either side of the discontinuity tend to become part of their respective, distinct surfaces. This process will become clearer by the end of the chapter. Figure 4.2 shows the basic concepts for a step

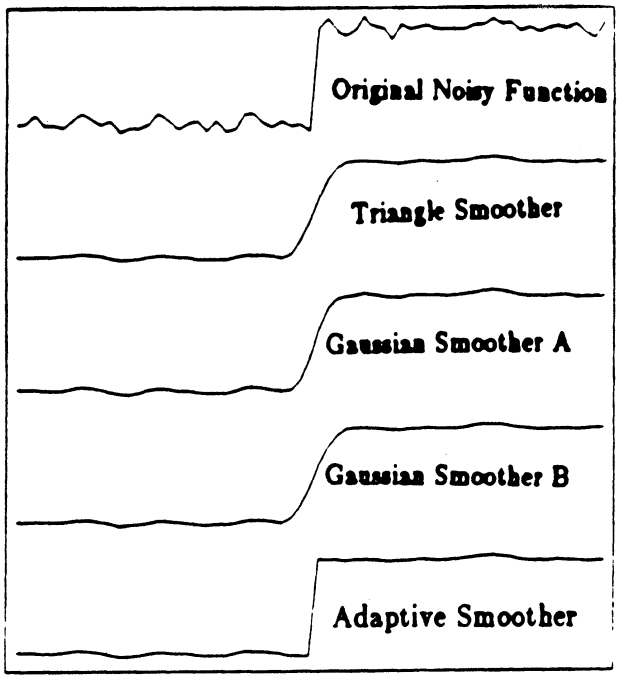


Figure 4.1. A Comparison of Different Smoothing Operators

edge, including smoothing, differentiation, sign of curvature, seed regions, and iterative fitting. Although smoothing algorithms that inhibit smoothing over depth discontinuities are useful, other approaches are also possible. Excellent results have been obtained on real image data as discussed in Chapter 7.

(2) **UNWANTED CONNECTIONS CAUSED BY NOISE:** In the presence of noise, HK-sign surface labels of one surface region tend to connect (in the sense of four-connectedness) with equivalent labels of neighboring, but distinct, surface regions. This fact is observed in the actual experimental results with real and synthetic data and was not anticipated by the theory for smooth curved surfaces. Therefore, it is not possible, in general, to simply isolate a four-connected region of pixels of a particular HK-sign type and identify that region as a single surface of the appropriate type. Even if it were possible to detect the unwanted connections

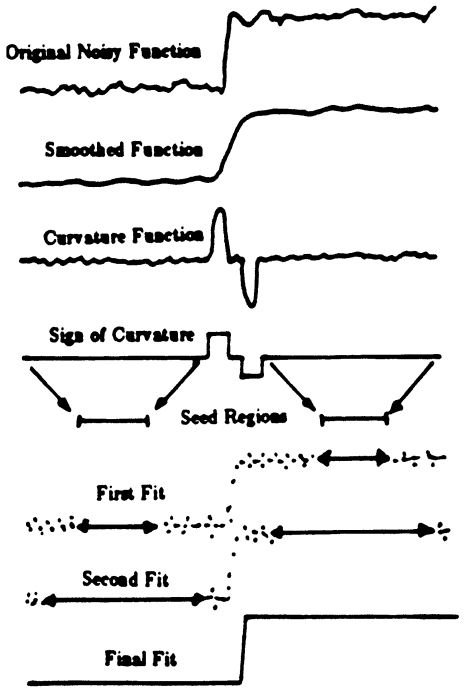


Figure 4.2. Basic Concepts of Surface-based Algorithm

and break them, the resulting connected regions are still found wanting because of their irregular shape, which does not accurately reflect the shape of the underlying surface. At this point, it is not clear if or how the HK-sign information can be used to provide a decomposition of digital surfaces into simple, basic surface primitives. It is proposed that connected regions should be isolated and contracted until a small, highly reliable, single-surface seed region is isolated. This seed region is then grown until it reaches its *natural limits* as defined by variations in the surface data. It will be shown that the HK-sign map provides useful surface primitive regions in conjunction with a seed region extraction algorithm and with an iterative region-growing algorithm based on variable-order surface fitting.

(3) **GLOBAL SURFACE PROPERTIES LACKING:** In order to perform surface matching between a range image description and a world model, it is advantageous to have an explicit, symbolic representation of a surface that possesses the global surface properties. A parametric equation for a surface is an example of an explicit symbolic representation. It is known that the mean curvature function of a surface in addition to a boundary curve for that surface can be used to reconstruct the original surface by solving an elliptic quasilinear partial differential equation. (This assumes that the original surface is sufficiently smooth.) Even though this is true in continuous mathematics, the differential equation can seldom be solved in closed form and numerical methods are required. Thus, significant computation may be necessary to "invert" mean curvature function information to obtain a description of the original surface even if the mean curvature function was known exactly. And, as the experimental results indicate, the estimates of the mean curvature function are good, but still noisy. Although surface curvature is extremely important for describing and processing surfaces in continuous and discrete form, it is

doubtful that accurate surface reconstructions could be computed by numerically solving partial differential equations.

Better mechanisms exist for describing global shape properties of digital surfaces. For example, if mean and Gaussian curvature are both zero on a given surface (which means that surface is planar), one might very much like to know the planar equation for that surface. A rich, symbolic, data-description must be able to provide such information. If all pixels corresponding to a given planar surface can be correctly grouped together, one could perform a minimum error fit of a plane to those pixels of the digital surface to obtain a good estimate of the planar equation of that surface. This would certainly be easier, more efficient, and more accurate than attempting to solve the minimal-surface partial differential equation numerically using the boundary pixels of the planar surface. This simple method may be fine for planes, but how can the shape of general, smooth, *curved* surfaces be described more directly and more precisely than through noisy estimates of the mean curvature function and the surface boundary? The original digital surface itself provides the shape information in a non-symbolic, high-dimensionality signal form. Although surface curvature is ***not*** the most convenient "handle" for many global surface shape properties, it is not clear what is. Past researchers have employed either polyhedral or quadric surface models to describe global surface properties, but have not successfully dealt with more complicated surfaces. Quadrics, though very useful, unnecessarily impose symmetry requirements on the image data, requirements that will not be satisfied by general surfaces. It will be shown that an effective, data-driven description of global properties can be computed using a small set of approximating functions and still allow flat surfaces to be flat and curved surfaces to be curved without imposing high-level conical or cylindrical

models as past research has done. For the purposes of grouping pixels into smooth surface regions using basic fitting techniques, first, second, and fourth order bivariate polynomials are used to quickly provide approximate parametric descriptions of graph surface regions.

These items summarize the main shortcomings of the HK-sign visible-invariant pixel-label surface-characterization output, and briefly introduce the approach described subsequently. The seemingly paradoxical motivation to compute a digital surface description for object recognition purposes without knowing anything about specific objects or even object classes prohibits us from bringing in model-based techniques to attempt to counteract the above difficulties. If suitable surfaces can be fitted to any possible HK-sign region, such surface fits would provide the desired general global surface information in symbolic form. As shall be shown, it is also possible to remove the effects of smoothing and unwanted connections using a post-surface-characterization, variable-order surface fitting approach to region growing.

4.2. Segmentation Algorithm Philosophy

A *stimulus bound* philosophy is proposed for computer vision algorithms in general and for range image segmentation and range image object recognition in particular as shown in Figure 4.3. It is recognized by many researchers that computer vision algorithms should function at several different levels using interacting vision modules to process signal and/or symbol information at each level. However, it is often implicitly assumed that each level's vision module will accept input only from the previous, lower level and provide output only to the subsequent, higher level. This assumption appears to be rooted in human visual models where retinal information is not directly available to the high level cerebral processes. However, human vision is a fundamentally dynamic

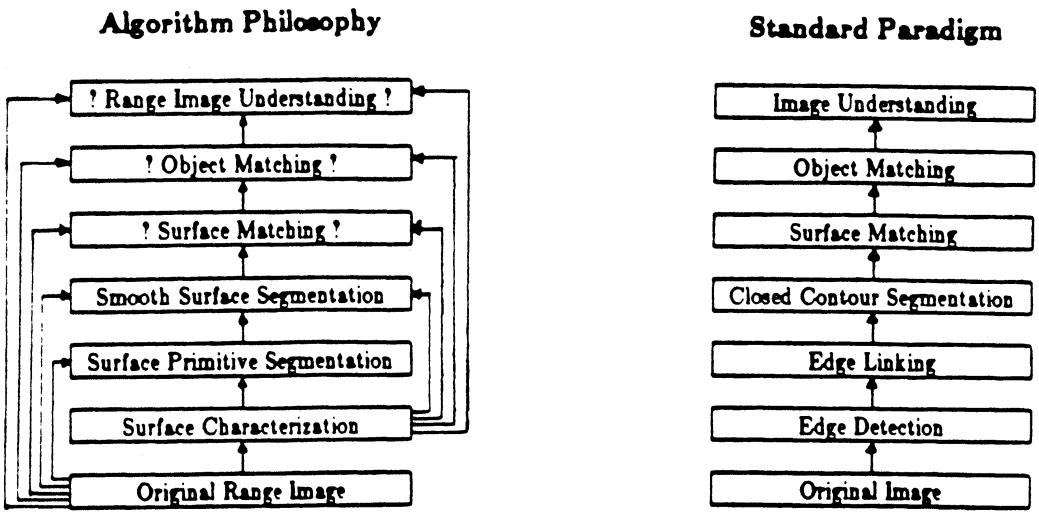


Figure 4.3. Stimulus-Bound Approach vs. Conventional Approach

perceptual process in which subsequent, highly-correlated "video frames" are always immediately available to the visual system after any given instant in time. Therefore, it may be inappropriate to apply dynamic human visual model principles to static computational vision problems. The stimulus bound philosophy states that the output from all lower level vision modules, including the sensor itself, should be available to any high-level vision module. In particular, the original image must be available to every vision module in a static scene vision system. An algorithm is *stimulus bound* if each stage of the algorithm uses the original input (the stimulus) and lower level inputs to influence the output of that stage. In Chapter 2, the rendering process, which provides high-to-low level feedback, was discussed as a process that computational vision systems should employ. If, in fact, there are only three levels in a vision system, Figure 2.3 is perhaps an adequate representation. However, assuming that a variety of intermediate levels must play a role in any successful vision system, the stimulus-bound philosophy is a more explicit expression of the rendering-feedback idea.

This topic has not arisen previously in this thesis because the surface characterization algorithm is the first level of processing and only has the original digital surface to work with. Confronted now with the higher-level segmentation problem, it must be decided if the segmentation algorithm is allowed access to the original image or if it must work with only the surface characterization output. Not allowing a higher level module to access the original image directly is self-contradictory because one is free to combine the given level module with all lower level modules to create a new lowest level module where all components of that new module have access to the input image. Hence, any attempts to restrict access based on module level arguments can obviously be bypassed by redefining the levels of a given system.

It is not that other researchers have serious reservations about using the original image data at higher levels (many algorithms do this), but just as many approaches seem to be locked into the standard, linear, vertical processing paradigm (also shown in Figure 4.3), and it seems worthwhile to belabor the point. Consider the ACRONYM system [Brooks 1983] for example, which involved several man-years of effort. Only the single, lowest level of that multilevel system has any direct contact with the original data. Moreover, the weakest components of the system all occurred at the lowest levels resulting in a non-robust system despite the major effort devoted to the highest system levels. Without feedback, a chain of interpretive vision modules is only as robust as the weakest module. A high-level manager that never checks the facts for himself is easily mislead by subordinate entities.

Another possible reason for the general lack of feedback in vision systems may be related to the dominance of the edge-detection paradigm in computer vision. In a surface-based approach, a hypothesized image surface may be verified by computing an

error measure over the individual errors made at each pixel in the image. This is easy because the sensed level at a pixel can be directly compared to the predicted level. In edge-based approaches, it is difficult to measure edge error directly against the original data because an explicit edge description is not present in the original data. This certainly does not imply that edge-based feedback is not possible, but only that it is less direct than surface-based feedback.

Therefore, the segmentation algorithm discussed below accepts the surface characterization output and the original digital surface as input. The next higher level should accept the segmentation output, the characterization output, and the original image as input. This process should continue as indicated in Figure 4.3 until the goals of the vision application have been achieved. The stimulus-bound philosophy is not generally followed by static scene computer vision researchers. This thesis presents an algorithm and empirical evidence in support of this philosophy in the area of early range image understanding.

4.3. Segmentation Algorithm Description

It is assumed that (1) an original range image and (2) the eight level HK-sign map are given as input to the segmentation algorithm. All other surface characterization results, though potentially useful, are ignored for segmentation purposes in this thesis. In practice, it will be advantageous to incorporate these other characteristics, especially edges, but the main goal of this thesis is to demonstrate the power of a purely surface-based approach. The results in Chapter 3 should be viewed as a statement of the quality of the differential geometric characterization method and an indication of potentially useful features that can complement the current segmentation approach.

The pixel label image (HK-sign map) has a reduced number of levels for each pixel compared to the original image, but retains the same spatial resolution. Hence, the label image is still a "signal" quantity even though it measures local neighborhood surface type rather than relative depth as in the original range image. But lower-dimensionality, symbolic information is needed. A standard connected-component algorithm (Appendix F) isolates four-connected regions of pixels of a fundamental surface type. As mentioned above, a given region may be undesirably connected to a distinct, adjacent region of the same type. Naive attempts to fit surfaces to connected regions may work occasionally, but fail miserably most of the time, especially in the presence of noise. It is possible to break these unwanted connections using a standard 3x3 contraction (or erosion) algorithm (Appendix F), but even then the HK-sign regions may still not correspond directly to meaningful surfaces because of preliminary smoothing distortions. A more elaborate strategy is called for.

In the surface-based segmentation algorithm, connected regions of pixels of a given surface type are extracted from the HK-sign map as ranked by decreasing size. Each extracted region is then contracted (eroded) until a sufficiently small number of pixels (greater than a fixed minimum value) is obtained in the largest connected subregion of the extracted region. This small region serves as the seed region to the iterative region growing algorithm. That algorithm is based on fitting variable-order surfaces to the original image data in the seed region and subsequent growth regions. The region growing process is controlled directly by (1) the surface fit error obtained at each iteration, (2) a pre-specified error tolerance, and (3) a regions test, which is a generalization of the one-dimensional runs test of nonparametric statistics. The iteration continues until the termination criteria are met at which point the computed surface region description is rejected or accepted. Rejection decisions cause the associated seed region to be marked

off in a writable copy of the HK-sign map so that subsequent attempts to use those pixels as a seed region will be prohibited. Acceptance decisions similarly cause the accepted surface region pixels to be marked off in the HK-Sign map copy. Accepted surfaces are also used to update (1) the image error buffer (E-buffer), (2) the best-fit so-far surface list (BFS-list), (3) the best-fit region label buffer (BF-RL-buffer), (4) the first-come, first-serve, within-tolerance region-label buffer (FW-RL-buffer), (5) the best-fit (BF) reconstructed image, and (6) the first-come, first-serve, within-tolerance (FW) reconstructed image. For each accepted surface primitive, the region-growing stage of the segmentation algorithm (the subject of this chapter) outputs (A) three different region descriptions in the form of the BF-RL-buffer, FW-RL-buffer, and the BFS-list, and (B) the coefficients of the graph surface (polynomial) equation and the fit errors (mean absolute error, RMS error, and the maximum error). Each of the three region descriptions may be considered as a single segmentation output, but better results are usually possible if these three descriptions are combined. These outputs are combined via a region-combination stage of the segmentation algorithm (described in Chapter 5) that computes a single region description for each surface primitive. The final stage (also described in Chapter 5) merges surfaces that join smoothly at their common boundary. A dataflow diagram for the entire process is shown in Figure 4.4. For easy reference, Figure 1.2 is duplicated here as Figure 4.5 to describe the sequential control flow of the algorithm.

Unmarked connected components of the HK-Sign map are considered sequentially as ranked by region size until the average error in the image error buffer falls below the prespecified error tolerance. If all unmarked connected components of the fundamental surface types above a given threshold size (30 pixels) have been tested by the algorithm and the image error threshold has not been met, the algorithm automatically coalesces all unmarked pixels into a "last-chance" binary image and reanalyzes all connected

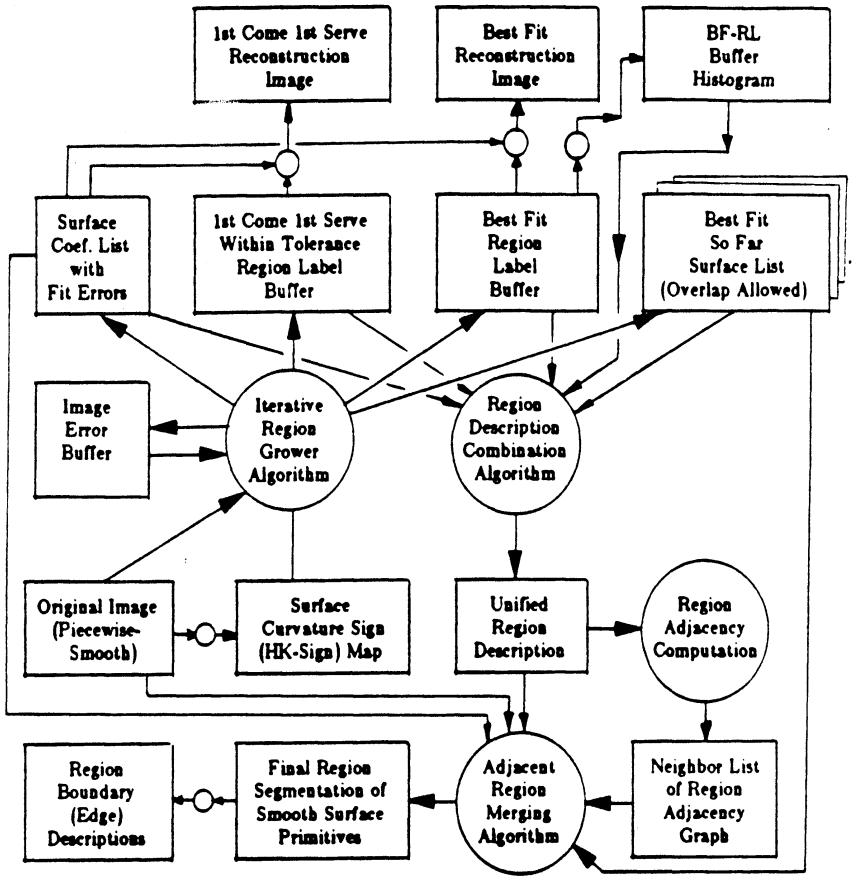


Figure 4.4. Dataflow Diagram for Segmentation Algorithm

regions in the last-chance image using the same approach outlined above. The last-chance image pixels are essentially regarded as another fundamental surface type that is processed separately from the eight fundamental surface types. The maximum fit error tolerance for a surface region acceptance is relaxed (doubled) at this last-chance stage in an attempt to recover useful information about the remaining pixels represented in the last-chance image that are, by definition, noisy or misgrouped. Eventually, the algorithm stops either when the pre-specified average image error threshold is met over the entire image or when all regions above the minimum region size threshold (30) in the last-chance image have been completely processed. At this point, the best-fit reconstructed image is very similar to the original image (if it possesses the surface coherence

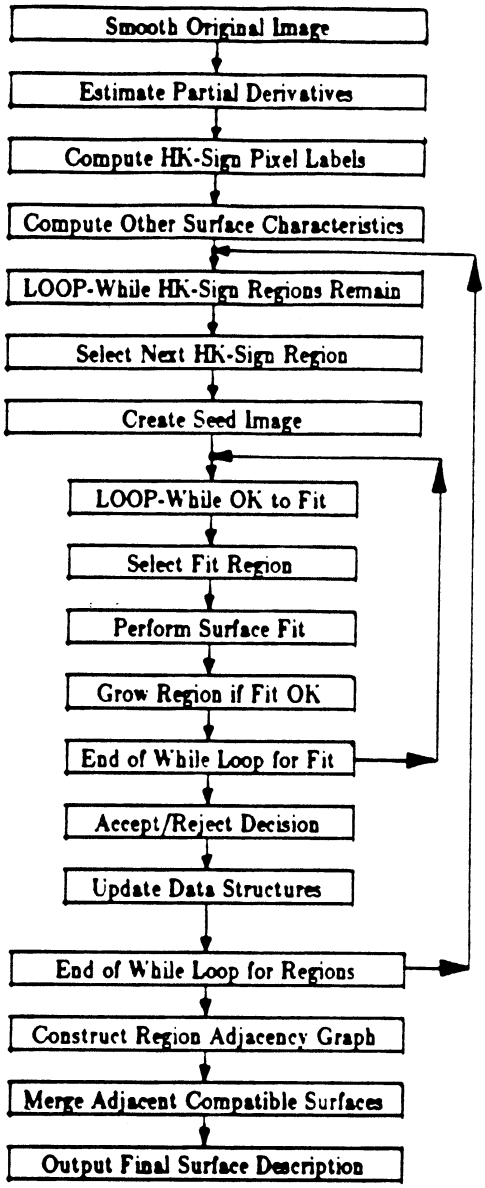


Figure 4.5. Sequential Control Flowchart

property), and a list of surface regions has been generated that qualify as a segmentation of the digital surface into approximate HK-sign surface primitives. However, smooth complicated surfaces, such as the visible surface of a torus, may be segmented into several regions due to the limitations of the approximating functions and the nature of the HK-sign surface primitives. Adjacent surface regions that join smoothly at the boundary separating the regions are then merged to create the final segmentation in

terms of smooth surface primitives.

Each part of the region-growing segmentation algorithm is described in detail subsequently. The material is divided into main sections on seed region extraction, variable-order surface fitting, parallel region growing, termination criteria, and surface acceptance and rejection decisions with data structure updates. The combination of the three region descriptions and surface merging at smooth surface joins are the subjects of Chapter 5. These algorithms are treated as a separate higher level vision module. Before entering into the full algorithm description, two preliminary discussions on approximating functions and error tolerance specification must first be presented.

4.4. Approximating Function Selection

As discussed in Chapter 2, the range image segmentation should consist of a set of approximating functions and a set of disjoint 2-D regions, one for each meaningful surface primitive in the original image. As indicated explicitly in Chapter 3, the eight fundamental surface shapes of the HK-sign surface primitives are relatively simple and exhibit even-order behavior. Arbitrarily complicated smooth surfaces can be decomposed into a disjoint union of such surface primitives. Hence, if these surface primitives can be approximated well, a composite surface description for arbitrary surfaces can be obtained. For dimensionality reduction, the approximating functions should be representable by a relatively small amount of data. For generality, the approximating surfaces must be well-defined over arbitrary connected regions. As stated in the description above, surface primitives are to be involved in an iterative region growing process. The approximating functions of the digital surface segmentation must be useful for extrapolation into neighboring areas of a surface in order for the region growing method to be successful. Interpolating capabilities are also useful. In addition, if the iterative

surface fitting approach to region growing is to be useful in the real world, the approximating functions must be able to be computed in a short amount of time on suitable hardware given the data to be fitted. The approximating functions should also be easily differentiable so that differential geometric shape descriptors can be recomputed from them and so that higher level processes may compare surface normals and other differential quantities. Finally, the set of approximating functions should be totally ordered (as opposed to partially ordered) so that each approximant is capable of describing lower order approximants exactly, but cannot approximate higher order functions.

These observations can be summarized in the following list of requirements. Note that for digital surface description purposes, general 3-D surface representation capability is not needed because digital surfaces are discrete representations of graph surfaces. A small finite ordered set of approximating surface functions is sought that

(1) Can approximate any smooth surface of constant HK-sign well over any type of connected domain,

(2) Can extrapolate accurately to arbitrary locations outside the data set used for fitting (this capability permits region growing),

(3) Allows a quick, computationally efficient surface fitting algorithm to be used,

(4) Can be represented by a small amount of data,

(5) Allows computation of partial derivatives for each member of the set,

(6) Can interpolate to locations within the fitted region.

Bivariate polynomials satisfy all of the above requirements, but what about other types of surface functions? Rational polynomial surface functions were investigated since they are able to provide a better fit to data than non-rational polynomials with the same number of coefficients. However, the rational fitting algorithms require substantially

more computation than non-rational approaches. Moreover, special care must be taken to avoid denominator zeros during surface extrapolation. Tensor-product spline surface functions are another possible choice for approximating surface functions, but arbitrary connected domain and extrapolation requirements present difficulties for these surface types. Quadrics and super-quadrics do not provide the explicit parametric forms that are desirable for fitting purposes. Other types of surface functions found in computer graphics and other literature, such as bicubic surface patches, triangular patch surfaces, Coon's patch surfaces, Steiner surface patches, Shepard's method interpolants, and hyperbolic multiquadrics, all present different sets of problems because they are mainly used for purposes distinctly different from what is needed here [Besl and Jain 1985]. Therefore, low-order least-squares-fitted bivariate polynomials have been chosen to satisfy the above requirements (Appendix C). It has been found that these polynomials work quite well for perceptual organization tasks with a comparatively miniscule amount of mathematical machinery. With all factors considered, they are the ideal choice for establishing the feasibility, the general characteristics, and the performance potential of the data-driven digital surface segmentation algorithm. If a better set of approximating functions were to be made available to the algorithm along with the appropriate fitting algorithm, the entire approach and theoretical arguments would still carry through with little or no change. To maintain generality in the algorithm statement, it is assumed only that there exists a set of approximating functions, denoted as $\mathbf{F}$, that contains $|F|$ discrete types of functions that can be ordered in terms of the "shape potential" of each type of surface function relative to the set of fundamental HK-sign surface primitives. Let $\mathbf{P}$ be the parameter space for the highest order function.

First-order (planar), second-order (biquadratic), and fourth-order (biquartic) bivariate polynomials are proposed as the set of approximating functions for HK-sign surface

primitives. Third-order (bicubic) and fifth-order (biquintic) polynomials add little surface representation capability for this particular problem because of the even-order (roughly symmetric) nature of the HK-sign surface primitives, and are therefore omitted from the set of approximating functions. Bivariate polynomials of order six and higher are avoided for data-fitting purposes owing to potential oscillatory and numerical difficulties. Therefore, $| F | = 3$ and the set of approximating functions $\mathbf{F}$ can be written in the form of a single equation:

$$\hat{g}(x,y) = a_0 + a_1 x + a_2 y + a_3 xy + a_4 x^2 + a_5 y^2 + a_6 x^2 y + a_7 x y^2 \quad (4.1)$$
$$+ a_8 x^3 + a_9 y^3 + a_{10} x^3 y + a_{11} x^2 y^2 + a_{12} x y^3 + a_{13} x^4 + a_{14} y^4 .$$

Planar surfaces are obtained by restricting the parameter vector space $\mathbf{P} = \mathbf{R}^{15}$ to three-dimensional subspace; biquadratic surfaces are restricted to a six-dimensional subspace. A least-squares surface fitting algorithm can compute the parameter vector $\mathbf{a}$ and the RMS fit error ϵ directly from the digital surface data in a region quickly and efficiently (Appendix C). Moreover, a least-squares approach allows surface region fits to be updated incrementally during the region growing process as new data points become available or as higher order approximations are needed [Golub and van Loan 1983].

As with any set of approximating functions, there are problems with these three bivariate polynomial surfaces. Many HK-sign surface primitives exist that cannot be exactly represented by bivariate polynomials. Perfect hemispheres and cylinder halves provide two good examples of common real-world shapes that fall into this category. It is argued that, for perceptual grouping purposes, a fourth-order polynomial can approximate a semicircle, or ellipse, closely enough that the casual observer cannot distinguish between them, especially without a reference figure at hand. For example, consider Figure 4.6(a), which compares an ellipse and a least-squares fitted fourth-order polynomial

(128 data points used). The shape approximation is really quite good: the average absolute error is 3.89 and the RMS error is 6.50 on a scale from 0 to 255. Consider also Figure 4.6(b), which compares an exponential function and a least-squares fitted fourth-order polynomial (128 data points used). The average absolute error is 2.34 and the RMS error is 4.87 on a scale from 0 to 255. Although exponentially shaped surfaces are not commonly encountered in the real world, the infinite series formula for the exponential is seen to be approximated quite well by only five terms (including the constant term) on a finite interval. A second-order approximation could not do nearly as well.

The main concern here is that the fit is good enough for perceptual organization tasks. Every application will have different types of relevant surfaces. In other words, if strong evidence is given that a peak surface primitive exists in a digital surface, the proposed bivariate polynomials can be used to group all the pixels of that peak surface into a coherent surface region, even if the underlying surface is a hemisphere. This statement regarding the hemisphere has been verified via experimental results and is given more

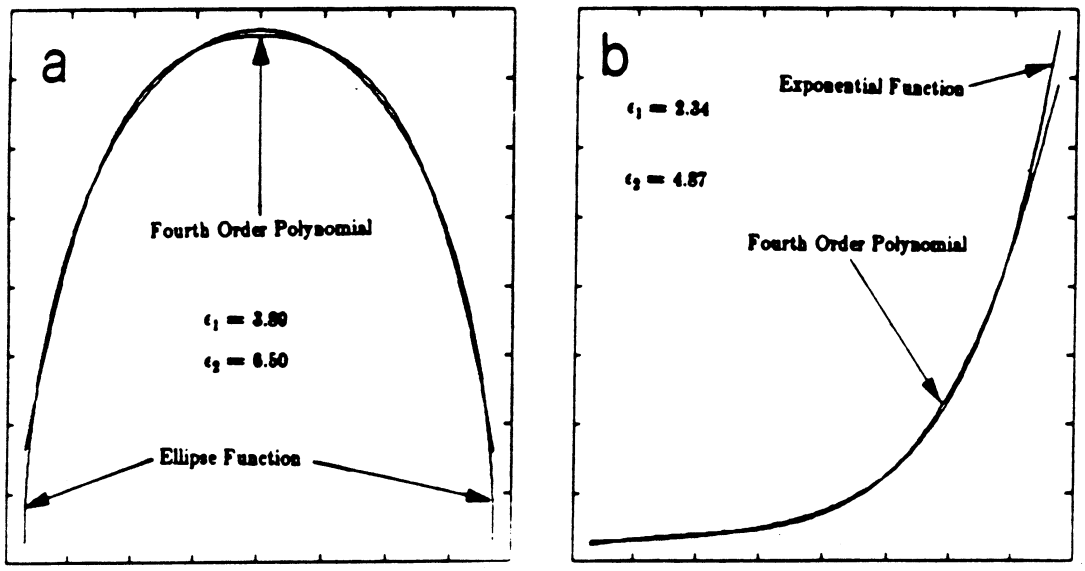


Figure 4.6. Ellipse & Exponential Compared to 4th Order Polynomial

attention later in this chapter. This type of approximate surface region description should provide no problems for the high-level processes that use it. For example, if a domain-specific algorithm were looking for a sphere, the existence of a round region that is well-approximated by a polynomial with approximately constant surface curvature would provide extremely strong evidence that a sphere is present. The surface matching algorithm is expected to use the object's surface model, the surface region description, *and the original data* in that region to verify that the surface is indeed a hemisphere.

4.4.1. Quadrics?

One might argue that a quadric surface function be included as one of the approximating functions since quadrics are so common in geometric modeling systems and in real world manufactured parts. There are several reasons not to include quadrics in this data-driven system. Although a quadric approximating function would handle problems with hemispheres or half cylinders, planar or quadric approximations to an exponential surface would not be nearly as good as the biquartic. Quadric approximations impose an implicit symmetry requirement on associated surface primitives, a requirement that cannot be expected to be met in a *general-purpose data-driven system.* In this sense, the biquartic polynomial approximant is more powerful (for grouping purposes) than the quadric approximant because the biquartic can come close to describing depth map projections of quadric surfaces, but quadrics cannot come nearly as close to describing other types of non-quadric surfaces that are also well approximated by the biquartic. Moreover, the standard least-squares quadric fitting approach (see e.g. [Hall et al. 1982]) does not minimize the error between the data points and the quadric surface as is done in polynomial surface fitting (see also [Faugeras et al. 1985]). Quadric fitting yields the best quadric model for the data, which does not necessarily yield the closest surface fit,

and is very sensitive to noise. Thus, it is necessary to go back and compute the fit error of the surface to the data when needed.

One might argue that a quadric approximant be included anyway along with the three bivariate polynomials, but then it is not clear what position the quadric would occupy in the total ordering of the approximating functions. If included, the quadric approximant would break the total ordering and require a Y-shaped partial ordering of approximating functions. Although possible, this would require a different theory from the theory developed here, which assumes a total ordering of approximating functions.

To summarize, 3-D quadric surfaces do not belong in a purely data-driven digital surface segmentation theory. The use of quadric surfaces should be delayed until later phases of processing. Quadrics are most appropriate at higher domain-specific matching levels to utilize and recognize certain types of symmetry in the data. For example, if the selected world modeler uses only planes and quadrics (as do many solid modelers), then quadrics will play a key role in the model-dependent surface-matching algorithm.

4.5. Noise Estimation and Error Tolerance Specification

Digital surfaces exhibit the property of "surface coherence" when sets of neighboring pixels are spatially consistent with each other; that is, a logical connection exists between neighboring pixels in that those pixels are sample points of some relatively smooth surface. This section discusses one way of estimating the average surface coherence in an image. The relationship of surface coherence to the error threshold that controls surface fitting and directly influences the overall image approximation error is discussed. Surface curvature zero thresholds, mentioned in Chapter 3, are also related to surface coherence. These issues are centered around the topics of *noise* and *signal quality*. In this section, a discussion about noise and the spatial extent of surfaces is

presented to complement the preceding section, which indirectly discussed the concept of surface shape as represented by approximating surface functions.

Sensor noise is a pervasive phenomenon. All sensors that operate above the temperature of absolute zero are at least subject to random thermal noise, not to mention other types of noise. Hence, algorithms that process real sensor data must be able to cope with noise in various forms. Theoretical approaches are sometimes developed that are useless in practical applications because signal quantities cannot be computed accurately enough. In communications theory, pure signals are transmitted over noisy channels to receivers that produce estimates of the pure transmitted signals. The additive noise model is commonly used to analyze this situation. Noise is modeled as a random process that can be measured when there is no signal. When a signal is sent, the random noise is modeled as simply being added to the signal. The job of the receiver is to reproduce the best possible version of the pure signal even though the received signal was corrupted by noise. The signal-to-noise ratio measures the ratio of the signal power in the absence of noise to the noise power in the absence of the signal. This single scalar quantity is used to measure the quality of received signals. Digital signal processing and digital image processing problems, especially edge detection, are typically formulated in similar signal and noise terms.

In Chapter 2, the digital surface segmentation problem was stated in terms of a signal plus additive noise model. What happens to signal quality concepts like the signal-to-noise ratio when the information in the signal is fundamentally geometric? It is still possible to compute power spectral densities in the 2-D spatial frequency spectrum and to compute noise power in bandwidths of interest, but geometric concepts are lost in this approach. One can clearly imagine an extremely high quality digital surface without

noise (i.e., high depth resolution samples of a smooth surface with no quantization and no added noise), but it is difficult to define or measure noise in the absence of a surface "signal" where range images are concerned. The reason for this difficulty is that flat surfaces orthogonal to the viewing axis appear in a range image as a constant plus measurement noise. Hence, the "no signal" state is equivalent to a "flat surface" state, which is a perfectly valid scene surface (signal). It is perhaps even more difficult to define the "power" in surface shape, which is the real content of a range image signal. Let us consider a specific instance of this problem.

Suppose range images are digitized for part of a step, part of a table top, and part of a curved surface, and assume the quantization noise and additive noise for these range images are practically equivalent. Figure 4.7 shows examples of the depth variations in these images as a function of position on a scanline. Standard edge detection models measure the signal strength of the step image as the height of the step. But how does one define signal strength for the table top and globe surfaces? How is the quality of a range image of a flat table top defined? It certainly appears futile to employ measures similar to the signal-to-noise ratio because the signal (as normally defined) is zero for a flat surface. Yet, if two 8-bit range images of a table top are given where both image A

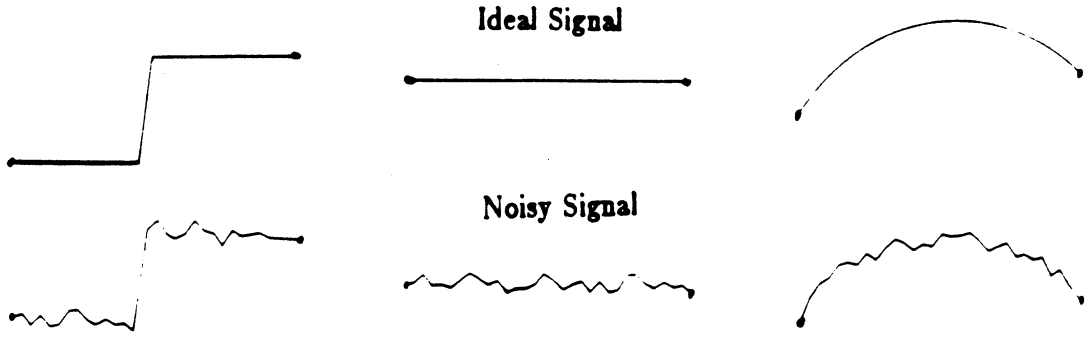


Figure 4.7. Depth Profiles for Step, Table Top, and Curved Surface

and image B have a mean value of 127, but image A has a standard deviation of 2 depth levels about the mean and image B has has a standard deviation of 40 depth levels about the mean, most people would say that image A is a higher quality range image than image B. It is assumed, of course, that the table is perfectly flat, and the range image is taken looking directly down on the table. See example depth profiles in Figure 4.8. Similarly, if a curved surface is substituted for the table top, one could still easily decide which range image was of higher quality by looking at the local fluctuations in the noise from pixel to pixel. Therefore, any quality measure of a range image of a smooth surface should be independent of the shape of that smooth surface. The quality measure should indicate properties of the measurement noise and not properties of the smooth surface. This in turn implies that quality measures of a range image are not interested in depth discontinuities or orientation discontinuities between surface regions, but rather in how smooth (how noisy) are the smooth surfaces (pure signals) in the image. A process that measures the quality of the step range image will then ignore the magnitude of the step and focus on the smoothness of the flat surfaces on both sides of the step. This is a complete reversal of standard edge detection models of image quality (signal quality).

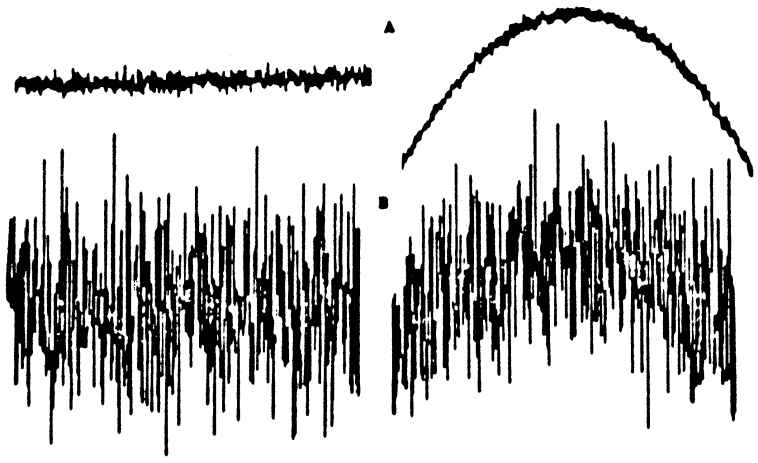


Figure 4.8. Different Noise Levels for Table Top and Curved Surface

Surface coherence is a concept that incorporates the "how smooth are the smooth surfaces in a scene" idea with the notion of the spatial extent of a surface. The key question is the following: How large must the visible surface area of a visible surface be in order that the visible surface may be said to be "coherent" (spatially consistent) and processed as such? If a distinct scene surface only occupies one pixel in a range image, it makes no difference how smooth or how textured or how noisy that surface is, it will only cause an ideal range sensor to register a value for the depth at that location. Unfortunately, noise spikes may occur in real images making it almost impossible to distinguish between a noise spike and a very small surface that occupies one pixel. However, even when noise spikes do occur, it is extremely unlikely that four or more spikes would occur in adjacent pixels and all lie in the same plane. Therefore, the coherence concept might be formulated in terms of regions as small as four pixels. But when convolution window operators are used for smoothing and derivative estimation, as in the surface characterization algorithm, it is difficult for the surface properties of small surfaces to remain intact in the HK-sign map. Therefore, in order to be certain that the proposed data-driven range image segmentation algorithm will produce meaningful results, a statement about the size of surfaces in the original image is needed. Ideally, an intelligent preliminary process might look at the range image and say, "The smallest meaningful region of interest is represented by a surface region approximately 7 x 9 pixels, and all smooth surface regions are corrupted by zero-mean additive Gaussian noise with a standard deviation of approximately 2 depth levels in the 8-bit dynamic range of the range image." Such an image would be an example of a range image exhibiting surface coherence, but unfortunately, it would be extremely difficult for a low-level process that is devoid of concepts about meaningful surfaces to issue such a statement. However, it is possible for a low-level process to yield useful information about image quality

and surface coherence. For the proposed segmentation approach, a preliminary measure of the overall "smooth surface fittability" of a digital surface is needed. Preferably, this measure could be computed quickly without actually fitting global surfaces and measuring fit errors.

The following approach has been implemented, tested, and found to be a useful indication of image quality and surface coherence. The method described below is very simple and does not account for everything that is implied by the notion of surface coherence. Specifically, the spatial extent of surfaces is not considered. The practical strength of this approach is that the existing surface characterization software can compute the proposed quality measure, which simplifies the overall system. Better methods of measuring smooth surface fittability, or surface coherence, are certainly possible. For example, autocorrelation functions or first-order tensor product surfaces may be useful for this purpose.

The proposed image quality measure algorithm is stated as follows:

(1) Perform a least-squares planar fit to every 3x3 neighborhood in an image using convolution window operators to get the slope of each plane.

(2) If the slope of the planar surface at a pixel is greater than a threshold slope (8), discard the pixel since it appears to be on or near to a depth discontinuity.

(3) Similarly, if the slope of the planar surface is exactly zero, the neighborhood of the pixel is likely to be flat-synthetic, saturated, or completely dark and should be discarded because it is not representative of actual smooth surfaces in an image.

(4) If the pixel has not been discarded, compute the planar fit error and add to average planar fit error. Increment the number of pixels.

(5) Finally, the average error after the entire image has been processed is the quality measure ρ.

This processing assumes that there are relatively few orientation discontinuity pixels compared to the number of visible surface interior pixels in the image. The equation for the quality measure ρ may be expressed as

$$\rho \approx \frac{1}{N_p} \sum_{\substack{surface \\ p \in interior \\ pixels}} \epsilon_{1,3}(p) \tag{4.2}$$

where $\epsilon_{1,3}(p)$ is the root-mean-square-error (RMSE) of the least-squares (1st order) planar surface in the 3x3 neighborhood of the pixel p and where N_p is the number of pixels contributing to the sum. Tests were done with 5x5 windows and higher-order surfaces, but the 3x3 planar fit measure produced the most useful results in that the

Image Quality Measures for Test Image Database					
Image Name	No.	ρ	Type	Source	Comments
Coffee Cup A	1	0.71	Range	ERIM	Excellent Quality
Coffee Cup B	2	0.84	Range	ERIM	Excellent Quality
Keyboard A	3	1.17	Range	ERIM	Good Quality
Keyboard B	4	1.35	Range	ERIM	Good Quality
Polyhedron	5	1.38	Range	ERIM	Good Quality
Ring on Polyhedron	6	1.61	Range	ERIM	Fair Quality
Curved Surface A	7	1.68	Range	ERIM	Fair Quality
Curved Surface B	8	2.15	Range	ERIM	Fair Quality
Road Scene A	9	0.72	Range	ERIM	Good Quality, but Litte Detail
Road Scene B	10	0.79	Range	ERIM	Good Quality, but Litte Detail
Auto Part	11	0.43	Range	INRIA	Image Formed from (x,y,z) List
Block A	12	1.82	Range	Synthetic	Low Noise Added
Block B	13	5.93	Range	Synthetic	High Noise Added
Two Cylinders	14	1.79	Range	Synthetic	Low Noise Added
Complex Object	15	1.83	Range	Synthetic	Low Noise Added
Light Bulb	16	3.45	Range	Synthetic	Medium Noise Added
Torus	17	2.59	Range	Synthetic	Medium Noise Added
Circular Waves	18	1.54	Range	Synthetic	No Noise Added
Space Shuttle	19	2.29	Intensity	Real	Fair Quality
Road Image	20	1.95	Intensity	Real	Fair Quality
USC Girl	21	2.63	Intensity	Real	Poor Quality (128x128)
U-Mass House	22	2.96	Intensity	Real	Good Quality, Highly Detailed

Figure 4.9. Image Quality Measures for Noisy Images

numbers were most highly correlated with subjective notions of image quality. Figure 4.9 shows the value of the quality measure for the images discussed in Chapter 7. There is a good qualitative agreement between the magnitude of the quality measure and humanly perceived visual quality of the images. More detailed images tend to have measures with higher values since the spatial extent of surfaces is not incorporated in the measure. Figure 4.10 displays the planar fit error image for the coffee cup image and the block with three holes image. The intensity at each pixel in these images is the magnitude of the planar fit error, but the images are contrast stretched so absolute magnitudes are not meaningful. All step-edge regions are blackened since they are not of interest to the quality measure operation. Figure 4.11 displays the planar fit error images for the Curved Surface A range image and the space shuttle intensity image. The quality measure ρ is the average of the non-zero pixels in these images.

The quality measure ρ allows us to tie the thresholds involved in the surface characterization and surface segmentation algorithms to the amount of noise in the image in an empirical manner. For example, let ρ_{cc} denote the image quality measure for the coffee cup image. The zero-curvature thresholds ϵ_H and ϵ_K for the surface characterization algorithm and the image fit error threshold ϵ_{max} for the surface segmentation algorithm are manually adjusted so that good segmentation results are achieved. Then, when an unknown image is presented to the system, the quality measure ρ is computed. No variable information is used since the internal slope threshold for the quality measure is fixed. If $\rho < \rho_{cc}$, then the empirically determined thresholds for the coffee cup should be lowered depending on the magnitude of the difference. If $\rho > \rho_{cc}$, then the empirically determined thresholds for the coffee cup can be raised depending on the magnitude of the difference. A simple linear fit of manually optimized error thresholds

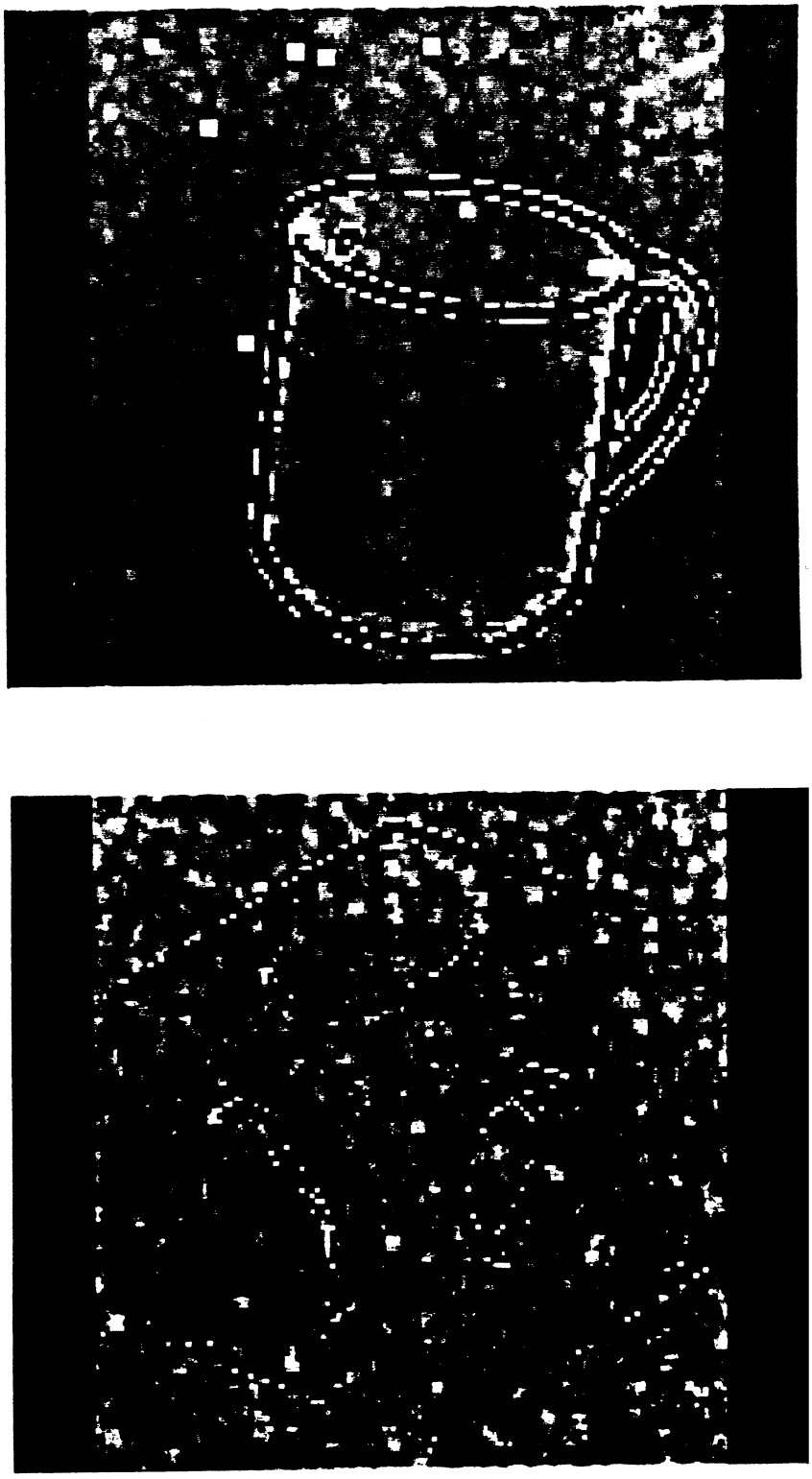


Figure 4.10. Planar Fit Error Images for Coffee Cup and Block

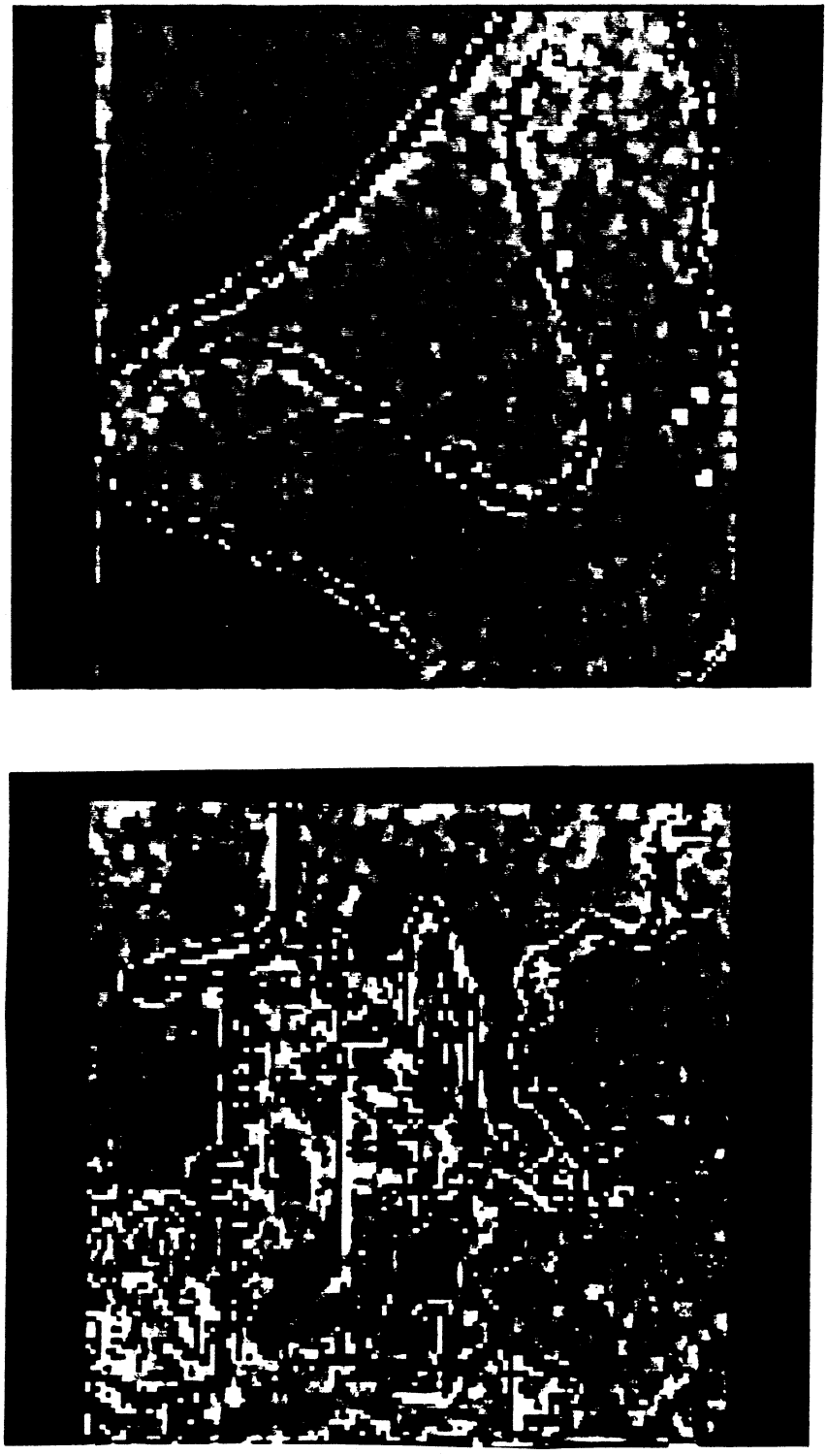


Figure 4.11. Planar Fit Error Images for Curved Surface A and Shuttle

for the region growing algorithm based on surface fitting yields a dependence of $\epsilon_{max} = 1.1 + 1.1\rho$. This technique, though not perfect, provides at least some basis for automated threshold selection for the overall segmentation algorithm. As shall be noted in Chapter 7, it is possible to achieve good results over a large set of images using the same fixed set of thresholds for each image as long as the quality measure is below some threshold (e.g. 3). It was found that it is ordinarily worse for the maximum fit error threshold to be too small than it is for it to be too large.

The quality measure ρ is proposed as a simple measure of surface coherence for digital surfaces. When the quality measure is small ($\rho \leq 1$ for 8-bit images), the 3x3 planar surfaces tend to fit the digital surface neighborhoods well and the segmentation algorithm is likely to do very well. Most synthetic images without added noise fall into this category and are not shown in the experimental results of Chapter 7 since they are usually easy to segment correctly. When the quality measure is in an intermediate range ($1 < \rho \leq 3$) as is the case for most of the images listed in Figure 4.9, the segmentation algorithm generally does well, but may not yield all meaningful surfaces as separate distinct regions. When the quality measure is in a higher range ($3 < \rho \leq 10$), the segmentation algorithm will yield some meaningful surfaces, but performance may vary widely from image to image depending on how large the noisy smooth surface regions are. One problem with ρ is that it does not yield any information about the spatial extent of the coherent surfaces in the image due to its very local nature. Despite this difficulty, it has been a useful tool for the wide variety of images discussed in Chapter 7.

4.6. Seed Region Extraction

Error threshold and approximating function selection have been discussed to set the stage for the actual algorithm description. The approximating function set $\{\hat{g}_i(x,y)\}$

and the error threshold ϵ_{max} shall appear as predetermined, but variable, elements in this description.

Given the original image and the HK-sign map, the algorithm begins by considering the HK-sign map. Most of the problems with the HK-sign map mentioned earlier are avoided by adopting the following strategy. First, the largest connected region of any fundamental HK-sign surface type in the image is isolated using an approach described below. If the isolated connected component region is contracted (eroded) repetitively using a 3x3 region contraction operator (Appendix F), the region will eventually disappear as long as the entire image is not of a single surface type. If the image is a single surface type, that case presents no problems as will be evident. For each contraction (erosion), there exists a largest four-connected sub-region of the original region. If the number of pixels in the largest connected sub-region is recorded as a function of the number of contractions, a *contraction profile* for the original region is created. Figure 4.12 shows the scaled contraction profiles for several different regions. Note that they all have similar shape, but that some go to zero faster than others. If a lower threshold is set for the number of pixels required to be in a seed region, there will be a minimum number of pixels in the contraction profile that is greater than or equal to the lower threshold. The largest connected sub-region with the minimum number of pixels greater than or equal to the lower threshold is assigned to be a *seed region* (or *kernel* region).

The lower threshold must be greater than or equal to the minimum number of points required for the simplest surface fit. If the threshold is equal to the minimum number of points, then the surface fit can respond strongly to measurement noise. Therefore, it is argued that the threshold should be somewhat greater than the minimum required number of pixels. Since three pixels are required to determine a plane and six

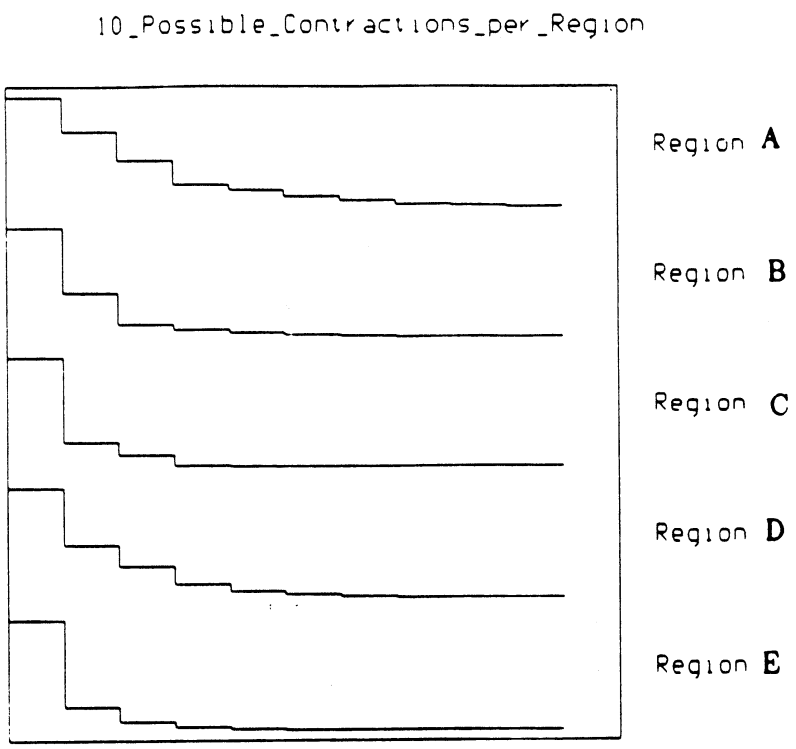


Figure 4.12. Contraction Profiles for Different Regions

pixels are required to determine a biquadratic, at least eight pixels in a region are required for the seed region. This provides a fair number of pixels for the planar fit, but also allows an increase in surface fit order if required. If it were possible to know the minimum spatial extent of meaningful surfaces in an image ahead of time, it would be advantageous to *increase this threshold as much as possible, thereby increasing the speed of the sequential algorithm.*

Since the fundamental purpose of the contraction profile computation is to find a small enough isolated region that (1) is not inadvertently connected to any adjacent regions, and (2) is far enough inside the boundaries of the actual surface primitive to have escaped the undesired side effects of smoothing and differentiation at surface boundaries, there is an upper limit on the number of necessary contractions. This limit is based on the window size of the smoothing and derivative operators used in the compu-

tation of surface curvature. For 11x11 smoothing and 9x9 derivative window operators, a limit of ten (10) contraction iterations has served perfectly thus far in experiments. (Ten contractions reduce a 20x20 binary block to nothing.) In a 9x9 convolution, a given pixel is potentially affected by the input data 4 pixels from it. If smoothing is applied first and then differentiation, a given pixel may be effected by data nine pixels away. Although nine pixels is all that is required, a limit of ten (10) contractions was chosen to err on the side of caution since it only costs one extra contraction, and extra contractions are only possible on the largest regions in an image. Extra contractions have no effect on the rest of the algorithm except that it is possible that extra region growing iterations may be required to grow back over the pixels lost in the contractions. All this should become clear as the algorithm description continues.

The brute force method of finding the largest region of any HK-surface type using a standard connected component analysis program with a single label buffer is to (1) isolate the four-connected regions of each of the eight types separately, (2) count the number of pixels in each region for each of the eight types, and (3) choose the region with the maximum number of pixels. The problem with this approach is that the standard connected component analysis algorithm is a two-pass algorithm and requires significant computations. This first step may take a long time for every time that a new seed region is to be computed. But the same single-label-buffer program can be used in a different way. On average, the following approach requires much less computation:

(1) Compute a surface type histogram, which is done with a simple one-pass algorithm,

(2) Sort the histogram, which only has eight pixel counts in the list corresponding to the eight fundamental surface types,

(3) Compute the connected components of the surface type with the largest histogram pixel count as determined by the sorted histogram and isolate the largest region, and

(4) Finally, compute the connected components of the next surface type in the sorted histogram and isolate the largest region until a surface type is reached that has a total number of pixels (in the histogram count) less than the largest connected region encountered so far for a particular surface type.

(5) At that point, quit and return the largest connected region encountered so far as the largest connected region of any surface type.

This approach takes advantage of the fact that the number of pixels in the largest connected component region of that type can never be larger than the number of pixels of that type in the image. At best, one histogram computation and one connected component analysis computation are required, saving seven expensive connected component analysis computations. At worst, the histogram computation does not help to save connected component analysis computations. On average, the histogram computation saved about five connected component analysis algorithm executions out of the possible eight, which is a 62.5% reduction over the brute force single-label-buffer method. It is possible to devise even faster methods that need more memory by using a special purpose connected component algorithm. A faster program could use three tag-bits for each pixel in the region label buffer and would only require two passes over the HK-sign map image.

Methods have been described for (1) quickly selecting the largest connected region of any surface type and for (2) contracting that region to obtain a seed region, which is subject to a smallest region size threshold. This ranking technique allows us to process the largest, most coherent, and hopefully, the most important and meaningful surfaces in

a range image first. The large-to-small surface-type region processing method has been a very useful approach for obtaining good seed regions. It is not without its weaknesses however. At the risk of getting ahead of the algorithm explanation, consider, for example, that it might prove fruitful to combine the size measure of a surface type region with the average error in the image error buffer over that region into one ranking measure. But it is not clear how region size and surface approximation error should be combined. The image error buffer records the error at each pixel for the best fit surface that has been encountered so far. This buffer is described in more detail in the later section on surface acceptance and rejection decisions. If it is possible, a smaller region of an image that has not been approximated yet may take precedence over a larger region that already has a fairly good approximation, but was not good enough to meet the error threshold requirement. More importantly, the existing segmentation algorithm is sequential at the control level and thereby has a strong order dependency despite attempts to alleviate that order dependency using the error buffer approach. However, the entire algorithm is parallelizable if the seed region selection process and the region growing process can be made parallel. It is not clear at this point in time that this can be achieved.

4.7. Iterative Variable Order Surface Fitting

Each isolated seed region is given as the input to the iterative region growing algorithm based on variable order surface fitting. The basic concepts of this algorithm are the following. First, it must be decided how well smooth surfaces should fit the data. The general-purpose noise estimation procedure discussed above provides an indication of the RMS error (a number) to be expected for smooth surface fitting. This number guides the choice of the maximum RMS error threshold ϵ_{max} for the iterative surface fitting algorithm. ϵ_{max} is also used for the overall average image error threshold used to

terminate the entire algorithm.

A plane is always fitted first to the small seed region using an equally weighted least squares approach described in Appendix C. If the seed region belongs to a surface that is not too highly curved, a plane will fit quite well to the original digital surface. This idea is basic to mathematics; if a small enough interval on the real line is considered, any smooth real-valued function of a single variable can be approximated well by a straight line on that interval. Indeed, linear approximations are used to help analyze many nonlinear phenomena locally when more direct general methods are not available. If the plane fits the seed region to within the maximum error threshold for the RMS error, then the seed is allowed to grow. If not, the seed is fitted with the next higher-order surface (the biquadratic in the current implementation) and the algorithm proceeds similarly. When the seed is allowed to grow, the functional description of the surface over the seed region is tested over the entire image to determine what pixels are compatible with the seed region (i.e., close enough to the surface fit of the seed region).

This process may be stated mathematically as follows. Let I be the rectangular image region over which the hypothetical piecewise smooth function $z = g(x,y)$ is defined. Let $\hat{R}_i^{(0)}$ be the first seed region provided by the seed region extraction algorithm that corresponds to the actual region R_i defined in Chapter 2. This seed region is hypothesized to always be a subset of the actual HK-sign surface primitive region R_i that will be determined as the seed region is grown:

$$\hat{R}_i^{(0)} \subseteq R_i \subseteq I \ . \tag{4.3}$$

The HK-sign map provides subsets of the desired segmentation regions (HK-sign surface primitive regions) through the seed region extraction process. The task at hand now is to convert $\hat{R}_i^{(0)}$ to a full region $\hat{R}_i$ that approximates the desired region R_i.

As discussed above, a specific finite set of approximating surface function types must be chosen for use in surface fitting. These surface functions must be able to provide for extrapolation into areas outside the local surface fitting region, and they must provide good approximation capabilities within the fitting region. The detailed interpolation capabilities of the surface functions are not of primary importance because of the implicit assumption of dense digital surface data. Let $\mathbf{a}_i^{(k)}$ be the parameter vector associated with the functional fit to the depth values in a given region $\hat{R}_i^{(k)}$ from the set of all connected surface regions. The superscript (k) denotes the k-th iteration in the iterative surface fitting process. Let $\mathbf{P}$ be the set of all parameter vectors for all forms of functions of interest. For the three bivariate polynomials, $\mathbf{P} = \mathbf{R}^{15}$. Let $|F|$ be the number of different types of surface functions to be used (in this case, $|F| = 3$). A particular function type (or fit order) is referred to as m_F where

$$m_F \in \left\{1,2,\cdots,|F|\right\}. \tag{4.4}$$

The actual fitting function of form m_F is denoted $z = \hat{g}(m_F,\mathbf{a},x,y)$. The surface fitting process, denoted $L_{\hat{g}}$, maps the original image $\mathbf{Q}(\mathbf{S}_I(g(x,y)))$, a connected region description $\hat{R}_i^{(k)}$, and the fit order m_F into the range space $\mathbf{P}\times\mathbf{R}^+$ where $\mathbf{R}^+$ is the set of possible errors (the set of non-negative real numbers):

$$(\mathbf{a}_i^{(k)},\epsilon_i^{(k)}) = \mathbf{L}_{\hat{g}}(m_F,\hat{R}_i^{(k)},g). \tag{4.5}$$

It has the property that the error metric (function norm)

$$\epsilon_i^{(k)} = |\hat{g}(m_F,\mathbf{a}_i^{(k)},x,y) - \mathbf{Q}(g(x,y))|_{\hat{R}_i^{(k)}} \tag{4.6}$$

is the minimum value attainable for all functions of the form specified by m_F. Equally-weighted least-squares surface fitting minimizes the error metric

$$(\epsilon_i^{(k)})^2 = \frac{1}{|\hat{R}_i^{(k)}|} \sum_{(x,y) \in \hat{R}_i^{(k)}} (\hat{g}(m_F, \mathbf{a}_i^{(k)}, x, y) - \mathbf{Q}(g(x,y)))^2 \qquad (4.7)$$

where $|\hat{R}_i^{(k)}|$ is the number of pixels in the region $\hat{R}_i^{(k)}$, or the area of the region. This may also be considered as the *norm* of the region as in Appendix E. Least-squares polynomial surface fitting is fast and direct, and the shape potential of polynomial surfaces is adequate for HK-sign surface primitives. Bivariate polynomial surfaces of orders 1, 2, 4 have provided excellent approximations to the basic surface types in small and even very large neighborhoods (regions).

In summary, given (1) a seed region $\hat{R}_i^{(k)}$ (where $k = 0$ for the original seed region), (2) the original image $\mathbf{Q}(g(x,y))$, and (3) the simplest interesting function type m_F (such that $\epsilon_i^{(k)} < \epsilon_{max}$), the parameter vector $\mathbf{a}_i^{(k)}$ and the RMS error $\epsilon_i^{(k)}$ are computed for the k-th iteration on the i-th HK-surface primitive region from the image. If the error is less than the predetermined maximum allowable error threshold ϵ_{max}, then the seed region is allowed to grow. If all three fit orders were tried and the error was never less than the threshold, the seed region is rejected by marking off the pixels in the writable copy of the HK-sign map, and then continuing by looking for the next largest connected region of any surface type. Surface rejection decisions are discussed in more detail later. At this point, it is also noted that even though a successful surface fit and region growth occur, the fit order for the next iteration may also be increased depending on the results of a "2-D regions test," a test that generalizes concepts of the 1-D runs test from non-parametric statistics. The regions test is discussed after the details of region growing are presented.

4.8. Parallel Region Growing

After a surface is fitted to a region, the surface description is used to grow the region into a larger region where all pixels in the larger region are connected to the original region and are compatible with the approximating surface function for the original region. On the k-th iteration for the seed region corresponding to the actual HK-surface primitive region R_i, the parallel region growing algorithm accepts the original digital surface $\mathbf{Q}(\mathbf{S}_I(g(x,y)))$, the approximating function $\hat{g}(m_F, \mathbf{a}_i^{(k)}, x, y)$ from the surface fitting algorithm, and the surface fit error $\epsilon_i^{(k)}$ from the fit to the seed region. *It does not use the seed region directly*, only the function derived from the seed region at this phase in the growth. For each pixel $p \in I$ (the entire image), the two values

$$z_{\hat{g}}(p) = \hat{g}(m_F, \mathbf{a}_i^{(k)}, x(p), y(p)) \quad \text{and} \quad z_g(p) = \mathbf{Q}(g(x(p), y(p))) \tag{4.8}$$

are compared to see if the pixel p is compatible with the approximating surface function. It is assumed that practically all pixels in the original region are compatible with the approximating surface function because the pixels in that region determined the surface fit. This subject is discussed in more detail when convergence issues are treated. If the magnitude of the difference between the function value and the digital surface value is less than the allowed tolerance value $w(\epsilon_i^{(k)})$, then the pixel p is added to the set of pixels $C(\hat{R}_i^{(k)}, \hat{g}, \epsilon_i^{(k)})$ that are compatible with the region $\hat{R}_i^{(k)}$; otherwise, it is incompatible and discarded. The result of this process is explicitly stated as

$$C(\hat{R}_i^{(k)}, \hat{g}, \epsilon_i^{(k)}) = \left\{ p \in I : \mid z_{\hat{g}}(p) - z_g(p) \mid \leq w(\epsilon_i^{(k)}) \right\}. \tag{4.9}$$

In order to stress all the implied dependencies, specifically note that the set of compatible pixels $C(\cdot)$ is dependent on (1) the region $\hat{R}_i^{(k)}$ and therefore also on the seed region extraction algorithm that created the first region $\hat{R}_i^{(0)}$, (2) the function $\hat{g}$ and therefore

on the function type m_F and all the properties of that function type, (3) the parameter vector $\mathbf{a}_i^{(k)}$ and error $\epsilon_i^{(k)}$, which are in turn dependent on the type of surface fitting algorithm and again on the original digital surface g and the seed region, and (4) the error tolerance increase function $w(\epsilon) > \epsilon$.

The compatible set of pixels also depends indirectly on all the surface regions previously discovered in the image. This will be clearer after surface acceptance and rejection decisions are discussed, but this indirect dependence plays a key role in the region growing algorithm so this material is introduced now.

For simple surfaces with precisely defined surface boundaries, the influence of other already determined surface regions is negligible or non-existent, but for noisy digital surfaces, the influence may be more prominent. When a surface region iteration terminates and the surface is accepted by successfully passing the test rules discussed subsequently, the magnitude of the error at each pixel of the grown surface region is stored in the image error buffer (E-buffer) to explicitly note the spatial distribution of the approximation errors. The surface region label for each pixel of the accepted surface region is stored in a region label buffer (RL-buffer) to explicitly note the pixels that the approximating surface fits to within the specified threshold. During the region growing process that forms the compatible pixel list, each pixel is not only checked to see if the error is less than $w(\epsilon_i^{(k)})$, but it is also checked to insure that either the pixel error is less than the error in the error buffer (the best fit error so far) or that the pixel has not yet been officially claimed by another surface. If neither of these conditions are satisfied, then the pixel is not considered to be compatible with the growing surface and is not marked as such despite the fact that the allowable error tolerance requirement is met. The error buffer approach provides a "soft inhibition" capability for pixels already associated with

a given surface primitive as opposed to strictly forbidding the reassignment of pixels to other surfaces once they have been assigned to one surface. This is done in an attempt to alleviate problems with the order dependent properties of this sequential algorithm. Also, the results of this approach are extremely useful for determining smooth surface merges between surface primitives in the presence of noise.

Other growth-inhibiting geometric structures could be used during the construction of the compatible pixel list. For instance, surfaces should not grow over depth discontinuities or orientation discontinuities by the definition of a smooth surface. Thus, if reliable estimates of step edges (depth discontinuities) and roof edges (orientation discontinuities) can be found, the resulting edge pixels could be used as a mask of pixels that are always incompatible for region growth. Since the main thrust of this thesis is surface-based segmentation, edge detection and combinations of edge-based and region-based approaches are not discussed here, but the excellent opportunity for combining edge-based results with the surface-based region-growing algorithm should be noted.

The error tolerance increase function $w(\cdot)$ is a function that increases the value of $\epsilon_i^{(k)}$ so that, if a pixel really lies on the smooth surface $\hat{g}(m_F, \mathbf{a}_i^{(k)}, x, y)$ and if $\epsilon_i^{(k)}$ is an estimate of the standard deviation of the measurement noise, then one is reasonably sure that all (or almost all) the pixels that belong to the smooth surface are correctly grouped with that surface. For example, a reasonable form of the error increase function $w(\cdot)$ is

$$w(\epsilon) = w_0 \epsilon \qquad \textit{where} \quad w_0 = 3 \tag{4.10}$$

because approximately 99% of all samples of the smooth surface corrupted by normally-distributed measurement noise will lie within this error tolerance. In this simple error increase function, the factor w_0 then controls the "aggressiveness" of each region

growing process and therefore, partially controls the speed and accuracy of the iterative surface fitting process. If w_0 is too large, a surface primitive can grow right over an orientation discontinuity if it is not sharp enough. If w_0 is too small, each iteration may only include a few more pixels making the iteration process quite slow and possibly cause actual compatible regions of the digital surface to be missed and labeled separately. Hence, there is a trade-off decision to be made between the two undesirable tendencies. A factor of 2.8 has been used to obtain excellent performance on good-quality images. Lower factors that are larger than 2.3 achieve similar results, but take many more region growing iterations to find the same surface region and are therefore much slower. For extremely noisy-images ($\rho > 4$), this factor must be dropped down to the neighborhood of 2.1 so that the growth is very careful (not aggressive).

Regardless of the factor w_0, the function $w(\cdot)$ must take on the following sort of hard limiting modification, or else no progress can be made when the fit is very good:

$$w(\epsilon) = \max(\ 1.1\ , w_0\epsilon\ )\ . \tag{4.11}$$

This is *the standard error increase function* used for almost all experimental results shown in Chapter 7. Since the levels of the digital surface are quantized to one depth level by the quantization operator **Q**, attempts to find compatible pixels with depth levels that are less than one depth level from the approximating surface are not too successful on average. The threshold 1.1 was chosen arbitrarily to satisfy the need for a threshold slightly greater than one; it has worked quite well in practice. The digital algorithm often requires attention to details not present in continuous domain mathematical analysis.

Several tests were performed with a more aggressive variant of the error increase function listed above. This function is referred to as $w_A(\cdot)$, and it involves the max-

norm (sup-norm) of the functional fit. Let the error ϵ_∞ represent the max-norm error between an approximant $\hat{g}(x,y)$ and the data $g(x,y)$:

$$\epsilon_\infty = \max_{(x,y)\in R} | \hat{g}(x,y) - g(x,y) | \tag{4.12}$$

Since the ϵ in the other definitions above is a Euclidean error norm, it is written as ϵ_2 in the expression for the aggressive error increase function:

$$w_A(\epsilon_2, \epsilon_\infty) = \max(1.1,\ w_0\epsilon_2,\ \epsilon_\infty) \tag{4.13}$$

This error increase function provided faster algorithm performance in general because region growth was more aggressive. Moreover, this error increase function is able to guarantee ideal fixed-point convergence of the iterative algorithm because regions can only get bigger (see Section 4.11). Slightly better algorithm results were obtained for specific cases, but much worse results were obtained in other cases. Linear, convex combinations of cautious and aggressive error increase functions were also tried so that seeds experience fast growth when the error is small, but established regions slowed their growth as their surface fit errors increased. The use of other types of functions was also explored. Experimental tests showed that the final segmentation results were often sensitive to the choice of the error increase function. The standard function mentioned above is the best choice for most images in the test image database presented in Chapter 7.

When the parallel region growing computation is complete, the pixel list $C(\hat{R}_i^{(k)}, \hat{g}, \epsilon_i^{(k)}) \subseteq I$ is complete. The largest connected region of this list that overlaps the seed region $\hat{R}_i^{(k)}$ must then be extracted to create the next region $\hat{R}_i^{(k+1)}$. This iterative process of region definition via largest, overlapping, connected region extraction is expressed using the function $\Lambda(\cdot)$:

$$\hat{R}_i^{(k+1)} = \Lambda\Big(C(\hat{R}_i^{(k)}, \hat{g}, \epsilon_i^{(k)}), \hat{R}_i^{(k)} \Big) = \Phi(\hat{R}_i^{(k)}) \tag{4.14}$$

where $\Phi(\cdot)$ represents all operations required to compute the region $\hat{R}_i^{(k+1)}$ from the region $\hat{R}_i^{(k)}$. The output region $\hat{R}_i^{(k+1)}$ must have the property that it is the largest connected region in the list of compatible pixels satisfying

$$\hat{R}_i^{(k)} \cap \hat{R}_i^{(k+1)} \neq \phi = \textit{Null Set} \ . \tag{4.15}$$

This constraint is required because it is possible to get larger connected regions in the compatible pixel list than the connected region corresponding to the seed region. This next region is then considered as a seed region and processed by the surface fitting algorithm

$$(\mathbf{a}_i^{(k+1)}, \epsilon_i^{(k+1)}) = L_{\hat{g}}(m_F, \hat{R}_i^{(k+1)}, g) \tag{4.16}$$

to obtain a new parameter vector and a new surface fit error. This region is allowed to grow again if $\epsilon_i^{(k+1)} < \epsilon_{max}$. The compatible pixel list is then recomputed

$$C(\hat{R}_i^{(k+1)}, \hat{g}(m_F, \mathbf{a}_i^{(k+1)}, x, y), \epsilon_i^{(k+1)}) \ , \tag{4.17}$$

the largest connected overlapping region of $C(\cdot)$ is extracted and so on until the termination criteria are met. The termination criteria are discussed shortly.

There is a slight variant of the **A** function that also insures ideal fixed-point convergence of the region growing iteration (see Section 4.11) when the standard error tolerance increase function is used. After the largest region overlapping a seed region is found, the seed region may not be completely contained in the new region if the standard error tolerance increase function is used. To guarantee convergence, it is possible to union (OR) the seed region with the new region to insure that this new unioned region contains the seed region. Thus, the growth of the region is monotonic and bounded by maintaining the inclusion of previous iteration's region. Although good results were obtained on

several images using this monotonicity guarantee, experiments showed that this was not the best policy for all images in the test image database. The algorithm must be able to let go of pixels that do not fit well in any given iteration. This topic is discussed in the section on termination criteria and convergence.

When the surface fitting process has yielded a fit error that is not acceptable (i.e., not below the maximum error threshold $\epsilon_i^{(k)} \geq \epsilon_{max}$), the algorithm immediately increments the surface type m_F to the next most general, flexible surface in the ordered, finite set of approximating functions and reattempts a surface fit of the higher order surface type if there are enough pixels in the connected region being fitted. If m_F reaches the maximum value $| F |$, the number of approximating functions in the set, and the surface still does not fit the data to within the maximum error threshold ϵ_{max}, or if there are not enough pixels in the region to attempt a higher order fit, then the iterative surface fitting algorithm attempts to accept the surface even though the ideal termination criteria have not been met. For example, if $\epsilon_{max} < \epsilon_i^{(k)} < 1.5\epsilon_{max}$, then the surface region is accepted because it is close enough to be acceptable. It has been found experimentally that this extra 50% margin of acceptance allows the surface fitting iteration to yield useful results that might not otherwise be obtained. Although the software implementation allows the user to set the acceptability factor, the default setting of 1.5 has never needed to be changed.

The algorithm is structured so that (1) seed region determination, (2) iterative surface fitting and region growing, and (3) surface acceptance functions are performed in a modular fashion. The termination criteria are incorporated in the iterative surface fitting module. Hence, the surface acceptance module attempts to accept surface regions that have met and not met the ideal convergence termination criteria. Therefore, the

termination criteria are treated before the surface acceptance operations.

It must be noted that this parallel region growing approach is entirely equivalent to a sequential spiraling region growing approach *until the last iteration.* At the last iteration, the processing of the compatible pixel list becomes an important feature of the segmentation algorithm as described in the section on surface acceptance and rejection decisions. By performing the parallel region growth at each iteration, it is possible to watch the compatibility of other image regions during each iteration. On a sequential machine, the parallel approach requires more computation, but the simplicity of the algorithm also has advantages. On a parallel machine, the time required to compute the compatible pixel list depends on the number of pixels that can be processed by each independent processor. A separate connected component processing step is required by the parallel approach, but it is assumed that this can be accomplished very quickly on special purpose hardware.

4.9. Regions Test

A fit error test decides if region growth for a particular fit order is allowed or not. However, it is possible for a lower order function to fit a higher order function well (within the error threshold) over a region even though the lower order fit is not appropriate. Figure 4.13 displays a curve that is fitted well by a line, but the signs of the residuals of the fit form long runs. If it were possible to detect that a higher order function is present in the data before the error tolerance exceeded the error threshold, the region growth process could proceed more quickly and accurately than otherwise. Indeed, it is possible to detect the presence of a higher order function in the data (without having to allow the fit error to increase up to the given error threshold) by analyzing the distribution of the sign of the fit errors (residuals) at each individual pixel

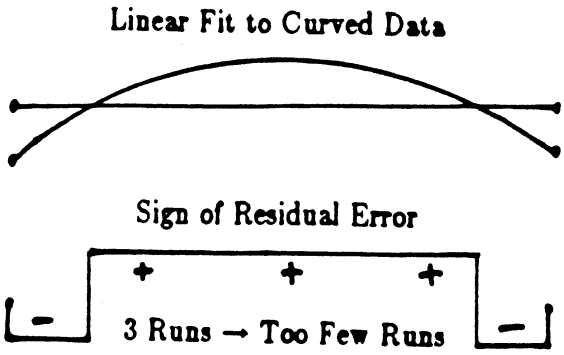


Figure 4.13. Evidence of Curve in Data Despite Good Linear Fit.

of the fit. First, the one-dimensional nonparametric runs test is described and then certain aspects of this test are generalized for two-dimensional purposes.

The nonparametric runs test is a general purpose tool to test for randomness in a sequence of data points. The runs test is used in regression analysis to test the pattern of sign changes in the residuals of a fit. Suppose that a set of data points that lie directly on a line is given. Suppose also that low-variance, independent, identically-distributed noise is added to each data point in the sample to simulate measurement noise in a sensor, a line is fitted to the noisy data points, and then each noisy data point is marked with a (+) if it lies above the fitted line and a (-) if it lies below the fitted line. Now the sequence of (+)'s and (-)'s corresponding to the noisy data points is examined. If the string (-)(+)(+)(+)(-) occurs at some point in the sequence, a *run* of (+)'s of *length* 3 is said to have occurred. Let n be the total number of data points in the sequence. Let n_1 be the total number of (+)'s and let n_2 be the total number of (-)'s. Thus, $n = n_1 + n_2$. Let r be the total number of runs of residual signs in the total sequence; this quantity serves as the test statistic. Given a particular level of statistical significance (e.g. 0.025), two functions may be derived $r_{min}(n_1,n_2)$ and $r_{max}(n_1,n_2)$ from probabilistic arguments. If $r_{min} < r < r_{max}$, then it is likely that the (measure-

ment) noise is random; otherwise, one must reject the random noise hypothesis at the given level of significance. For example, for 40 fitted data points with 20 (+)'s and 20 (-)'s, the number of runs is expected to be between 14 and 28 at the 0.025 level of significance [Daniel 1978]. Note that there is an inverse relationship between the number of runs and the lengths of runs on average: fewer runs imply longer runs, and more runs imply shorter runs.

When fitting surfaces over an arbitrary connected subset of a regularly spaced rectangular pixel grid, the residual signs of a surface fit form 2-D regions, not 1-D runs. One way to generalize the 1-D runs test to create a 2-D regions test is to perform horizontal and vertical runs tests on every scan line and every column of the arbitrary shape fitted region. Such a generalization decomposes a 2-D problem into two sets of 1-D problems. Another method of generalization has been chosen that maintains a 2-D "flavor." The first stage of the regions test algorithm is stated as follows: After the approximant $\hat{g}(m_F, a_i^{(k)}, x, y)$ has been computed for the region $\hat{R}_i^{(k)}$, form two error-sign binary images B^+ and B^- based on residual signs. The pixel $p \in I$ is set to white in the B^+ image if $p \in \hat{R}_i^{(k)} \subseteq I$ and $Q(g(x(p), y(p)) > \hat{g}(m_F, a_i^{(k)}, x(p), y(p))$; otherwise, the pixel is set to black. Similarly, the pixel $p \in I$ is set to white in the B^- image if $p \in \hat{R}_i^{(k)} \subseteq I$ and $Q(g(x(p), y(p)) < \hat{g}(m_F, a_i^{(k)}, x(p), y(p))$; otherwise, the pixel is set to black. A four-connected region in the B^+ image is directly analogous to a run of (+)'s in the 1-D example just as a four-connected region in the B^- image is directly analogous to a run of (-)'s. Figure 4.14 shows examples of the binary images for a particular fit. Let $n = |\hat{R}_i^{(k)}|$, the number of pixels in the fit region. Let $n_1 = |B^+|$, the number of white pixels in the binary image B^+, and let $n_2 = |B^-|$, the number of white pix-

Figure 4.14. Binary Residual-Sign Images for Surface Fit

els in the binary image B^-. The framework so far is a precise generalization of the runs test. The question is, "What should be used as the test statistic?" For example, the number of four-connected regions in the two binary images could be used. Attempts to use this test statistic were unsuccessful because relatively large connected regions may result even in the presence of random binary patterns of pixels.

What is really needed from a regions test? The 1-D runs test checks for too many regions and too few regions. The most runs are obtained when (+) and (-) residuals always alternate; every run is of length one and $n_1 \approx n_2$. The most (four-connected) regions are obtained when both error-sign images are perfect checkerboard patterns; every region is of size (or area) one and $n_1 \approx n_2$. Although such error patterns are not random, regression analysts are not usually interested in the too-many-runs (regions) case when the question of interest is the order of the fit. (Only low-order polynomial fits are being considered here.) Consider a linear fit to a section of a parabola with no noise added as in Figure 4.13. There will be exactly three runs independent of the number of

data points and independent of the fit error. The (+)'s and (-)'s *cluster* into *long intervals*. Consider a planar fit to a section of an elliptic paraboloid with no noise added. There will be exactly two regions independent of the number of data points and independent of the fit error. The white pixels in the error-sign binary images will *cluster* into *large regions* (or blobs). Based on this line of reasoning, it is proposed that a 2-D regions test for regression analysis should try to detect large homogeneous regions of white pixels in the error-sign images. This test must decide if a higher-order function in the approximating function set is needed.

The second stage of the regions test algorithm is as follows. Once the error-sign images B^+ and B^- are formed, contract (or erode) each error-sign image once. Then compute the size of the largest region in each eroded error-sign image. Divide these two sizes by the total number of pixels in the fitted region to obtain the two percentages. Choose the larger of the two percentages as the test statistic. If the test statistic exceeds a given threshold (e.g. 2%), then a higher-order surface approximation function is needed; otherwise, the sign of the residuals (fit errors) are said to be random. This test can be written in mathematical notation as follows. Let $E(\cdot)$ be the 3x3 binary image erosion (contraction) operator that maps binary images to binary images, and let $\Lambda(\cdot)$ (with only one argument) be the largest connected region operator that maps binary images to four-connected regions. The test statistic for the regions test is defined as

$$r(B^+, B^-) = \frac{1}{n} \max(\; |\Lambda(E(B^+))|, \; |\Lambda(E(B^-))| \;) . \qquad (4.18)$$

In the iterative region growing algorithm based on variable order surface fitting, the regions test plays a fundamental role in increasing the fit order m_F when required. The regions test is phrased as a rule:

REGIONS TEST RULE: IF $r(B^+, B^-) > 0.02$ AND $m_F < |F|$, THEN increment the fit order m_F by one for the next surface fit iteration; ELSE do nothing.

This rule says that when the fitted surface lies above (or below) a sufficiently large number of four-connected pixels, a higher-order surface is needed. If the highest order surface is already being used, do not bother to perform the test because the results will have to be ignored for order incrementing purposes. The regions test is not absolutely necessary to obtain good results for all images in the test image database if one can afford to spend time searching for just the right fit-error threshold ϵ_{max}. However, when the regions test is used, many surface regions converge more quickly, most curved surface segmentations improve, and a large range of fit-error thresholds may be used providing the same segmentation results. Despite the additional computations required per region-growing iteration for planar and biquadratic fits, the benefits far outweigh the costs.

The regions test is a heuristic test based on the statistical runs test. The 2% threshold mentioned above was determined empirically after experimental testing was performed, not by theoretical means. Thus, a specific level of statistical significance is not claimed for this threshold as can be done in the 1-D runs test. Nonetheless, the regions test performs the necessary task extremely well with an algorithm based solely on 2-D concepts. Moreover, only the basic operations of binary image contraction and connected component analysis are needed to do the test. It should also be noted that, although the regions test is not applied to fourth order surfaces during the fitting iterations, this test has proved useful in evaluating the final fourth-order surface fit to a large set of data. If sufficiently large residual-sign regions are detected for a given fourth-

order surface region, then multiple surface types may be present even though the surface fit error is small. If the regions test fails on the final surface, it may be necessary to analyze the region more closely if detailed surface structure is required for particular applications.

4.10. Hemisphere Surface Example

In an attempt to make the above ideas on region growing and variable order surface fitting clearer, consider the actions of the algorithm on a simple, hemispherical surface as a specific example of all the topics discussed so far. The surface characterization algorithm labels almost all the pixels on the spherical surface as type PEAK assuming the zero curvature thresholds are not too large (in which case the FLAT label may occur). The spherical surface is necessarily bounded by pixels of other fundamental surface types, by the edges of the image, or is bounded by both. These spherical surface pixels form a connected region of PEAK pixels in the HK-sign map that are identified at some point in the large-to-small seed region extraction process. When this peak region is chosen by the seed region extraction algorithm, it will first perform the contraction sequence on it to counteract smoothing effects and possible unwanted connections to other adjacent peak regions. A small seed region is isolated somewhere on the spherical surface. It will be located at the center of the peak region if no region holes have been caused by noise; otherwise, it will be located at another position within the peak region.

As mentioned earlier, the currently implemented algorithm uses only three types of surfaces: planar, biquadratic, and biquartic polynomial surfaces. First, a plane is fitted to the small seed region. If the seed region is small enough, then (1) the fit is good, (2) the region is grown, and (3) the regions test and the fit error threshold test do not trigger an increase in the fit order right away. Depending on the error increase function

and the size of the hemispherical surface, a few planar-fit (order 1) region-growing iterations are computed. If the noise level in the data is not large compared to the fit error threshold, the regions test eventually causes an increase in the fit order after sufficient growth. For example, this might take place after the third planar iteration. This biquadratic surface type will fit well at first, yielding the best-fit elliptic paraboloid. But as the second-order surface grows, the fourth order (biquartic) surface will eventually be invoked by another regions test. If the entire hemispherical surface is visible and large enough, even the fourth-order surface-fit error may eventually exceed the maximum fit error threshold although not by much (it will remain within the 50% acceptability range mentioned above). This surface will be accepted as a surface primitive. Almost all pixels on the hemispherical surface will be identified as part of the surface primitive except for those close to the boundary. Even most of those close to the boundary will be assigned to the biquartic surface during the surface acceptance process described later. In that process, the accepted surface primitive region is dilated and neighboring pixels that are closer to the biquartic surface than any other surface are assigned to that surface. This example shows how quadric surface pixels are effectively grouped together even without a quadric approximant.

4.11. Termination Criteria and Convergence

An iterative algorithm requires termination criteria so that the algorithm halts when the desired useful information has been obtained from the iteration. If the termination criteria for an iterative algorithm are improperly formulated, the iteration is capable of continuing forever (until interrupted by other means). Such a computational process cannot *guarantee* convergence even though the algorithm might converge for most inputs. Any iterative process can be made to terminate after a maximum finite

number of steps, but from a mathematical point of view, it is clearly preferable to use algorithms that provably converge to a unique limit. Unfortunately, provably convergent algorithms do not always yield the desired useful information one may want. From a computer vision point of view, the quality of a vision algorithm's output is its most important property. Unfortunately, vision algorithm output quality is usually evaluated subjectively by the human mind, making it difficult to arrive at consistent mathematical models for optimizing algorithm performance.

As pointed out earlier, convergence is guaranteed if the region growing algorithm insures that

$$\hat{R}_i^{(k)} \subseteq \hat{R}_i^{(k+1)} \subseteq I \ . \tag{4.19}$$

This is called the *superset requirement* for growing regions. The sequence of regions is monotonically increasing and bounded above by the entire image region I. Thus, a given seed region must stop growing eventually. It will yield either a correct meaningful region or an incorrect meaningless region, which could be the entire image. Experiments with provably convergent, monotonic-growth versions of the region growing segmentation algorithm were performed and it was found that, although these versions are more attractive mathematically, they do not always produce the good results that are obtainable with other versions of the algorithm that do not attempt to insure convergence via purely monotonic growth of the seed regions. The fundamental problem with requiring that each subsequent region be a superset of the given iteration's region is that bad pixels that are picked up in one iteration can never be dropped in subsequent iterations. When the standard error increase function $w(\epsilon) = \max(1.1, w_0\epsilon)$ is used and *the superset requirement is not used*, bad pixels can enter a region on one iteration and leave on a later iteration. This is a desirable property, which is perhaps more desirable than easily

proven convergence. Hence, the effect of the bad pixels is not cumulative in this case. In summary, experimental tests have shown that insuring monotonic growth via the superset requirement can be counterproductive to the process of obtaining a good segmentation. Although the current algorithm has always terminated after a finite number of iterations and produces good segmentations when the other version could not, it has been difficult to prove convergence without the superset requirement.

Iterative algorithms have naturally received a great deal of attention in the optimization and numerical analysis literature. Some analysis requires the assumption of continuous or differentiable functions. Two representative theoretical results are quoted from Dahlquist and Bjork [1974] and Luenberger [1984]. This is intended to provide an interesting view of difficulties involved in attempting to analyze the convergence properties of the region growing segmentation algorithm using existing mathematical theory.

First, the contraction mapping approach mentioned in [Dahlquist and Bjork 1974] is briefly discussed to point out the concept of ideal fixed-point convergence, which is normally sought. Suppose an *initial vector* $\vec{x}_1$ from a fixed dimension real vector space $\mathbf{R}^n$ and the iterative process Φ that maps $\mathbf{R}^n \rightarrow \mathbf{R}^n$ are both given:

$$\vec{x}_{k+1} = \Phi(\vec{x}_k) \,. \tag{4.20}$$

Suppose that there exists a fixed point $\vec{x}_0$ such that

$$\vec{x}_0 = \Phi(\vec{x}_0) \,, \tag{4.21}$$

and that *all first partial derivatives* of Φ exist in an n-ball $B_r(\vec{x}_0)$ of finite radius r around the fixed point $\vec{x}_0$. The (i,j)-th element of Jacobian matrix $\mathbf{J}_\Phi(\vec{x})$ is given by the partial derivative $\dfrac{\partial \Phi_i}{\partial x_j}$ where x_j is the j-th component the $\vec{x}$ vector and Φ_i is the i-th component function of the Φ vector function. If $\Phi(\cdot)$ is a contraction mapping with

respect to some matrix norm on the set $B_r(\vec{x}_0)$ (i.e., if the matrix norm is less than one), then the iteration will converge to the unique fixed point $\vec{x}_0$ for any choice of the initial point $\vec{x}_1 \in B_r(\vec{x}_0)$ via the fixed point theorem for complete metric spaces (metric spaces where all Cauchy sequences converge):

$$\vec{x}_0 = \lim_{k \to \infty} \vec{x}_{k+1} = \lim_{k \to \infty} \Phi(\vec{x}_k) = \Phi(\lim_{k \to \infty} \vec{x}_k) = \Phi(\vec{x}_0) \,. \tag{4.22}$$

A necessary condition that the iteration converge is that the spectral radius (maximum eigenvalue) of the Jacobian matrix evaluated at the fixed point is less than one.

Luenberger [1984] proves the following, more general Global Convergence Theorem. Let X be an arbitrary metric space and let **A** be an algorithm that maps points $\vec{x}$ in X to subsets of X. In this case, the iterative operation is written as

$$\vec{x}_{k+1} \in \mathbf{A}(\vec{x}_k) \tag{4.23}$$

to allow the same starting point $\vec{x}_1$ to evolve into different states depending upon other initial conditions. Let $\Gamma \subseteq X$ be a given set of valid solutions to the iterative process. It is shown in Luenberger [1984] that the limit of *any convergent subsequence of* $\{\vec{x}_k\}$ *is a solution if*

(1) All points in $\{\vec{x}_k\}$ are contained in a compact set $S \subseteq X$,

(2) There is a *continuous* function $Z : X \to \mathbf{R}$ (known as a descent function) such that

(a) if $\vec{x} \notin \Gamma$, then $Z(y) < Z(x)$ for all $\vec{y} \in \mathbf{A}(\vec{x})$,

(a) if $\vec{x} \in \Gamma$, then $Z(y) \leq Z(x)$ for all $\vec{y} \in \mathbf{A}(\vec{x})$,

(3) the algorithmic mapping **A** is closed at all points outside the solution set Γ.

The required closed property of the point-to-set mapping A is the generalization of the continuity property of point-to-point mappings [Luenberger 1984].

Can these existing theoretical results be applied to the convergence analysis of the iterative region-growing segmentation algorithm? Instead of vectors of real numbers most often used in optimization and numerical analysis, this algorithm works with four-connected regions in images. If I is the set of pixels in an image, then 2^I is the power set of the set of all pixels, which is the set of all subsets of pixels of the image. Many of the sets in the power set are not four-connected sets of pixels. Therefore, let C^I represent the set of all four-connected subsets of pixels in an image. Hence,

$$C^I \subseteq 2^I \tag{4.24}$$

is the space of the region quantities for the iteration process. Every region R in the set C^I (or 2^I) may be represented by a binary vector (a bit string) of length $|I|$, the number of pixels in the image I. The explicit constraint on all binary vectors $R \in C^I$ is that the number of four-connected component regions is exactly one. The region-growing iterative algorithm may then be viewed as a mapping Φ of connected regions to connected regions:

$$\Phi : C^I \rightarrow C^I \qquad \hat{R}_i^{(k+1)} = \Phi(\hat{R}_i^{(k)}) \tag{4.25}$$

This connected region space is a finite space although the size of the space $|C^I|$ is quite large.

The convergence question can now be phrased as follows: **Given** the original digital surface, denoted simply g, which is the noisy, sampled, quantized version of a piecewise-smooth function as stated in Chapter 2

$$g(x,y) = \sum_{i=1}^{N_R} g_i(x,y)\chi_{R_i}(x,y) + n(x,y) \tag{4.26}$$

and **given** a seed region $\hat{R}_i^{(0)}$, which is a subset of the actual region R_i that corresponds to a relatively smooth surface in the real-world scene represented by the

digital surface, show that

$$\lim_{k \to \infty} \hat{R}_i^{(k)} = \hat{R}_i \approx R_i \tag{4.27}$$

where $\hat{R}_i^{(k+1)} = \Phi(\hat{R}_i^{(k)})$. $\hat{R}_i \approx R_i$ implies that the distance between the two region descriptions is small:

$$d(\hat{R}_i, R_i) = |\hat{R}_i \Delta R_i| < e(\rho) . \tag{4.28}$$

Note that the pair (C^I, d) form an actual metric space where Δ is the symmetric difference operation for sets, which is defined by

$$A \Delta B = (A - B) \bigcup (B - A) . \tag{4.29}$$

This statement is proved in Appendix E. $e(\rho)$ is the maximum amount of region estimation error given some quality measure ρ of the digital surface. Note that when regions are represented as bit strings (binary vectors), region distance (as discussed in Appendix E) is identical to the *Hamming distance* between the bit strings.

The implementation of the $\Phi(\cdot)$ operation involves several intermediate operations and quantities: (a) least-squares fitting of a bivariate polynomial of order 1, 2, or 4 to the original digital surface g restricted to the seed region, which creates a parameter vector $\mathbf{a}$ and a fit error ϵ; (b) an error threshold ϵ_{max}, which is the maximum amount of RMS error tolerated for any given surface fit; (c) the error increase function; (d) parallel region growing operation over the entire image to create the compatible pixel list $C(R, \hat{g}, \epsilon)$; (e) a largest connected component operation $\mathbf{A}$ that returns the largest region overlapping the growth region; and (f) a heuristic regions test that causes the fit order m_F to be incremented.

The question is now posed in a framework similar to the framework required to apply existing theories. Does a fixed-point region in the region space C^I exist such that

$R = \Phi(R)$? Note that the task of computing the Jacobian matrix for Φ is not defined since the concept of a derivative is not defined. This is because each element of the binary region vector only takes on two values: zero (for pixel not in region) or one (for pixel in region). Moreover, it would be quite difficult to write down the function Φ so that it could be analyzed even if derivatives could be defined. Therefore, the Jacobian matrix approach is not very useful in this case.

The Global Convergence Theorem appears to be potentially useful. Since the region space is a finite metric space (Appendix E), let the solution set Γ_i be the set of regions that are within some distance $e(\rho)$ of the actual region as above. The descent function may be defined as the distance from any region to that set:

$$Z(\hat{R}_i^{(k)}) = \min_{R \in \Gamma_i} d(R, \hat{R}_i^{(k)}) \tag{4.30}$$

so that if each iteration can be shown to yield a region strictly closer to the solution set than the last region, then the descent function constraints would be satisfied. The only problem is that, even though this has always happened as the iterative process proceeds, it is difficult to mathematically demonstrate that this will always be true for any digital surface.

Although a good theory is worth a thousand computer runs, and although a finite metric-space model has been established, it is unclear at this point in time that such a complicated process can be effectively analyzed for convergence purposes. As a counter-proof against ideal fixed-point mathematical convergence (i.e., a unique limit region exists) for every possible seed region on every possible digital surface, counterexamples have been observed experimentally where, at some point in the iterative process, two regions are obtained such that $R_1 = \Phi(R_2)$ and $R_2 = \Phi(R_1)$, which would go on *ad*

infinitum unless specifically detected. However, both regions were good approximations to the final desired region in which case it could be said that both regions were members of the solution set and that the iteration did converge to the solution set in the sense of the Global Convergence Theorem. For the region-growing segmentation algorithm, it was found that (1) detecting when a region description may have been visited before during the iteration and then (2) terminating at that point has yielded excellent results.

To summarize, ideal, unique limit, fixed point ($R = \Phi(R)$) convergence is not provable for the current version of the iterative region-growing algorithm without the superset requirement because counterexamples have been observed. However, the best results on the test image database have been obtained using this version of the algorithm. This algorithm has always converged in the sense that the iteration for every original seed region terminates after a finite number of steps and ordinarily yields useful results. It may be possible to prove convergence to an approximate solution set in the sense of the Global Convergence Theorem [Luenberger 1984], but this has not yet been accomplished. It has been found that the quality of the segmentation output depends greatly on the maximum fit error threshold, the error tolerance increase function, and the zero curvature thresholds used to obtain the HK-sign map. A complete mathematical theory of the process would have to account for all these phenomena. Unfortunately, imposing the superset requirement on the region iteration does provide ideal fixed-point convergence properties, but has resulted in poor segmentation performance on several images in the test image database. There are many open problems in this area because the fundamental goal is to compute high-quality segmentations that are useful and meaningful, and a human mind is currently providing the evaluation function for these quality measures of usefulness and meaning. Levine and Nazif [1985] have tested a low-level segmentation performance evaluation function that uses uniformity and contrast

measures of lines and regions in an attempt to put this evaluation in the computer instead of in the head of the observer. It is not clear that such research will solve any of the problems stated here.

4.11.1. Termination Rules

The termination criteria for the iterative surface fitting algorithm are formulated in terms of the following quantities:

(1) The maximum error threshold: ϵ_{max}

(2) The total number of pixels in the compatible pixel list: $| C(\hat{R}_i^{(k)}, \hat{g}, \epsilon_i^{(k)}) |$

(3) The total number of pixels in the four-connected region of interest: $| \hat{R}_i^{(k)} |$

(4) The current mode of fitting: m_F

These quantities are used at each iteration to decide whether to continue fitting and growing or to stop.

The termination criteria are expressed as a set of rules and the reasoning behind each criterion is summarized.

(1) **RULE 1:** IF $\epsilon_i^{(k)} > \epsilon_{max}$ AND $m_F \geq | F |$, THEN Stop!

This rule is required due to the limitations of a finite set of approximating functions (fitting orders). The larger the set of fitting orders, the less likely it will be that this rule is needed, but any finite set of fitting orders requires this check for digital surfaces that bend and flex more than the most flexible function type. In the current implementation, $| F | = 3$. That is, three types of bivariate polynomial surface functions are used: planar, biquadratic, and biquartic. Because the surface fitting algorithm responds to the fit error exceeding the error threshold by increasing the order of the surface fit, this criterion expresses the fact that the

algorithm must quit when it runs out of fitting orders and the fit is not good enough. If there are not enough pixels in the growth region to support the fitting of a higher order surface, then the algorithm must stop also. Hence, $|F|$ in Rule 1 is actually dependent on $|\hat{R}_i^{(k)}|$, the number of pixels in the growth region. (That is, if the region size is less than 6, then $|F| = 1$. If the region size is less than 15, then $|F| = 2$.) But, since most regions contain more than 15 pixels, this subtlety is seldom encountered with digital surfaces that exhibit the surface coherence property.

(2) **RULE 2:** IF $|\hat{R}_i^{(k)}| \approx |\hat{R}_i^{(j)}|$ for any $j < k$, THEN Stop!

When the growing seed region stops growing, it is reasonable to quit. The spatial arrangement of the digital surface data prohibits further growth. The statement "for any $j < k$" is used to help prevent grow-shrink oscillations that are possible when the superset requirement is not used. It has been observed empirically that the surfaces obtained by stopping at such iterations are meaningful surface primitives. Note that this rule applies only to the connected growth region. This is a necessary, but not sufficient, condition for ideal mathematical convergence. The two region descriptions are not checked for equivalence on a pixel by pixel basis.

(3) **RULE 3:** IF $|\hat{R}_i^{(k)}| = |C(\hat{R}_i^{(k)}, \hat{g}, \epsilon_i^{(k)})|$, THEN Stop!

When the compatible pixel count and the connected pixel count are the same, there is no use in continuing the region growing algorithm because region growth has stopped. The next iteration's region must be a subset of the compatible list and therefore less than or equal to the current iteration's region size. This condition can occur in ideal convergence, but is not necessary.

(4) **RULE 4:** AT LEAST TWO (2) Iterations are required for a given surface fit order m_F before the algorithm is allowed to stop due to Rules 2 or 3.

This metarule (a rule about other rules) is exercised only occasionally, but it is important to include it. If it is not included, the fitting potential of the higher order surface function is not used effectively, and the algorithm terminates prematurely. This rule states that pixel counts from one fitting order should not be compared directly with pixel counts of a higher fitting order.

These four rules state all the essential concepts involved in terminating the surface fitting iteration. Of course, there is a maximum limit on the number of possible iterations to prevent extremely long iterations and guard against the possibility of infinite loops. In all tests done to this point, the maximum limit of 30 iterations has never been reached and the average number of iterations for all results in Chapter 7 is approximately seven. This statement is contingent upon the approximate equality statements made below.

While convergence to equality conditions in Rule 2 has been observed, it often takes much longer to achieve exact equality than to achieve good approximate equality. Moreover, it is difficult for the human observer to discriminate between approximately and exactly equal iteration results as used to describe a meaningful region in an image. Therefore, only approximate equality is required for termination in Rule 2. That is, if the quantities differ by less than 5 pixels or by 0.2% after the first five iterations, the quantities are assumed to be equal for termination criteria purposes.

The current algorithm terminates in a small, finite number of steps for each region, and produces useful surface segmentation results in almost all cases. Although convergence to a unique fixed-point limit is a desirable mathematical property, provably con-

vergent versions of the algorithm have been witnessed that produce regions containing significant errors in terms of scene surface content whereas the current algorithm, which may or may not be provably convergent, produces excellent (meaningful) regions even though grow-shrink oscillations might have occurred if the appropriate check were not incorporated in Rule 2.

4.12. Surface Acceptance and Rejection Decisions

After the surface growing iterations have terminated, one is left with the set of compatible pixels and the connected surface region itself along with the function parameters and the fit error. For growth surface regions that exceed the error threshold ϵ_{max}, but not by much, an acceptance zone above the error threshold is defined such that surface regions within the acceptance zone are accepted. The acceptance threshold used for the experiments is 50% greater than ϵ_{max}. Surface regions with fit errors beyond the acceptance zone are rejected.

When a surface region is rejected for any reason, the seed region responsible for the surface region is marked off in a writable copy of the HK-sign map as having been processed, which prohibits the use of the same original seed region again. When a surface region is accepted, all pixels in that region are similarly marked off in the HK-sign map so that they are not considered for future seed regions. In this respect, surface rejection and surface acceptance are similar. However, the surface acceptance process is much more elaborate and must update several other data structures besides the writable copy of the HK-sign map: the image error buffer, the best-fit region label buffer, the first-come, first-serve, within-tolerance region label buffer, and the best-fit so far surface list. In addition, two reconstruction images are maintained to allow visual interpretation of the results of the algorithm during execution and afterwards: the best-fit reconstruction

image and the first-come, first-serve, within-tolerance reconstruction image.

To describe how these data structures interact with the rest of the algorithm, their purpose and initialization conditions are described first, followed by the surface acceptance update procedure.

(1) The image Error buffer (E-buffer) is an array the size of the image used to define the smallest error encountered so far at each pixel between the best-fit surface and the original digital surface subject to connectivity constraints. The average of the contents of the E-buffer is computed after each surface acceptance to check if the total image error is less than the threshold ϵ_{max} in which case the entire algorithm terminates. The E-buffer is initialized by setting each pixel to a large value. The large value should be large enough to cause the algorithm to try to find a surface for almost every pixel, but it should also be small enough that a few pixels won't cause the average image error to be too high. The value 1024 was used by the program.

(2) The Best-Fit Region Label buffer (BF-RL-buffer) is an array the size of the input image that keeps track of the surface regions that are represented in the E-buffer. When a surface pixel has less than the current error in the E-buffer at that pixel, the new error is assigned to the E-buffer and the identification number of that surface (the region label) is assigned to corresponding pixel in the BF-RL-buffer. Also, the best-fit (BF) reconstruction image pixel is assigned the value of the surface function at the point. Thus, the BF-RL-buffer, the E-buffer, and BF reconstruction image form a closely related triplet of image buffers that monitor and influence the iterative process in a way that alleviates some difficulties caused by order dependencies in the sequential algorithm. The BF-RL-buffer is initialized to the

NO_LABEL state in which each pixel contains a marker denoting that no region owns that pixel. The BF reconstruction image is initialized to 0, or black.

(3) The First-come, first-serve, Within-tolerance Region Label Buffer (FW-RL-buffer) keeps track of the first surface region to fit the data at a pixel where the RMS error for the surface region was within the error threshold ϵ_{max}. Hence, once a pixel in the FW-RL-buffer is assigned, it is never changed, but it is known that the fit at the given pixel was quite good. Also, the first-come, first serve, within-tolerance (FW) reconstruction image pixel is assigned the value of the surface function at the point. The FW-RL-buffer is initialized to the NO_LABEL state in which each pixel contains a marker denoting that no region owns that pixel. The FW reconstruction image is initialized to 0, or black.

(4) The Best-Fit so far Surface List (BFS-list) is a third data structure used that allows for multiple surfaces to be defined at the same pixel. When regions grow, they are inhibited by the error of the best surfaces fit so far (the E-buffer), are controlled by the maximum error threshold ϵ_{max}, and are influenced by the FW-RL buffer, as described in the section above on region growing: if the surface error at a pixel is less than the maximum fit error threshold, and if either it is less than the current E-buffer value or that pixel is not marked in the FW-RL-buffer, then the pixel is assigned to the given surface. Because later surfaces can grow "under" the best-fit error of earlier surfaces (if they fit better), it is possible for different surface regions to have owned the same pixel at different stages in the algorithm. When surfaces grow substantially into one another, this is strong evidence of a smooth surface merge. This surface merging evidence motivates the maintenance of this third type of surface description mechanism. Since a region label buffer cannot be used,

separate run-length-encoded region descriptions are written to an intermediate file in our implementation of the algorithm. Since this list is built as the algorithm proceeds, no initialization is required. The BFS-list is a set of snapshots of each region as it is accepted.

The combination of the two region label buffers, the surface list, the error buffer, and the two reconstruction images provide an effective means for storing and monitoring the actions of the iterative region-growing algorithm as it operates on different seed regions in an image.

The surface acceptance module receives the compatible pixel list and the last version of the connected growth region as input. The compatible pixel list first undergoes a 3x3 window region refinement operation that (1) connects over one-pixel-size holes in regions (and removes isolated pixels), (2) fills in pixel-size notches in regions, and finally (3) trims off pixel-size protrusions. The form of these operations is the following: for each pixel in the binary image representing the set of compatible pixels, examine the pattern of the eight neighboring pixels. Modify the given pixel based on pre-decided allowable shapes for 3x3 neighborhoods in binary images. These operations are shown in Figure 4.15. They greatly enhance the humanly-perceived quality of the resulting region

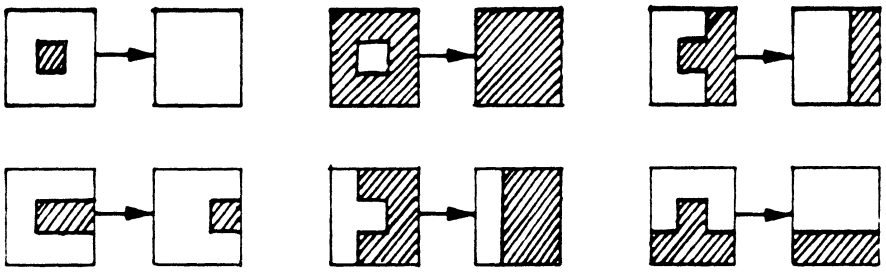


Figure 4.15. Example 3x3 Window Region Refinement Operations

descriptions. Under the assumption of surface coherence, all one pixel regions cannot be meaningful, are associated with noise effects, and are therefore to be ignored.

The largest connected region of the refined compatible pixel list that overlaps the previous version of the connected growth region is the final connected region of interest. This region is accepted unconditionally after the error tolerance check ($\epsilon_i^{(k)} < 1.5\epsilon_{max}$) and an absolute-minimum region size check ($|\hat{R}_i^{(k)}| >$ 4 pixels). When this region is accepted, the E-buffer is updated with the absolute value of the surface fit error at each pixel (the residuals), and the dynamic (regions-may-change) BF-RL-buffer, the static (regions-are-fixed) FW-RL-buffer, and their corresponding reconstruction images are updated with the surface region labels and surface depth values at each pixel respectively. The best-fit so far surface region description is also written to the static BFS-list mentioned above.

At this point, the updated region is dilated (expanded) N_D times to include all neighboring pixels of the region that are sufficiently close to the region of interest. These pixel values, which lie on the fitted surface, are compared to the original image and are used to update the E-buffer, BF-RL-buffer, and the BF reconstruction image *under the condition* that the absolute value of the error at each pixel is less than the error encountered so far (the error at each pixel in the E-buffer) and that the error is within a loose tolerance ($5w(\epsilon_i^{(k)})$). This step allows pixels that belong to meaningful surfaces, but do not fit within the specified growth error tolerance $w(\epsilon_i^{(k)})$, to be associated with a given surface as long as a better fitting surface does not come along subsequently and claim the pixels. This procedure remedies some of the shape limitations of the bivariate polynomial approximants and improves the quality of both the segmentation and the best-fit reconstruction image.

As discussed in the hemispherical surface example above (Section 4.10), the boundary region pixels at the steepest sloped area of the hemisphere may not fit the biquartic surface to within the specified tolerance. This extra expansion step allows the correct association of most pixels with the hemispherical boundary region despite the inability of a fourth order polynomial to bend exactly like a semicircle. Note that none of these neighboring pixels (that do not fit the surface within the specified growth error tolerance) are marked off in the HK-sign map; they are not considered to be correctly associated with a surface unless no other grown surface fits better.

The value of N_D, the number of dilations, presents yet another requirement for a heuristically or analytically determined constant. For larger values of N_D, more computation is required and the probability of incorrectly assigning a pixel to a wrong surface is increased. The smaller the value of N_D, the greater the chance that a pixel belonging to a particular surface may be missed. For the current implementation of the algorithm, $N_D = 6$ has served quite well. The value 4 also worked well in experiments.

The remaining set of compatible pixels that resulted from the last region growing iteration may contain regions that logically belong to the same physical surface, but are not directly connected to the grown seed region in context of the image. These regions must be evaluated to decide if they logically belong to the same physical surface associated with the grown seed region or if they have happened by a chance intersection of the fitted surface with the data in another area of the image. One very useful rule has been formulated regarding the acceptance of compatible regions that are not four-connected directly to the grown seed region, but seem to be a logical part of the grown seed region because of the shape consistency with the fitted surface.

DOUBLE CONTRACTION SHAPE STABILITY RULE: IF a non-connected, compatible region is stable against (does not vanish with) double contractions (erosions) and has a fit error to the surface function of the original region less than or equal to the error threshold met by the original region, THEN accept the region unconditionally as part of the original region even though they are not four-connected. Note: The original region either meets the error threshold condition specified by ϵ_{max} or the acceptable zone condition $1.5\epsilon_{max}$.

Double contraction shape stability is a heuristic criterion developed empirically by closely observing the performance of the region growing algorithm on hundreds of surfaces. If a region is contracted twice and still has more than enough points to fit a plane (four), that region is said to be double-contraction stable. Such a region must have at least 25 connected pixels and must have a smoothly shaped boundary and have few or no holes. There can be smaller logically connected regions that do not satisfy the above rule, but those that do satisfy it are almost always logically connected to the original surface. This rule has performed remarkably well, but it is sometimes too strict to unconditionally accept regions that should have been accepted on this status.

If a non-seed-related connected region of the compatible pixel list is double-contraction stable and the error is small as described in the rule, then this region is given the exact same treatment as the seed-related growth region: the E-buffer, the RL-buffers (and associated images), and the surface list are updated unconditionally, then the region is dilated, and the neighboring pixels are used to conditionally update the E-buffer and the BF-RL-buffer (and associated image) as described above.

If a non-seed-related connected region of the compatible pixel list is double-contraction stable and the error is NOT small enough to satisfy the above rule, then the

region is used to conditionally update only the E-buffer and the BF-RL-buffer (and the BF reconstruction image). The FW-RL-buffer and the BFS-list are NOT updated. The region is then dilated and the neighboring pixels are also used to conditionally update the E-buffer and the BF-RL-buffer (and associated image). This is a safe practice because the HK-sign map is not updated and the errors that go into the error buffer are generally larger than the maximum fit error tolerance. Thus, it is easy for subsequently processed surfaces to claim pixels already assigned during this phase if those surfaces fit the data well.

If a non-seed-related connected region is NOT double contraction stable, then the region is discarded. Regions of this type are fairly common owing to the nature of the parallel region growing process. For example, if a plane is fitted to a given region, that plane usually slices through the rest of the digital surface yielding many compatible pixels that are close to the plane, but they do not form large connected regions of a regular shape. The double contraction stability requirement is used to filter out these regions. It acts as a combination region-size and region-shape filter.

As noted in the section on region growing, the parallel region growing algorithm is logically equivalent to a sequential spiraling region growing algorithm *except for this final stage* in which compatible regions that are not connected may be assigned the same surface label as the main surface grown from the seed. In the coffee cup image in Chapter 7, the flat background visible through the handle of the cup is correctly assigned to larger background surface without high-level knowledge, only a surface consistency criterion. This is an excellent example of the data-driven pixel grouping capabilities of this segmentation algorithm.

4.13. Chapter Summary

The iterative region-growing algorithm based on a variable-order approximating surface function accepts (a) the HK-sign map and (b) the original image as input and produces as output (1) three separate surface region descriptions for each region as well as (2) the surface equation and the surface fit errors for each region. Two reconstruction images are also created that allow the user to visually evaluate the quality of the surface approximations. Segmentation into meaningful surface primitives is not yet complete because it is necessary to integrate the three separate region descriptions into one consistent region description for each surface primitive. At that point, surface primitives that join smoothly at their shared boundaries should be merged together to create the final data-driven surface region description. These topics are discussed in Chapter 5.

CHAPTER 5

DESCRIPTION COMBINATION AND REGION MERGING

The iterative region-growing algorithm isolates surface regions *sequentially* based on the original HK-sign map regions. As discussed in Chapter 4, three different region descriptions are recorded for each region so that the effects of the sequential processing might be accounted for in the final segmentation. The static first-come first-serve, within-tolerance and the dynamic best-fit region description computed by the algorithm attempt to provide dual region decompositions that can be integrated by a higher level process that understands how the region descriptions were created. The first-come, first-serve, within-tolerance (FW) region description is obtained by a sequential "monotonic" computational process that employs strict connectivty constrains and strict updating procedures The best-fit (BF) region description as obtained via a flexible, "non-monotonic" computational process that controls connectivty only by region formation and does not controls the connectivty of existing regions as affected by the relatively liberal best-fit updating. The best-fit so far (BFS) surface list complements these two dual region description by retaining a snapsnot of the best-fit regions of the time of region definition.

There are many different ways that these three complementary region descriptions could be combined to create a final region description for each region conceivable that different applications might require different integration algorithms depending on

the end goal. Whatever methods might be used, these methods could be categorized as monotonic and non-monotonic in the sense used above. Monotonic algorithms move straight ahead, never look back, and never modify past decisions, such as the decision to assign a given pixel to a particular region; they are therefore simpler on average than algorithms that are not monotonic. Excellent results have been obtained using a simple monotonic algorithm to integrate the three region descriptions. Although a monotonic integration algorithm may not in general produce the best possible results for every possible image, the extreme simplicity of this region-description combination algorithm merits its consideration given the high quality of the experimental results obtained using it.

After the presentation of the region-description combination algorithm, the region adjacency graph computation, which yields the list of adjacent regions needed for merging, is discussed briefly. Once the list of adjacent regions is computed, each adjacency relationship is tested for merging based on smooth surface join criteria. All appropriate regions are then merged to create the final region description. Each region description then consists of one or more polynomial surface patches associated with original surface regions. These surface patches are hypothesized to be meaningful scene surfaces for recognition purposes in range images even though only data-driven processing has been used to obtain them.

5.1. Region Description Combination

None of the three region descriptions are always satisfactory considered on their own. The best-fit region label buffer (BF-RL-buffer) is often fragmented; that is, regions can lose their original connectivity properties as better fitting surfaces are encountered. Examples of this fragmentation are shown in Chapter 7. The first-come, first-serve, within-tolerance region label buffer (FW-RL-buffer) is not fragmented due to strict

connectivity constraints, but regions often have ragged edges and do not connect well with neighboring regions due to strict error requirements. The best-fit so far surface list (BFS-list) generally contains overlapping region descriptions that need to be modified for the final segmentation output, which requires that each pixel belong to one and only one region.

A simple solution to the combination problem that has yielded excellent results is achieved by computing the union of the three descriptions and by not allowing any pixels already associated with another region. Let $\hat{R}_i^{BFRL}$ be the region description obtained for the i-th surface primitive from the best-fit region label buffer (BF-RL-buffer). Let $\hat{R}_i^{FWRL}$ be the region description obtained for the i-th surface primitive from the first-come, first-served, within-tolerance region label buffer (FW-RL-buffer). Let $\hat{R}_i^{BFS}$ be the region description obtained for the i-th surface primitive from the best-fit surface list (BFS-list). Then $\hat{R}_i^{U}$ is defined as the union of the three surface descriptions:

$$\hat{R}_i^{U} = \hat{R}_i^{BFRL} \bigcup \hat{R}_i^{FWRL} \bigcup \hat{R}_i^{BFS} \tag{5.1}$$

The final, combined region description is then obtained from the union description by masking off all regions defined thus far:

$$\hat{R}_i = \hat{R}_i^{U} - I_R \tag{5.2}$$

where

$$I_R = \bigcup_{j=1}^{i-1} \hat{R}_j \tag{5.3}$$

and where

$$A - B = A \bigcap NOT(B) \tag{5.4}$$

is the standard set difference operation. The universal set for complementation is the set of all image pixels I.

The ordering of the regions affects the final descriptions of all regions because of the differencing operation in Equation (5.2). For lack of a better philosophy in such a sequential approach, a "bigger is better" assumption is used to order the regions before they are processed by the combination algorithm. Since the BF-RL-buffer possesses the best segmentation in terms of the surface fit error only (without considering region connectivity), the regions are ordered according to their size in the BF-RL-buffer. Hence, a histogram of the BF-RL-buffer is computed and sorted, and then it is used to order the region combination process. This re-ordering based on best-fit region size tends to suppress difficulties that may be encountered owing to the order dependency of the sequential processing of the iterative region-growing algorithm.

The proposed method is simple and insures that no pixel is assigned to more than one region. It is a monotonic algorithm in that no pixel is ever reassigned to another region once it has been assigned to a given region. This method assumes that the first region is a meaningful region. Since the first region is chosen because it is the largest region represented in the BF-RL-buffer, the first region is typically a well-defined, reliable, meaningful region without extremely jagged edges (i.e., a high quality region). Because of the nature of this algorithm, if the first region is of high quality, then the other regions that follow it tend to also be of high quality.

5.1.1. Final Region Surface Primitives

Once all the regions have been defined by the sequential, monotonic, region-description combination algorithm, an intermediate list of surface primitives exists as the first-pass segmentation output. There is a correspondence between these bivariate polynomial surface primitives and the ideal theoretical HK-sign surface primitives, but it is not a one-to-one correspondence. All planar surface primitives correspond directly (one-

to-one) to flat HK-sign surface primitives. Since biquadratic surfaces must have constant Gaussian curvature sign (the second derivatives are constants), biquadratic surface primitives correspond directly (one-to-one) to curved HK-sign surface primitives if they are peaks, pits, ridges, or valleys. The one exception here is that saddle-shaped biquadratic surfaces may consist of combinations of saddle ridges, saddle valleys, or minimal saddle surfaces depending on the exact shape of the surface. Although many fitted biquartic surfaces correspond to single HK-sign surface primitives in the same way the biquadratic surfaces do, the flexibility of the fourth order surface allows it to bend, to grow into other adjacent, smoothly joined regions, and to describe more than just one of the basic HK-sign surface primitives. This does not generally cause problems in terms of the final segmentation results, but causes the intermediate list of surface primitives to not coincide exactly with the HK-sign surface primitive list that one might compute analytically.

An attempt was made to constrain fourth-order surfaces so that the sign of the Gaussian curvature (K-sign) remained constant over the entire surface. Such a constraint would yield a better correspondence between theoretical HK-sign surface primitives and the polynomial surface primitives produced by the algorithm. Since a biquadratic is always fitted to a region before the biquartic, the sign of the Gaussian curvature (K-sign) of the biquadratic was computed and assigned to the biquartic surface before the fit was made. During the region-growing procedure, all compatible pixels of a different K-sign were noted and not allowed to be compatible based on that difference. This inhibited the growth of the fourth order surface into areas that would take place otherwise in an non-inhibited algorithm. This inhibition requires the extra computational expense of evaluating the Gaussian curvature at each compatible pixel, which is considerable for fourth-order surface polynomials. This effort to obtain better HK-surface primitives did not work as planned for most images. The problem was that once

a zero curvature threshold was decided, there was always a region in some image with a little too much noise that caused the fourth order surface to warp slightly to conform to the data, and thereby exceed the curvature threshold. These pixels, which did indeed belong to the surface and were considered part of that surface under the biquadratic surface fit, were now disregarded at this advanced stage in the region growing process as incompatible. They were left unclaimed by the surface that they did logically belong to, and later became one or more separate surface primitives. In conclusion, the desired effect was achieved for several images in that the growth of the fourth order surfaces was inhibited into regions of different Gaussian curvature sign. But in other images the added constraints caused many regions that were previously correctly interpreted to subdivide into meaningless regions. This type of region growing constraint was abandoned for this practical reason and the reason discussed below.

If an approximating polynomial surface is capable of growing from one HK-sign surface primitive into another adjacent HK-sign surface primitive, then these two surface primitives must be approximately smoothly joined because polynomial surfaces cannot kink. Why should the algorithm bother to separate smoothly joined regions of different HK-sign types in one stage of the algorithm if the next stage will simply join them together anyway? Despite the theoretical appeal of a possible one-to-one correspondence between HK-sign surface primitives and the fitted bivariate polynomial surfaces, it was decided that biquartic surfaces should be allowed to grow without constant surface curvature sign constraints so that the algorithm did not make extra work for itself during an intermediate stage. The primary interest is in a final, meaningful, smooth surface segmentation, not in the theoretical, intermediate, mathematical decomposition of surfaces. This non-restriction of biquartics has caused a few relatively minor problems in detecting small orientation discontinuities in noisy images as shall be pointed out in the

experimental results of Chapter 7.

5.2. Region Adjacency Graphs

A region adjacency graph is a set of regions accompanied by a set of adjacency relations between the regions in that set. The regions are the nodes of the graph, and the adjacency relationships are the undirected arcs of the graph. The region description combination algorithm produces a single region list, but an adjacency algorithm is needed to compute the adjacency relationships of the region list and store them in a neighbor list. An efficient way to compute the neighbor list for images with touching adjacent regions using only two intermediate image-size buffers is as follows:

(1) Put the finalized region list in the form of a region label buffer, where each pixel contains the label of the surface region to which it belongs and initialize the second scratch buffer to all zeros.

(2) Start with $i = 1$. For each i-th surface region, do the following using i as the search label: for each pixel with the label i, examine the 3x3 neighborhood to see if any neighbor pixel has a different value than i. If it does, mark that neighbor pixel by putting its value in the scratch buffer at the same location as it occupies in the region label buffer.

(3) Compute the histogram of the scratch buffer. Each label with a non-zero count (besides zero) represents an adjacency relationship, which is added to the neighbor list if it is not already there. The size of the count indicates the strength of the adjacency, which is the (digital) arc length of the separating boundary.

(4) Reinitialize scratch buffer to zeros, increment the search label to $i + 1$, compute the neighboring pixels while adding them to the scratch buffer, and then compute the histogram and neighbor list for the next region label.

When this process is completed for each possible region label, the neighbor list (and the region adjacency graph) is complete.

An alternative approach requires more memory, but can bridge gaps of n pixels between regions without requiring a large window size for label comparisons and is directly useful in the context of the surface merging algorithm discussed in the next section. For example, the FW-RL-buffer contains gaps between regions such that the algorithm in the above paragraph would not be effective using a 3x3 window. Each region should be stored as efficiently as possible. For the software implementation, a bitmap was used as a tradeoff between storage space and ease of use. Regions are dilated (or expanded) by an appropriate number of times depending on the size of the gaps that may need to be bridged and are then stored in an efficient manner. *Double dilations* were used so that touching adjacent regions would have a four-pixel wide overlap for the surface merging computations to be discussed in the next section. Three-pixel wide gaps can be bridged for adjacency checks; this would require 7x7 comparison windows using the method described above. The dilated regions (in bitmap or other condensed form) are then checked for overlap (non-null intersection) to decide if two regions are adjacent. If the min-max box for each region is stored along with each region description, many overlap checks are quickly dismissed with negligible computation. When an overlap check indicates overlap, the adjacency relationship is noted, and the overlap region, the two surface equations, and the fit errors are then given to the smooth surface merge decision module (along with original image) to decide if a smooth surface join, an orientation discontinuity, a depth discontinuity, or some combination of the above exists between the adjacent regions.

In summary, two algorithms have been described for computing region adjacency graphs: a standard algorithm and a special purpose algorithm for bridging gaps and feeding the required information to the surface region merging algorithm described next.

5.3. Region Merging

Two surface primitive regions should be merged if and only if they are adjacent to each other AND they join smoothly at the connecting boundary curve between the surfaces. Continuous, differentiable surface functions join *smoothly* along a boundary curve if the local surface description of both surfaces is the same at all points on the curve. In other words, the first and second fundamental forms (or the metric and the shape operator) of the two surfaces should be equal along the boundary curve. If these quantities are equal along the curve, then the following statements are also true: (1) both sets of surface normals along the curve are completely aligned, and (2) the line integral of the norm of the surface difference vector along the curve is zero (due to equality of surfaces at boundary).

Unfortunately, there are no boundary conditions placed on the approximating functions since they are used for extrapolation during the region growing procedure. Hence, two grown surface primitives may belong to the same smooth surface, but may have different local surface behavior at the boundary between the surfaces because of the lack of explicit boundary conditions during the surface fitting process. Consider the break in the parameter curve from the best-fit reconstructed range image of a noisy torus as shown in Figure 5.1. The top function shows cross-sections of two polynomial surface approximants that should have joined smoothly given that the data joins smoothly. The bottom function represents the actual depth data along that parameter curve. The maximum fit error threshold was larger than it needed to be for this image ($\epsilon_{max} = 4.2$),

which allowed the polynomial surfaces to wander farther from the data than they should have for the given noise level. This break could be attenuated somewhat via the use of a weighted least squares fit where large weights are associated with the pixels near the boundary of the surface region, but this may undesirably change the surface fit on the interior of the region. But no matter what approximations are made to obtain surface equations near the boundary, it is certain that ideal equality of surface parameters at the boundary curve will never be obtained very reliably given noisy data. Hence, notions of approximate equality and thresholds are necessary.

Since thresholds are necessary anyway, surface regions are not refitted to obtain better boundary pixel descriptions. Instead, an attempt is made to judge the smoothness of the join between two surfaces based on (1) the approximations obtained during the region growing process (equally weighted least squares approximants), and (2) the original data.

Let us now assume that surface equation A, surface equation B, the overlapping region (intersection) of the dilated region descriptions R_O, and the pixel values of the original image in the overlapping region are all given. A decision must be made regarding the boundary between surfaces A and B. The smooth surface join question is the

Figure 5.1. An Anomalous Break in Parameter Curve for Noisy Torus

primary issue since the two regions are either to be merged or not merged. A secondary issue is the depth discontinuity or orientation discontinuity or combination discontinuity question. Figure 5.2 shows a top view of two adjacent regions and the corresponding overlap region R_O as well as side views of the three possible continuity states across the boundary. A set of rules have been developed to make the region merging decision based on several different quantities related to the geometry of the two surfaces. The quantities of interest are presented below along with the corresponding rules.

5.3.1. Region Merging Decision Rules

The region merging decision rules are formulated to determine if two approximating surfaces join smoothly at a boundary. The boundary is represented by a boundary region, the overlapping region contained in the dilated region descriptions of both surfaces. This region may be as much as four pixels wide. Let us consider this boundary region as the compact support for two surface functions (approximants). If the two sur-

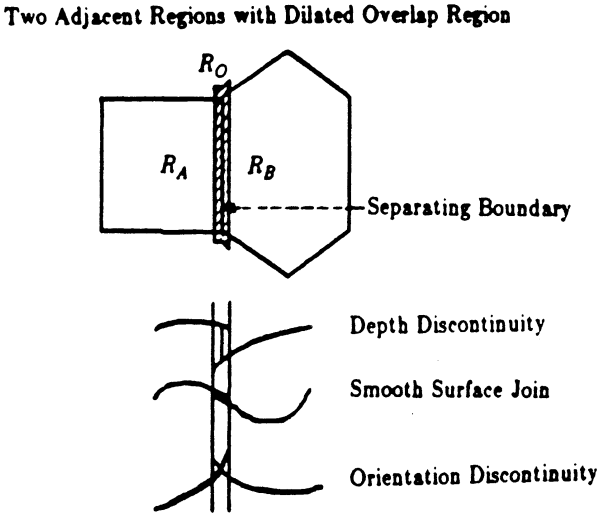


Figure 5.2. Adjacent Region Boundary Terminology

faces do not intersect at all over the support region and the mean separation distance between the two surfaces is sufficiently large, one may conclude that the two surfaces do not join smoothly along the boundary curve. In this case, a pure depth discontinuity exists at the boundary. If the two surfaces do intersect or if the mean separation distance is small, then there is not a pure depth discontinuity and there is a possibility that surfaces join smoothly. Then surface normal information is used to evaluate the compatibility of the two surface orientations over the support region. If the orientations of the surfaces are compatible and no depth discontinuity exists, a smooth surface join is detected. Two quantities are used: (1) the average of the angular separations of the normal vectors at each point, and (2) the angular separation of the average normal vectors of the two surfaces over the support region. If both of these quantities are in the same range and at least one of them is less than a specified threshold, then the two surfaces have roughly the same orientation in space over the support region. In accordance with the stimulus bound approach, the angular threshold depends on the surface fit errors to the original data over the support region. If both surface fits are good, the angular threshold is small. If either or both surface fits are poor, the angular threshold is large. This simple compensation scheme has been effective in providing meaningful smooth surface joins in a variety of noise levels over a wide variety of images. To summarize the logic of the decision making process, a boundary is either a pure depth discontinuity or it is not. If it is not a depth discontinuity, the boundary is then either a smooth surface join or it is not. If it is neither a depth discontinuity or a smooth surface join, then it may be an orientation discontinuity or some combination of all three states. For merging purposes, no further distinction is needed.

The first decision parameter is the easiest to compute, the most exact, and perhaps the most meaningful. Let R_O be the overlapping region obtained by intersecting the

dilated region descriptions for surfaces A and B. Let $z_A(p)$ be the surface value at pixel p considered as part of surface A and let $z_B(p)$ be the surface value at pixel p considered as part of surface B. Note that the extrapolating properties of the bivariate polynomials are again being used. In general, an approximating function over region A need not be defined over region B. The average error, referred to as the *mean separation*, is defined as

$$e_0^{AB} = \frac{1}{|R_O|} \sum_{p \in R_O} (z_A(p) - z_B(p)) . \tag{5.5}$$

Note that the mean separation is a ***signed quantity***; it may positive or negative, depending on whether or not surface A is mostly over surface B or mostly under surface B. The average absolute error, referred to as the ***mean separation distance***, is defined as

$$e_1^{AB} = \frac{1}{|R_O|} \sum_{p \in R_O} |z_A(p) - z_B(p)| . \tag{5.6}$$

Note that the mean separation distance is always a non-negative quantity. Thus, the mean separation is not the same as the mean separation distance as defined here. In general, it will always be true that

$$|e_0^{AB}| \leq e_1^{AB} \tag{5.7}$$

which is proved by simple inspection. Let the first decision parameter be d_1, which is defined as

$$d_1 = \frac{e_1^{AB} - |e_0^{AB}|}{e_1^{AB}} \geq 0 . \tag{5.8}$$

Note that if surface A is above surface B at every point in R_O, then d_1 is exactly zero because $e_0^{AB} = e_1^{AB}$. But if one pixel of surface B goes above surface A, these quantities cease to be equal. Similarly, if surface A is below surface B at every point in R_O,

then d_1 is still exactly zero because $e_0^{AB} = -e_1^{AB}$. If one pixel of surface A goes above surface B, these quantities cease to be equal. Hence, $d_1 = 0$ for a pure depth discontinuity *independent of the magnitude of the mean separation distance!* It increases as the surfaces intersect one another and is bounded by the value 1.0, which can be obtained only when $e_0^{AB} = 0$. This can occur when the two surfaces clearly intersect at the boundary, and the mean separation on one side of the surface intersection exactly cancels the mean separation on the other side of the intersection. Hence, d_1 is a useful, meaningful, descriptive parameter of a boundary curve. These results are stated explicitly as a rule.

RULE 1: IF $d_1 = 0$, THEN surfaces A and B do not intersect each other AND a depth discontinuity exists at the boundary between surface A and B. IF $e_1^{AB} > 2.5$ depth levels, a sufficiently high contrast step-edge exists AND the boundary is classified as a depth discontinuity. If not, the boundary is a very low contrast step edge, and it may actually be a smooth surface join. In this case, a decision is not made until other rules are applied.

The threshold 2.5 was chosen for its performance over a wide set of images. In practice, this threshold might be set using domain specific assumptions about depth discontinuities. Also, d_1 was not required to be exactly zero for a depth discontinuity. As long as the decision parameter was smaller than a threshold of 2%, it was found to be small enough to imply the existence of a depth discontinuity. Note that, for object recognition algorithms, it is possible to tell which region is occluding and which region is occluded using the sign of the mean separation.

The alignment of surface normals of the two surfaces forms the basis for the two second decision parameters d_{20} and d_{21}. These parameters are designed to be useful for

arbitrary surface joins. Let $\vec{n}_A(p)$ be the unit surface normal at the pixel p computed from the bivariate surface polynomial for surface A. Let $\vec{n}_B(p)$ be the unit surface normal at the pixel p computed from the bivariate surface polynomial for surface B. The first of the two decision parameters based on normal vectors is computed on a pointwise basis as the arc cosine of the average dot product between surface normals at each point:

$$d_{20} = \cos^{-1}\left(\frac{1}{|R_O|} \sum_{p \in R_o} (\vec{n}_A(p) \cdot \vec{n}_B(p)) \right) \tag{5.9}$$

This quantity averages the angular difference of the surface normals at each point. If the orientation of the two surfaces is approximately the same over the support region around the boundary, then d_{20} will be small. It is noted that another parameter could be based on the mean absolute value of the surface normals inner product.

The second of the two surface normal decision parameters is computed by averaging the normal vectors over each surface and then computing the arc cosine of the dot product of the average normals:

$$d_{21} = \cos^{-1}\left(\frac{\vec{n}_A \cdot \vec{n}_B}{|\vec{n}_A| \, |\vec{n}_B|} \right) \tag{5.10}$$

where

$$\vec{n}_A = \frac{1}{|R_O|} \sum_{p \in R_o} \vec{n}_A(p) \tag{5.11a}$$

$$\vec{n}_B = \frac{1}{|R_O|} \sum_{p \in R_o} \vec{n}_B(p) . \tag{5.11b}$$

Hence, this quantity averages the normal vectors for the two surfaces separately over the support region, and then renormalizes the average surface normals to find the angular separation of the average normals. If the orientation of the two surfaces is approximately the same over the support region around the boundary, then d_{21} will also be

small. If both d_{21} and d_{20} are small simultaneously, then the surfaces are constrained to have approximately the same orientation. These two parameters are used together in the following rule:

RULE 2: IF Rule 1 does not apply AND if $e_1^{AB} \leq 15$ depth levels AND IF either d_{20} OR d_{21} is less than or equal to 25 degrees AND IF $| d_{20} - d_{21} | < 25$ degrees, THEN the two surfaces A and B are merged.

If A and B are both planar surfaces, then $d_{21} = d_{20}$. In this special case, these two decision parameters should not be computed and the planar equation should be used to determine the normal vectors instead of Equations (5.11). Also for intersecting planes, the 25 degree threshold can be made as small as is needed (within the constraints of the computer) before a merge is allowed. For example, this would be useful in a polyhedral domain. Two planar surfaces also allow the formation of a perfectly defined dihedral edge in 3-D space. For the current software implementation, no surfaces were given special treatment, and the same angular thresholds were used regardless of surface type. The thresholds of 15 depth levels and 25 degrees were chosen for their good performance over the entire test image database. In practice, domain-specific knowledge could be used to assign these thresholds. Note that, for orientation discontinuities, $\vec{n}_A$ and $\vec{n}_B$ can be used to label straight or curved edges as predominantly convex or concave. Such labels may be useful in matching.

The second rule is a good rule as long as the local approximations for surface A and surface B fit the data well in the overlapping region R_O. However, if the surfaces do not fit well, it is necessary to be considerably more tolerant of differences in normal direction. Toward the goal of detecting the goodness of the fit over the support region, the average absolute errors for both surfaces are computed:

$$e_1^A = \frac{1}{|R_O|} \sum_{p \in R_O} |z_A(p) - z(p)| \tag{5.12a}$$

$$e_1^B = \frac{1}{|R_O|} \sum_{p \in R_O} |z_B(p) - z(p)| \tag{5.12b}$$

where $z(p)$ is the depth level from the original image. If either or both of these two values are greater than a certain large percentage of the mean separation distance, the surface fit is not considered adequate to finalize a strict decision that the surfaces do not join smoothly. Therefore, the following rule, which is based on the original image data, was also included:

RULE 3: IF Rules 1 and 2 do not apply AND IF both surface A and surface B are not planar AND IF $e_1^{AB} < 1.3 \max(e_1^A, e_1^B)$, THEN re-apply Rule 2 using a doubled angular threshold of 50 degrees and a doubled depth threshold of 30 depth levels.

For particularly noisy surfaces, it was necessary to include such a rule in order to get meaningful surface merges on complex surfaces. When there is not much noise present, the rule has no impact on merging decisions because it is not invoked. However, in noisy images there is a tradeoff in that certain orientation discontinuities that are not very prominent are merged across and are not present in the final segmentation results.

A fourth rule is needed to terminate the decision tree structure of the rules in this chapter:

RULE 4: IF Rules 1, 2, and 3 do not apply, THEN the boundary between the surfaces A and B is not a smooth surface merge and it is not a pure depth discontinuity. It may be a pure orientation discontinuity, or it may be any combination of the three possible states of surface boundaries.

Since the boundary is not a smooth surface join, the segmentation output does not

require further analysis of the edge. Rule 1 provides easily computed, reliable *negative* evidence about the existence of a smooth surface join whereas Rules 2 and 3 provide *positive* evidence. In object recognition systems, it may be advantageous to label each region boundary segment as an occluded depth discontinuity, an occluding depth discontinuity, a convex orientation discontinuity, a concave orientation discontinuity, or a smooth surface join. The decision parameters above provide sufficient information for such labeling.

It has been observed empirically that if two regions in the BFS-list overlap by a significant amount, then this is strong positive evidence that the two surfaces should be merged because each surface approximates the other well (within the specified maximum error tolerance) over a large number of pixels. This condition is actually checked first before any of the above rules are applied.

RULE 0: If $| \hat{R}_A^{BFS} \bigcap \hat{R}_B^{BFS} | > 0.35 | R_O |$, then surfaces A and B should be merged and Rules 1, 2, 3, 4 need not be checked.

This rule is discussed last because it is not based on general-purpose 3-D surface shape arguments, but on the algorithm-specific nature of the region growing process.

These merging rules represent simple criteria for deciding if two surfaces should be merged or not. They are not intended as an optimal set of statistical decision rules, but rather a set of geometry-based rules that capture the basic ideas for testing the smoothness of joins of the polynomial approximation functions in the presence of noise. The segmentation approach is becoming less data-driven at this stage in the processing. Although the algorithm is still based on general properties of surfaces, the necessary thresholds are more related to the interests of perceiving entity than to the properties of the data. The advantages of the approach are that it provides a clear-cut entrance for

the usage of domain-specific information.

1.1. Final Region Segmentation Output

Three region descriptions are combined to create a unified list of polynomial-surface-primitive regions that is generally of better quality than any of the three region descriptions considered individually. Adjacency relationships are then computed from the unified list, and each pair of neighbors is tested for merging by applying the set of rules described above. If a smooth surface merge decision is affirmative, then the (non-dilated bitplane) polynomial-surface-primitive region-descriptions for the two regions are merged to create a new smooth-surface-primitive region. After this process has been done for every pair of neighbors, the final smooth-surface-primitive region list is formed consisting of all non-merged and merged polynomial-surface-primitive regions. This smooth-surface-primitive region list is the final output of the segmentation algorithm. The surface equations and the unified region descriptions of the polynomial surface primitives are maintained to allow the correct surface equations of the merged smooth surface regions to be used with the appropriate pixels within the merged regions. If adjacency relationships of the final regions are required, the final output is rerun through a region adjacency algorithm creating the final smooth-surface-primitive region neighbor list. If data-driven parametric descriptions of region edges are required to complement the data-driven parametric descriptions of the graph surface primitives, then the boundary pixels of each region can be fed to the sign-of-curvature edge fitting algorithm (discussed in the next chapter), which generates polynomial edge interval descriptions. If only a segmentation plot is required, this can be computed directly from the final region list

without further computations. In the experimental results section, both parametric edge descriptions and segmentation plots are used to display the segmentation algorithm output.

CHAPTER 6

REGION EDGES

Edges play a dominant role in computer vision research, but have received little attention thus far in this thesis. Although independent edge detection and edge linking algorithms are not of direct concern to a discussion of the surface-based data-driven range image description algorithm, the topic of parametric edge description is of significant interest for the following reason. The approximate shape of smooth surfaces in images is given *parametrically* by the approximating graph surface functions $z = \hat{g}_i(x, y)$. However, the 2-D image regions associated with these approximants are still specified by a pixel list, a bitplane, or some other *non-parametric* form according to the algorithm expressed thus far. The boundary of an image region can be encoded as a chain code, which is a general, but non-functional, method for describing regions in a discrete image.

A functional edge description offers the same advantages of the functional surface description: groups of pixels are represented concisely, mathematically, and symbolically for higher-level processes. For example, if the boundary of a planar region happens to form a quadrilateral, it is preferable to have a parametric linear description of the linear edges for higher-level processing rather than a chain code. There are many methods, such as the Hough transform [Duda and Hart 1972], for isolating linear edges in edge maps, but a data-driven edge description algorithm that characterizes straight and

curved edges equally well without a bias towards either type of edge is desired. An edge segmentation algorithm is sought that will partition an arbitrary curve into meaningful intervals and generate an explicit representation for each interval. In this chapter, the *sign-of-curvature paradigm* developed for surface description is used to describe the arbitrary edges that bound the arbitrary four-connected regions created by the surface-based segmentation algorithm. By reducing the dimension of the sign-of-curvature approach from 2-D to 1-D, two goals are accomplished: (1) a data-driven region-boundary (edge) description algorithm is derived to complement the smooth surface primitive description obtained thus far, and (2) the general, dimension-independent nature of the sign-of-curvature surface algorithm is presented in a simpler context that provides perhaps a more illuminating vantage point and several interesting contrasts.

The input to the edge description algorithm is an eight-connected set of pixels that form a one-pixel wide path through the image. The approach does not require a closed edge path although closed paths arise naturally when boundaries of connected regions form edges. Hence, any edge detection/linking algorithm that yields eight-connected one-pixel-wide edges can provide input to this edge description algorithm. Before describing the edge algorithm in detail, a useful framework is established for discussing edges in different dimensions since the surface-based edge detection mechanism that provides input to the edge description algorithm is quite different from most existing edge detectors. Also, although a higher dimensional segmentation algorithm will not be discussed, the following discussion hints at how the sign-of-curvature paradigm can be generalized to higher dimensions.

6.1. Edge Detection and Dimensionality

The concepts of edges, boundaries, discontinuities, and dimensionality are extremely important in computer vision, but are sometimes confused by the use of conflicting terminology. Consider terms like "3-D images" and "3-D edge detectors" that may imply totally different mathematical entities depending on the context in which they are used. For example, the term "3-D image" is used to refer to range images [Snyder and Bilbro 1985] that measure depth as a function of two spatial variables and is used by others to refer to stacks of density images that measure density as a function of three spatial variables. Also, the 3-D edge detector of Zucker and Hummel [1981] is really a surface detector, not an edge detector in the usual sense of the term. The following is an attempt to provide a common framework for several computer vision terms.

An n-D *image* (not the usual mathematical term "image") represents a piecewise smooth mapping $f(\vec{x})$ from $\mathbf{R}^n \rightarrow \mathbf{R}^m$ where $\vec{x} \in \mathbf{R}^n$. One usually deals with n-D *scalar* images where $m = 1$, but n-D *m-vector* images are also encountered where m is an integer larger than one. Specifically, a 1-D scalar image represents a mapping of the type $y = f(x)$ (a function of one variable), a 1-D 2-vector image represents a mapping of the type $(x,y) = (f(s),g(s))$ (a planar curve), a 2-D scalar image represents a mapping of the type $z = f(x,y)$ (a graph surface), and a 3-D scalar image represents a mapping of the type $w = f(x,y,z)$. Hence, intensity images and range images are both scalar 2-D images in this terminology. This does not really contradict the popular "2.5-D image" term applied to range images since half of a dimension is not defined here, but it does set of guideline for not referring to range images as 3-D images. Range images are 2-D images just like intensity images, only the *interpretation* of the pixel values make any difference. The term *3-D image* is reserved for real three dimensional

images, such as those obtained by stacking CAT-scan data in biomedical imaging. Any long sequence of 2-D images, such as those encountered in dynamic scene analysis, may be considered as a 3-D image entity. Subsets of the domains of 1-D, 2-D, and 3-D images are *intervals, regions, and volumes* respectively. The size of these subsets are *lengths, areas, and volumes* respectively.

A simple order-0 discontinuity point in an n-D image (an $(n-1)$-D *step-edge* point) is a point at which there exists a unit-norm direction vector $\vec{u} \in \mathbf{R}^n$ ($|\vec{u}| = 1$) such that, for α strictly non-negative,

$$\lim_{\alpha \to 0} f(\vec{x} + \alpha\vec{u}) \neq f(\vec{x}) . \tag{6.1}$$

It is assumed that simple order-0 discontinuities in an n-D image form connected $(n-1)$-D geometric entities; more pathological arrangements of discontinuities are not considered because they are not of interest to computer vision researchers. For a piecewise smooth 1-D image (function) $y = f(x)$, simple order-0 discontinuity points form 0-D sets (points) that are the boundaries of the continuous pieces of the image. These discontinuity points represent the "0-D step edges" in the 1-D image. For a piecewise smooth 2-D image (function) $z = f(x,y)$, simple order-0 discontinuity points form 1-D sets (curves) that are the boundaries of the continuous pieces of the image. These discontinuity points represent the "1-D step edges" in a 2-D image: depth discontinuities in a 2-D range image and intensity discontinuities in a 2-D intensity image for example. For a piecewise smooth 3-D image (function) $w = f(x,y,z)$, simple order-0 discontinuity points form 2-D sets (surfaces) that are the boundaries of the continuous pieces of the image. In this case, the "2-D step edges" of a 3-D density image are the *surfaces of solid objects.*

This is the fundamental concept in this discussion. Since the real world may be modeled as a 3-D density image, a rangefinder (mainly a piece of hardware) determines the 2-D step edges in a 3-D image to create a 2-D (range) image that represents the 2-D (surface) boundaries of 3-D objects in a scene. According to the usual edge detection paradigm, an algorithm (software) must then detect the 1-D step edges in the 2-D image to find the 1-D (edge) boundaries of the 2-D surfaces in the (range) image. That is, the *range sensor directly detects surfaces*, but edge (boundary of surface) information must be *extracted computationally* from the range image. Video cameras, on the other hand, detect light reflected from object surfaces (the 2-D step edges of a 3-D image) to create a 2-D (intensity) image, which has a completely different meaning than a range image. The assumption (which is not always true) that intensity 1-D edges coincide with meaningful scene edges motivates the desire to detect (via computation) the simple order-0 discontinuities in intensity images. This is the edge detection problem addressed so often in the literature. For range and intensity imaging, digital surfaces are created by the sensor, but digital edges are usually *computed* from the digital surface and are not ordinarily sensed directly by hardware. The meaning of an edge or a surface depends on the type of physical quantity being sensed by the sensor.

A simple order-1 discontinuity point in an n-D image (an $(n-1)$-D *roof-edge* point) is a point $\vec{x}$ at which there exists a unit-norm direction vector $\vec{u} \in \mathbf{R}^n$ such that the directional derivative at the point $\vec{x}$, defined as

$$\lim_{\alpha \to 0} \frac{f(\vec{x} + \alpha\vec{u}) - f(\vec{x})}{\alpha} = \vec{\nabla} f(\vec{x}) \cdot \vec{u} \ , \tag{6.2}$$

is not a well-defined quantity because the value depends on whether α approaches zero from the positive or negative side. It is assumed that simple order-1 discontinuities in an n-D image form connected $(n-1)$-D geometric entities; again more pathological

arrangements of these discontinuities are not considered. For a piecewise smooth 1-D image (function) $y = f(x)$, simple order-1 discontinuity points form 0-D sets (points) that are the boundaries of the *differentiable* (smooth) pieces of the image. These discontinuity points represent the "0-D roof edges" in the 1-D image. For a piecewise smooth 2-D image (function) $z = f(x,y)$, simple order-1 discontinuity points form 1-D sets (curves) that are the boundaries of the differentiable pieces of the image. These discontinuity points represent the "1-D roof edges" in a 2-D image, i.e. the orientation discontinuities in a 2-D image. For a piecewise smooth 3-D image (function) $w = f(x,y,z)$, simple order-1 discontinuity points form 2-D sets (surfaces) that are the boundaries of the differentiable pieces of the image. In this case, the "2-D roof edges" of a 3-D (density) image are found in the interior of objects.

Higher order discontinuities in n-D images exist, but do not correspond to changes in image smoothness. Order-0 and order-1 discontinuities in n-D images form the boundaries of *smooth* image regions and are therefore the most interesting and most prominent types of discontinuities in the perception of piecewise smooth images.

If range image segmentation is to be accomplished using the edge detection paradigm, both depth discontinuities (step edges) and orientation discontinuities (roof edges) must be detected and *integrated* to create a range image partition of smooth surface primitives. Although the edge detection approach is a valid approach and has been used by many (e.g. [Bolles et al. 1983][Tomita and Kanade 1984]), it certainly is not the only possible approach. It appears that an appropriate mix of edge-based and surface-based approaches will provide the best results obtainable for (range) image segmentation. However, this thesis is intended to explain and demonstrate the power of surface-based segmentation and does not digress to discuss separate edge-based techniques.

As an example of the differences between edge- and surface-based approaches, consider a circular hole drilled in a flat surface viewed obliquely as shown in Figure 6.1. A circular edge would be visible in a range image of the hole. One half-circle is a step edge whereas the other half-circle is a roof edge. An edge-based method must first detect both types of edges and then link them together to get the set of pixels that form the circle. In contrast, the surface-based approach finds and fits the pixels belonging to the flat surface, and a circular boundary is created around the hole (where the range image pixels do not fit well with the flat surface description). Hence, *edges are indirectly detected in the surface-based approach as boundaries of smooth surfaces.* Roof edges and step edges do not need to be discriminated, computed explicitly, or linked together.

The dimensionality of different types of images has been defined along with the dimensionality of two types of edges that occur in each type of image. Order-0 and order-1 discontinuities (step and roof edges) were defined and isolated as the key edge types for range image perception of scenes with smooth surfaces. Order-0 discontinuities are the primary type of edges given attention in intensity image processing work. Imaging sensors create 2-D images, but range sensors explicitly detect the 2-D step edges (sur-

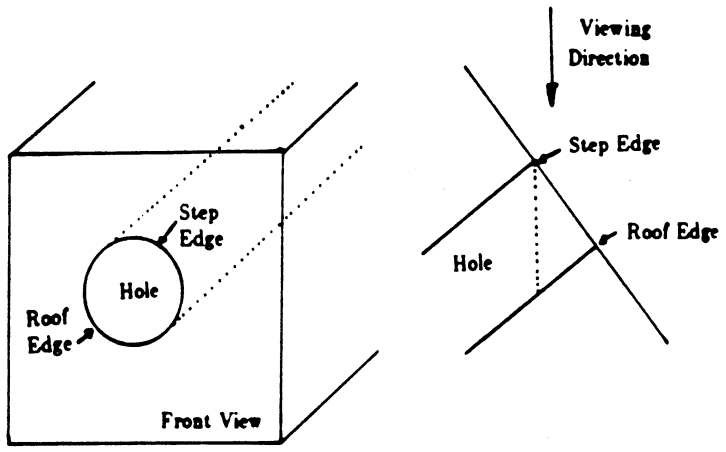


Figure 6.1. Circular Hole in Flat Surface Has Two Edge Types

faces) of 3-D density images (objects in field of view). Explicit digital computation is normally required to detect 1-D edges in any type of digital 2-D image.

6.2. Edge Algorithm Description

The input to the edge description algorithm is the definition of a 2-D four-connected region associated with a smooth surface. The output is an explicit approximate functional representation of the region boundary. As an example of the output for several regions, the set of all boundary descriptions for a view of the block with three holes is shown in Figure 6.2. Since all image pixels belong to one and only one region, region boundary pixels are not shared by separate regions. Therefore, two region boundaries, one for each region, are explicitly drawn for every physical edge. The internal processing steps of the edge algorithm mirror the steps of the surface characterization algorithm and the iterative, surface-based, region-growing algorithm. Those two algorithms together produced parametric descriptions of surfaces. In this chapter, the edge characterization algorithm and the iterative, function-based, interval growing algorithm

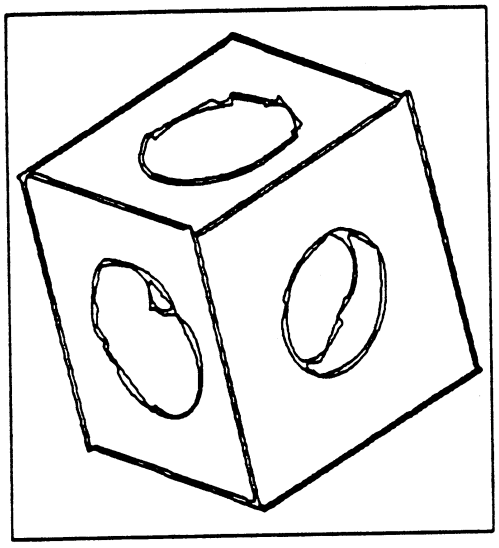


Figure 6.2. Edge Descriptions of Region Boundaries for Block

are used together to produce parametric descriptions of edges.

Each part of the edge description algorithm is described in detail subsequently, emphasizing the differences required for edges as opposed to surfaces. The material is divided into main sections on edge definition, edge characterization, seed interval extraction, edge fitting, interval growing, termination criteria and interval acceptance, and interval merging at smooth interval joins.

6.3. Edge Definition

Edges are defined directly from a 2-D four-connected region description in the following manner: if a region pixel is four-adjacent to a pixel that is not in the region, then that pixel is an edge pixel. This process creates a one-pixel-wide eight-connected edge map representing the boundaries of the region. If there are no arms or holes of the region that are only one pixel wide, then the edge map contains only simply-followable, non-branching, closed edge paths: one path for the outline of the region and one path for each hole in the region. If one-pixel-wide region arms do exist, as shown in Figure 6.3, then the edge map is still followable except that the one-pixel-wide edge path(s) are not closed. It is assumed that the input to the edge algorithm is a single simply followable path, but it is not required that it closes on itself. If necessary, one-pixel-wide arms can be eliminated using multiple applications of the 3x3 burr-trimming operator discussed in Chapter 4.

Since the edge input is assumed to be simply followable, an edge following algorithm can be used to create two parameterization functions (or edge vectors) $x(s)$ and $y(s)$ that describe the curve exactly as a function of arc length. This representation is straightforward except for the diagonal vs. non-diagonal point spacing issue. In order to use discrete orthogonal polynomials for the least-squares derivative estimation as in

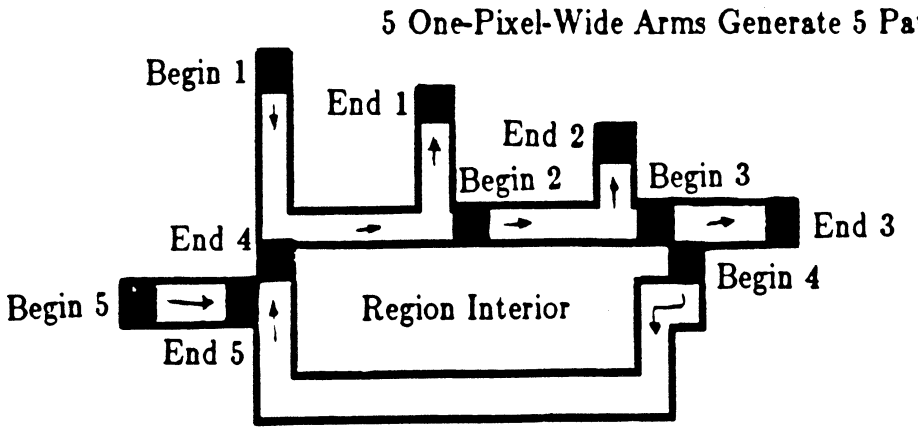


Figure 6.3. Regions with One-Pixel-Wide Arms Cause Multiple Paths

Chapter 3, equally spaced points are needed. However, the Northwest, Northeast, Southeast, and Southwest neighbors of a pixel are $\sqrt{2}$ pixel units from the central pixel of 3x3 neighborhood whereas North, South, East, and West neighbors of a pixel are only one pixel unit away. Hence, eight-connected edge pixels are not equally spaced as an edge is followed. Also, since edges are computed as boundaries of four-connected regions in the surface-based segmentation approach, it is desirable that a North neighbor move followed by a West neighbor move is equivalent to a Northwest neighbor move. For these reasons, the diagonal pixel spacing is labeled as two (2.0). For example, a Northwest neighbor is two pixel units away from a given center pixel instead of $\sqrt{2}$. This assignment of two distance units to each diagonal pixel spacing allows for equally spaced data points, which considerably simplifies edge vector processing. A more accurate technique that maintains equal spacing is to replace horizontal and vertical moves by 5 distance units and diagonal moves by 7 distance units [Shahraray and Anderson 1985].

An edge following algorithm can be stated as follows for a simply followable edge: Given a starting point, make it the current pixel and look for a 4-connected neighbor first. If a 4-connected neighbor is found, add the new pixel to the edge vectors describ-

ing the edge. If no 4-connected neighbors are found, look for an 8-connected neighbor that is not 4-connected. If an 8-connected neighbor is found, add two new (x,y) locations to the vectors describing the edge. The first of these is half-way between the current pixel and the eight-connected neighbor; the second is the eight connected neighbor. Hence, the length of an equivalent chain code (the number of edge pixels) and the arc length of the $(x(s), y(s))$ edge vector description are not the same in general. These few extra steps in the edge definition phase save significant processing time in the edge characterization phase. Figure 6.4 shows a set of simply followable edge pixels and the x and y edge vectors plotted as a function of arc length for the outside boundary of the block.

6.4. Edge Characterization

Once the edge description vectors have been formed, the data in these vectors is smoothed independently to reduce random noise. Then x and y derivatives are computed independently using discrete orthogonal polynomials as described in Chapter 3.

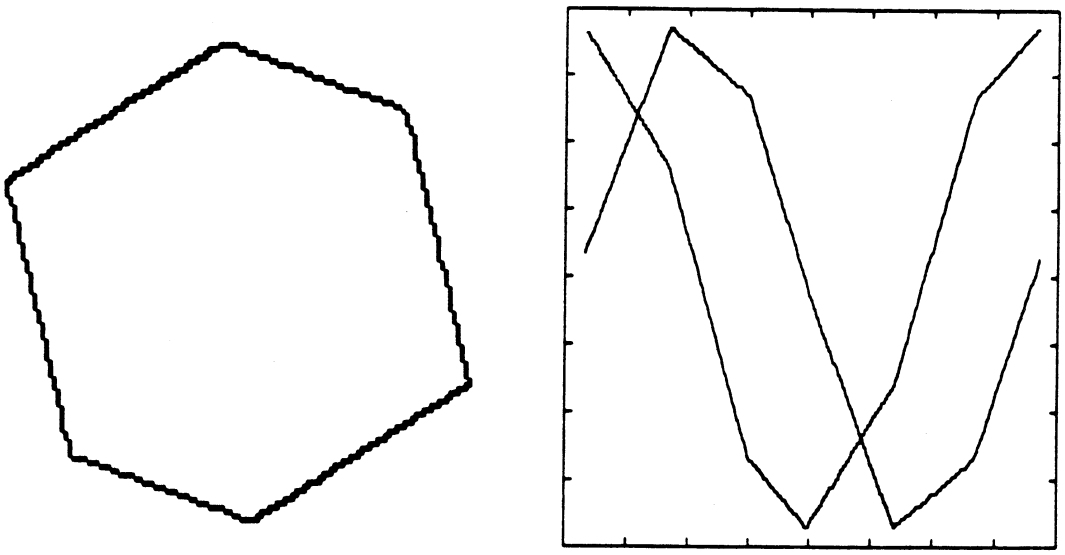


Figure 6.4. Original Edge Pixels and x and y Functions

The tangent angle function vector is then formed as follows:

$$\phi(s) = \tan^{-1}\left(\frac{dy/ds}{dx/ds}\right). \tag{6.3}$$

Special processing is required to handle phase wrapping, but the final approximation to the tangent angle function represents a smooth differentiable function due to the initial smoothing of the data. The curvature function for plane curves is then computed directly as the derivative of the tangent angle function:

$$\kappa(s) = \frac{d\phi(s)}{ds} \tag{6.4}$$

Curvature can also be computed directly from an equivalent formulation in terms of the first and second derivatives of x and y with respect to s [Faux and Pratt 1979]. The sign-of-curvature function $sgn(\kappa(s))$ is then computed using a preset zero threshold ϵ_0:

$$\begin{aligned} sgn(\kappa(s)) &= 0 && \text{if } |\kappa(s)| \leq \epsilon_0 \\ &= \frac{\kappa(s)}{|\kappa(s)|} && \text{if } |\kappa(s)| > \epsilon_0 \end{aligned} \tag{6.5}$$

The sign-of-curvature function of this edge description algorithm is exactly analogous to the HK-sign map of the surface description algorithm. This function is then given to the iterative edge fitting algorithm along with the original unsmoothed edge vectors $(x(s), y(s))$.

Figure 6.5 shows the x and y derivatives of the block boundary overlaid on top of the edge parameterization vectors as well as the curvature function of the block boundary with overlays. The curvature function has the zero level and the ϵ_0 level marked at the bottom of the plot. The sign-of-curvature function is also overlaid on top of the curvature plot. Each peak in the curvature function yields a square pulse in the sign-of-curvature function.

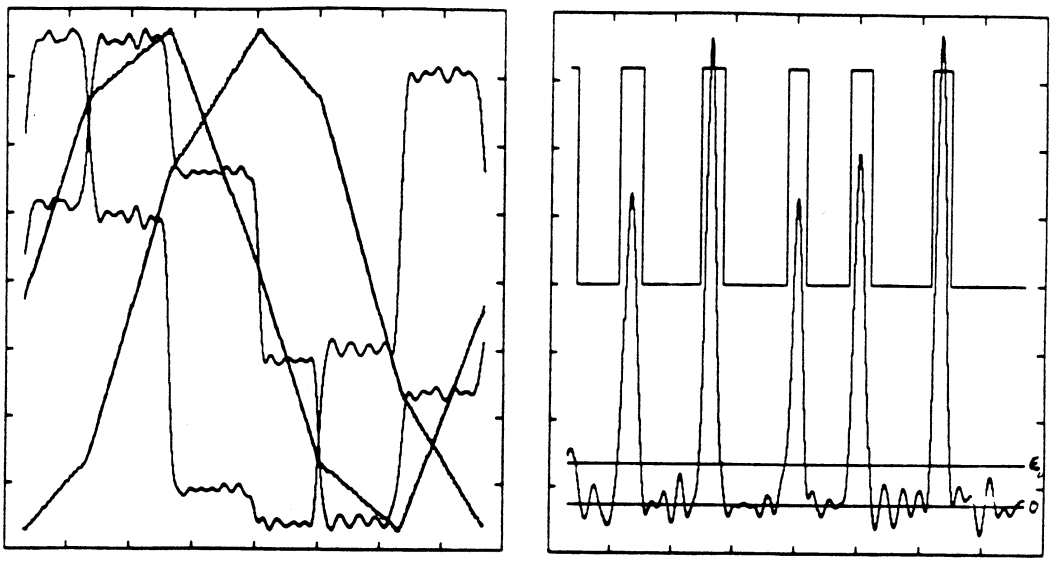


Figure 6.5. x and y Derivatives and Curvature Function of Block

6.5. Approximating Function Selection

Just as approximating functions were needed for surface fitting, a finite set of approximating functions are needed for edge fitting. However, instead of eight HK-sign surface primitives, there are only three sign-of-curvature edge primitives: a convex curve (positive curvature), a concave curve (negative curvature), and a straight line (zero curvature). Again, the shapes of the sign-of-curvature edge primitives are relatively simple and exhibit even-order behavior. Again, arbitrarily complicated piecewise smooth curves can be decomposed into a disjoint union of such surface primitives. If these edge primitives can be approximated well, then the entire edge can be approximated well.

The requirements of approximating functions for surface primitive were summarized earlier, but are resummarized here for edges. A small, ordered, finite set of approximating functions is sought that

(1) Can approximate any smooth curve of constant sign-of-curvature well over an interval of finite length,

(2) Can extrapolate accurately to arbitrary points outside the interval used for fitting (this permits interval growing),

(3) Allows a quick and computationally efficient surface fitting algorithm to be used,

(4) Can be represented by a small amount of data,

(5) Can be differentiated, and

(6) Can interpolate to locations within the region for fitting.

First-order (linear), second-order (quadratic), and fourth-order (quartic) univariate polynomials have been selected as the set of approximating functions for sign-of-curvature edge primitives. Therefore, the number of approximating functions $|F| = 3$ and the set of approximating functions $\mathbf{F}$ for edge primitives can be written in the form of two scalar equations:

$$\hat{x}(s) = a_0 + a_1 s + a_2 s^2 + a_3 s^3 + a_4 s^4 . \qquad (6.6)$$
$$\hat{y}(s) = b_0 + b_1 s + b_2 s^2 + b_3 s^3 + b_4 s^4 .$$

These two scalar polynomials may be considered as a single polynomial with vector coefficients. In the terminology set forth above, an edge is a 1-D 2-vector image. The goal of the edge description algorithm is to segment that 1-D edge image into meaningful intervals. Linear edges are obtained by restricting the parameter vector space $\mathbf{P} = \mathbf{R}^5$ to a two-dimensional subspace; quadratic surfaces are restricted to a three-dimensional subspace. A least-squares edge fitting algorithm can compute the parameter vectors $\mathbf{a}$ and $\mathbf{b}$ and the RMS fit error $\epsilon = \max(\epsilon_x, \epsilon_y)$ from the edge vectors in an interval very quickly. The key difference in the current implementation between edge fitting and surface fitting is that edge fitting amounts to performing convolutions since discrete orthogonal polynomials may be used for any interval of equally spaced data; since surfaces do

not always form rectangular or regular regions, the general least squares problem must be solved as in Appendix C.

The problems with these polynomial approximants are the same as in the case of surfaces. But as stated earlier, the main concern is that the functional fit is good enough for perceptual organization of edge pixels. If an application is interested in circles or ellipses, the data-driven edge description output of this algorithm can be used to isolate curved intervals of approximately constant curvature to guide the testing of hypotheses about circles and ellipses. All of the other arguments of the previous section against quadric surfaces apply here against conic curves.

6.6. Error Tolerance Specification

Just as the surface description algorithm required an error threshold for the RMS error that one was willing to tolerate for surface fits, the edge description algorithm requires an error threshold for edge fitting. This threshold could also be tied back to an image quality measure mentioned previously since noisier images are going to have noisier edges. All of the edge results in Chapter 7 have been obtained with a fixed RMS error threshold of 0.96.

6.7. Seed Interval Extraction

Given the original edge vectors and the sign-of-curvature vector, the algorithm begins by looking for the largest interval of any type { -1, 0, +1 } in the sign-of-curvature vector. Rather than contracting this interval with an erosion (contraction) algorithm as is necessary in the surface region case, this interval is simply shrunk by a factor of two about the center point of the interval if there are more than eight pixels in the interval; otherwise, the interval is not shrunk. This new interval is the seed interval.

Hence, the difficulties and computational expense in finding and maintaining connected component descriptions of regions are not present in applying the sign-of-curvature paradigm to edges. This simplification due to "regions" of reduced dimensionality makes the edge technique much faster on sequential machines than it would be otherwise.

6.8. Iterative Variable Order Edge Interval Fitting

Each isolated seed interval is given as the input to the iterative edge fitting algorithm. The basic concepts of this algorithm are the following. First, it is decided how well smooth curves should fit the edge data. This is given by the error threshold ϵ_{max}. A line is always fitted first to the seed interval using the standard equal weighting least squares approach based on discrete orthogonal polynomials (see Section 3.4). If the seed interval belongs to an edge that is not too highly curved and not too noisy, a line will fit quite well to the original edge data. If one considers a small enough interval on the real line, any smooth real-valued function of a single variable can be approximated very well by a straight line. If the line fits the seed interval to within the maximum error threshold for the RMS error, then the seed is allowed to grow. If not, the seed is fitted with the next higher-order curve (the quadratic in this implementation) and the algorithm proceeds similarly. When the seed is allowed to grow, the functional description of the curve over the seed interval is tested against all edge pixels in the edge vectors for the interval growing operation.

As discussed above a specific finite set of approximating surface function types is chosen for use in edge fitting. These edge functions must be able to provide for extrapolation into areas outside the local edge fitting interval, and they must provide good approximation capabilities within the fitting interval. Let $\mathbf{a}_k$ be the parameter vector associated with the functional fit to the edge x coordinates in a given interval R_k. Let

$\mathbf{P}$ be the set of all parameter vectors for all forms of functions of interest. For the selection of the three univariate polynomials, $\mathbf{P} = \mathbf{R}^5$. Let $| F |$ be the number of different types of edge functions to be used (in this case, $| F | = 3$). A particular function type (or fit order) is referred to as m_F where $m_F \in \left\{ 1,2, \cdots, | F | \right\}$. The actual fitting function of form m_F is denoted $\hat{x}(s) = \hat{g}(m_F, \mathbf{a}, s)$ and $\hat{y}(s) = \hat{g}(m_F, \mathbf{b}, s)$. The edge fitting process, denoted $L_{\hat{g}}$, maps the original edge data $\{x(s), y(s)\}$, an interval definition R_k, and the fit order m_F into the range space $\mathbf{P} \times \mathbf{P} \times \mathbf{R}^+$ where $\mathbf{R}^+$ is the set of possible errors (the set of non-negative real numbers):

$$(\mathbf{a}_k, \mathbf{b}_k, \epsilon_k) = \mathbf{L}_{\hat{g}}(R_k, m_F, x(s), y(s)) . \tag{6.8}$$

The fit error term is given by $\epsilon_k = \max(\epsilon_k^x, \epsilon_k^y)$ where the error metrics

$$\begin{aligned} \epsilon_k^x &= | \hat{g}(m_F, \mathbf{a}_k, s) - x(s) |_{R_k} \\ \epsilon_k^y &= | \hat{g}(m_F, \mathbf{b}_k, s) - y(s) |_{R_k} \end{aligned} \tag{6.9}$$

are the minimum values attainable for all functions of the form specified by m_F. Equally-weighted least-squares surface fitting minimizes the two error metrics

$$\begin{aligned} (\epsilon_k^x)^2 &= \frac{1}{| R_k |} \sum_{s \in R_k} (\hat{g}(m_F, \mathbf{a}_k, s) - x(s)))^2 \\ (\epsilon_k^y)^2 &= \frac{1}{| R_k |} \sum_{s \in R_k} (\hat{g}(m_F, \mathbf{b}_k, s) - y(s)))^2 \end{aligned} \tag{6.10}$$

where $| R_k |$ is the length of the interval R_k. Least-squares polynomial curve fitting of equally-spaced data points is extremely fast and direct and the "shape potential" of polynomial surfaces is adequate for sign-of-curvature edge primitives. The polynomial curves of orders 1, 2, 4 provide excellent approximations to the basic edge types in small and even very large intervals.

Given a seed interval R_k, the original edge vectors $\{x(s), y(s)\}$, and the simplest interesting function type m_F such that $\epsilon_k < \epsilon_{max}$, the parameter vectors $\mathbf{a}_k$ and $\mathbf{b}_k$ and the error (RMS) $\epsilon_k = \max(\epsilon_k^x, \epsilon_k^y)$ are computed. When the error is less than the predetermined maximum allowable error threshold ϵ_{max}, then the seed region is allowed to grow. If all three fit orders were tried and the error was never less than the threshold, the best highest order fit is accepted as a good approximation to the pixels on that interval. If the highest order fit is not sufficiently good, intervals could be rejected by the edge description algorithm to maintain theoretical parallels with the surface based approach. Interval rejections have never occurred with the test image database, and it is proposed that it is not worthwhile to include such an acceptance/rejection test for edges. This appears to be due to the fact that $x(s)$ and $y(s)$ cannot change by more than one level for any given unit change in arc length. Digital surfaces rarely exhibit such controlled variation. Hence, the edge functions are always very well behaved because the maximum possible slope is one.

6.9. Edge Interval Growing

After a polynomial curve is fitted to an interval, the original edge description is used to grow the interval into a larger interval where all pixels in the larger interval are connected to the original interval and compatible with the approximating edge function for the original edge. The same approach from the surface description algorithm is used for the edge description algorithm.

For each edge pixel $s \in E$, the entire set of edge pixels, the two values

$$\hat{x}(s) = \hat{g}(m_F, \mathbf{a}_k, s) \qquad (6.11)$$
$$\hat{y}(s) = \hat{g}(m_F, \mathbf{b}_k, s)$$

are computed and compared to $x(s)$ and $y(s)$ respectively to see if the edge pixel s is compatible with the approximating edge function. It is assumed that almost all pixels in the original interval are compatible with the approximating edge function because the pixels in the region were used to obtain the edge interval fit. If the maximum of the magnitude of the difference between the computed edge pixel locations and the original edge pixel location is less than the allowed tolerance $w(\epsilon_k)$, then the pixel s is added to the set of pixels $C(R_k, \hat{g}, \epsilon_k)$ compatible with the region R_k; otherwise, it is incompatible and discarded. The result of this process is explicitly stated as

$$C(R_k, \hat{g}, \epsilon_k) = \left\{ s \in E : \max(\mid \hat{x} - x \mid, \mid \hat{y} - y \mid) \leq w(\epsilon_k) \right\}. \qquad (6.12)$$

Thus, the set of compatible edge pixels $C(\cdot)$ is directly dependent on (1) the seed interval R_k and therefore also on the seed interval extraction algorithm, (2) the function $\hat{g}$ and therefore on the function type m_F, (3) the two parameter vectors $\mathbf{a}_k$ and $\mathbf{b}_k$ and the error $\epsilon_k = \max(\epsilon_k{}^x, \epsilon_k{}^y)$, which are in turn dependent on the edge fitting algorithm and again on the original edge data $\{x(s), y(s)\}$ and the seed interval, and (4) the error tolerance increase function $w(\epsilon) > \epsilon$.

The compatible set of edge pixels also depends indirectly on the all the edge intervals previously discovered in the edge vectors. For precisely defined edge interval boundaries, the influence of other already determined edge intervals is negligible or nonexistent, but for noisy edges, the influence may be more prominent. When an edge interval iteration terminates and the edge is accepted, the error at each pixel of the grown edge interval is stored in an edge error-buffer (E-buffer) to explicitly note the spatial distribution of the approximation errors, and the edge interval label for each pixel of the accepted edge interval is stored in an interval label buffer (IL-buffer) to explicitly

note the pixels that the approximating edge fits to within the specified threshold. During the interval growing process that forms the compatible edge pixel list, each pixel is also checked to insure that the pixel error is less than the error-buffer error (the best fit error so far). If this condition is not satisfied, then the pixel is not considered to be compatible with the growing edge and is not marked as such.

When the interval growing computation is complete, the pixel list $C(R_k, \hat{g}, \epsilon_k) \subseteq E$ is complete. The largest interval of this list that overlaps the seed interval R_k must then be extracted to create the next interval R_{k+1}. This iterative process of interval definition via largest overlapping interval extraction is expressed using the function $\Lambda(\cdot)$:

$$R_{k+1} = \Lambda\Big(C(R_k, \hat{g}, \epsilon_k), R_k \Big) = \Phi(R_k) \tag{6.13}$$

where $\Phi(\cdot)$ represents all operations required to compute the interval R_{k+1} from the interval R_k. The output interval R_{k+1} must have the property that it is the largest interval in the list of compatible pixels satisfying

$$R_k \cap R_{k+1} \neq \phi = \textit{Null Set} \ . \tag{6.14}$$

This constraint is required because it is possible to get larger connected intervals in the compatible pixel list than the connected interval corresponding to the seed interval. Experience indicates that this does not happen as often in edge processing as in surface processing.

This next interval is then considered as a seed interval and processed by the edge fitting algorithm

$$(\mathbf{a}_{k+1}, \mathbf{b}_{k+1}, \epsilon_{k+1}) = L_{\hat{g}}(R_{k+1}, m_F, x(s), y(s)) \tag{6.15}$$

to obtain a new parameter vector and a new edge fit error. If this interval is allowed to

grow again ($\epsilon_{k+1} < \epsilon_{max}$), then the compatible edge pixel list is recomputed

$$C(R_{k+1}, \hat{g}, \epsilon_{k+1}) , \tag{6.16}$$

the largest interval of $C(\cdot)$ is extracted and so on until the termination criteria are met.

A runs test is performed at each iteration for linear and quadratic fits to see if a higher order function is necessary. This involves computing two residual error sign functions for both the x and y function fits. The number of runs in each residual error sign function is computed, and the minimum number of runs (the smaller of the two) is used as the test statistic. Given the number of edge pixels in the fit and the test statistic, a one-sided runs test is performed to see if there are too few runs. If there are too few runs, a higher order function is needed and value of m_F is incremented for the next iteration.

If the edge fitting process has yielded a fit error that is not acceptable (i.e., not below the maximum error threshold $\epsilon_k \geq \epsilon_{max}$), then the algorithm increments the edge type m_F to the next most general, flexible edge in the finite approximating function set and reattempts a edge fit of the higher order edge type if there are enough pixels in the connected interval being fitted. If m_F reaches the maximum value $|F|$ and the edge still does not fit the data to within the maximum error threshold, or if there are not enough pixels in the interval to attempt a higher order fit, then the iterative edge fitting algorithm accepts the edge even though the ideal termination criteria have not been met. As mentioned above, edge intervals are never rejected because the edge functions are known to be very well behaved functions owing to the edge vector construction algorithm.

6.10. Termination Criteria and Interval Acceptance

The termination criteria are very simple for the edge description algorithm.

(1) **RULE 1:** IF $\epsilon_k > \epsilon_{max}$ AND $m_F \geq |F|$, THEN Stop!

This says that if the edge fit error increases above the preset threshold, and if no more fit orders remain in the finite set, then terminate.

(2) **RULE 2:** IF both of the end pixels of an interval did not move since the last iteration, THEN Stop!

This says, of course, that the iteration should stop when interval growing stops.

Quick, *ideal* convergence has been experienced with the edge description algorithm for every edge interval ever tested (thousands of edge intervals). In addition to the excellent behavior of the edge functions (maximum slope of unity), note that interval boundaries only have two degrees of freedom, the start and stop end points. In contrast, discrete image regions have a very large finite number of degrees of freedom.

All grown edge intervals are currently accepted. Intervals in the compatible pixel list are accepted if the intervals exceed a certain length and if the RMS fit errors are small enough. A threshold of three or more edge pixels is recommended.

Whenever any interval is accepted, it is marked off in a writable copy of the sign-of-curvature function so that none of the interval pixels are capable of being involved in another seed region.

An error buffer and the interval label buffer are updated when an interval is accepted. These two buffers are used to monitor and control growth during the interval growing process in the same way as in the surface algorithm. Since gaps between intervals are usually very small and easily handled by straight line connections, it is not necessary to maintain two separate interval label buffers as in the surface algorithm.

6.11. Interval Merging and Interval Connections

Arbitrary piecewise-smooth planar curves (region boundaries, edges) consist of smoothly curved edge-interval primitives which are bounded by order-0 and order-1 discontinuities. Smooth edge-interval primitives consist of sign-of-curvature edge-interval primitives, which are approximated by (up to) fourth-order vector polynomials. A set of merging rules similar to those in the surface algorithm are defined to control the merging of adjacent smoothly joined sign-of-curvature edge-interval primitives. The rules are based on the average differences in functional descriptions and average differences in edge tangents (or normals) within a small interval of overlap where two edge intervals join. The boundaries of the smooth edge-interval primitives are simply the breakpoints between the intervals that do not join smoothly.

When displaying the results of the edge description algorithm, intervals must be connected with each other to obtain visually appealing edges. The end pixels of adjacent intervals are connected with straight lines in the simplest scheme. This is adequate for current purposes although the final results do appear somewhat jagged at times.

6.12. Chapter Summary and Algorithm Simplifications

The iterative interval-growing algorithm based on variable-order approximating edge functions accepts the sign-of-curvature function and the original edge vectors as input and produces parametric descriptions of basic edge intervals. A list of edge-interval primitive polynomial coefficients, fit errors, and smooth edge-interval end points are output. This type of edge description method is also data-driven, handles arbitrary edge shapes, and can be used with any edge detector/linker that creates the appropriate type of digital edges: one-pixel-wide, 8-connected, simply followable edges. It is interesting, not only for its own merits as an edge description mechanism, but also because the

same sign-of-curvature paradigm provides useful algorithms in one and two dimensions. This sign-of-curvature paradigm can also be extended to higher dimensions in which $(n-1)$-D geometric entities are embedded in n-D space. For 3-D piecewise-smooth images, such as density images, there are 27 different sign-of-curvature volume-primitive types, and the fourth-order trivariate polynomial requires 35 coefficients. The first and second fundamental form matrices become 3x3 matrices. It will be interesting to apply the sign-of-curvature paradigm to segment dynamic scenes, but such a digression would not be appropriate here. For digital surfaces, a unified approach to edge and surface description in images is important theoretically and practically in that improvements in one algorithm are immediately available to the other via the appropriate modifications.

Simpler variants of this edge description algorithm exist that are also useful. Instead of parallel interval growing, seed intervals can very easily be grown outward from each interval end point pixel. Growing arbitrary edge intervals outward is much simpler than growing arbitrary regions outward using the spiraling region growing method. Sequential outward growth is equivalent to the parallel growth approach except at the last iteration. The only difference is that non-connected compatible edge pixels are not detected directly by this method and must be grown from separate seed intervals. Also, it is not absolutely necessary to maintain an error buffer or an interval label buffer. The only disadvantage here is that several pixels at interval joins may not be associated with the best-fit interval. As with the surface algorithm, the edge algorithm will also function without the runs test. The net effect of all these simplifications is some minor jaggedness between edge intervals.

CHAPTER 7

EXPERIMENTAL RESULTS

This chapter is dedicated to the discussion of the surface-based segmentation algorithm's performance on a set of twenty-two test images containing real and synthetic range images and intensity images. These images were selected from a group of over thirty images used as a test image database for the experiments. All input images are 128 x 128 pixels with eight bits per pixel, except for one 256 x 256 (8-bit) intensity image of a house scene. All real range images were acquired by the Environmental Research Institute of Michigan's (ERIM) phase-difference laser range finder except for the Renault Auto Part Image, which was created by converting a range data xyz-file from INRIA (courtesy of Prof. T. Henderson) into a range image using a special purpose program written by the author. All results have been obtained using software written by the author in the C programming language on the UNIX 4.2 BSD operating system running on a VAX/11-780.

7.1. Output Format

For each input image, the following set of images are displayed in the following order:

(1) Original Image (Gray Scale)

(2) Best-Fit Reconstructed Image (Gray Scale)

(3) HK-Sign Map (Gray Scale)

(4) Final Region Label Buffer (Gray Scale)

(5) Final Segmentation Plot Overlaid on Original Image (Gray Scale)

(6) HK-Sign Map Segmentation Plot (Binary)

(7) Best-Fit Segmentation Plot (Binary)

(8) Final Labeled Segmentation Plot (Binary)

(9) Edge Description Plot (Binary)

The original image is the only variable input to the surface-based segmentation algorithm. All input thresholds and window sizes were fixed for the main body of results presented in this chapter. Such results are referred to as ***standard results***. Occasionally, noticeably better results were obtainable using different thresholds. These exceptional cases are mentioned explicitly and are referred to as ***modified threshold*** results.

The eight-level HK-sign Map results demonstrate the intermediate surface characterization output and the input to the iterative region-growing algorithm. This format is different than in Chapter 3 in which separate $sgnH$ and $sgnK$ images were displayed. The best-fit reconstructed image shows the visual quality of the approximate surface representation. The best-fit segmentation plot shows the underlying segmentation in the best-fit region label buffer of the best-fit reconstructed image. The final segmentation plots show the quality of the final segmentation output alone and overlaid on the original image. The edge description plots show the edge interpretations obtained by applying the sign-of-curvature method to edges as described in Chapter 6. This set of images graphically describes the information available from the algorithm.

The experimental results shown below were obtained using a *fixed set of input thresholds* (except for those results noted explicitly). Several of the fixed numerical inputs required by the algorithm are noted below.

(1) Pre-Smoothing Operator Window Size for Surface Characterization: (11 x 11)

(2) Derivative Operator Window Size for Surface Characterization: (9 x 9)

(3) Curvature Smoothing Operator Window Size: (7 x 7)

(4) Zero Curvature Thresholds: 0.015 (Mean) and 0.006 (Gaussian).

(5) Regions Test Threshold: 2.0%

(6) Error Threshold ϵ_{max} : 4.2 levels on 8-bit scale

(7) Error Increase Factor: 2.8

(8) Acceptability Zone: 50%

This set of inputs summarizes several potentially variable algorithm inputs. Any other inputs, thresholds, or constants required by the algorithm that are not mentioned were assigned to their default values mentioned previously in the text.

7.2. Real Range Image Results

The figures referred to in the text below are all grouped at the end of this chapter. For almost all images, two figures are presented. The first figure contains the original and best-fit reconstruction images, the HK-sign (surface curvature sign) map, and the final region label buffer in one photograph, and the final segmentation plot overlaid on the original image in the second photograph. The second figure consists of a set of four plots: the initial segmentation in the HK-sign map, the best-fit region label buffer segmentation, the final region segmentation, and the boundary edge descriptions.

7.2.1. Coffee Cup Range Images (ERIM)

The two coffee cup range images from the ERIM range sensor are very high quality images. The original range images, their best-fit reconstruction images, and the final segmentation overlay are shown in Figures 7.1.1 and 7.2.1. Note the excellent likeness of the reconstructed images showing that the surface primitives interpreted by the algorithm do accurately represent the image data. In Figures 7.1.2 and 7.2.2, four separate images are represented. Figures 7.1.2(a) and 7.2.2(a) show the initial segmentations obtained by computing the signs of the mean and Gaussian curvature at every pixel and drawing boundaries around connected regions of the same surface curvature sign. Figures 7.1.2(b) and 7.2.2(b) represent the segmentations in the best-fit region label buffer (BF-RL-buffer). After the region combination and region merging processes have operated on the region growing algorithm output, the final labeled region segmentations are obtained, which are shown in Figures 7.1.2(c) and 7.2.2(c).

Figures 7.2.2(b),(c),(d) show that the cup handle is divided into two separate surfaces, the top of the handle and the inside of the handle. The fixed set of thresholds used for all test images represents a compromise for some images. The perhaps more appealing result shown in the segmentation plot of Figure 7.2.1 is a modified threshold result; the handle is segmented as one region. Figures 7.1.2(d) and 7.2.2(d) display the functional edge descriptions of the edge intervals of the region boundaries in the final region segmentations. Note that in Figure 7.2.2(d) straight edges are identified as straight and curved edges are identified as curved. In Figure 7.1.2(d), similar edge descriptions are obtained.

Figure 7.1.3 displays plots of four different quantities as a function of the number of iterations in the region-growing process for the four surface primitives in the coffee

cup A image. Automatically scaled plots are shown for (1) the RMS surface fit error, (2) the number of compatible pixels, (3) the size of the growing region, and (4) the order of the fitting surface. Note how the RMS fit error drops each time the fit order is incremented.

Figure 7.1.4 shows the exact segmentation information for the coffee cup A image. The data includes the surface type of the seed region, the type of the final fitted polynomial, the coefficients of the bivariate polynomial, the three different fit errors, and a run-length encoded version of the region. The edge parameterizations of the region are not shown here.

The final segmentations in both images clearly delineate the outside cylindrical surface of the cup, the background table surface (recognized in two parts despite the topological separation of the small patch visible through the handle in Figure 7.2.2(c)), the inside cylindrical surface of the cup, and the cup handle surface, which is represented either as one symbolic primitive or two depending upon the input thresholds (compare segmentation plot of Figure 7.2.1 and Figure 7.2.2(c)). Although the hole in the body of the cup in the first image is not big enough to merit its own symbolic surface region description, it is clearly visible in the error image. Figure 7.2.3 shows a thresholded error image overlaid on the original range image. This output image type is included only for this view of the coffee cup to show that such small features can be isolated even though they do not form a coherent surface region in the image. Note how clearly the hole in the coffee cup body stands out. Figure 7.2.4 displays a reconstruction image obtained by applying the surface equations to the final region segmentation. Hence, this is slightly different than the best-fit reconstruction and the FW reconstruction mentioned in the text.

Remember that this algorithm knows nothing about *cylinders*, much less coffee cups, yet all surfaces are meaningful to a person who understands the semantic content of the image. Except for the small hole in the side of the cup, this range image is the epitome of a coherent digital surface. As expected, excellent results are obtained.

Note the straightness of the edges bounding the outside cylindrical surface region in the final segmentation output in Figure 7.2.2(d) as compared to the straightness in the original image. The edges in the original image are not very straight owing to noise effects from the range finder at steeply sloped surfaces. The net straightness of the edge description is due to the sign-of-curvature method combined with region refinement techniques that use 3x3 window mappings as discussed in Chapters 4 and 5.

The quality measure ρ is 0.84 for the image where the handle is clearly visible and 0.70 for the other image where it is not. These values are higher than those for synthetic images without noise, and lower than those for the synthetic images with $\sigma = 2.3$ additive white pseudo-Gaussian noise.

Figure 7.2.5 displays clearly the notion of surface coherence. The original image and the reconstruction image are shown in gray scale in Figure 7.2.1 and appear to be very similar. If a random gray scale lookup table is used to transform the gray levels in these two images, the two new images in Figure 7.2.5 are obtained. These new images may be thought of as contour plot images in which each depth level is assigned a special gray level. The noisy top image is the original image, and the clean bottom image is the noise-free reconstruction image. The original data does exhibit noticeable surface coherence in spite of the measurement noise. This surface coherence in the data allows the surface-based segmentation algorithm to produce a perfectly coherent reconstruction image.

7.2.2. Computer Keyboard Range Image (ERIM)

A similar set of experimental results are presented for two range images of a computer keyboard in Figures 7.3.1, 7.3.2, 7.4.1, and 7.4.2. The gray scale images (original and reconstructed) in Figures 7.3.1 and 7.4.1 may appear washed out due to the low contrast of this range image. Note the intermixing of the keyboard surfaces in the best-fit region label buffer as shown in Figures 7.3.2(b) and 7.4.2(b). This intermingling (or overlapping) of the surface primitive descriptions is strong evidence for a smooth surface merge as discussed in Chapter 5. Figures 7.3.2(c) and 7.4.2(c) show that the surfaces are correctly merged in the final region segmentation. Despite the low contrast of the original image, excellent results have been obtained using the fixed set of thresholds.

Two different versions of the same image are included to show the small variations in the output that are possible with a slight variation in the input. The quality measure for the 7.3.1 keyboard image is 1.17 as compared to 1.35 for the 7.4.1 image. Slightly different parts of the keyboard are visible in the two images. Despite minor differences in the output for the two images, the overall consistency is quite good. The actual key surfaces are combined together into single regions due to the large fit error tolerance (4.2). Lower thresholds can be used to detect individual key surfaces, but the segmentation becomes more ragged.

7.2.3. Polyhedron (ERIM)

A range image of a polyhedron resting on a table top is shown in Figure 7.5.1 along with the best-fit reconstructed image. Note the refinement of the edges between the best-fit RL-buffer segmentation and the final region segmentation in Figure 7.5.2. All facets of the polyhedron are correctly segmented. Many approaches to range image analysis in the literature are only able to handle simple scenes like this.

7.2.4. Ring on Polyhedron (ERIM)

A ring was placed on top of the polyhedron for the next ERIM range image. Note in Figure 7.6.1 how the steeply slanted surfaces in the image are quite noisy compared to the other images shown here. Despite these noisy areas, the overall image segmentation is quite good. A slight region merge error has occurred here: Region No. 6 on the polyhedron in Figure 7.6.2(c) has acquired a small part of Region No. 3. The algorithm is capable of distinguishing between these two regions when both were completely visible as was shown in Figure 7.5.2(c). However, the occlusion by the ring created a boundary that caused Region No. 6 to grow into Region No. 3 during the region growing phase. Also, the small portion of the background visible through the ring did not receive the same label as the background surface for this set of thresholds. Modified thresholds results have shown that the region can be recognized as part of the background.

Figure 7.6.3 shows a pair of surface coherence images similar to those shown in Figure 7.2.5. The original image and the reconstruction image are mapped through random gray scale lookup tables to obtain the top image and the bottom image respectively. It is very difficult to see any structure in the noisy top image, but the surface-based segmentation algorithm has generated a perfectly coherent reconstruction image.

7.2.5. Curved Surface A (ERIM)

For lack of a better name, the surface in the range image in Figure 7.7.1 is called Curved Surface A. Owing to the regular shape of this surface, the polynomial surface primitives were able to describe the data quite well as indicated in the similarity between Figure 7.7.2(b) and Figure 7.7.2(c).

7.2.6. Curved Surface B (ERIM)

Curved Surface B is significantly different than Curved Surface A. Note that during the processing, the best-fit region label buffer became quite fragmented as shown in Figure 7.8.2(c). This curved surface is not well approximated by the bivariate polynomials and contains a cusp. Nevertheless, a variety of a surfaces grew and mixed with each other. The final segmentation in Figures 7.8.2(c) shows that the surface primitives join together correctly during the region merging stage.

7.2.7. Road Scenes (ERIM)

Two range images were selected at random from a sequence of range images taken from the Autonomous Land Vehicle with the ERIM range sensor. The edges of the road are clearly visible in Figures 7.9.2(b) and 7.10.2(b), which show the best-fit region label buffer segmentation. The disturbing line that appears across the middle of the road is due to the limited bending capability of the fourth order surface and the extreme warping of the flat road surface beyond that point due to equal-angle increment sampling (see Appendix D) and the viewing angle of the surface. The surfaces of the road and the road shoulders are completely merged together during the region merging process to create a final region segmentation that is not very useful. In this case, the segmentation algorithm could be stopped prior to region combination and region merging by using application-specific assumptions about the geometry of the road. This has been done here by displaying the intermediate best-fit segmentation, not the final segmentation, in the overlaid plot images in Figures 7.9.1 and 7.10.1.

7.2.8. Renault Auto Part (INRIA)

The original data for the auto part is formatted as a list of (x,y,z) triplets. Although the data is easily divided into scan lines, a different number of pixels occurred on each scan line, and the pixels were not always regularly spaced. This data could be converted naturally to a 64x64 range image, which was then expanded to a 128x128 range image, and smoothed to create the input image used here. Although the best-fit reconstruction image in Figure 7.11.1 and the best-fit region label segmentation images are quite good, the final region segmentation is not meaningful due to the high maximum fit-error threshold of 4.2. Note that the image quality measure is 0.43, which means that the image data is not very noisy. By reducing the fit-error threshold to 1.7, the modified threshold results shown in the overlaid plot image of Figure 7.11.1 were obtained.

7.3. Synthetic Range Image Results

Although synthetic images generally lack the realism of actual data and do not provide test results that represent algorithm performance on real data, it has been found that synthetic range images with added noise are quite realistic and do adequately test almost all algorithm capabilities. The range image synthesis problem is much simpler than the intensity image synthesis problem because it depends only upon geometry. The SDRC/GEOMOD solid modeler was used to create several different (non-convex) object models on Apollo workstations. The object models were stored in ASCII files and ported over to the VAX/UNIX system where a depth buffer display program, written by the author, was used to convert the object models to range images. The use of synthetic range images was very useful for this research and will be even more critical to object recognition research since arbitrary views of 3-D objects can be obtained much more

easily using the depth buffer algorithm than could be obtained otherwise.

7.3.1. Block with Three Holes

The block with three holes drilled through it provides an interesting non-convex combination of flat and cylindrical surfaces. Figures 7.12.1 and 7.13.1 show the block with two different levels of added noise: 2.3 and 9.0 respectively. The corresponding image quality measures are 1.82 and 5.93. These images were also analyzed in Chapter 3. If the equation $\epsilon_{max} = 1.1 + 1.1\rho$ guides the selection of a maximum fit-error threshold, the ϵ_{max} thresholds of 3.1 and 7.6 would be used. It is usually not a serious problem to use a threshold larger than that predicted by the simple equation. The problem with the auto part segmentation is representative of what may happen. Because 3.1 is less than the standard fixed 4.2 threshold, the results shown in Figure 7.12.2 are expected to be fairly good. However, 7.6 is significantly greater than the 4.2 threshold and poor results are expected. When the maximum fit-error threshold is not large enough, higher order surfaces are often invoked when not needed. The shape potential of the higher order surfaces is able to respond to the noise in the data rather than to the structure in the data. Figure 7.13.2 shows the failure of the algorithm on the noisier block. However, by increasing the ϵ_{max} threshold to 8.0, the results shown in the overlaid segmentation plot image in Figure 7.13.1 are obtained. These results are quite good considering the noise level in the image. A key point here is that the image quality measure can be used to *predict the failure* of the algorithm for a fixed threshold and *suggest a more appropriate threshold.*

7.3.2. Two Joined Cylinders Embedded in Plane

Figure 7.14.1 shows two cylinders with different diameters embedded in a sloped plane. The sloped plane kinks and becomes flat in the upper left and lower right hand corners of the image as is indicated in Figure 7.14.2(b). The same noise level used with the less noisy block image is also used here ($\sigma = 2.3$). The image quality measure is 1.79, and good results are expected using the standard fixed set of thresholds. The results are good, but curious things have happened. Region 3 in Figure 7.14.2(c) really consists of two different surfaces: the round body of the smaller cylinder and the flat bottom of the larger cylinder. A second order surface was fitting the round body of the smaller cylinder, but included pixels in the bottom of the large cylinder. A fourth order surface was invoked, and it fluted outward to fit the bottom even though the object model had two perpendicular surfaces. The bottom of the small cylinder was such a small surface that it caused the fourth order surface to bend slightly to include it. Finally, the kinks in the sloped plane, which are visible in Figure 7.14.2(b), are not present in Figure 7.14.2(c) due to the fixed dihedral angle threshold of 25 degrees for arbitrary surface joins. Note that the planar equations for all three planar background areas are available from the iterative region growing algorithm if special case processing for plane-plane intersections were desired. As a result, most of the important orientation discontinuities in the scene are detected, but some less important orientation discontinuities are missing due to region growing and region merging factors in the presence of noise.

7.3.3. Object with Protrusions over Sphere

Figure 7.15.1 shows a brick-shaped object with two protrusions suspended above a flat and hemispherical background surface. One protrusion consists of a base cone

joined with a cylinder whereas the other consists of a base cylinder joined with a sphere. The same amount of noise (that was used for the two cylinders) was added to this range image ($\sigma = 2.3$). Note that the hemispherical background surface is clearly segmented in Figure 7.15.2(b), but that it was merged with flat pieces of the background, as shown in Figure 7.15.2(c), owing to a small dihedral angle at the joining boundaries in the noisy image.

Due to the flexibility of the fourth order surface, the base cone surface of one protrusion grew together with the round body of the joined cylinder. Similarly, the cylindrical base of the other protrusion grew together with the joined sphere surface. Attempts were made to constrain the fourth order surface from bending in ways that included multiple fundamental surface types, but no methods were found that provided suitable performance over the whole test image database in various levels of noise. The perceptual organization of the pixels in the segmentation output is usually excellent, but small orientation discontinuities in noise were generally very difficult to isolate. Such orientation discontinuities were easily isolated in synthetic images without noise.

7.3.4. Light Bulb

The light bulb in Figure 7.16.1 is an interesting surface because it demonstrates a smooth join between a positive Gaussian curvature surface (the spherical part of the bulb) and a negative Gaussian curvature surface (the stem of the bulb connecting the base with the spherical part). Figure 7.16.2(b) shows that one (fourth-order) surface grew out to fit both positive and negative Gaussian curvature surface primitives. Another surface was required to approximate the threaded metal base of the light bulb. These two surfaces merge smoothly to yield a simple figure-ground segmentation. Even though it would have been trivial to obtain this segmentation by thresholding the

original image, it is important to note that the flexibility of the surface approximants is adequate to handle such non-convex curved surfaces.

7.3.5. Torus

A torus is a more complicated surface not easily approximated by planes, quadrics, or low-order bivariate polynomial graph surfaces. It also demonstrates a smooth join between a positive Gaussian curvature surface (the outside of the torus) and a negative Gaussian curvature surface (the inside of the torus). The torus in Figure 7.17.1 incorporates $\sigma = 3.4$ additive noise. The surface-based segmentation algorithm (with all the same fixed thresholds) grew several surfaces to approximate the complicated shape as shown in Figure 7.17.2(b). The region merging algorithm noted the smooth joins of those surfaces and coalesced them to create the correct final region description shown in Figure 7.17.2(c).

7.3.6. Circular Waves

A surface that is even more complicated and more difficult to approximate with planes, quadrics, or low-order polynomials is the circular waves surface shown in Figure 7.18.1. This surface is defined by the equation

$$z(x, y) = \sin^2(\omega\sqrt{x^2 + y^2}) \; . \tag{7.1}$$

No noise was added to this image. Quantization noise is the only noise present, but it is substantial because the quality measure is 1.54. This surface consists of smooth joins of many different fundamental HK-surface types, all arranged in a circularly symmetric manner around the center of the image. The best-fit region buffer segmentation in Figure 7.18.2(b) shows that a large number of approximating surfaces were needed to approximate the complicated surface. This segmentation exhibits many interesting sym-

metries on the noiseless data, but is not perfectly symmetric. The region merging algorithm correctly merged almost all of the surface primitives, but three surfaces were obtained instead of one as might be expected. A narrow annulus, located at the first trough of the wave moving out from the center, did not merge with the rest of the surface regions, but it is circularly symmetric. See Figures 7.18.2(c) and (d) and Figure 7.18.1. This may be due to the fact that the zero depth level is not allowed for internal programming reasons, and all zero levels in an image are mapped to the value of one before processing. This usually makes no difference at all, but it may have increased the orientation discontinuity of the surface join beyond the threshold for merging in this no-noise added image.

7.4. Intensity Image Results

The entire segmentation algorithm is based only on the knowledge of surfaces. Since intensity images are also digital surfaces (just as range images are digital surfaces), the algorithm can be applied to intensity images for symbolic-surface-primitive segmentation purposes. And just as the only difference between range images and intensity images is in the meaning of the values at each pixel (depth vs. light intensity), the difference in the algorithm output lies in how the surface segmentation is interpreted. That is, an intensity smooth surface primitive does not necessarily correspond to the 3-D surface primitives of objects in a scene although it may. Intensity image surface primitives have an entirely different meaning than range image primitives. However, it appears that the algorithm may be useful for intensity image segmentation purposes as long as this fundamental difference is kept in mind.

7.4.1. Space Shuttle Intensity Image

The original shuttle image, the reconstructed image, and the overlaid segmentation plot image are shown in Figure 7.19.1. The reconstructed image lacks detail whenever the detail in the original image consists of only a few pixels (10 or less). For example, the windows on the cockpit of the shuttle only occupy about three pixels and are therefore missed in the image reconstruction. The surface-curvature sign segmentation in Figure 7.19.2(a) appears to be completely meaningless when compared to the shuttle image. This is unlike most range images where some semblance of image structure is perceivable. However, it provided enough significant information to the region growing algorithm to create the best-fit region label buffer segmentation shown in Figure 7.19.2(b). The final segmentation is displayed in Figures 7.19.2(c) and 7.19.1. Considering that the algorithm was developed for range image analysis based on surface concepts, the segmentation for the shuttle image is extremely good. This is expected because the quality measure is only 2.29. The sky and many smoke clouds are isolated as intensity-image smooth-surface-symbolic primitives. It is emphasized that the exact same fixed set of thresholds were used here as were used for all the range images. This is an extremely strong testimonial to the soundness of the digital surface approach when the same program with the same set of input thresholds and window sizes can segment the coffee cup images, the keyboard images, the block with holes image (low-noise), and the space shuttle intensity image without a single modification.

7.4.2. Road Image

Figure 7.20.1 shows an intensity image of a road scene from an Autonomous Land Vehicle sequence. Again, the surface-curvature sign segmentation appears almost structureless. The best-fit region label segmentation is somewhat fragmented in Figure

7.20.2(b). However, the reconstructed image is good, and the results in Figure 7.20.2(c) appear to be useful. The right edge of the road is particularly well-defined, and the sky, hills, and grass are segmented very well. The image quality measure is 1.95.

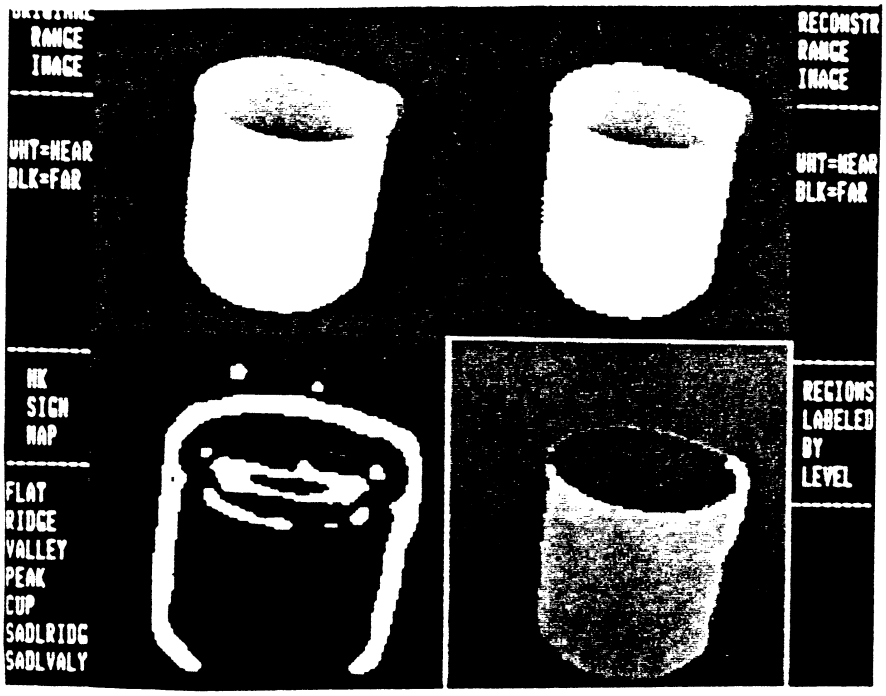
7.4.3. Girl (USC)

For comparison purposes, two standard images from the image processing literature were selected. The first of these is the USC girl image. A 128 x 128 (8-bit) version of the original 512 x 512 (5-bit) image was used as the input to the segmentation algorithm. Two shrinking stages were used where four pixels are averaged and replaced by a single pixel with the average value. This input image, the standard algorithm reconstruction image, and the overlaid segmentation plot image are shown in Figure 7.21.1. The segmentation results in Figure 7.21.2 are not nearly as good as the space shuttle results, but are still reasonable for such a complicated image. The facial features caused significant difficulties. The image quality measure is poor (2.63), and a lot of small detail is needed for an adequate description of the image. Figure 7.21.3 (a best fit reconstruction image) and Figure 7.21.4 (an overlaid plot image) show segmentation algorithm results obtained using the aggressive error tolerance increase function discussed in Chapter 4. These results are much better than those obtained with the standard algorithm. This significant improvement with the larger error tolerance function is probably due to the fact that the original image was digitized to only five bits. Even though the smoothing during the shrinking process helped, the effect of the original quantization is still present as determined by histogram analysis.

7.4.4. House (Univ. of Mass)

The second standard image from the literature is a U-Mass house scene shown in Figure 7.22.1 along with the best-fit reconstruction image. A slightly different format is

adopted for this set of results because of the complexity of the image. This image is 256 x 256 pixels (8-bits) obtained by a simple averaging of the original red, green, and blue images. As usual, the surface curvature sign segmentation in Figure 7.22.2(a) appears to be a jumbled mess of small regions except for some large pieces of the sky. The best-fit region label segmentation in Figure 7.22.2(b) provides enough image detail that one can distinguish the outline of the house and the pants of the person walking in front of the house. The final region segmentation, obtained from the region merging process, is much easier to interpret. The road surface is successfully merged into two large regions separated by the white line in the center of the road. Even the shadow of the person walking in front of the house is isolated. The garage door is segmented into a shadowed portion and an unshadowed portion. The lawn is represented by only three different large image regions. One major unexpected difficulty, however, is that the roof of the house is not merged into a single region. Figure 7.22.3 shows the overlaid segmentation plot image for the standard thresholds and a different overlaid segmentation plot image obtained using a slight modification in thresholds. The front lawn, the side of the house, and the sideyard tree are less fragmented in the modified threshold results. The modified threshold results for the four image format and the best-fit reconstruction image are shown in Figure 7.22.4. The algorithm stopped in both cases when it reached the maximum number of regions (currently limited to 254), which caused the large black (unfitted) regions in the reconstructed images. The trees in this image and the USC girl's hair show that the algorithm can successfully segment large regions of similar texture.



(a) Standard Results for Coffee Cup A

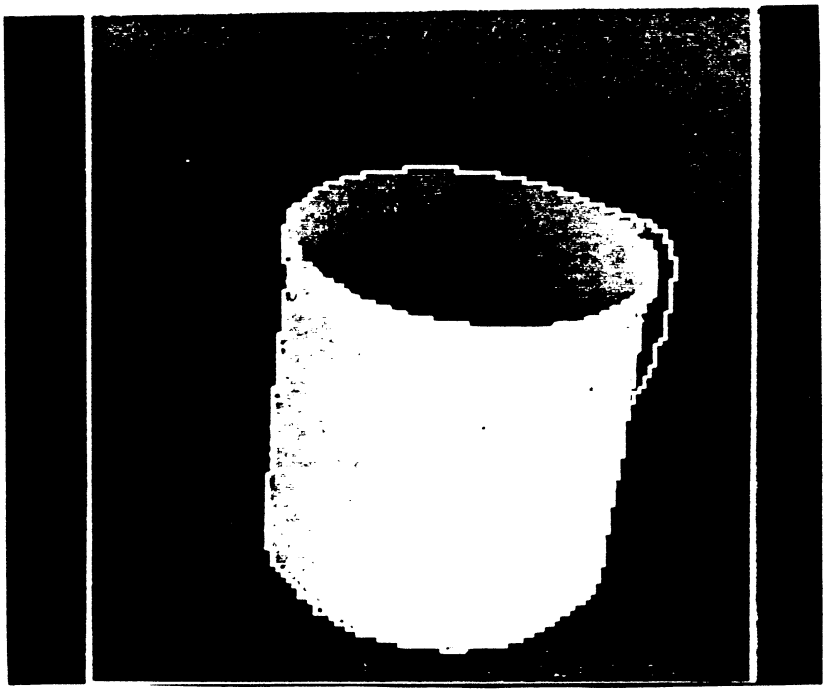
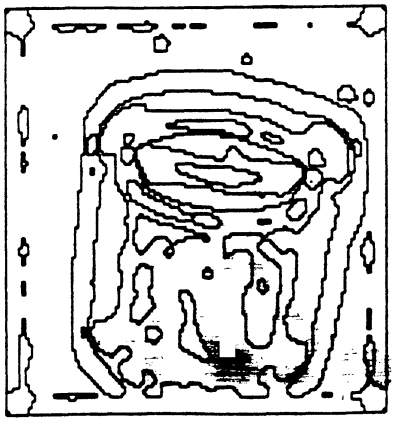
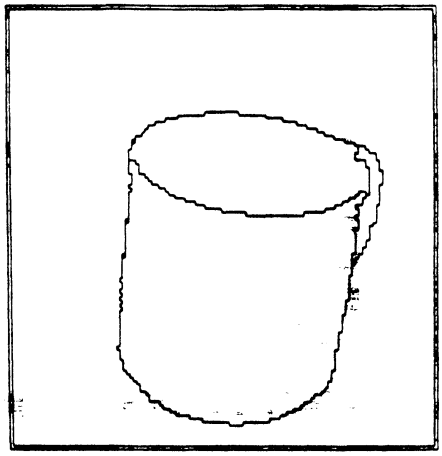


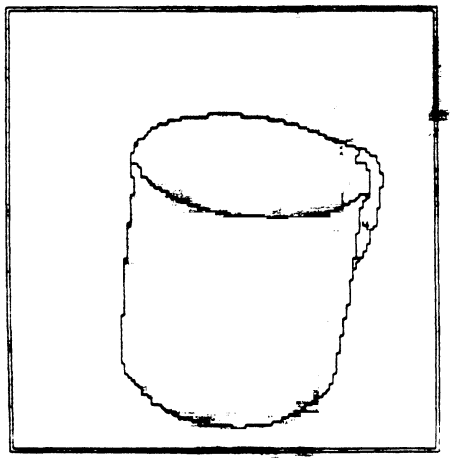
Figure 7.1.1.(b) Standard Results for Coffee Cup A



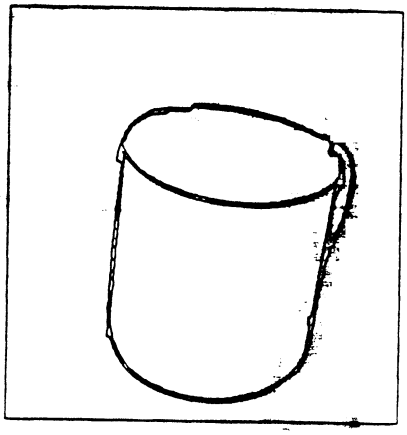
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation



(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.1.2. Standard Results for Coffee Cup A

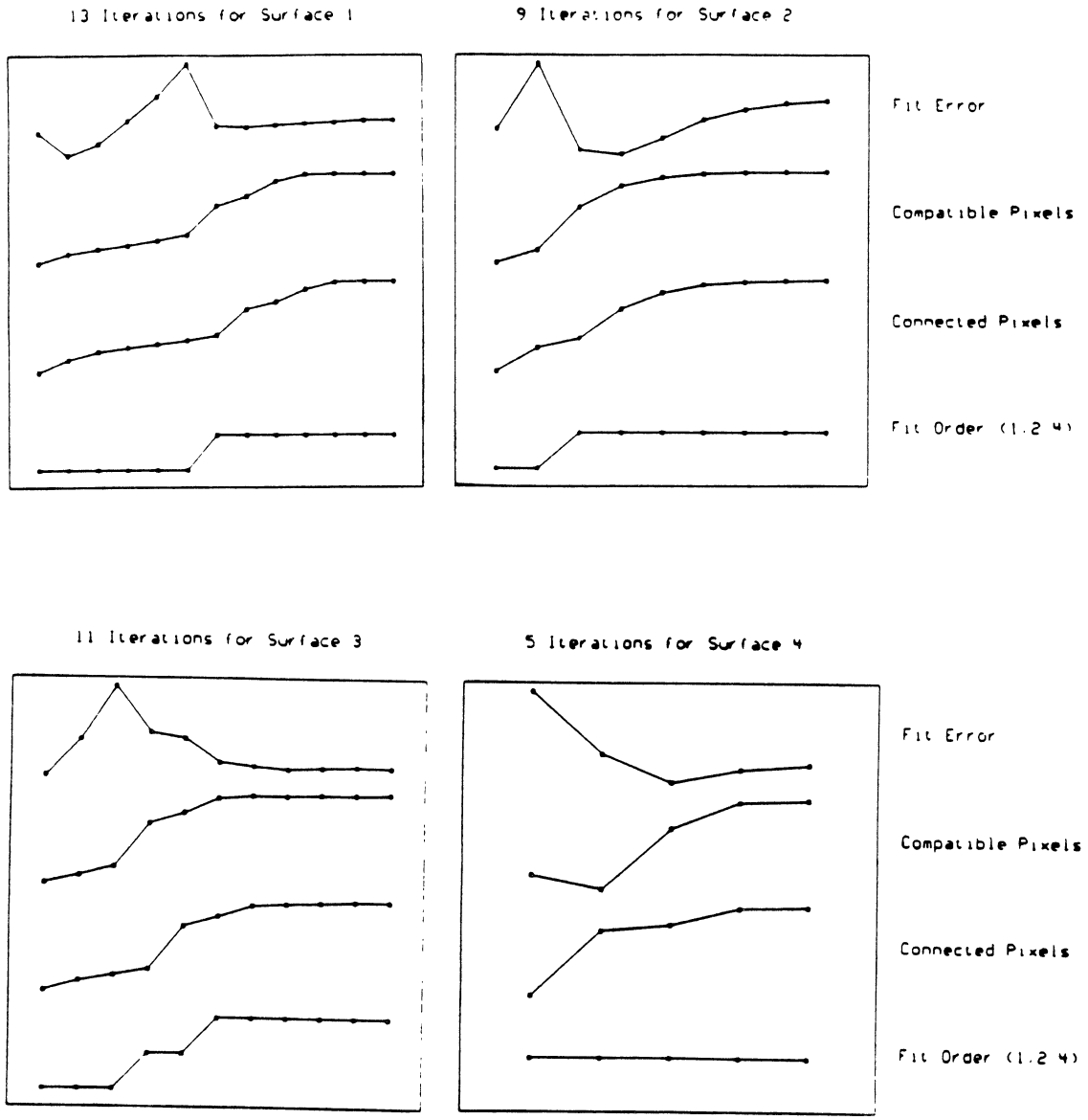


Figure 7.1.3. Iterative Quantities for Coffee Cup A

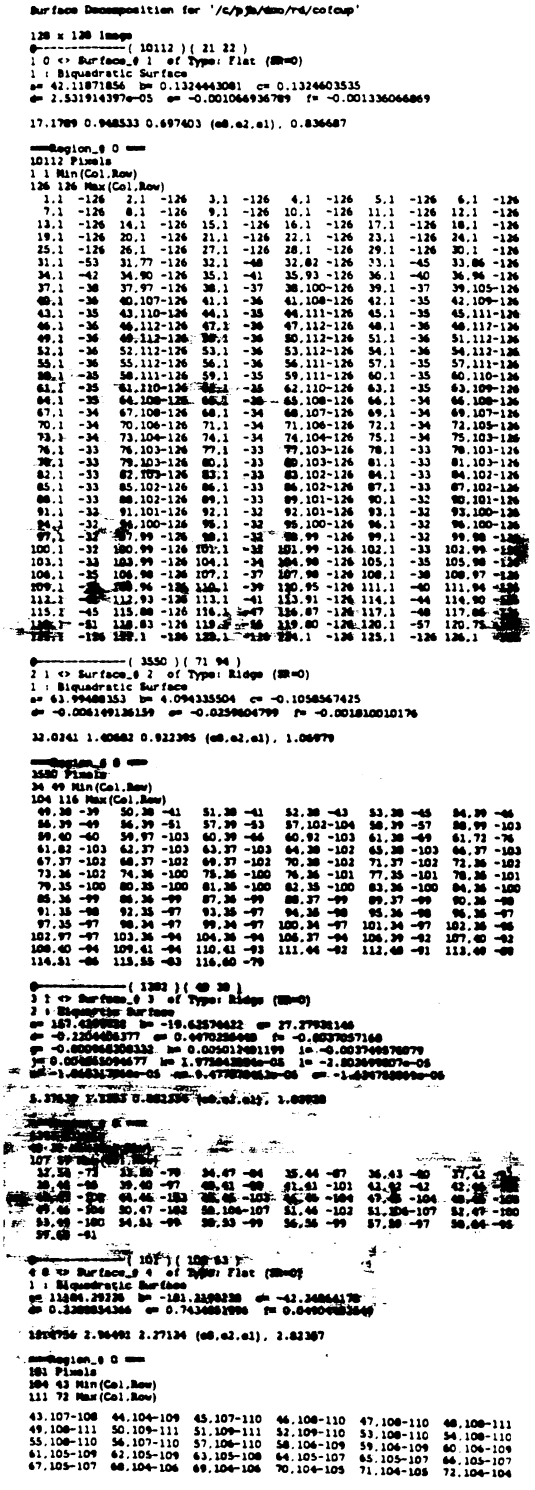


Figure 7.1.4. Surface Primitive Information for Coffee Cup A

(a) Standard Results for Coffee Cup B

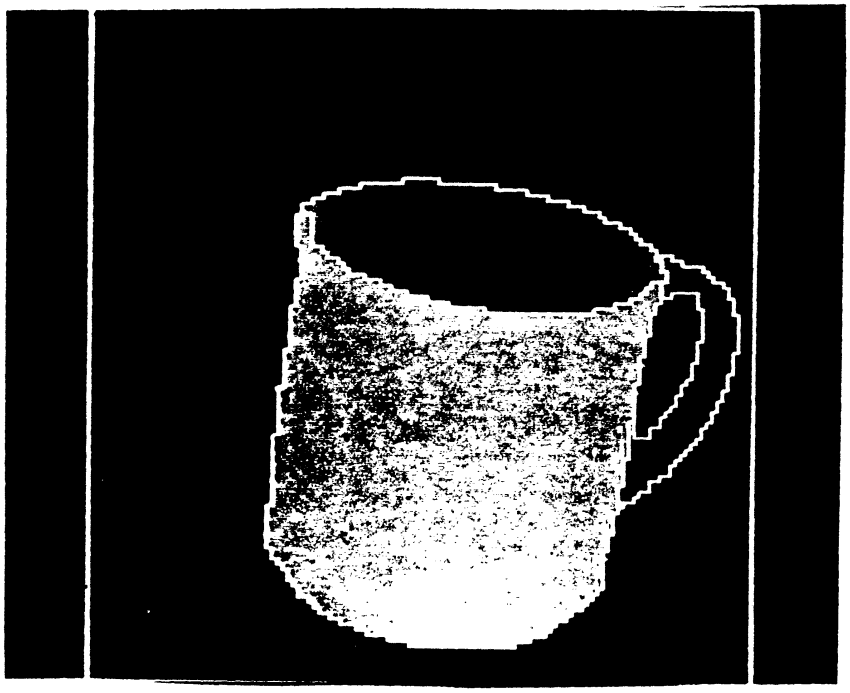
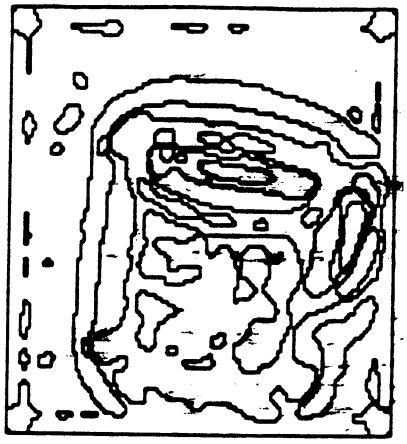
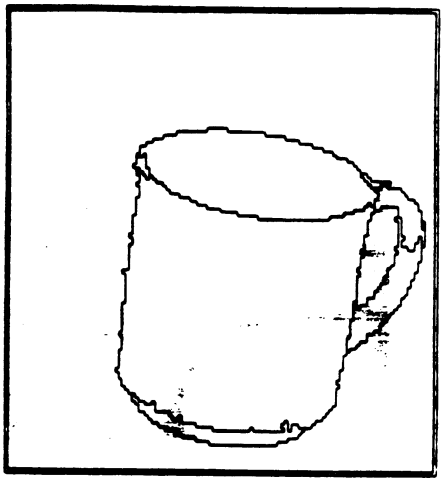


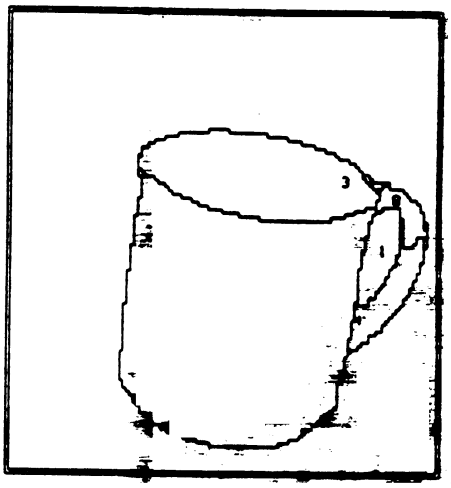
Figure 7.2.1.(b) Modified Threshold Results for Coffee Cup B



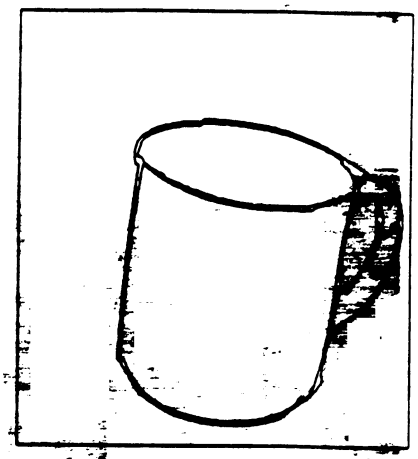
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation



(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.2.2. Standard Results for Coffee Cup B

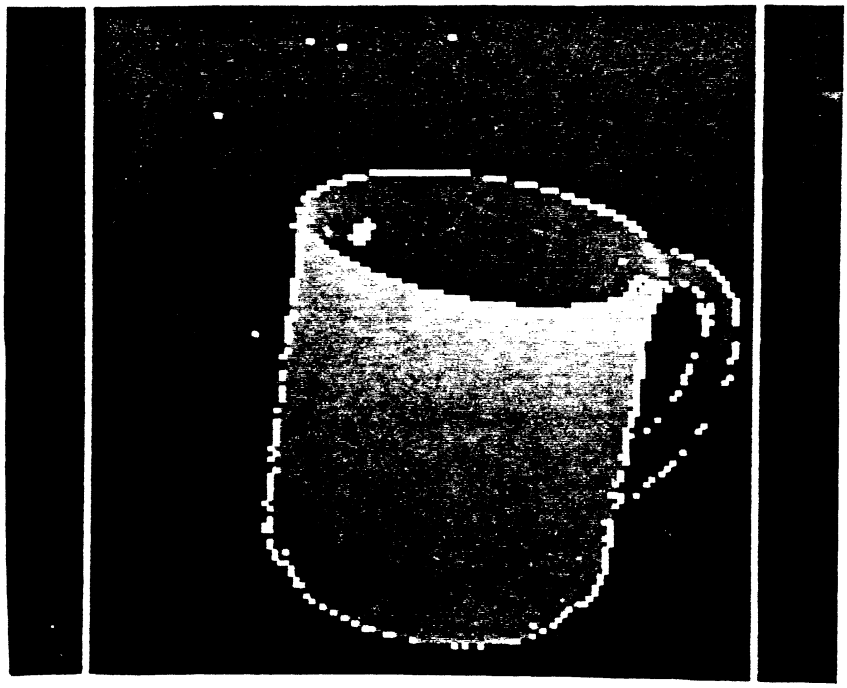


Figure 7.2.3. Error Threshold Image Overlaid on Original Image

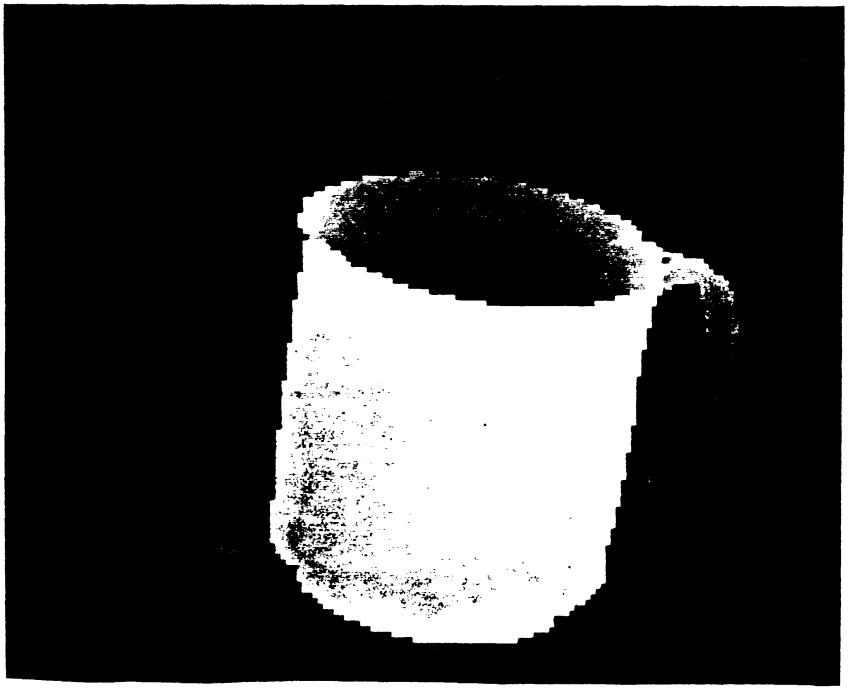


Figure 7.2.4. Final Region Label Buffer Reconstruction Image

(a) Original Coffee Cup B Image Mapped Through Random Lookup Table

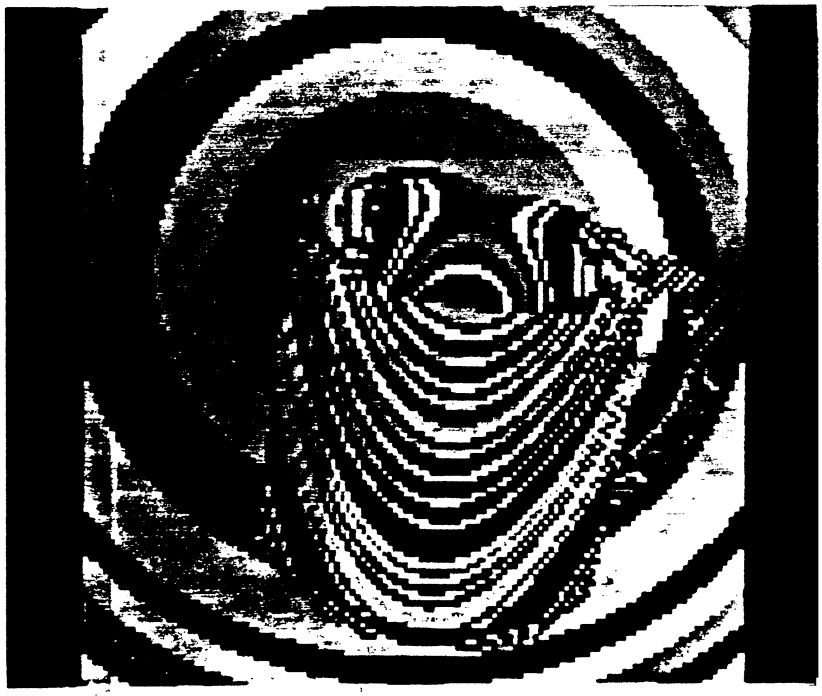
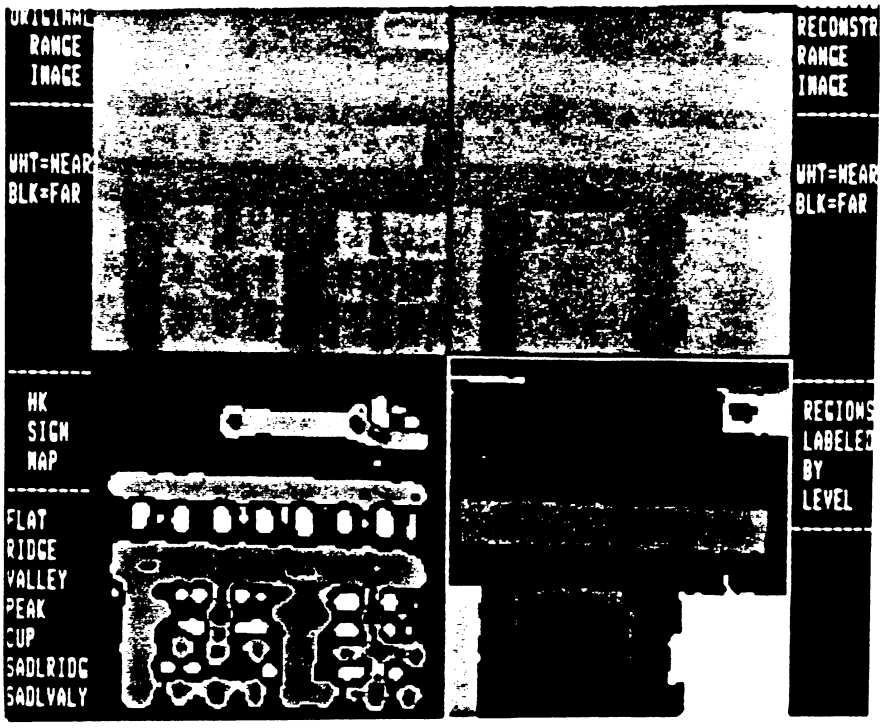


Figure 7.2.5.(b) Reconstruction Image with Random Gray Levels



(a) Standard Results for Keyboard A

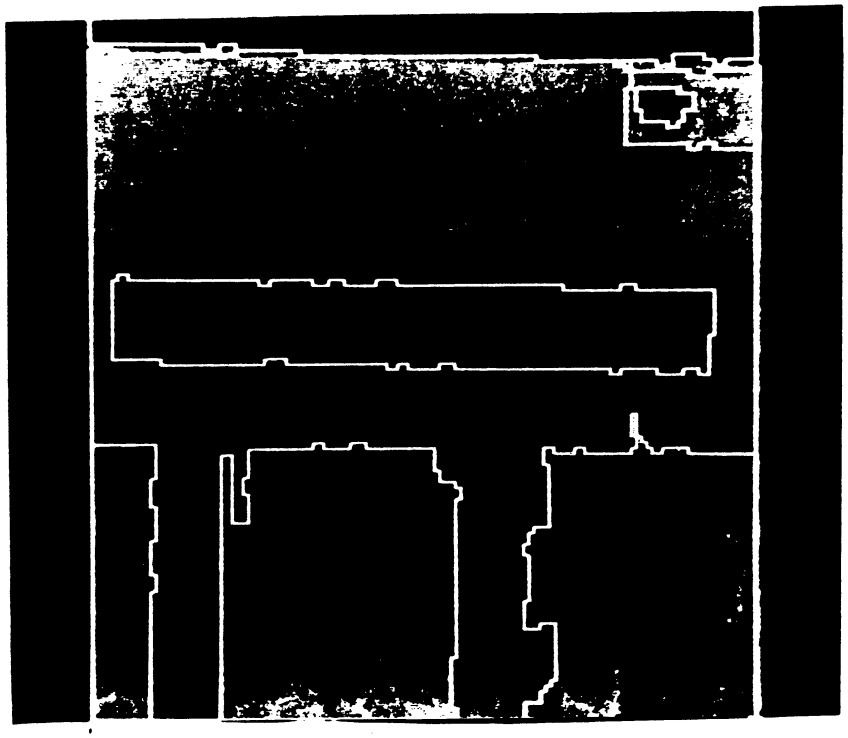
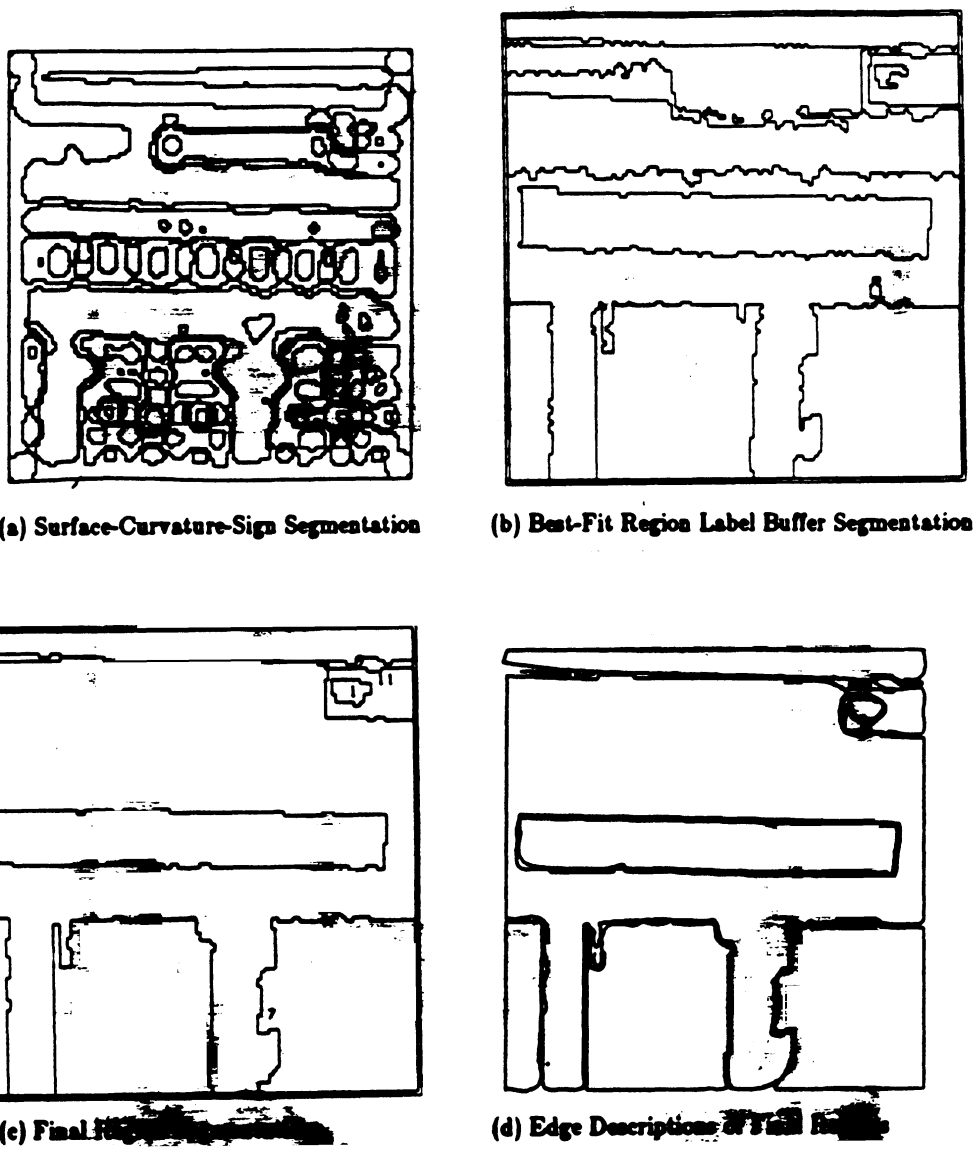


Figure 7.3.1.(b) Standard Results for Keyboard A



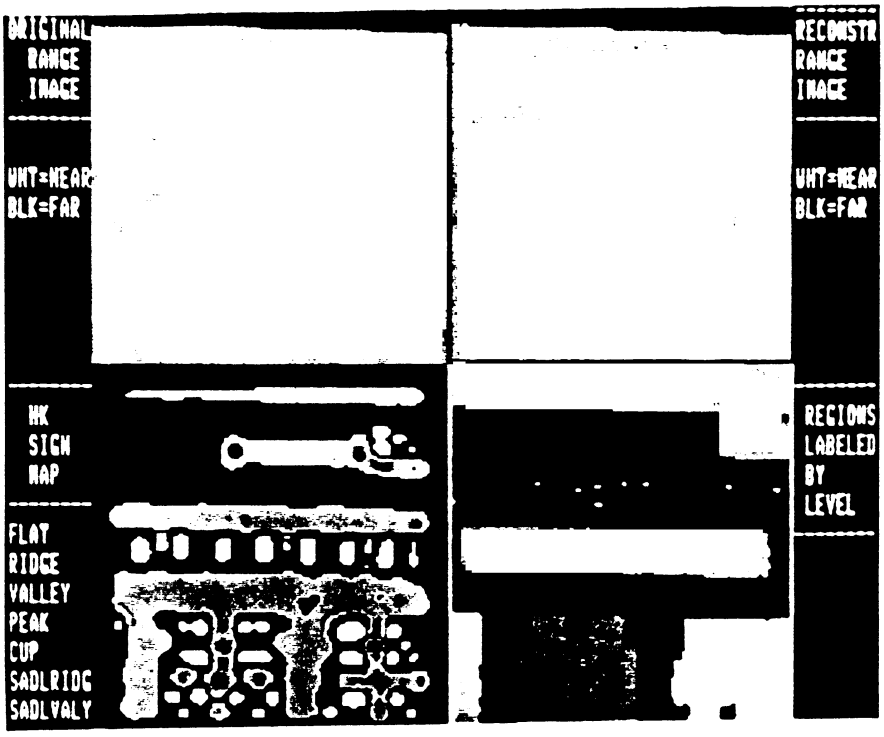
(a) Surface-Curvature-Sign Segmentation

(b) Best-Fit Region Label Buffer Segmentation

(c) Final

(d) Edge Descriptions of Final

Figure 7.3.2. Standard Results for Keyboard A



(a) Standard Results for Keyboard B

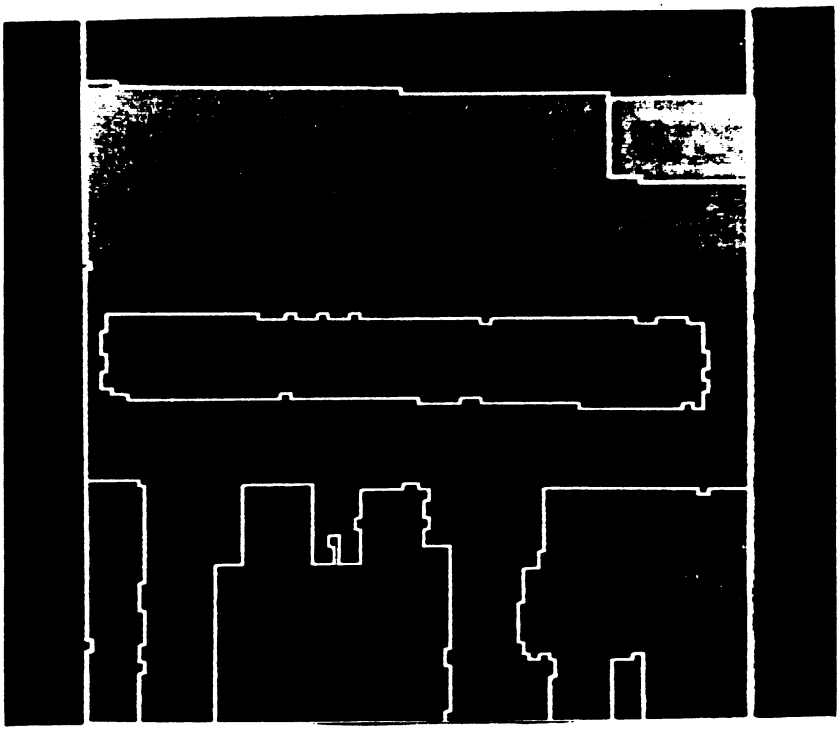
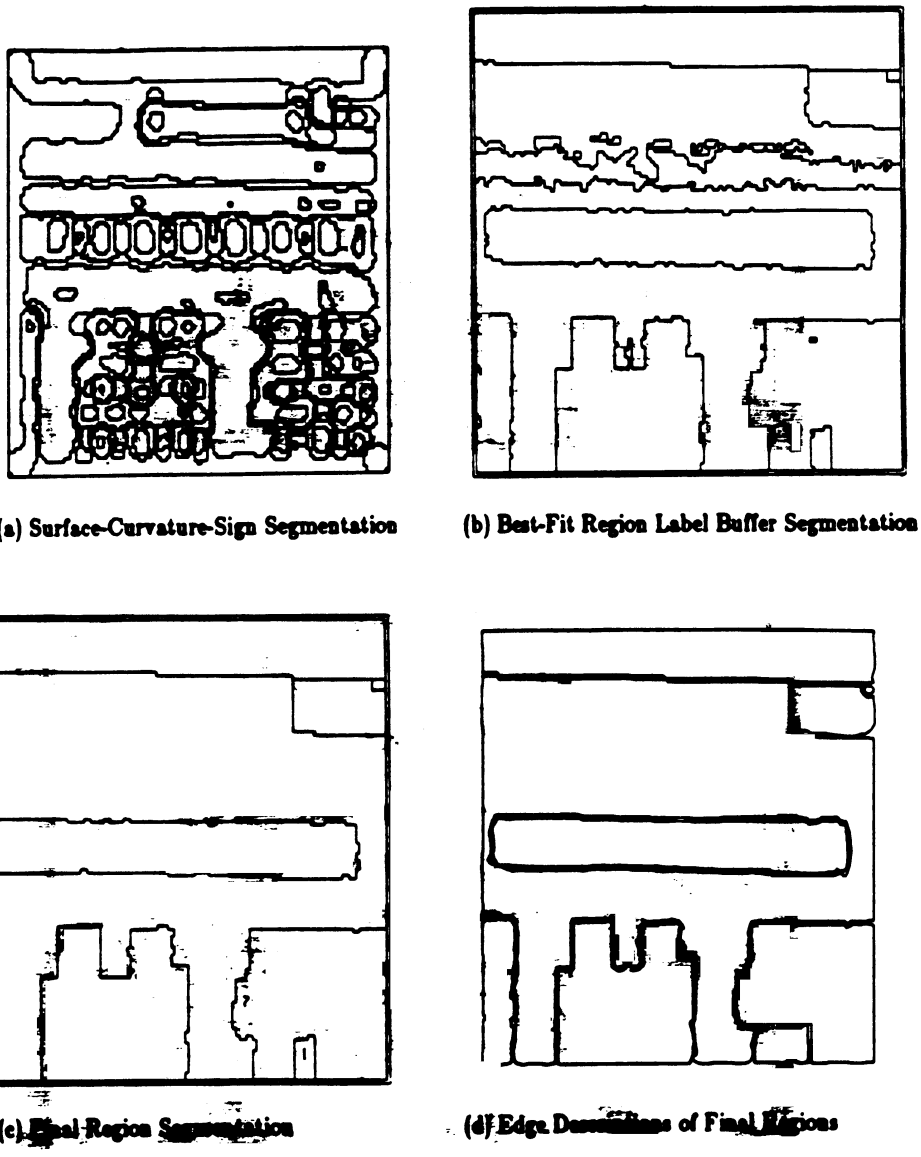


Figure 7.4.1.(b) Standard Results for Keyboard B



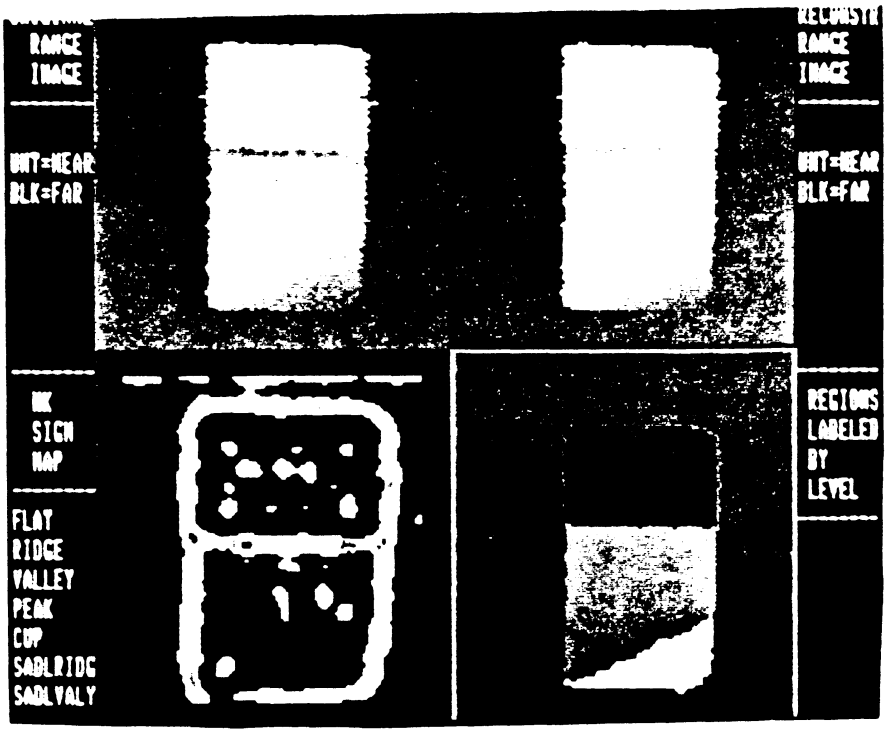
(a) Surface-Curvature-Sign Segmentation

(b) Best-Fit Region Label Buffer Segmentation

(c) Final Region Segmentation

(d) Edge Description of Final Regions

Figure 7.4.2. Standard Results for Keyboard B



(a) Standard Results for Polyhedron

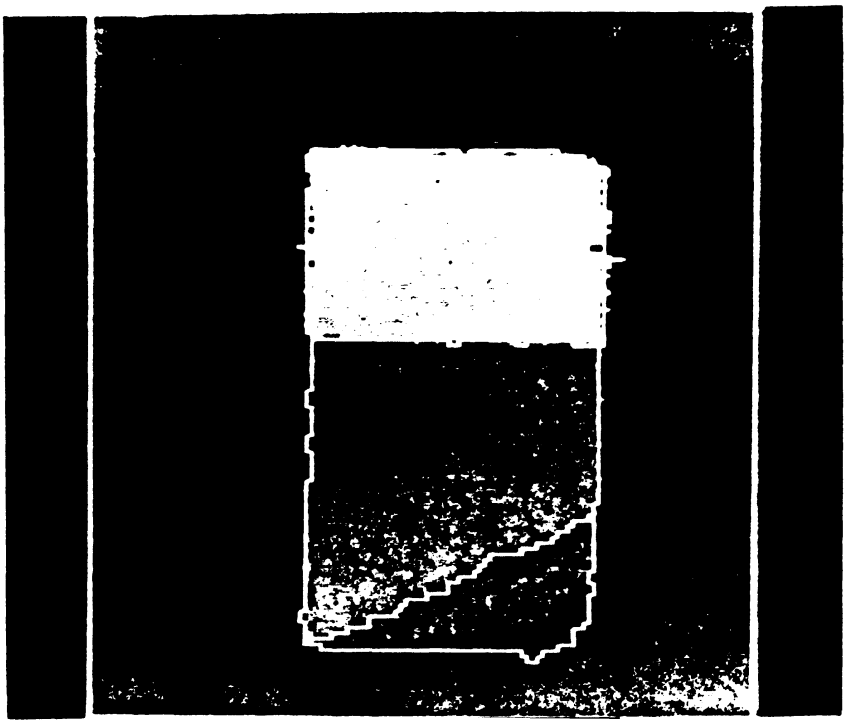
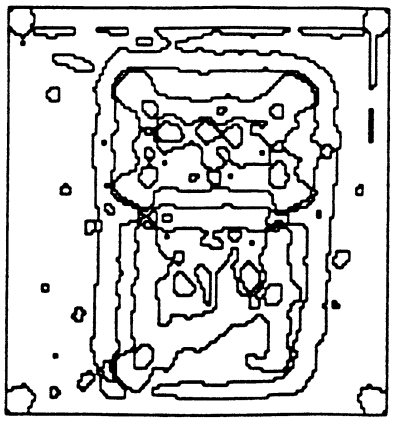
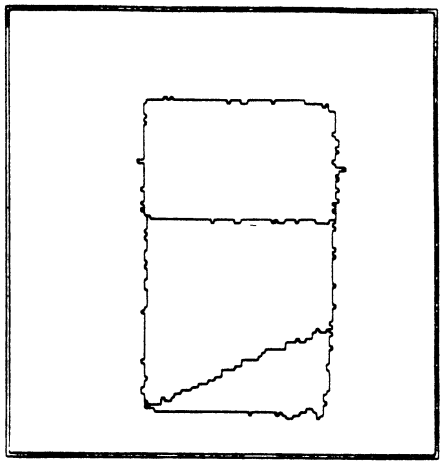


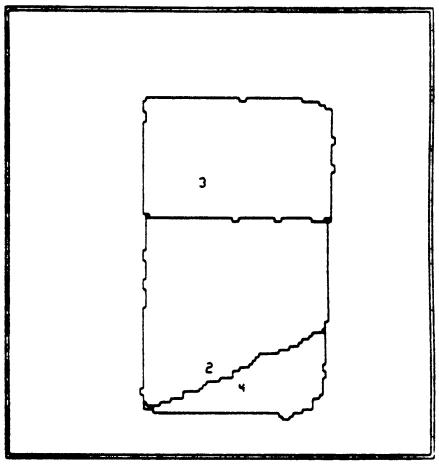
Figure 7.5.1.(b) Standard Results for Polyhedron



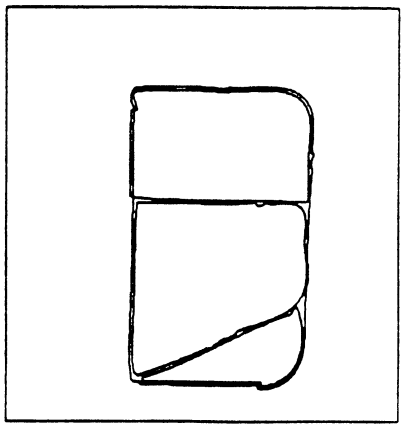
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

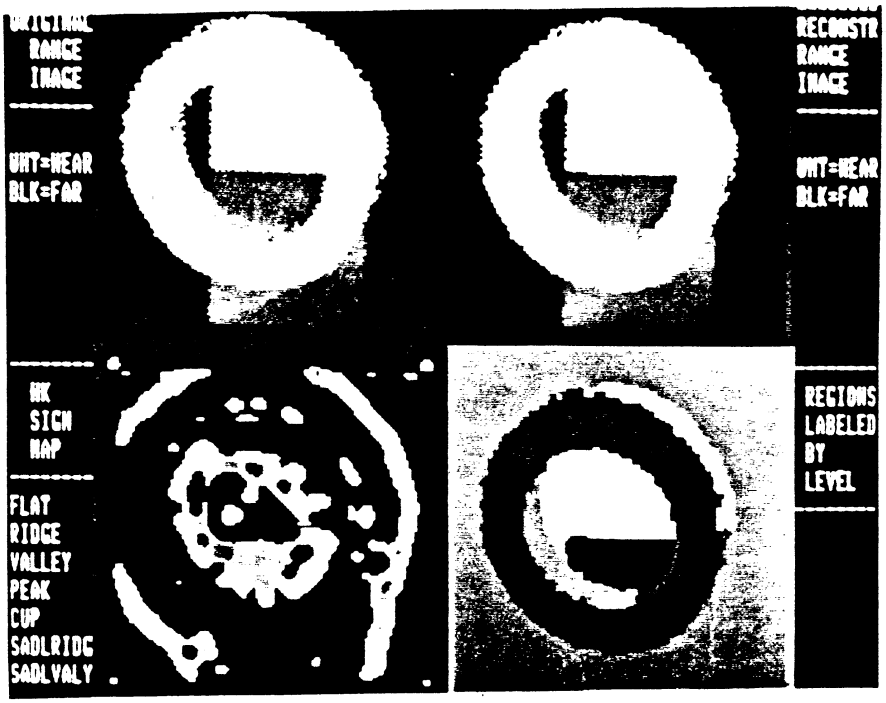


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.5.2. Standard Results for Polyhedron



(a) Standard Results for Ring on Polyhedron

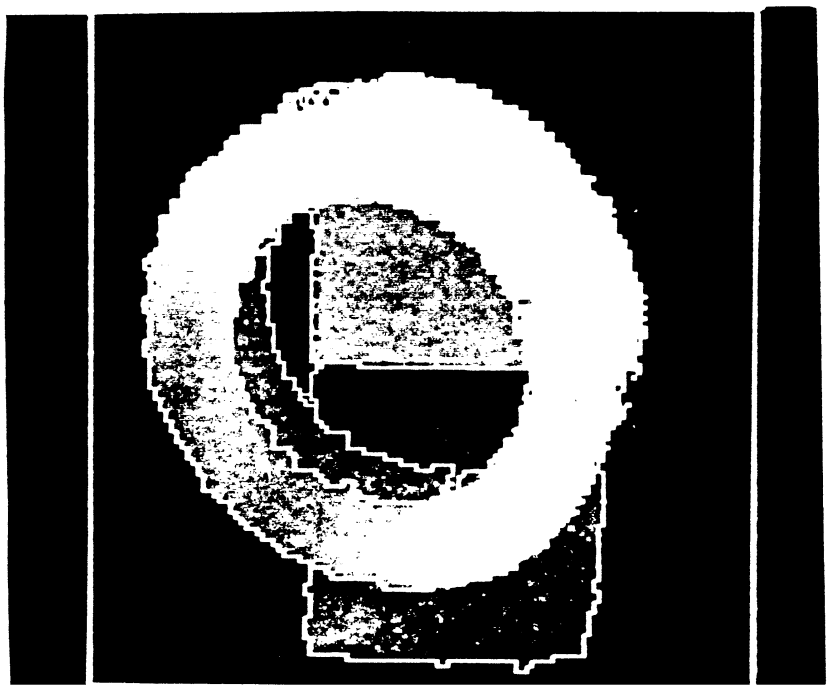
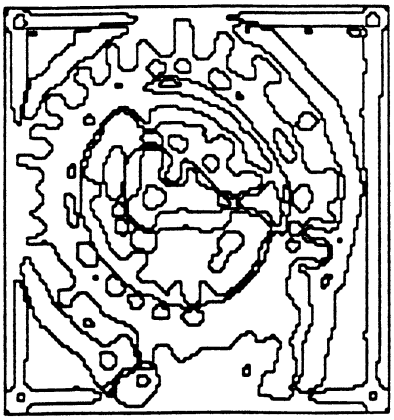
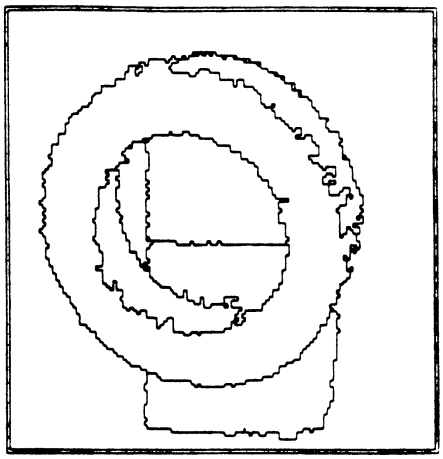


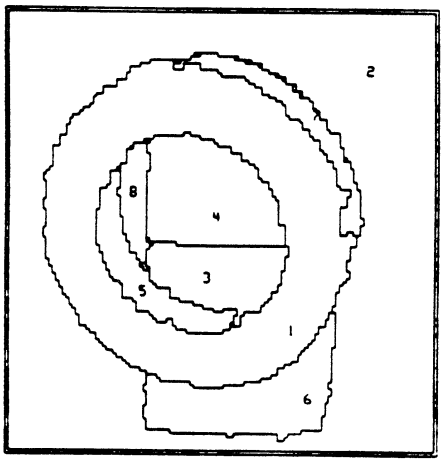
Figure 7.6.1.(b) Standard Results for Ring on Polyhedron



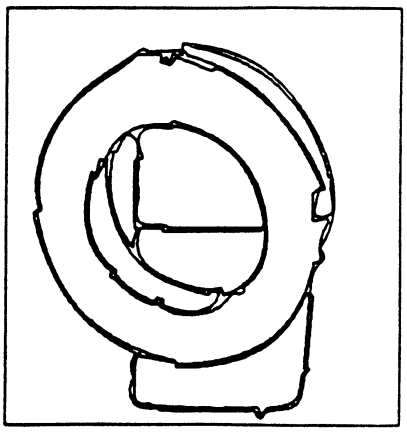
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

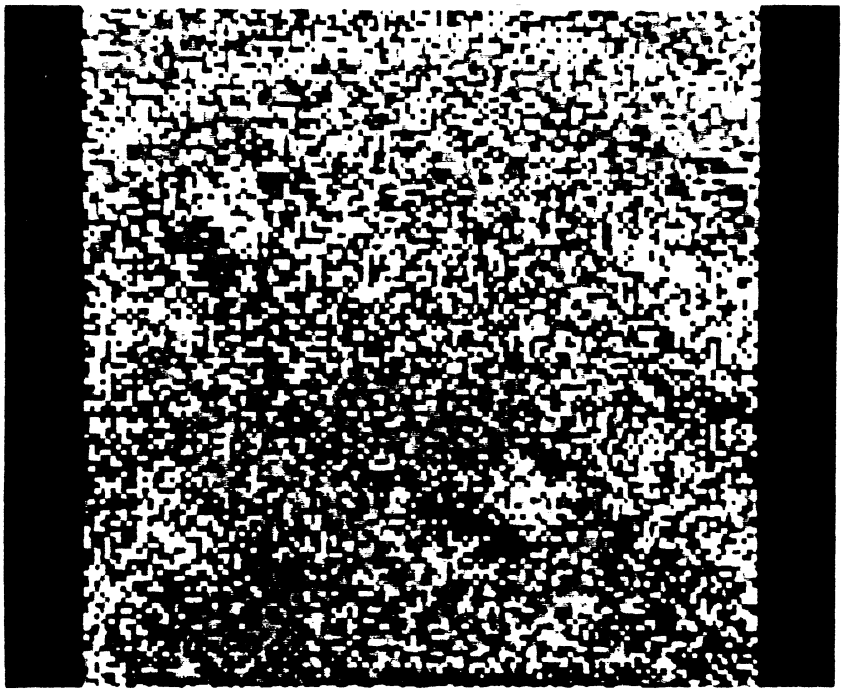


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.6.2. Standard Results for Ring on Polyhedron



(a) Original Ring on Polyhedron Image Mapped Through Random Lookup Table

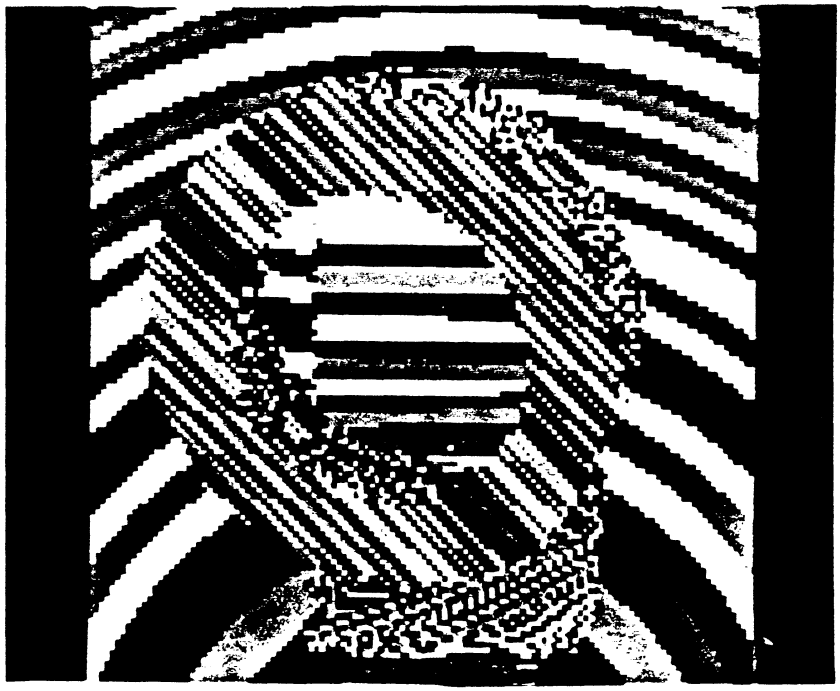
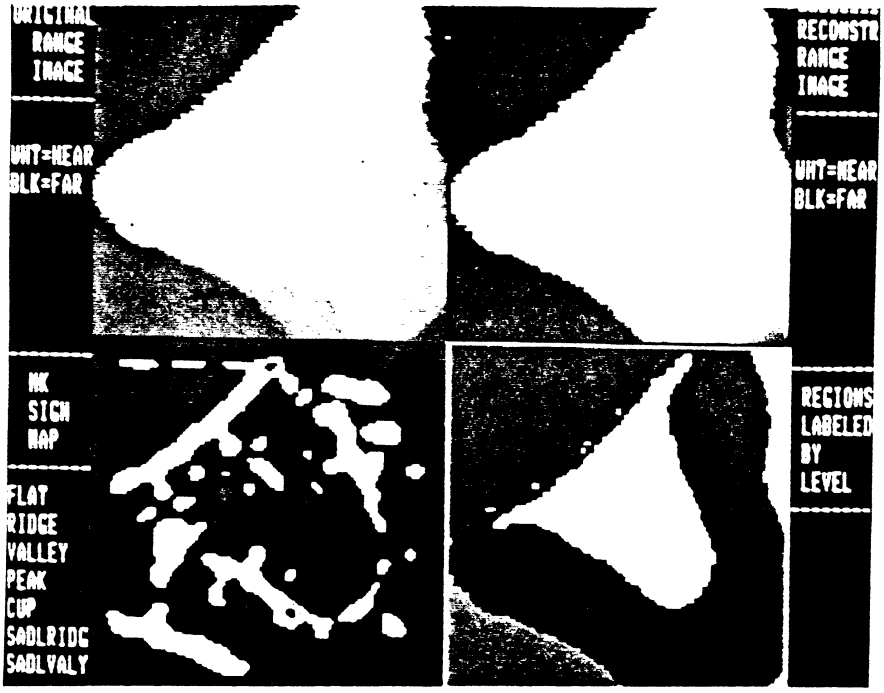


Figure 7.6.3.(b) Reconstruction Image with Random Gray Levels



(a) Standard Results for Curved Surface A

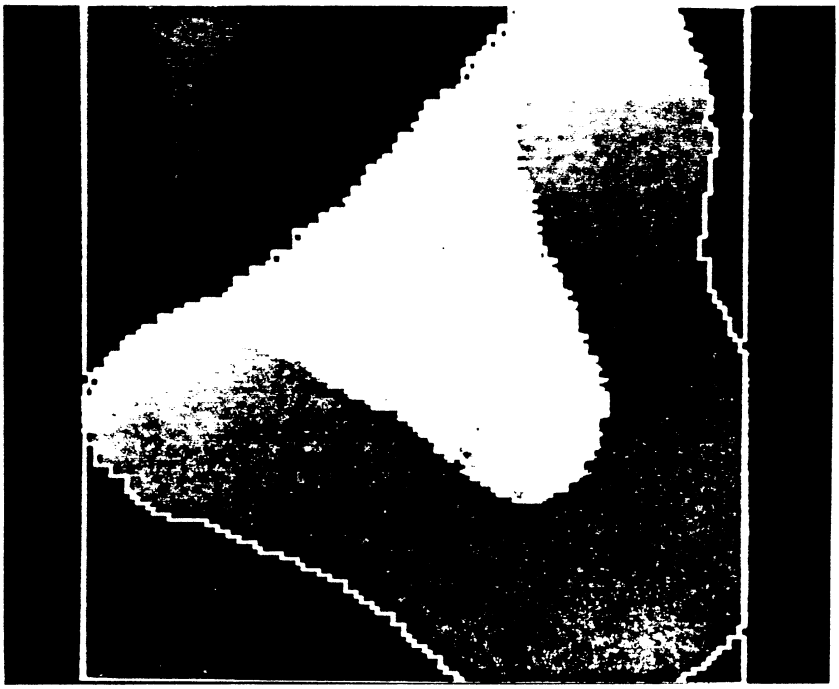
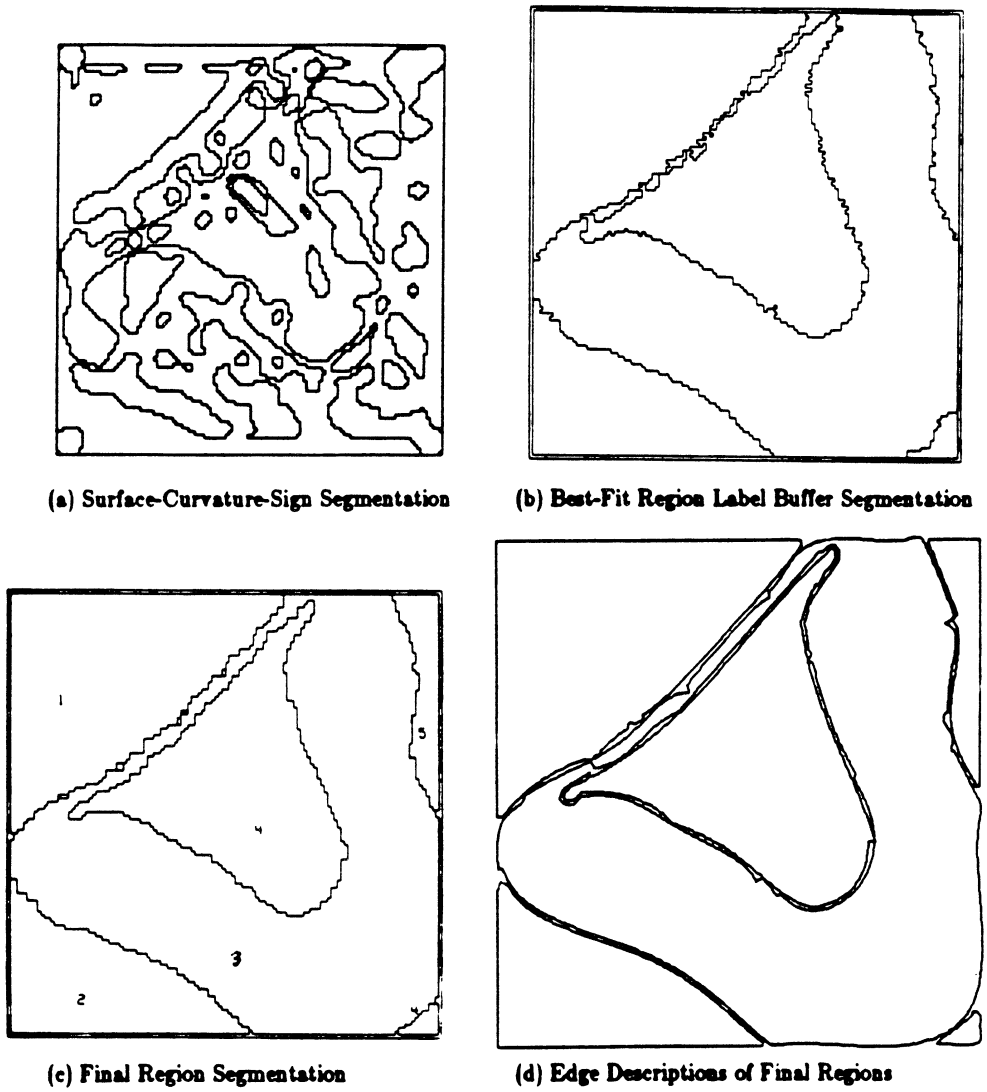


Figure 7.7.1.(b) Standard Results for Curved Surface A



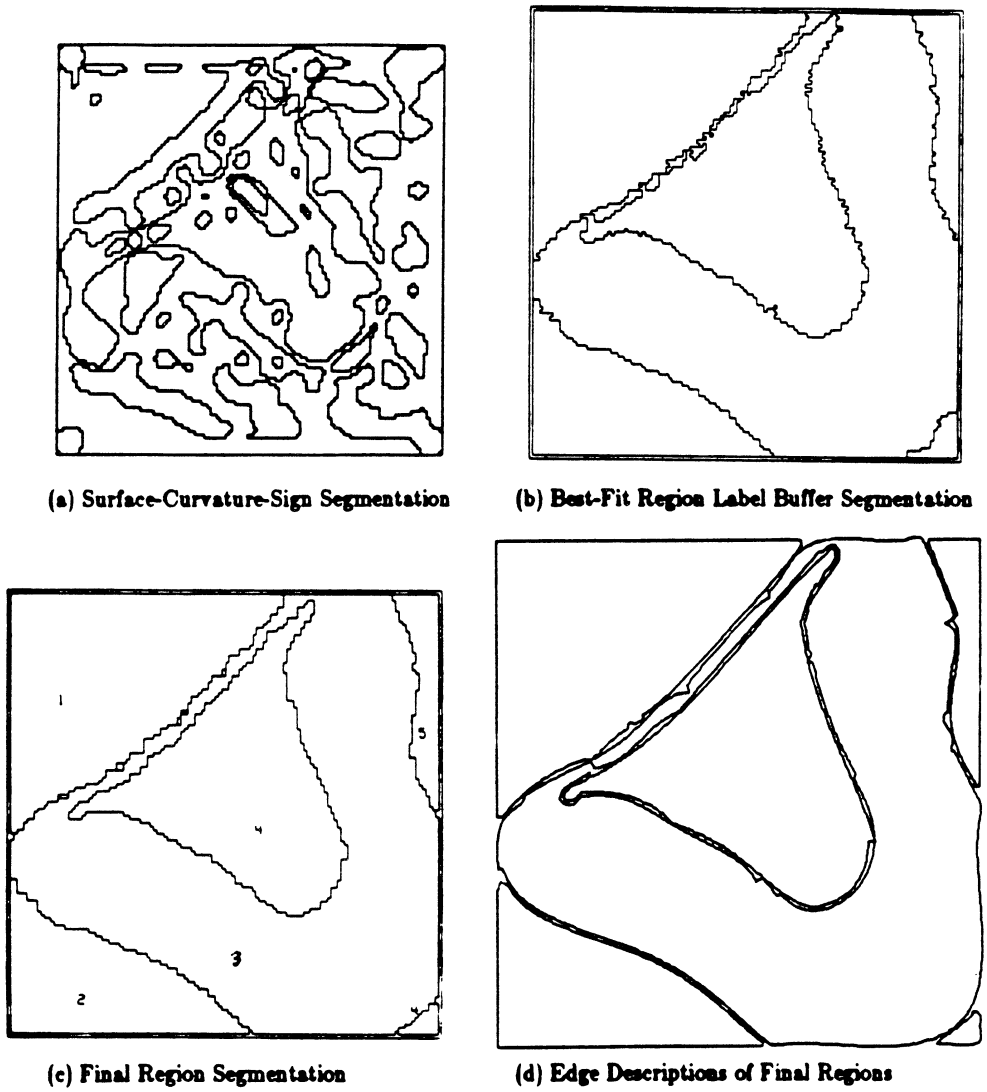
(a) Surface-Curvature-Sign Segmentation

(b) Best-Fit Region Label Buffer Segmentation

(c) Final Region Segmentation

(d) Edge Descriptions of Final Regions

Figure 7.7.2. Standard Results for Curved Surface A



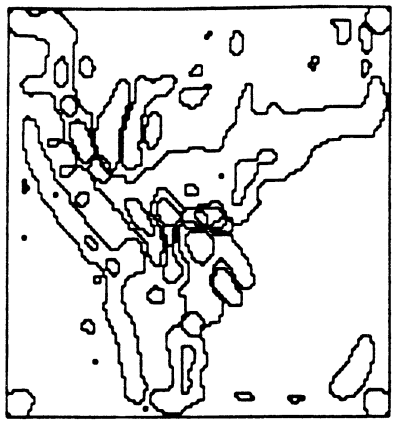
(a) Surface-Curvature-Sign Segmentation

(b) Best-Fit Region Label Buffer Segmentation

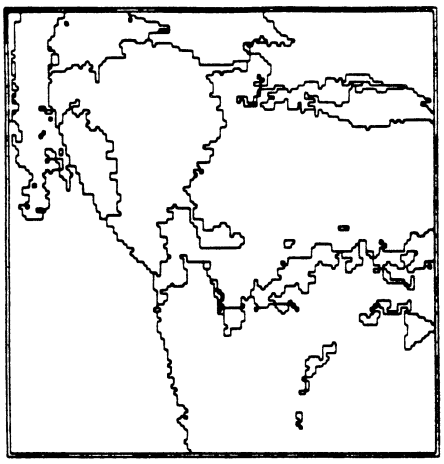
(c) Final Region Segmentation

(d) Edge Descriptions of Final Regions

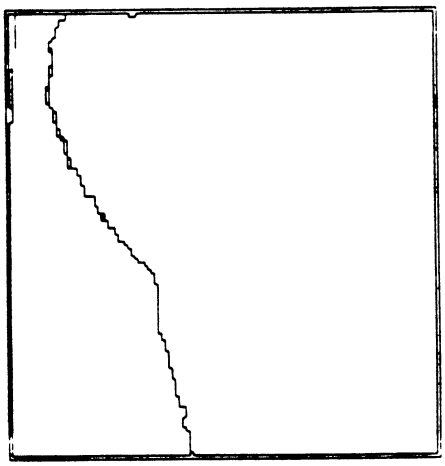
Figure 7.7.2. Standard Results for Curved Surface A



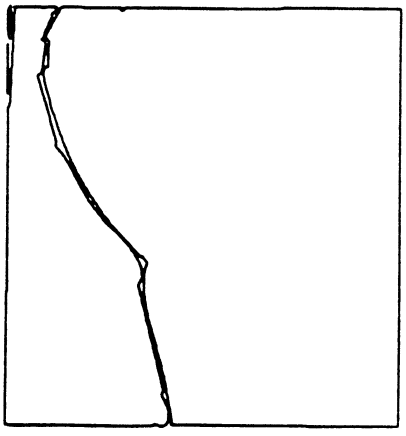
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

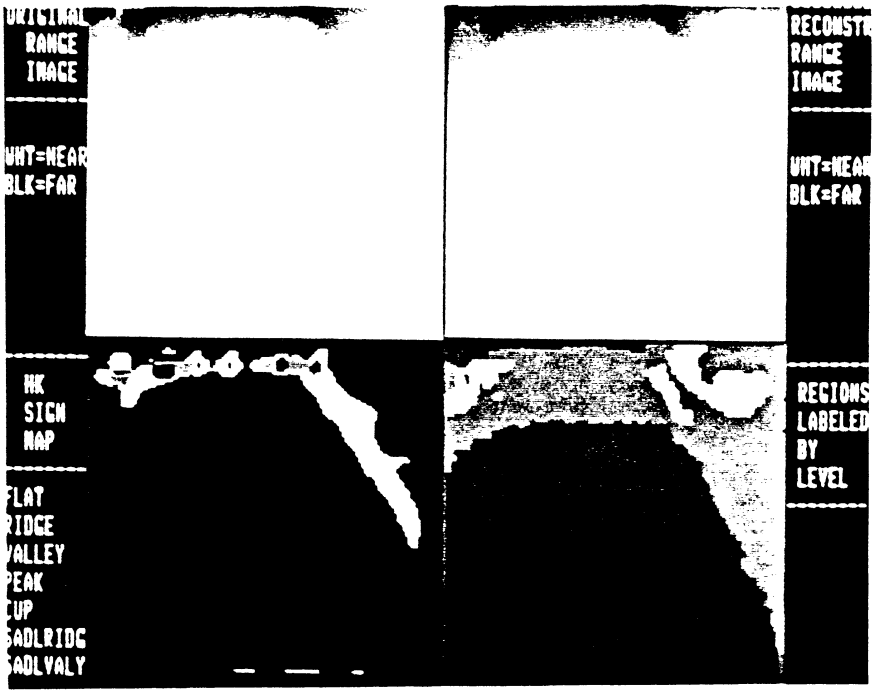


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.8.2. Standard Results for Curved Surface B



(a) Standard Results for Road Scene A

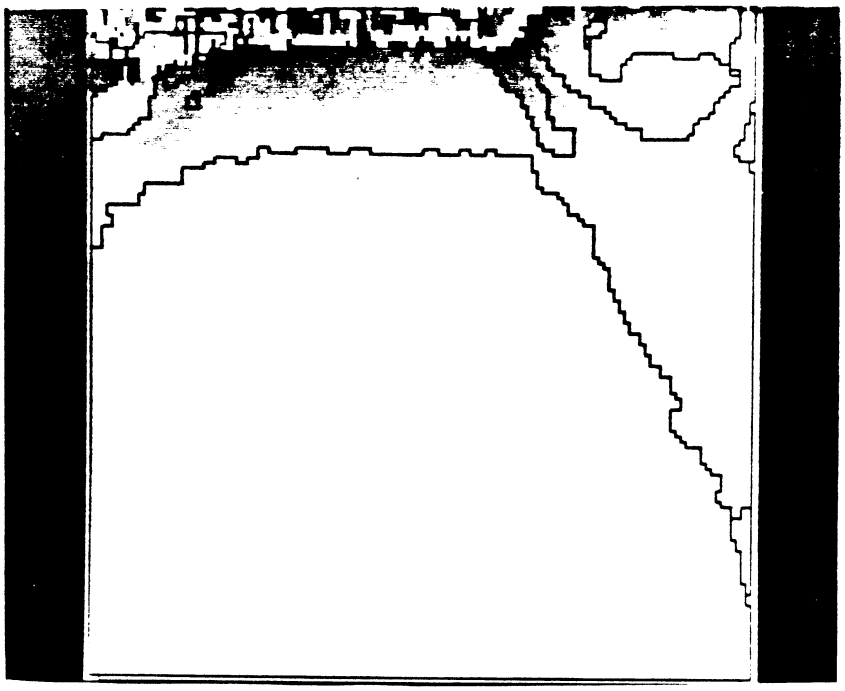
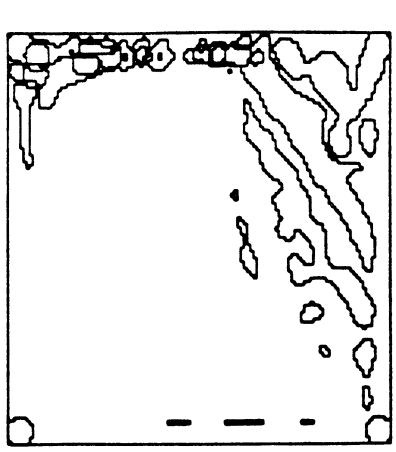
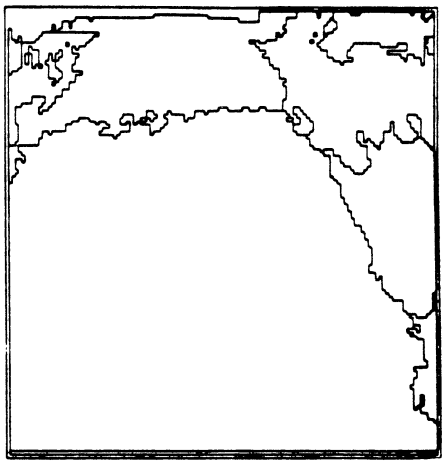


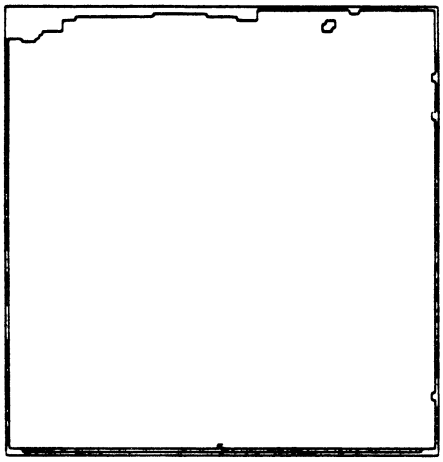
Figure 7.9.1.(b) Standard Results for Road Scene A



(a) Surface-Curvature-Sign Segmentation

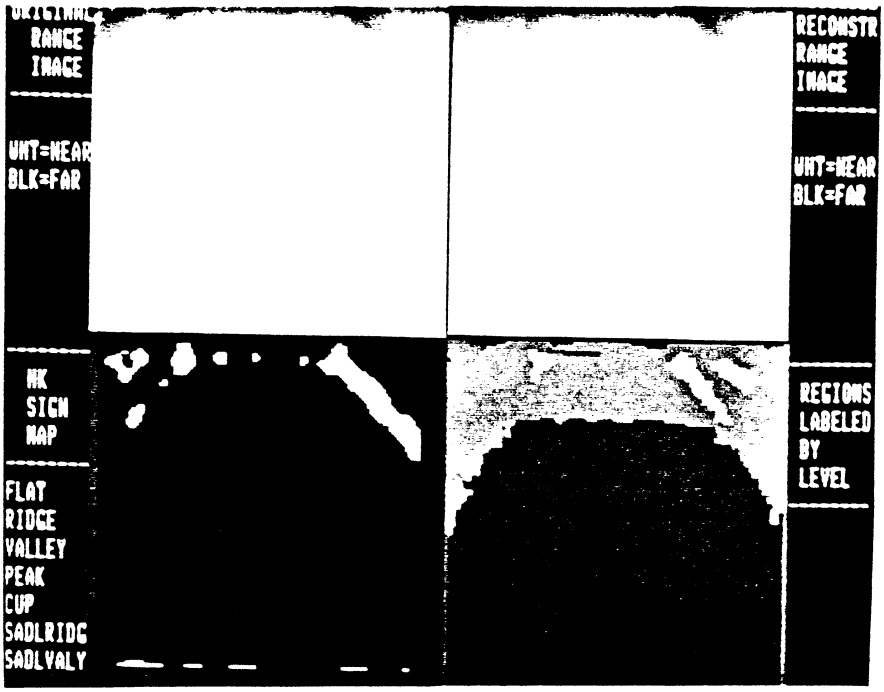


(b) Best-Fit Region Label Buffer Segmentation



(c) Final Region Segmentation

Figure 7.9.2. Standard Results for Road Scene A



(a) Standard Results for Road Scene B

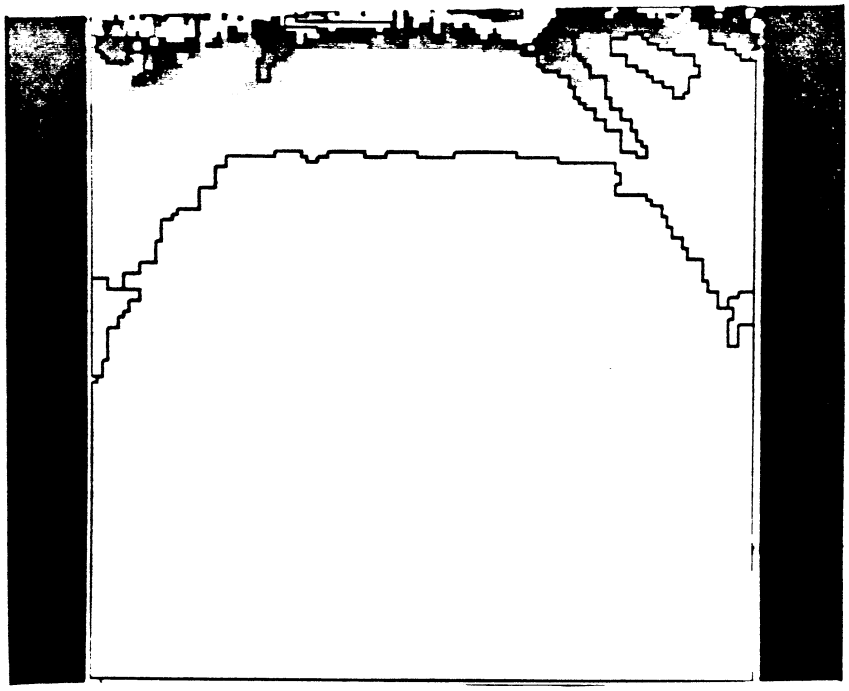
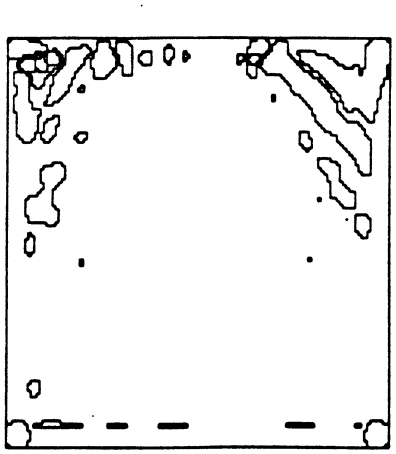
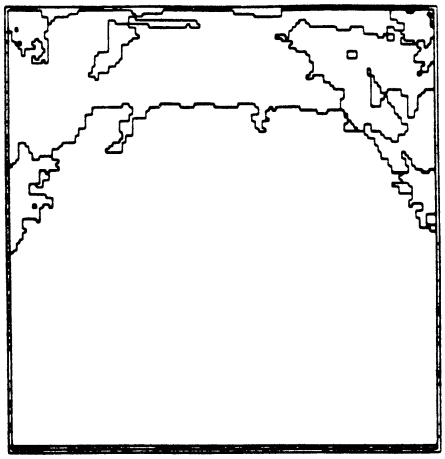


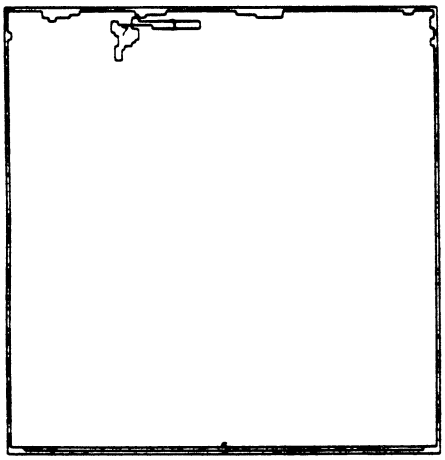
Figure 7.10.1.(b) Standard Results for Road Scene B



(a) Surface-Curvature-Sign Segmentation

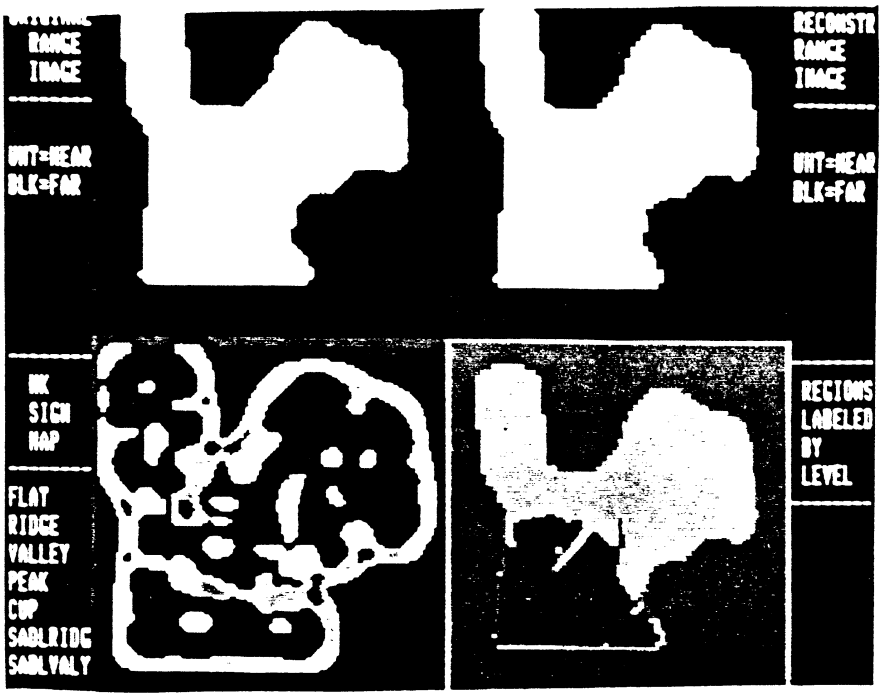


(b) Best-Fit Region Label Buffer Segmentation



(c) Final Region Segmentation

Figure 7.10.2. Standard Results for Road Scene B



(a) Standard Results for Auto Part

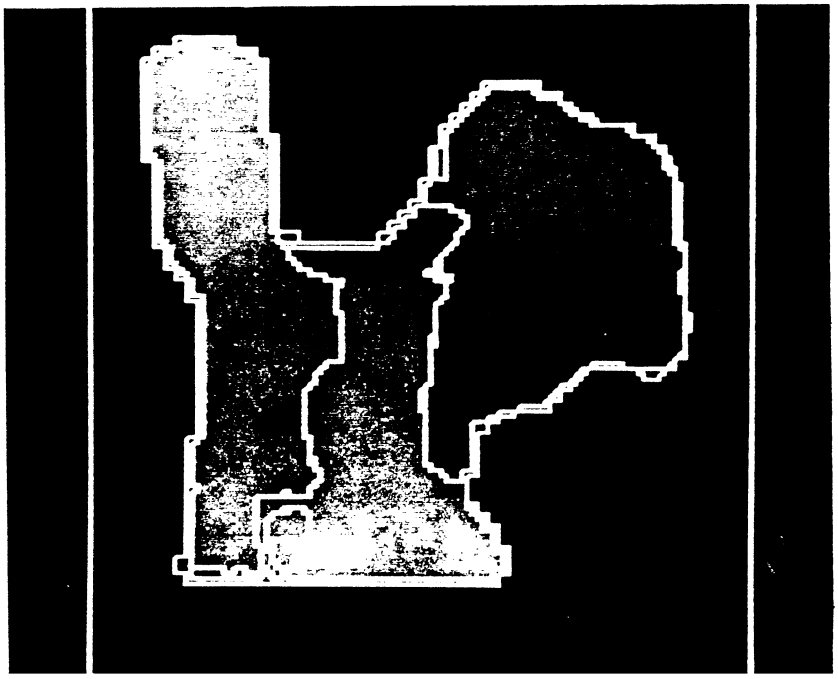
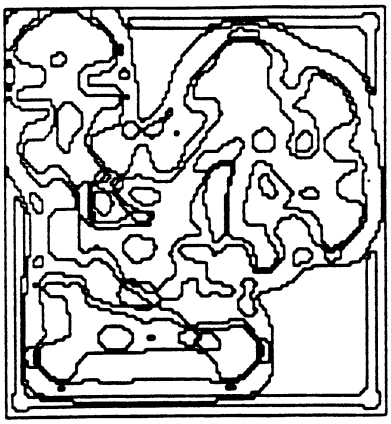
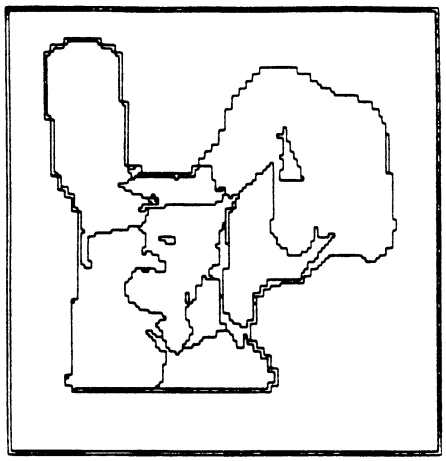


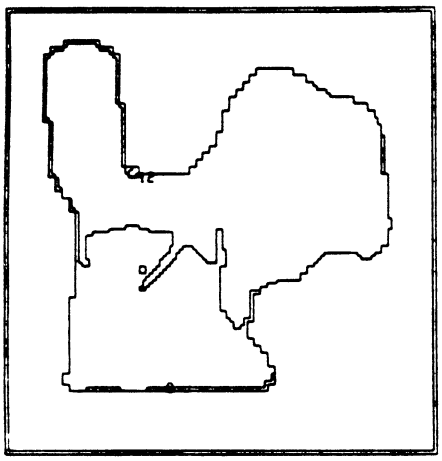
Figure 7.11.1.(b) Modified Threshold Results for Auto Part



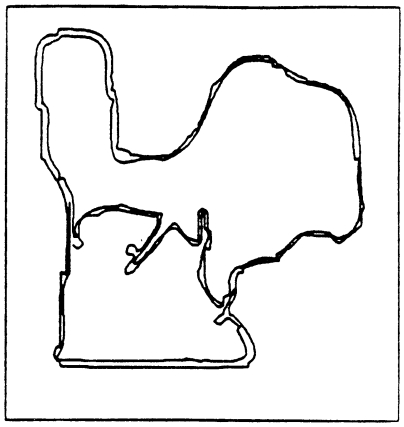
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

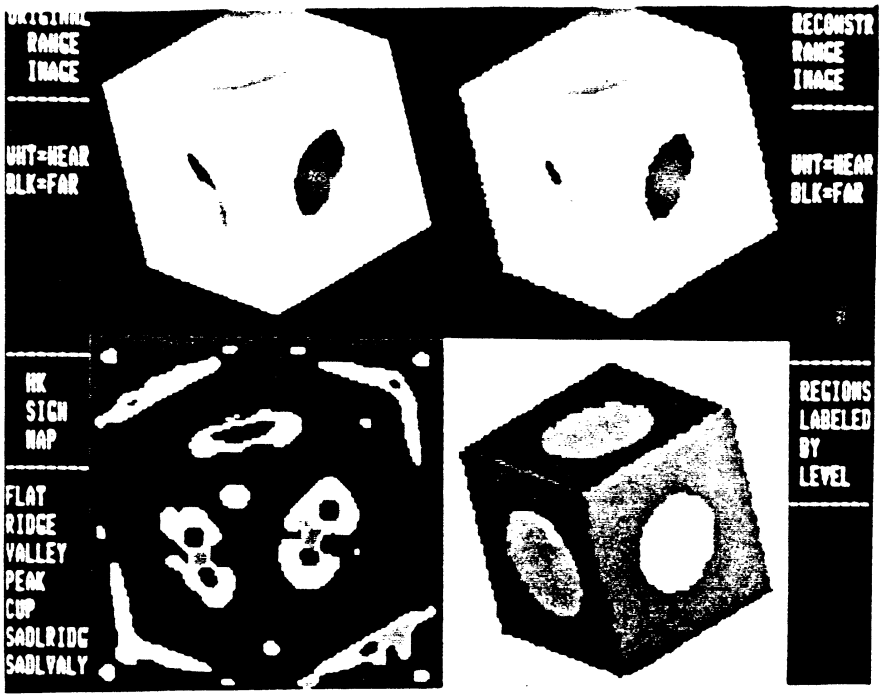


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.11.2. Standard Results for Auto Part



(a) Standard Results for Block with Holes (Low-Noise)

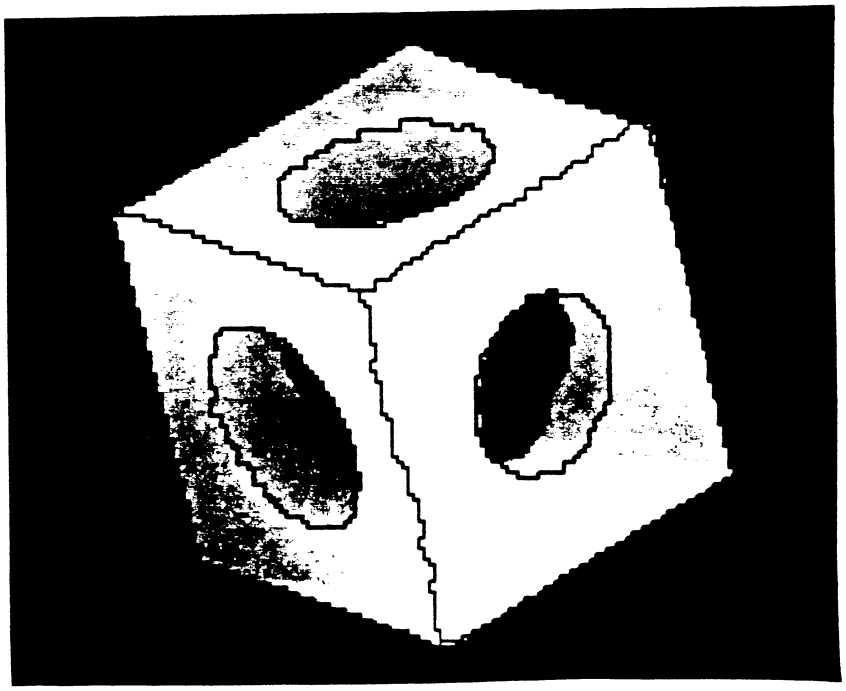
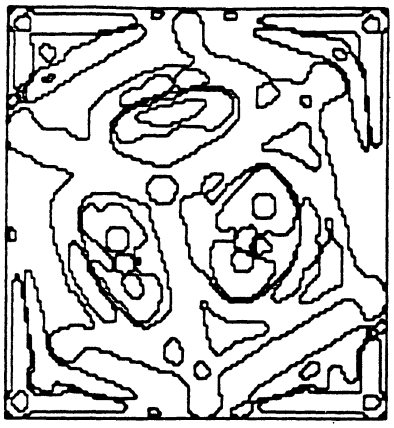
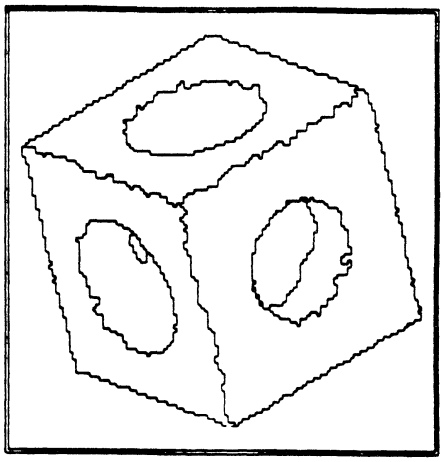


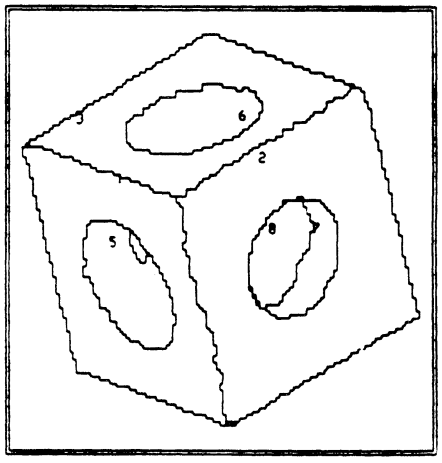
Figure 7.12.1.(b) Standard Results for Block (Low-Noise)



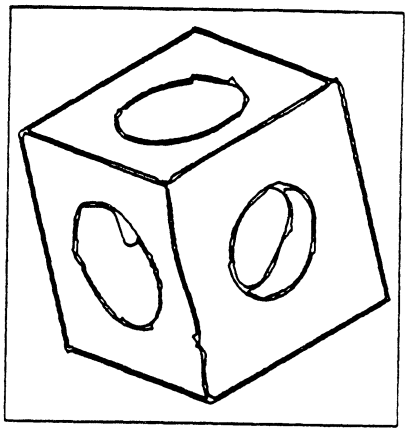
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

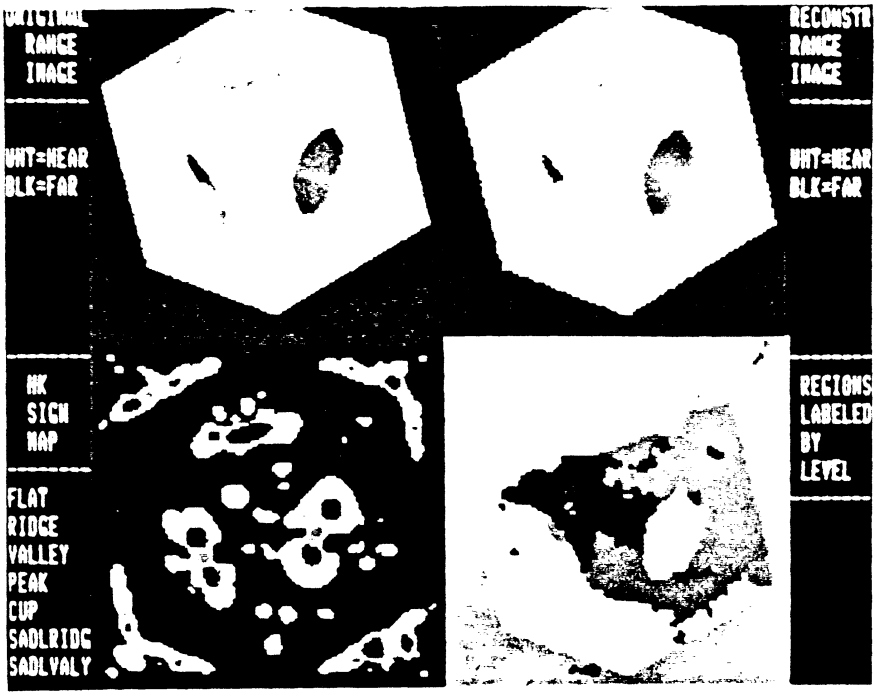


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.12.2. Standard Results for Block (Low-Noise)



(a) Standard Results for Block (High-Noise)

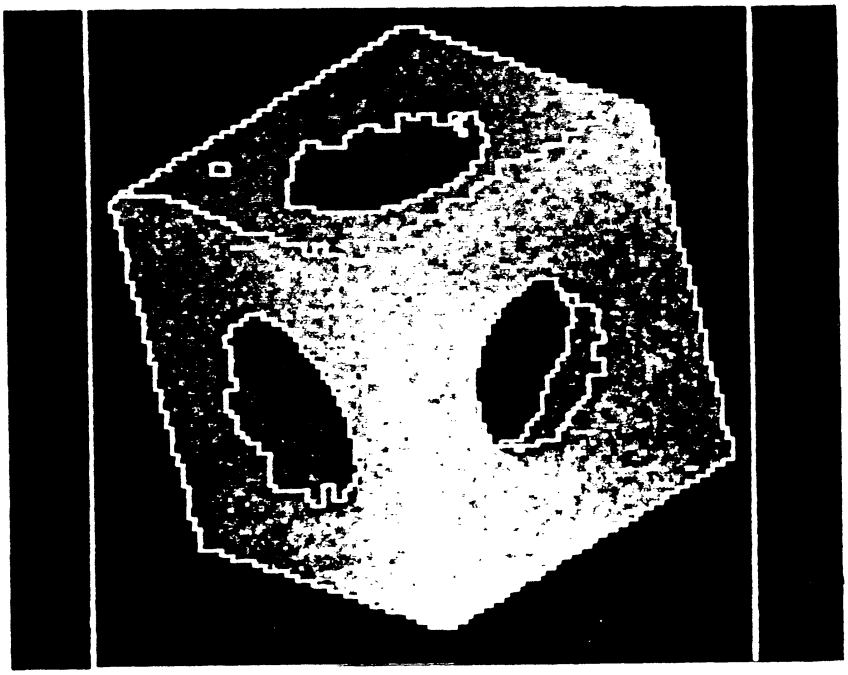
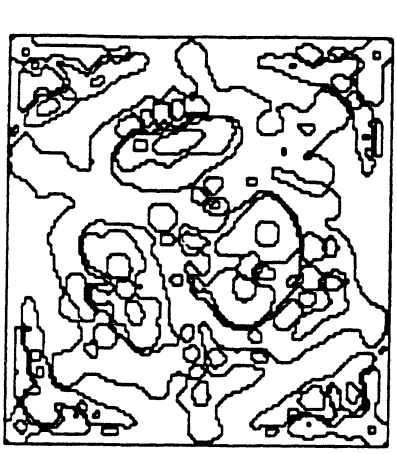
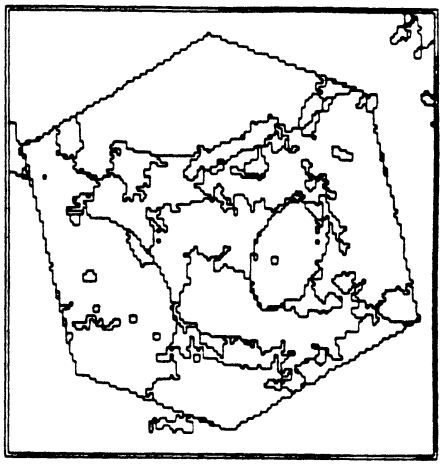


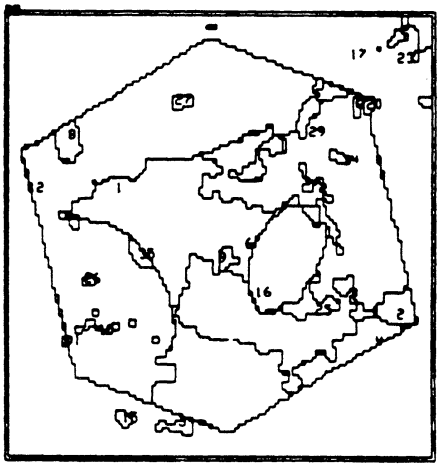
Figure 7.13.1.(b) Modified Threshold Results for Block (High-Noise)



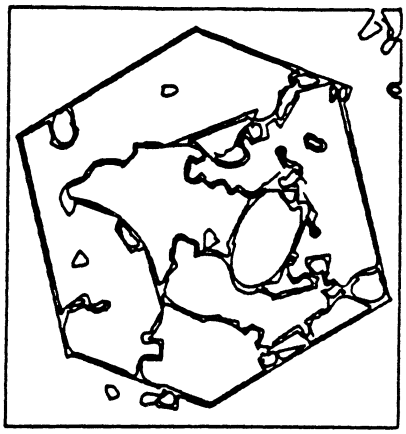
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

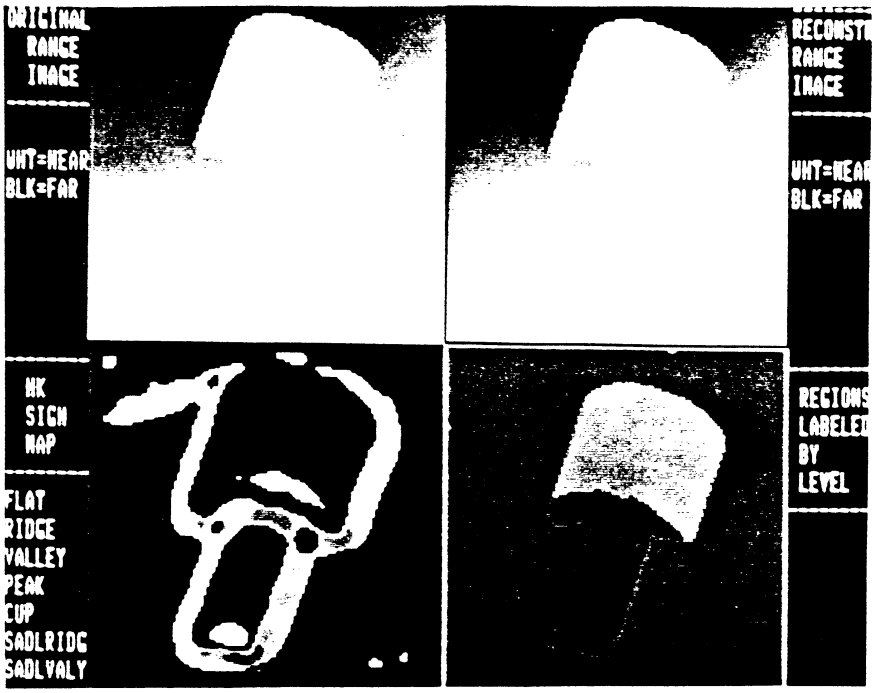


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.13.2. Standard Results for Block (High-Noise)



(a) Standard Results for Two Cylinders

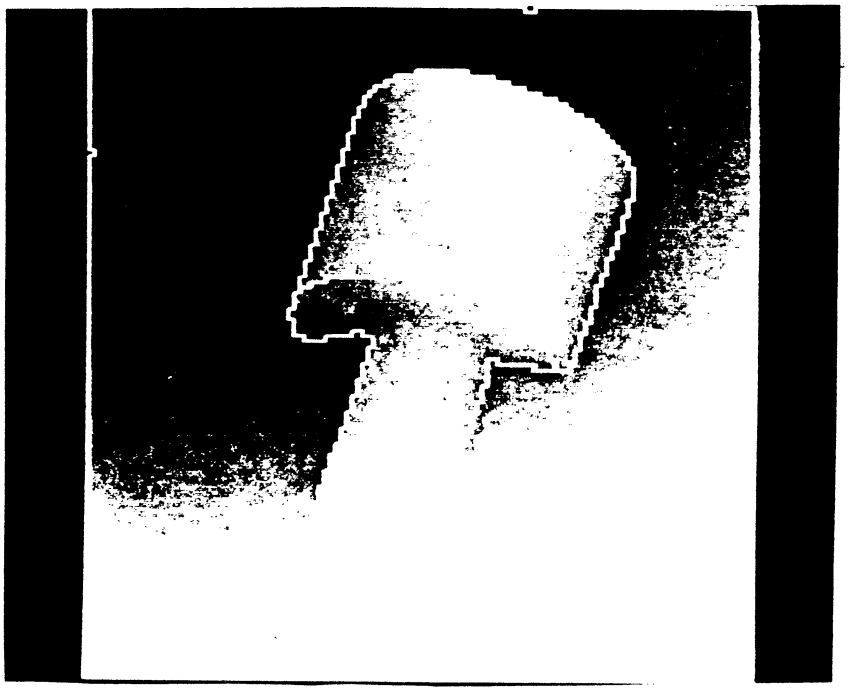
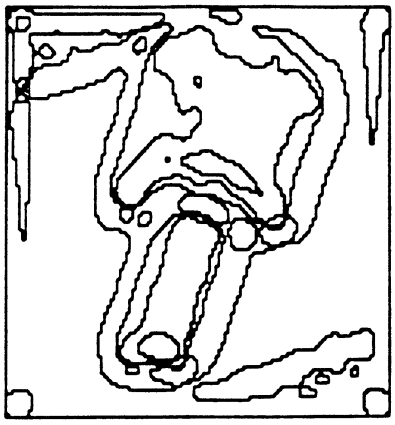
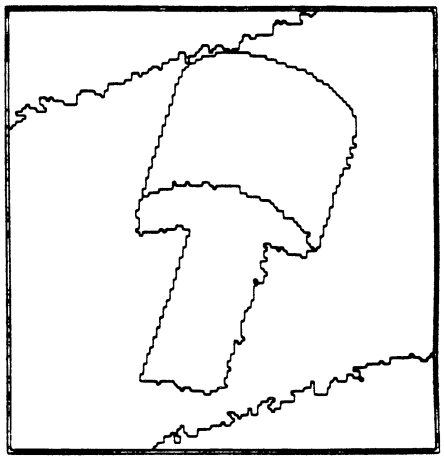


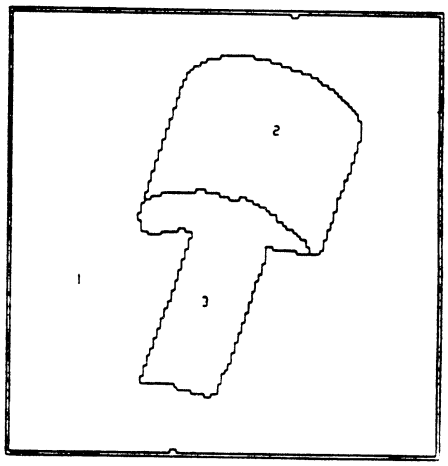
Figure 7.14.1.(b) Standard Results for Two Cylinders



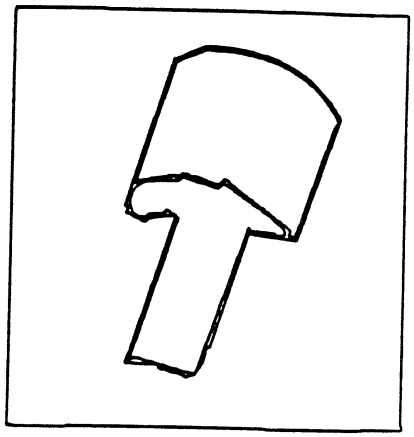
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

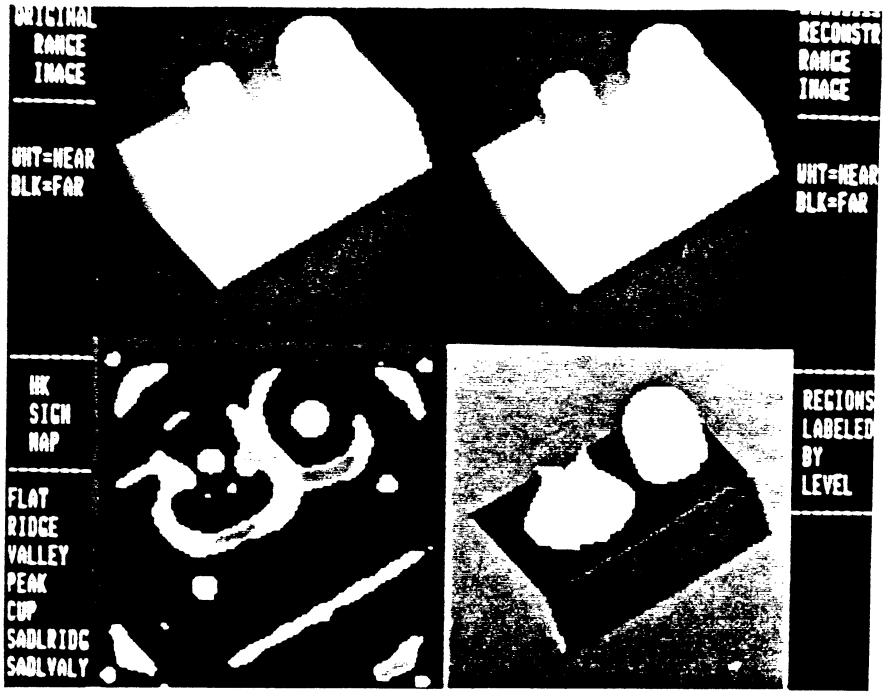


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.14.2. Standard Results for Two Cylinders



(a) Standard Results for Object with Protrusions

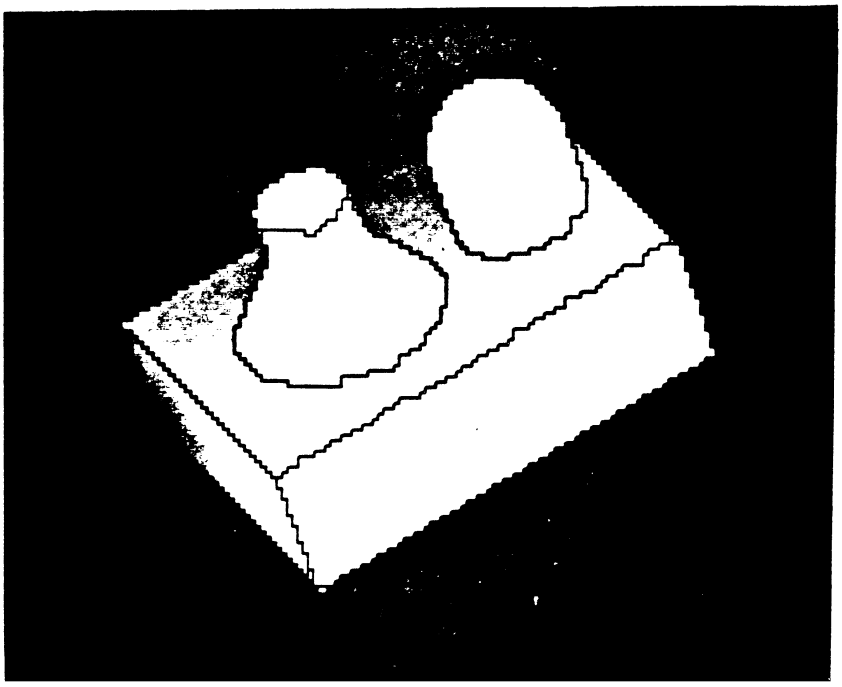
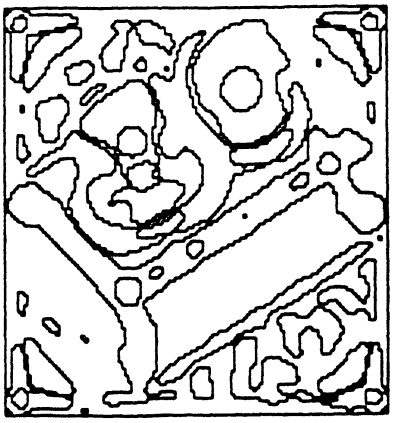
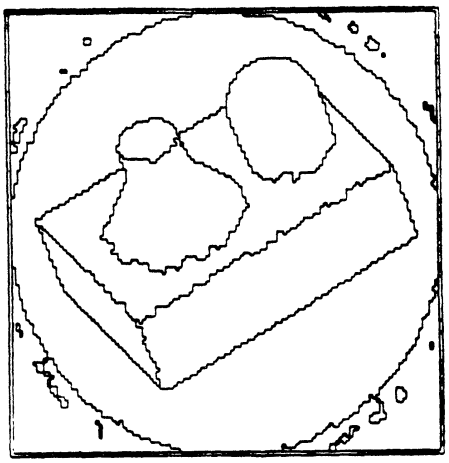


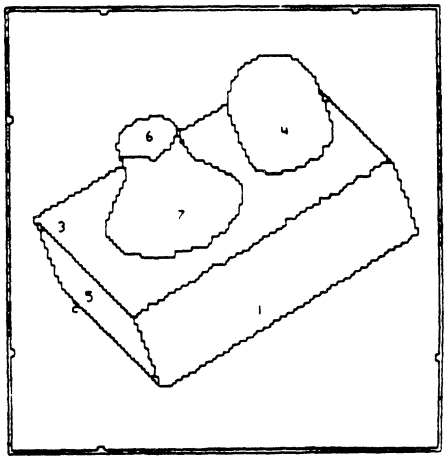
Figure 7.15.1.(b) Standard Results for Object with Protrusions



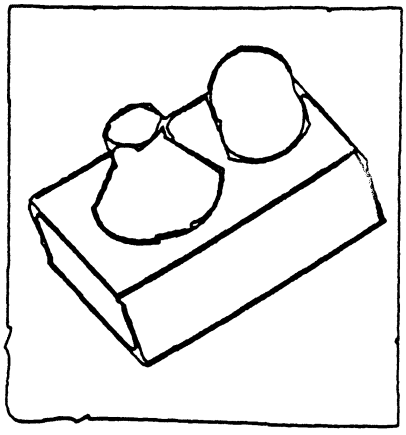
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

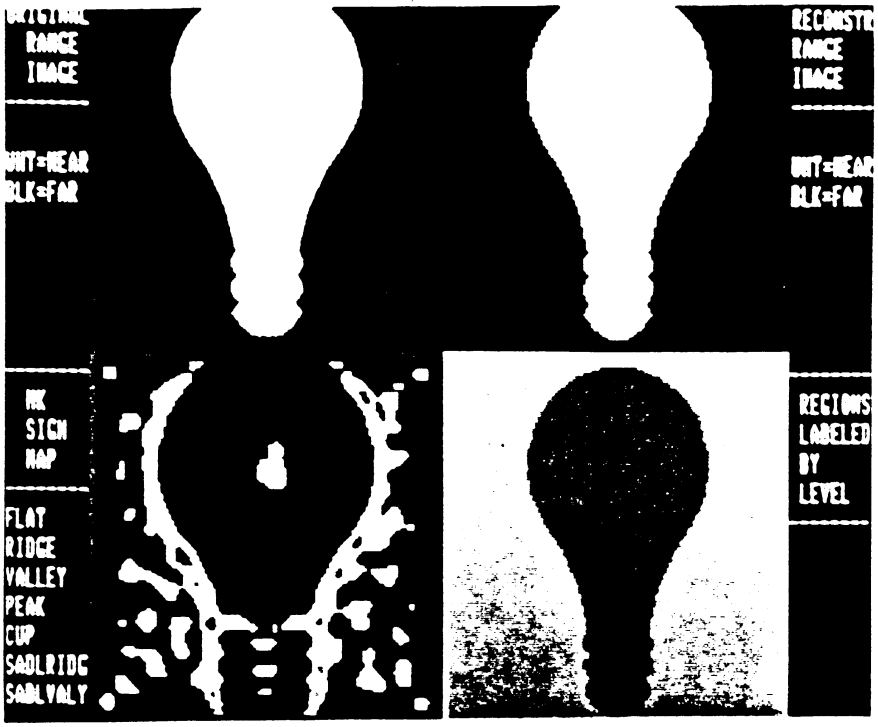


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.15.2. Standard Results for Object with Protrusions



(a) Standard Results for Light Bulb

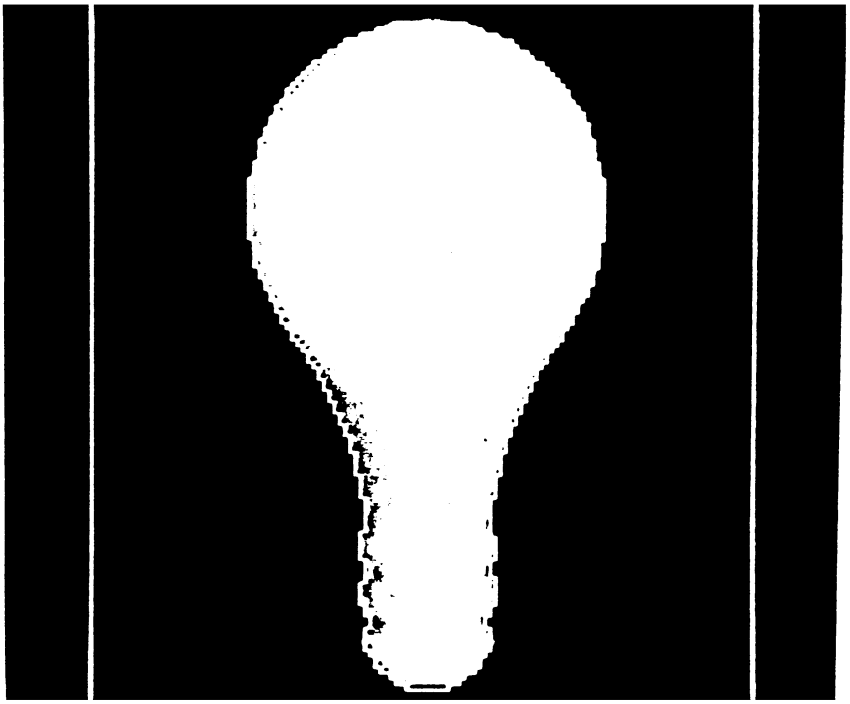
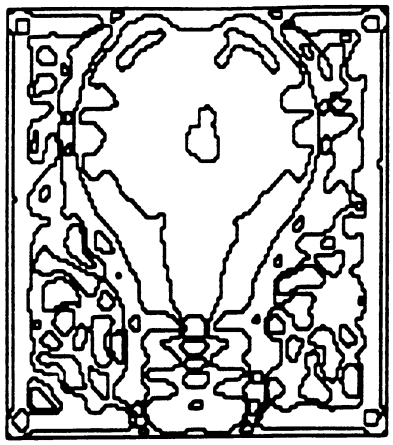
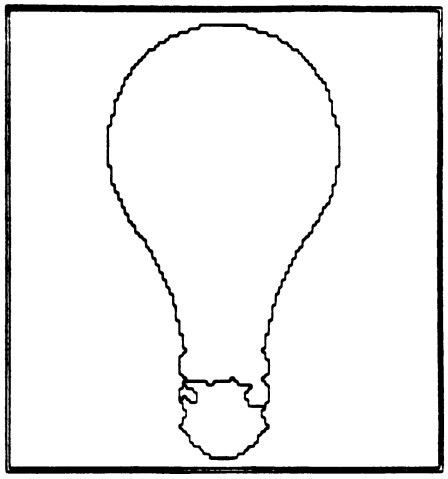


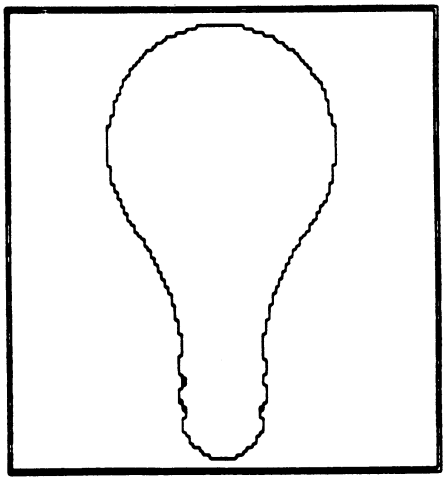
Figure 7.16.1.(b) Standard Results for Light Bulb



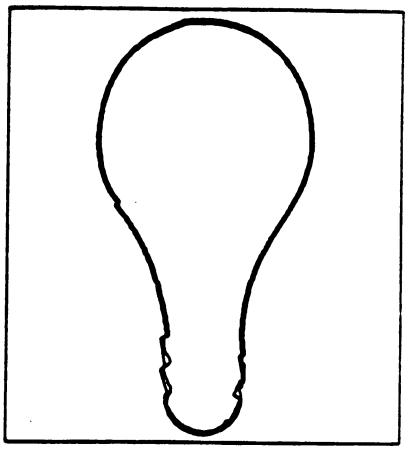
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

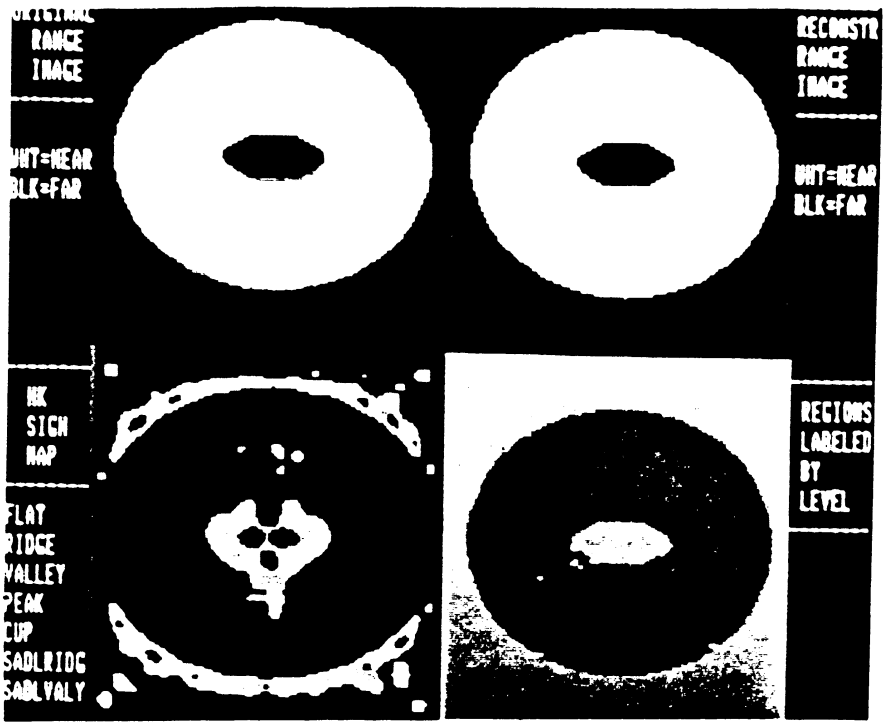


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.16.2. Standard Results for Light Bulb



(a) Standard Results for Torus

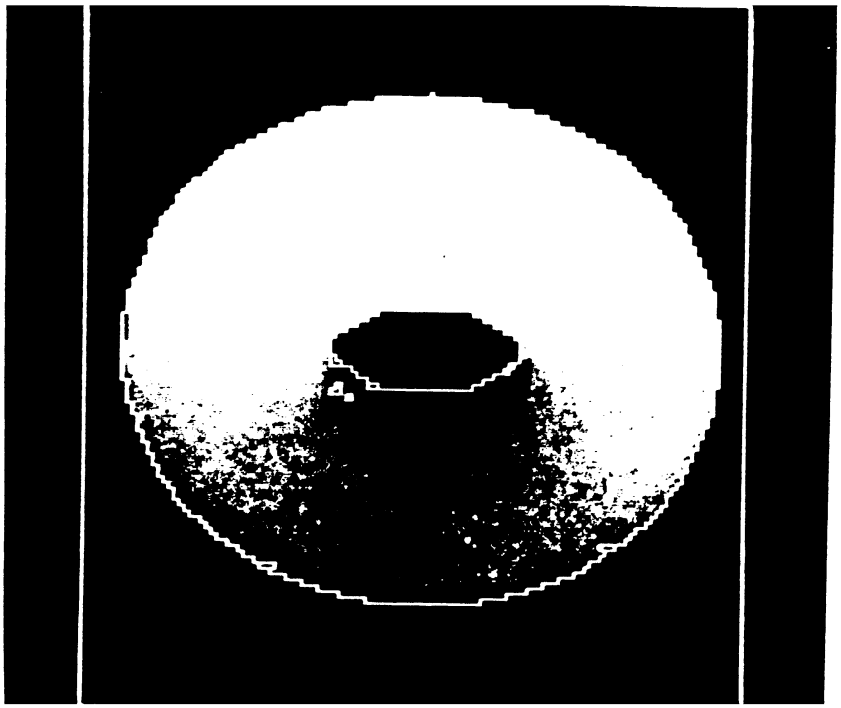
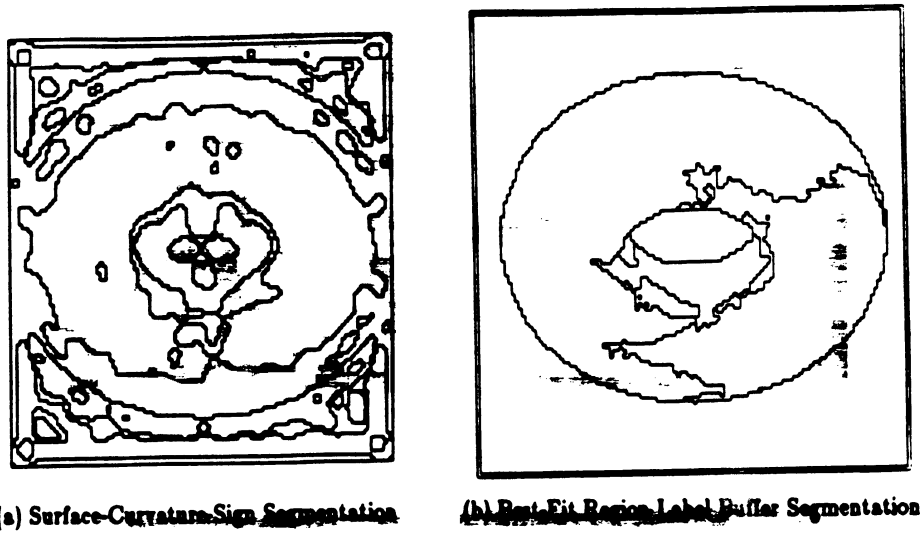
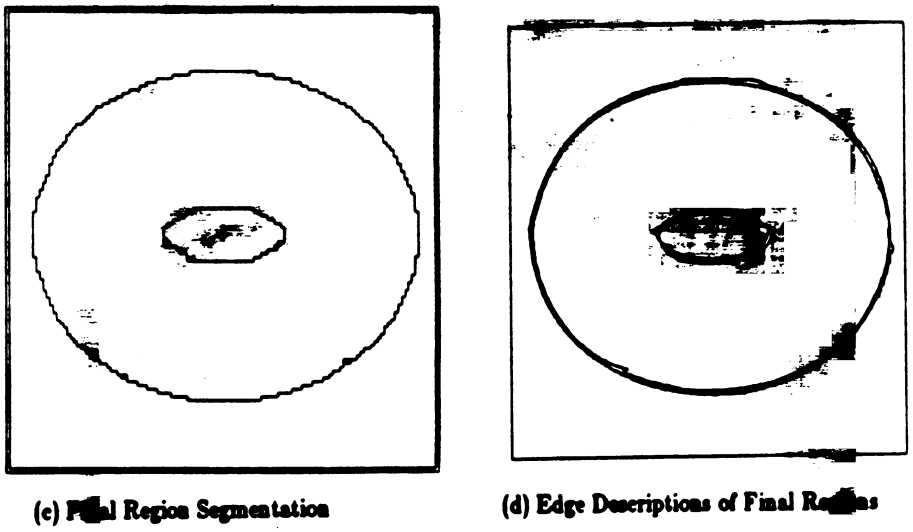


Figure 7.17.1.(b) Standard Results for Torus



(a) Surface-Curvature-Sign Segmentation

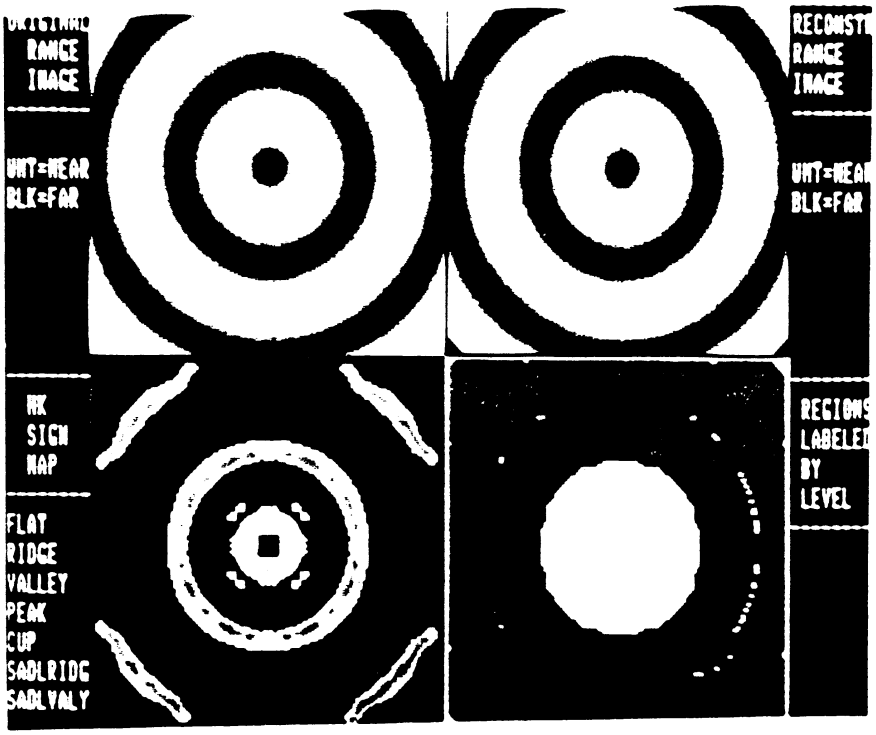
(b) Best-Fit Region-Label Buffer Segmentation



(c) Final Region Segmentation

(d) Edge Descriptions of Final Regions

Figure 7.17.2. Standard Results for Torus



(a) Standard Results for Circular Waves

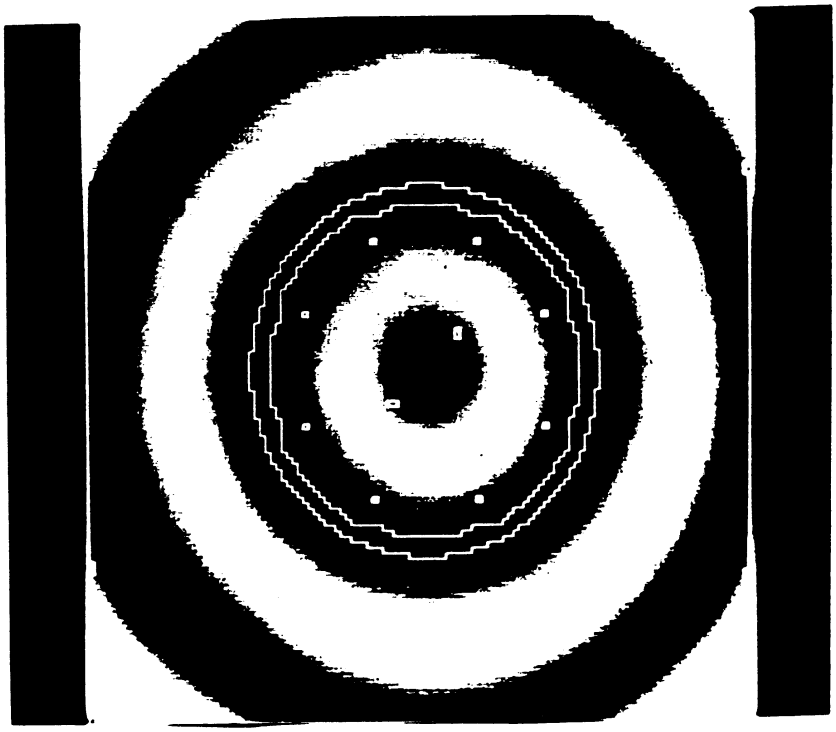
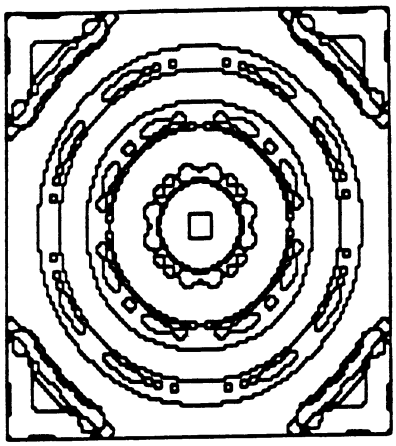
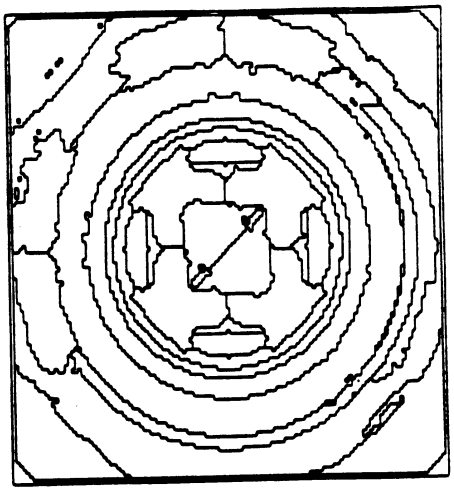


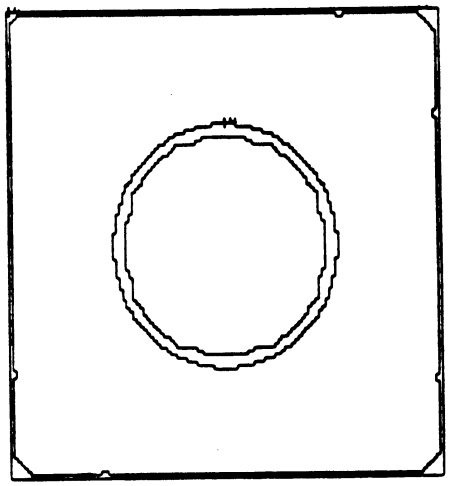
Figure 7.18.1.(b) Standard Results for Circular Waves



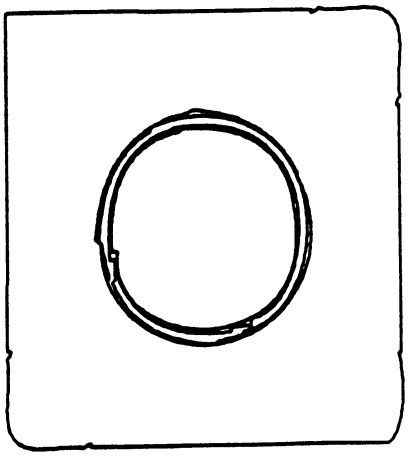
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

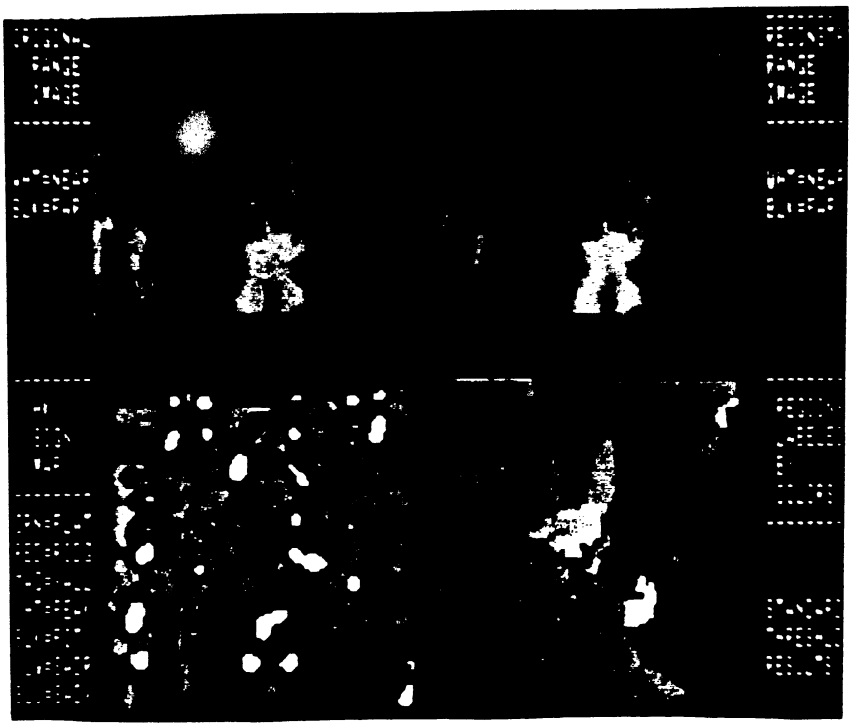


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.18.2. Standard Results for Circular Waves



(a) Standard Results for Space Shuttle

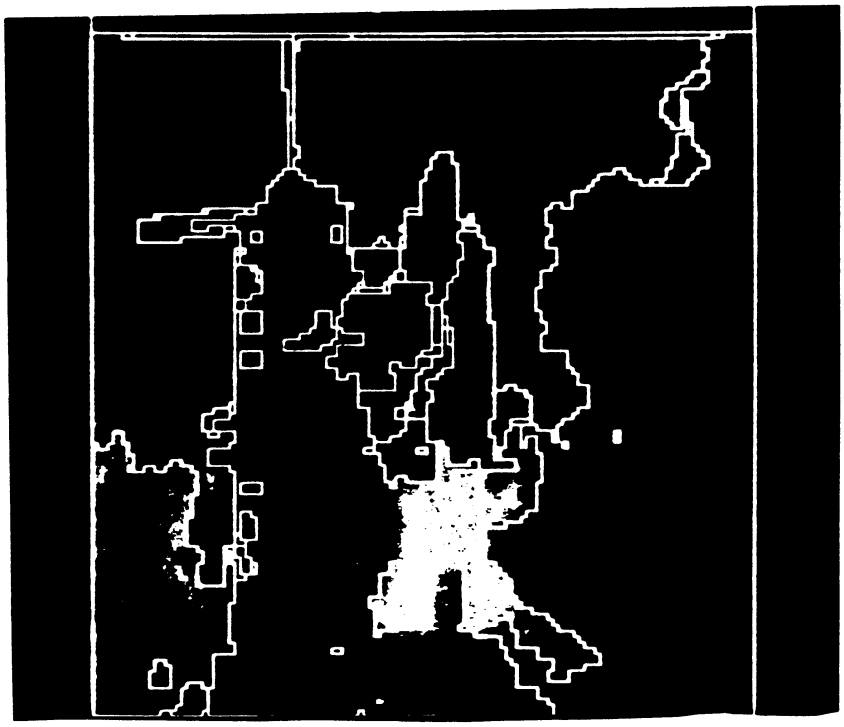
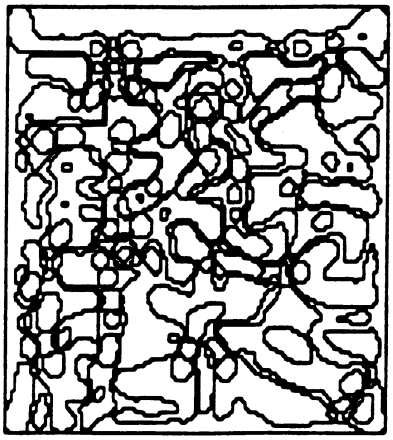
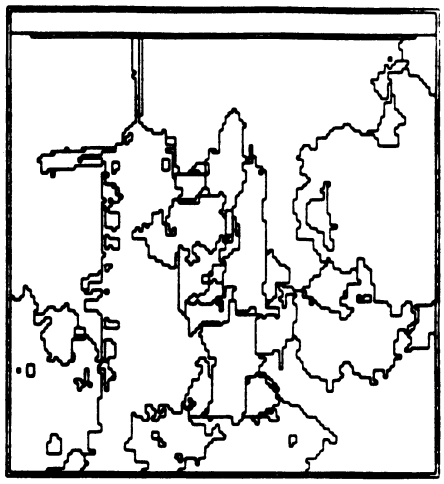


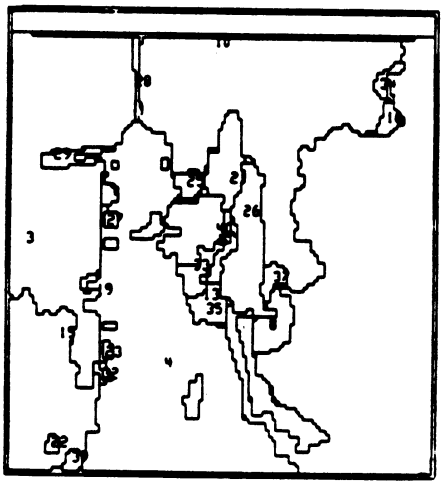
Figure 7.19.1.(b) Standard Results for Space Shuttle



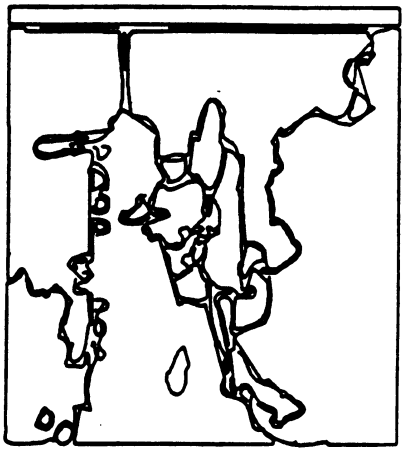
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

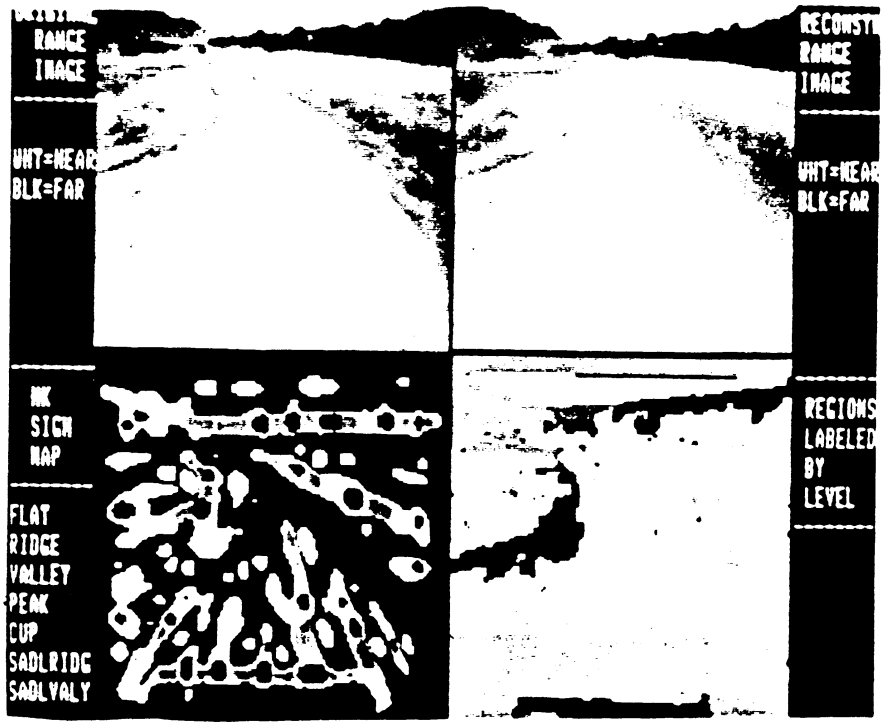


(c) Final Region Segmentation



(d) Edge Descriptions of Final Regions

Figure 7.19.2. Standard Results for Space Shuttle



(a) Standard Results for Road Image

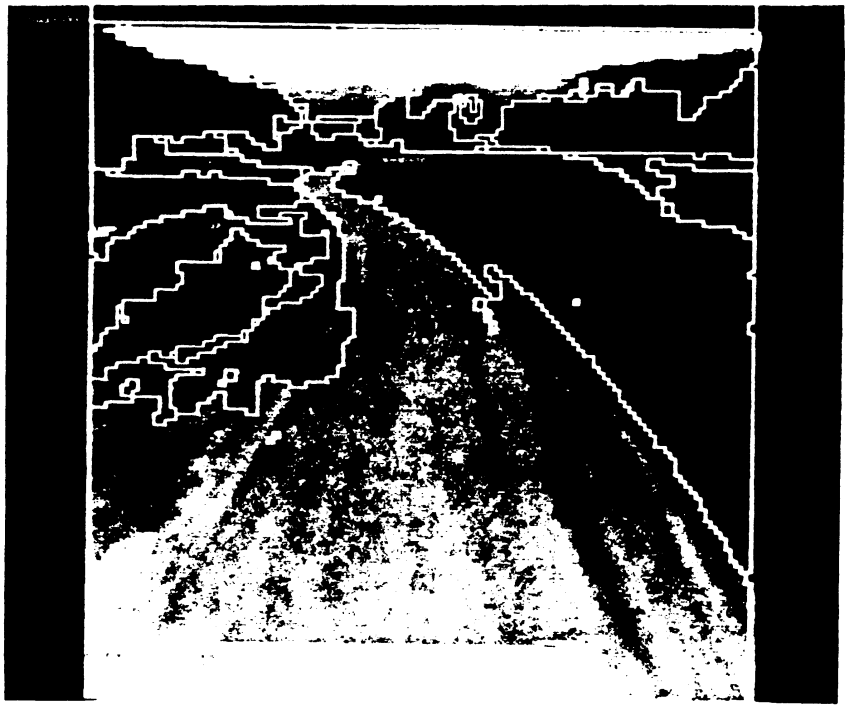
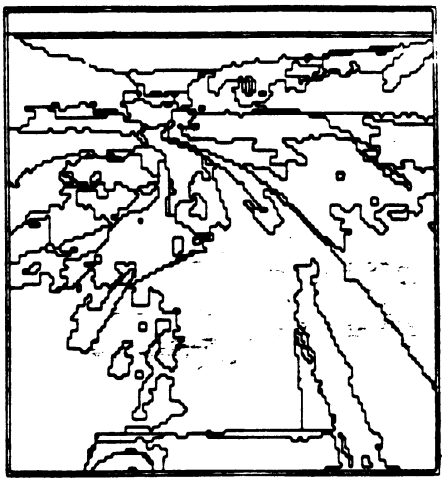
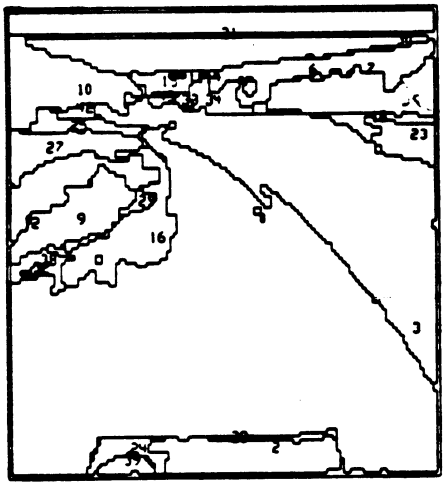


Figure 7.20.1.(b) Standard Results for Road Image

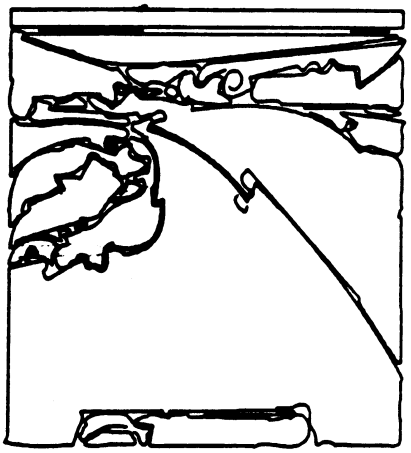
(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation



(c) Final Region Segmentation

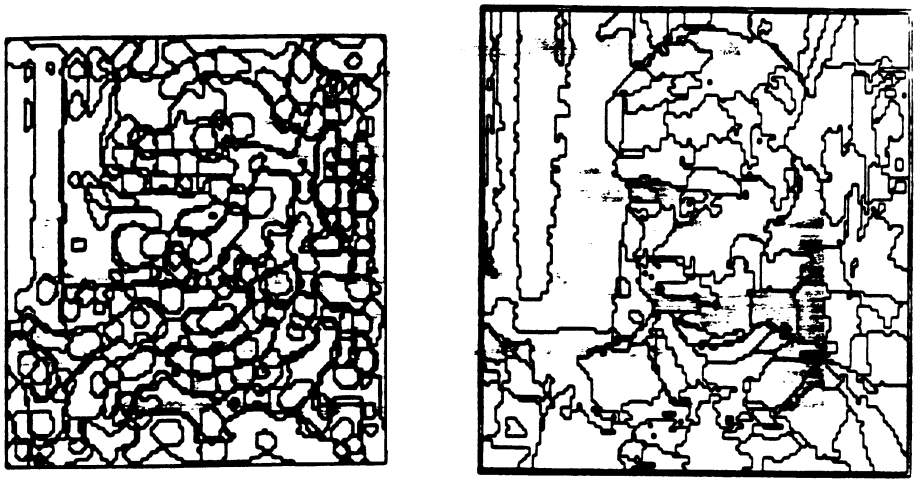


(d) Edge Descriptions of Final Regions

Figure 7.20.2. Standard Results for Road Image

(a) Standard Results for USC Girl

Figure 7.21.1.(b) Standard Results for USC Girl



(a) Surface-Curvature-Sign Segmentation

(b) Best-Fit Region Label Buffer Segmentation

(c) Final Segmentation

(d) Edge Descriptions of Final Regions

Figure 7.21.2. Standard Results for USC Girl

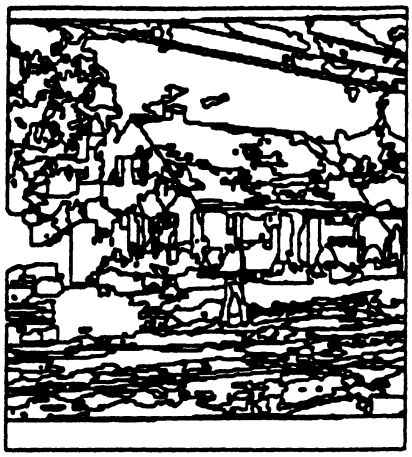
Figure 7.21.3. Modified Algorithm Reconstruction for USC Girl
(Aggressive Error Tolerance Increase Function)

Figure 7.21.4. Modified Algorithm Segmentation for USC Girl
(Aggressive Error Tolerance Increase Function)

(a) Original Intensity Image of U. Mass. House (256x256)

Figure 7.22.1.(b) Standard Reconstruction for U. Mass. House

(a) Surface-Curvature-Sign Segmentation



(b) Best-Fit Region Label Buffer Segmentation

(c) Final Region Segmentation

Figure 7.22.2. Standard Results for U. Mass. House

Figure 7.22.3.(a) Standard Segmentation Results for House

Figure 7.22.3.(b) Modified Threshold Segmentation for House

Figure 7.22.4.(a) Modified Threshold Results for House

Figure 7.22.4.(b) Modified Threshold Reconstruction for House

CHAPTER 8

SUMMARY AND FUTURE DIRECTIONS

The perception of surfaces plays a key role in 3-D object recognition. Recognition of 3-D objects from 2-D images is performed by matching *perceived surface descriptions* with stored models of objects. The first step in general-purpose object recognition is the *segmentation* of a digital image into image regions that correspond to physical scene surfaces. If the pixels of a digital surface can be correctly grouped into symbolic surface descriptions that have meaningful interpretations based only on general knowledge of surfaces, this grouping process would provide a fundamental service to higher level matching (or recognition) processes. An algorithm for segmenting images into symbolic surface descriptions has been presented in this thesis. This approach may be considered *stimulus bound* in that it considers that the ultimate truth is the sensed image data, and hence the output of all processes should conform to the sensed values. Range images offer a simpler domain to study grouping processes for segmentation and scene analysis than intensity images because the depth values at each pixel depend only on geometry, not on reflectance and illumination. Visual image domain cues are actually physical scene domain cues in the range image context. Since the problems of range image segmentation are also encountered in the more difficult intensity image segmentation, an effective solution to grouping problems will be extremely useful to both areas. The experimental results of Chapter 7 indicate excellent segmentation performance on range

images and show that techniques for range image segmentation may be successfully applied to intensity images. The success of the algorithm on both range and intensity images is due to the generality of the underlying digital surface model, which makes no assumptions about the meaning of pixel values.

The surface-based approach to segmentation and early image understanding is very promising. In the past, many image analysis and segmentation techniques have focussed on *discontinuity (edge) detection*. Although such techniques are worthwhile and although good research in region growing has been done, the alternative dual approach of *continuity detection* has not received equal attention. If one considers only the amount of pixel evidence available for supporting various hypotheses about detected entities, the number of pixels representing continuity or spatial coherence in an image is typically much larger than the number of pixels representing discontinuity or spatial variation. One alternative to compressing image data into a small set of edge pixels is to represent large smooth surface primitives in an image using a small set of functional parameters. This thesis has stressed the power of continuity detection in the surface-based segmentation approach and has (1) provided a theoretical framework for object recognition and image segmentation, (2) devised algorithms for image segmentation based on object recognition requirements, and (3) demonstrated a successful software implementation via experimental results on real data.

The theoretical framework for range image understanding is summarized, and the fundamental concepts are italicized. Object recognition in range images is a *generalized inverse set mapping* that maps a range image into a set of compatible image interpretations based on world knowledge. The largest primitive element in a range image that is independent of object information is a *smooth surface primitive*. The *surface coherence*

predicate is a logical predicate for stating the range image segmentation problem in the standard form [Horowitz and Pavlidis 1974] [Zucker 1976]. A signal plus noise model is proposed for *piecewise smooth digital surfaces* that allows a very specific statement of the segmentation problem in terms of image regions and functional approximations. Preliminary grouping for segmentation is based on concepts from the differential geometry of surfaces and curves. Just as curvature and speed characteristics uniquely specify the shape of 1-D smooth planar curves in 2-D Euclidean space, the *Weingarten mapping and the metric tensor* uniquely specify the shape of 2-D smooth surfaces in 3-D Euclidean space. *Visible-invariant surface characteristics* are quantities relevant to 3-D object recognition that are invariant to rotations, translations, and changes in parameterization. *Mean and Gaussian curvature* are visible-invariant surface characteristics based on the Weingarten mapping and the metric tensor. The *sign-of-curvature paradigm* states that connected groups of pixels with constant mean and Gaussian curvature sign provide an initial segmentation of a range image that has visible-invariant properties. Many techniques attempt to characterize shape using local extrema of curvature (e.g. [Asada and Brady 1986]); such approaches must reliably compute curvature magnitude correctly, which is difficult. The sign-of-curvature paradigm only requires that the sign of the curvature function is correctly computed for most of the pixels in a region; therefore, the approach is insensitive to noise. The initial segmentation provides input to the *iterative, parallel region-growing algorithm based on variable-order surface-fitting.* This process is insensitive to grouping errors in the initial segmentation due to the *seed region extraction process* and the iterative surface-fitting technique, which shares common ideas with the RANSAC method [Bolles and Fischler 1981]. A single final region-merging step joins adjacent surface primitives based on geometric considerations.

Several capabilities and properties of the surface-based segmentation algorithm are summarized.

(1) The sign-of-curvature approach is unified, very general, and independent of geometric dimension. Flat surfaces are described explicitly as being flat, and arbitrary curved surfaces are described as being curved within the context of the same algorithm. Straight lines are described explicitly as being straight, and arbitrary curved edges are described as being curved. The method is also generalizable to higher dimensions. No *a priori* assumptions about convexity, symmetry, or object shape are used.

(2) The symbolic description is driven by the content of the data, not by expected high-level models as is done in many other approaches as discussed in the literature survey of Chapter 1. Moreover, the exact same surface algorithm with the exact same set of thresholds is shown to be useful for segmenting and describing range images and intensity images. The exact same edge algorithm with the exact same set of thresholds is shown to be useful for segmenting and describing edges in a wide variety of images.

(3) The algorithm allows for arbitrary four-connected image regions, which are determined by the data during region growth. This is superior to the blocky regions that result from fixed-window-based segmentation methods.

(4) The parallel region-growing allows logical association between non-adjacent, non-connecting regions that are part of the same physical surface without high-level knowledge. Isolated regions were given the same label as another surface in the coffee cup B image and the keyboard B image in Chapter 7.

(5) The final segmentation output is a rich, information-preserving data-driven description of an image. It allows reconstruction of a noise-cleaned image that has almost all of the structural properties of the original image depending on the amount of surface coherence in the data. Depth discontinuities and orientation discontinuities between adjacent smooth surface primitives are enhanced in the final data description. This description could be referred to as a "surface primal sketch" in the terminology of Brady [Brady et al. 1985][Ponce and Brady 1985].

(6) Image quality and image surface coherence can be roughly estimated by a simple process at the first stage of the surface-based segmentation algorithm. The image quality measure predicts the success or failure of the standard set of input thresholds and suggests more appropriate threshold levels.

No other known algorithm provides all the capabilities mentioned above in a theoretically unified manner.

The final data description of the surface-based algorithm contains a list of smooth surface primitives. Each smooth surface primitive consists of one or more polynomial surface primitives and has a support region bounded by edges that consist of smooth edge-interval primitives. Each polynomial surface primitive is defined by at most fifteen coefficients over a given sub-region contained in the support region of the smooth surface primitive. The given sub-region may be represented either by a run-length code or by the bounding edges. The surface fit errors (mean absolute, RMS, maximum) are also given for each polynomial surface primitive to indicate the quality of the approximation over the sub-region. Each smooth edge-interval primitive consists of one or more vector polynomial edge-interval primitives. Vector polynomial edge-interval primitives are represented by at most ten coefficients, and each associated sub-interval of the smooth

edge-interval is represented by its endpoints. The edge-interval fit errors (mean absolute, RMS, maximum) may also be given for each vector polynomial edge-interval primitive to indicate the quality of the approximation of the sub-interval. This type of representation may be considered as symbolic, geometric data compression as opposed to other types of data compression based on statistics or differential codes. The primary goal is not a reduction in the number of bits needed to represent the image even though that may occur, but rather an extraction of potentially meaningful symbolic surface region primitives.

Differential-geometric surface characterization provides an initial segmentation (or pixel grouping) for the region-growing algorithm based on local differences in the data, but it is also interesting in its own right. Although the Gaussian curvature function has received the most attention in previous literature, the mean curvature function is the most interesting quantity discussed in Chapter 3. The mean curvature uniqueness theorem states that the shape of graph surfaces is captured by the mean curvature function and a boundary curve. The zero-crossings-of-the-mean-curvature operator acts as an isotropic step-edge detector very similar to the zero-crossings-of-the-Laplacian operator. The mean curvature and Laplacian operators are both second-order elliptic differential operators, but are quasilinear and linear respectively. A local-maxima-of-the-mean-curvature operator acts as a roof-edge detector. It is noted also that the metric-determinant-square-root operator acts like a Sobel edge detector, but it is also useful for estimating the surface area of image regions. A Gaussian curvature operator acts as a corner or surface-kink detector. Hence, the surface curvature characteristics for region-growing are also useful as edge and corner detectors.

The *stimulus bound* philosophy is an important aspect of the surface-based segmentation algorithm. Most decisions made by the algorithm are checked against the original image data for verification. Each module anticipates the possibility of poor information from the lower-level module, and attempts to adjust its decisions based on feedback from the original data.

The dichotomies of Section 1.2 are summarized below. The input signals to the algorithm are digital surfaces, which may be intensity images or range images. The output symbols are the smooth surface primitives and their associated geometric data. The segmentation algorithm is data-driven, but model-driven processes will be needed for object recognition and image understanding. This early (low-level) process creates a rich, surface-based description that is useful to later (higher-level) processes. Local difference information is computed via partial derivative estimation and curvature computations and is used to provide global similarity information via region growing based on the original data. Grouping errors are reduced via the seed region extraction process, and measurement errors are handled via iterative surface fitting. The experimental results in Chapter 7 show that it is *possible to extract* from an image those entities that are *desirable to compute* from an image without high-level object knowledge.

Future research is needed in many areas related to the topics presented in this thesis. It is necessary to determine an effective matching algorithm and modeling scheme that will enable object recognition given the output from the segmentation algorithm and the original image. The segmentation output provides enough information that arbitrary non-convex objects ought to be recognizable with an appropriate matching algorithm. Research in qualitative and quantitative world model representations may be influenced by the availability of symbolic (polynomial) surface primitives.

Future research is also necessary to refine the segmentation algorithm. The current algorithm is sequential and, in certain cases, the order dependency of the region processing (large regions to small regions) adversely affects final region descriptions despite attempts to avoid it by the use of the image error buffer and region label buffers. If a proper parallel processing approach can be developed, there would be no such order dependency. The current algorithm does not commit to a particular surface interpretation until the iterative region-growing process terminates and the surface is accepted as described in Chapter 4. In the facet model iteration of Haralick and Watson [1981], their algorithm commits to a particular surface interpretation almost immediately. As discussed in Chapter 7, small orientation discontinuities are not easily detected in the presence of noise using the current algorithm. This roof-edge detection failure may be occurring because the surface-based algorithm never commits to an interpretation until the end of the iteration. This allows the noise in the pixel data to influence the surface fitting process all during the region-growing iteration even though it is highly likely that, in the absence of noise, many pixels participating in the surface fitting process lie almost exactly on the surface being fitted. It seems possible that a *gradual commitment* to a surface interpretation may be able to solve the problems encountered with small orientation discontinuities in the presence of noise.

In addition, the region merging algorithm may require more research. The algorithm in Chapter 5 is extremely simple and is susceptible to errors. For intensity image segmentation, it will be especially important to develop clean and flexible ways to incorporate specific domain knowledge so that the final region descriptions are representative of physical scene surfaces.

Sensor integration is also a fruitful area for future research. Range images, intensity images, thermal images, and tactile images are all digital surfaces that can be processed using the surface-based segmentation algorithm. Multi-sensor integration of the individual outputs will increase the reliability and accuracy of the multiple image interpretation.

The sign-of-curvature paradigm can be applied to 3-D volumes embedded in 4-D space, or to similar entities in higher dimensional spaces. A sign-of-curvature algorithm is potentially useful for segmenting 3-D density images, such as those obtained from CAT scan sensors, or dynamic scenes (image sequences).

In Chapter 2, the notions of quantitative segmentation and recognition error metrics were introduced. These preliminary ideas will be very useful for system evaluation if they are refined and applied to several different segmentation programs and object recognition systems.

If surface curvature estimates can be improved, the initial HK-sign map segmentation will be improved, which could reduce the "effort" required by the iterative region-growing algorithm. Signal processing research may be able to discover better methods of estimating partial derivatives from digital surface data. Performance analysis for the HK-sign map pixel labeling method should be included in such research to determine the accuracy of the pixel labels in different amounts of noise. Range finders might be improved to yield better depth resolution; this would improve surface curvature estimates even using the existing approach.

Future research might also be directed toward a post-processing analysis of the highest-order (fourth-order) surface functions (based on the regions test) that will indicate the accuracy of the highest order fit. If the regions test indicates that large regions

of the fitted surface lie above or below the data, the surface region may need to be subdivided into smaller surface primitives. A complete evaluation of the applicability of other surface function types is another related research area.

For real-time applications, least squares computations might be substantially reduced by using an incremental update QR algorithm or an orthogonal multinomial approach as discussed in Appendix C.

It has been shown that the segmentation of range images into scene surfaces can be data-driven and need not involve higher level knowledge of objects. Range image object recognition systems will be much more flexible if this type of segmentation algorithm is used. The perceptual organization capabilities of the surface-based range image segmentation algorithm are also worthwhile capabilities for intensity image segmentation as is shown via experimental results. More research in the area of shape from shading is needed to determine how higher level knowledge should be used in relating intensity-image smooth-surface primitives to real scene surfaces.

APPENDICES

APPENDIX A

INVARIANCE PROOFS FOR SURFACE CURVATURE

The rotation-translation and parameterization invariance properties of the mean and Gaussian curvature of a smooth surface are discussed in Chapter 3 without proof. Since these properties are the most important aspects of visible-invariant features and the direct proofs of these invariance properties are brief and straightforward, they are included here for reference. The two basic theorems as presented here were not found explicitly in texts on differential geometry, but were assembled from proofs of other theorems found in Lipschutz [1969] and O'Neill [1966].

THEOREM A.1: Mean curvature $H(u,v)$ and Gaussian curvature $K(u,v)$ are invariant to rotations and translations of the smooth surface represented by $\vec{x}(u,v)$ where $(u,v) \in D \subseteq \mathbf{R}^2$.

Proof: Let $\vec{x}'(u,v)$ be a rotated, translated version of the surface $\vec{x}(u,v)$. Hence, we can write

$$\vec{x}'(u,v) = \mathbf{U}\,\vec{x}(u,v) + \vec{t} \tag{A.1}$$

where the matrix $\mathbf{U}$ is a fixed pure rotation matrix

$$\mathbf{U}\mathbf{U}^T = \mathbf{U}^T\mathbf{U} = \mathbf{I} = \textit{Identity Matrix} \tag{A.2}$$

and $\vec{t}$ is a fixed translation vector. Since the rotation and translation are both fixed, the partial derivatives of the surface parameterization may be written as

$$\vec{x}_u'(u,v) = \mathbf{U}\,\vec{x}_u(u,v) \tag{A.3}$$

$$\vec{x}_v'(u,v) = \mathbf{U}\,\vec{x}_v(u,v)$$

$$\vec{x}_{uu}'(u,v) = \mathbf{U}\,\vec{x}_{uu}(u,v)$$

$$\vec{x}_{vv}'(u,v) = \mathbf{U}\,\vec{x}_{vv}(u,v)$$

$$\vec{x}_{uv}'(u,v) = \mathbf{U}\,\vec{x}_{uv}(u,v)$$

The first fundamental form coefficient functions are defined via inner products of the first partial derivative vectors.

$$\begin{aligned} E' &= (\vec{x}_u'(u,v))^T \vec{x}_u'(u,v) \\ &= \vec{x}_u^T(u,v)\mathbf{U}^T\mathbf{U}\,\vec{x}_u(u,v) \\ &= \vec{x}_u^T(u,v)\,\vec{x}_u(u,v) = E \end{aligned} \tag{A.4}$$

Similarly, $F = F'$ and $G = G'$. Hence, the E,F,G functions (the metric tensor components) are invariant to rotations and translations.

The proof for the second fundmental form coefficients is not much different. The unit normal to the surface at each point is defined as

$$\vec{n}(u,v) = \frac{\vec{x}_u \times \vec{x}_v}{|\,\vec{x}_u \times \vec{x}_v\,|} = \textit{Unit Normal Vector}\ . \tag{A.5}$$

Since pure rotation (orthogonal) matrices preserve lengths and relative angles of vectors, we have

$$\vec{x}_u' \times \vec{x}_v' = (\mathbf{U}\,\vec{x}_u) \times (\mathbf{U}\,\vec{x}_v) = \mathbf{U}\,(\vec{x}_u \times \vec{x}_v)\,, \tag{A.6}$$

which directly implies that

$$\vec{n}'(u,v) = \mathbf{U}\,\vec{n}(u,v)\,. \tag{A.7}$$

The second fundamental form coefficient functions are defined via inner products of the second partial derivative vectors and the surface normal.

$$\begin{aligned} L' &= (\vec{n}'(u,v))^T \vec{x}_{uu}'(u,v) \\ &= \vec{n}^T(u,v)\mathbf{U}^T\mathbf{U}\,\vec{x}_{uu}(u,v) \\ &= \vec{n}^T(u,v)\,\vec{x}_{uu}(u,v) = L \end{aligned} \tag{A.8}$$

Similarly, $M = M'$ and $N = N'$. Hence, the L,M,N functions are also invariant to rotations and translations. Since H and K depend only on the six E,F,G,L,M,N functions as expressed in equations (3.38) and (3.39) in Chapter 3, mean and Gaussian curvature are invariant to arbitrary 3-D rotations and 3-D translations. •

Mean and Gaussian curvature are also invariant to changes in the parameterization of a smooth surface. This type of invariance is equally important for visible-invariants because of the implicit reparameterization of surfaces in sampled depth map projections from different views. The invariance property is stated as a theorem.

THEOREM A.2: Mean curvature $H(u,v)$ and Gaussian curvature $K(u,v)$ are invariant to changes in parameterization of the smooth surface represented by $\vec{x}(u,v)$ where $(u,v) \in D \subseteq \mathbf{R}^2$ as long as the Jacobian of the parameter transformation is non-zero.

Proof: Suppose that there exist two parameters, (η,ξ), and two differentiable functions, $u = u(\eta,\xi)$ and $v = v(\eta,\xi)$, such that the composite surface parameterization $\vec{x}'(\eta,\xi)$ over the domain $D' \subseteq \mathbf{R}^2$ describes exactly the same surface as $\vec{x}(u,v)$ over the domain D. Using the chain rule, we can write expressions for the partial derivative vectors:

$$\begin{aligned} \vec{x}_u'(\eta,\xi) &= \vec{x}_\eta' \eta_u + \vec{x}_\xi' \xi_u \\ \vec{x}_v'(\eta,\xi) &= \vec{x}_\eta' \eta_v + \vec{x}_\xi' \xi_v \end{aligned} \tag{A.9}$$

The E,F,G functions transform as follows:

$$\begin{aligned} E(u,v) &= \vec{x}_u \cdot \vec{x}_u = \vec{x}_u^{\,*} \cdot \vec{x}_u^{\,*} \\ &= \vec{x}_\eta^{\,*} \cdot \vec{x}_\eta^{\,*} \eta_u^2 + 2\vec{x}_\eta^{\,*} \cdot \vec{x}_\xi^{\,*} \eta_u \xi_u + \vec{x}_\xi^{\,*} \cdot \vec{x}_\xi^{\,*} \xi_u^2 . \\ &= E^* \eta_u^2 + 2F^* \eta_u \xi_u + G^* \xi_u^2 \end{aligned} \tag{A.10}$$

Similarly, we have

$$F(u,v) = E^* \eta_u \eta_v + F^* (\eta_u \xi_v + \eta_v \xi_u) + G^* \xi_u \xi_v \tag{A.11}$$

$$G(u,v) = E^* \eta_v^2 + 2F^* \eta_v \xi_v + G^* \xi_v^2 \tag{A.12}$$

Thus, the E,F,G functions are *not* invariant to changes in parameterization of a smooth surface.

The term $EG - F^2$ is common term in the expressions for mean and Gaussian curvature. Therefore, it is evaluated since it will be needed for the final invariance statements. After several algebraic manipulations, we have

$$EG - F^2 = (E^* G^* - F^{*2})(\eta_u \xi_v - \eta_v \xi_u)^2 \tag{A.13}$$

The second multiplicative term is the square of the determinant of the Jacobian matrix of the parameter transformation, known as the *Jacobian*. The usual notation for the Jacobian is the following:

$$\frac{\partial(\eta,\xi)}{\partial(u,v)} = (\eta_u \xi_v - \eta_v \xi_u) \tag{A.14}$$

Next, the second fundamental form coefficient functions are examined. By definition of the normal vector to the surface, the tangent vectors $\vec{x}_u$ and $\vec{x}_v$ are always perpendicular to the normal vector.

$$\vec{x}_u \cdot \vec{n} = 0 \qquad \vec{x}_v \cdot \vec{n} = 0 \tag{A.15}$$

If these two zero functions are differentiated, the resultant derivative functions must also be zero. Using the inner product derivative rule, it is found that

$$L(u,v) = \vec{x}_{uu} \cdot \vec{n} = -\vec{x}_u \cdot \vec{n}_u \tag{A.16}$$

$$N(u,v) = \vec{x}_{vv} \cdot \vec{n} = -\vec{x}_v \cdot \vec{n}_v$$

$$M(u,v) = \vec{x}_{uv} \cdot \vec{n} = \vec{x}_{vu} \cdot \vec{n} = -\vec{x}_v \cdot \vec{n}_u = -\vec{x}_u \cdot \vec{n}_v$$

Next, note that the chain rule can also be applied to the partial derivatives of the normal vector:

$$\vec{n}_u'(\eta,\xi) = \vec{n}_\eta' \eta_u + \vec{n}_\xi' \xi_u \tag{A.17}$$

$$\vec{n}_v'(\eta,\xi) = \vec{n}_\eta' \eta_v + \vec{n}_\xi' \xi_v$$

The L,M,N functions transform as follows:

$$\begin{aligned} L &= \vec{x}_u \cdot \vec{n}_u = \vec{x}_u' \cdot \vec{n}_u' \\ &= \vec{x}_\eta' \cdot \vec{n}_\eta' \eta_u^2 + 2\vec{x}_\eta' \cdot \vec{n}_\xi' \eta_u \xi_u + \vec{x}_\xi' \cdot \vec{n}_\xi' \xi_u^2 . \\ &= L' \eta_u^2 + 2M' \eta_u \xi_u + N' \xi_u^2 \end{aligned} \tag{A.18}$$

Similarly, we have

$$M = L' \eta_u \eta_v + M'(\eta_u \xi_v + \eta_v \xi_u) + N' \xi_u \xi_v \tag{A.19}$$

$$N = L' \eta_v^2 + 2M' \eta_v \xi_v + N' \xi_v^2 \tag{A.20}$$

As is easily seen by comparison, the exact same transformation occurs for both the E,F,G functions and the L,M,N functions. Since the transformation is exactly the same, this implies that

$$LN - M^2 = (L'N' - M'^2)\left(\frac{\partial(\eta,\xi)}{\partial(u,v)} \right)^2 \tag{A.21}$$

Since $K = (LN - M^2)/(EG - F^2)$, it is seen that the squared Jacobian term will cancel out in numerator and denominator. Therefore, $K(u,v) = K'(\eta,\xi)$ where $u = u(\eta,\xi)$ and $v = v(\eta,\xi)$.

More work is required to show that the mean curvature is invariant. The numerator of the mean curvature expression transforms as follows:

$$EN + GL - 2FM = (E^*N^* + G^*L^* - 2F^*M^*)\left(\frac{\partial(\eta,\xi)}{\partial(u,v)}\right)^2 \quad \text{(A.22)}$$

Thus, since $H = (EN + GL - 2FM)/(2(EG - F^2))$, it is clear that the squared Jacobian term will also cancel out in the numerator and denominator. Therefore, $H(u,v) = H^*(\eta,\xi)$ where $u = u(\eta,\xi)$ and $v = v(\eta,\xi)$. In summary, mean and Gaussian curvature are invariant to changes in the parameterization of a surface as long as the Jacobian of the parameter transformation is non-zero at every point on the surface.

Note that Theorem A.2 shows that the mean and Gaussian curvature possess an invariance property that none of the fundamental form coefficient functions E,F,G,L,M,N possess.

APPENDIX B

INTRINSIC PROPERTIES WITHOUT PARTIAL DERIVATIVES

It is interesting to note that there is at least one way to compute Gaussian curvature and other intrinsic surface properties without using explicit partial derivative estimates [Lin and Perry 1982]. This technique originates in the Regge calculus of general relativity in which geometry is analyzed without coordinates. A discrete triangularization of a surface (a piecewise flat surface approximation) must first be obtained in order to use this technique. Consider a particular point $\vec{x}_k$ on the triangularized surface; it is the vertex for N different triangles. See Figure B.1 for an example of the geometry. The lengths of the sides of the $i-th$ triangle are denoted 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 $\vec{x}_k$ is then given by

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

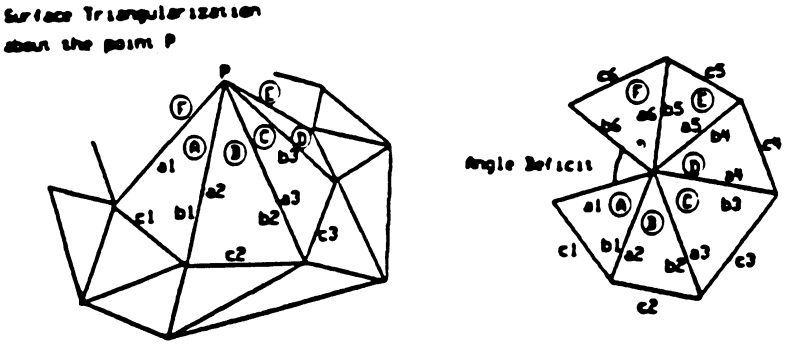


Figure B.1. Discrete Gaussian Curvature at a Point using Angle Deficit

The Gaussian curvature at a point is computed from the angle deficit as follows:

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

where

$$A_i = \sqrt{s\,(s - a_i)(s - b_i)(s - c_i)} \quad \text{and} \quad s = \frac{a_i + b_i + c_i}{2} \tag{B.3}$$

and where $\delta(\cdot)$ is the Dirac delta function. This method is related to the parallel transport formulation of Gaussian curvature mentioned in Chapter 3. Note that surface area estimates are computed during the Gaussian curvature computation.

Extremely accurate estimates of the Gaussian curvature of a sphere of radius r have been obtained using only five points on the surface of a hemisphere using the following derived formulation:

$$\hat{K} = \frac{2\pi - 4\cos^{-1}\left(1 - \dfrac{x^2}{a^2(x)} \right)}{x\sqrt{2a^2(x) - x^2}} \tag{B.4}$$

where

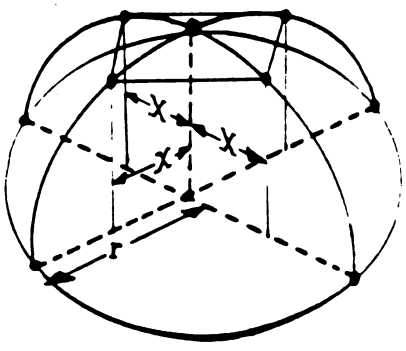


Figure B.2. Five Points on Hemisphere for Curvature Approximation

$$a(x) = \sqrt{2}r \cdot \left[1 - \left[1 - \frac{x^2}{r^2} \right]^{\frac{1}{2}} \right]^{\frac{1}{2}} \qquad (0 \leq x \leq r) \tag{B.5}$$

The five points on the hemisphere are shown in Figure B.2. Four of these points move from the (north) pole to the equator as a function of the variable x. The excellent results of example calculations for $r = 1$ are shown in Figure B.3. Nonetheless, the current computational method based on least squares biquadratic surface fitting is used instead of this method because the current method determines Gaussian curvature *in addition to* many other characteristics, both intrinsic and *extrinsic*, with comparable results and comparable noise sensitivity. It is impossible to compute mean curvature and other extrinsic surface characteristics with an approach of this sort since the

Gaussian Curvature Estimates		
Unit Sphere (Radius = 1)		
Five Symmetrically Spaced Points		
x	K	Percent Error
0.0002	0.975231	-2.5%
0.0004	0.998412	-0.16%
0.0005	0.999415	-0.06%
0.001	0.99992	-0.01%
0.005	1	0%
0.01	1	0%
0.1	1	0%
0.2	1.00007	+0.01%
0.3	1.00035	+0.04%
0.4	1.00116	+0.12%
0.5	1.003	+0.30%
0.6	1.00672	+0.67%
0.7	1.01385	+1.4%
0.8	1.02755	+2.8%
0.9	1.05666	+5.7%
0.95	1.08714	+8.7%
0.97	1.10796	+10.8%
0.99	1.14454	+14.4%
1	1.2092	+20.9%

Figure B.2. Example of Gaussian Curvature Estimates

embedding of the piecewise flat manifold in 3-D space is not known.

APPENDIX C

LEAST-SQUARES SURFACE FITTING

Low-order bivariate polynomials and least-squares surface fitting have been chosen as the surface approximation mechanism for the iterative region growing algorithm. Other types of approximation might also serve in the place of least-squares, but least squares is generally the easiest to use and the fastest type of approximation for many applications. Least-squares fitting and low-order bivariate polynomials have been selected to demonstrate the feasibility and the power of the surface-based range image segmentation approach.

If all regions of interest in an image were always regularly shaped, preferably rectangular, there would be no need for a special section on least-squares surface fitting because the description given in Chapter 3 would have been sufficient if the third and fourth order discrete orthogonal polynomials were also given. However, all image regions are not regularly shaped, and a data-driven algorithm should adapt itself to the regions in the data and not vice-versa. Therefore, an approach is needed that will be numerically stable over arbitrary regions and provide quick approximations.

Bartels and Jezioranski [1985] have very recently published a paper on least-squares fitting using orthogonal polynomials in several variables. Their approach is quite complicated because it is formulated for arbitrary inner products and for any number of independent variables, but it would allow one to compute least-squares fits of surfaces to arbitrary regions in a computationally efficient and numerically stable manner. The method has these properties because the basis functions are computed recursively using a three term recurrence relation and the coefficients for any fit are obtained via inner

products, not SVD or QR computations. The orthogonal polynomials used by this approach are essentially customized for each set of data and are not dependent upon the assumption of equally-spaced data points. This technique is recommended for future research, but was not known to the author prior to the completion of this thesis. In this appendix, an approach is presented that handles arbitrary regions using a fixed set of orthogonal polynomials that are dependent on the assumption of equally-spaced data points. This approach was used for all experimental results shown in Chapter 7 and did not present any numerical difficulties.

A least-squares algorithm can be applied directly to the image data to obtain the coefficients of the bivariate polynomial of interest. However, the finite word length of the computer may present difficulties when fourth-order polynomials are used over large regions. In an attempt to provide reasonable least-squares computations without numerical difficulties, discrete orthogonal polynomials are used as basis functions for the least squares computation even though the region of the fit is not rectangular or regular. This approach attempts to formulate a least squares problem in which the square matrix of the corresponding normal equations is diagonally dominant. As a compromise, the min/max box of a region and the center of mass of the region are used to specify the discrete orthogonal polynomial basis functions' exact form. The coefficients obtained for the basis functions may then be modified to provide a new set of coefficients that could be used to evaluate the polynomial at any point in powers of x and y. This conversion process is numerically unstable, but it is discussed in detail to point out the equivalence of the two representations.

Let (x_{min}, y_{min}) and (x_{max}, y_{max}) represent the corners of the min/max box that surrounds a closed bounded connected region of any shape. Let

$n_x = x_{max} - x_{min} + 1$ and $n_y = y_{max} - y_{min} + 1$ denote the dimensions of the min/max box. Let (μ_x, μ_y) represent the center of mass for that region. The discrete orthogonal polynomial basis functions for fitting a surface to the image data in that region are given as follows:

$$\phi_0(x) = 1 \tag{C.1}$$

$$\phi_1(x, \mu) = x - \mu \tag{C.2}$$

$$\phi_2(x, \mu, n) = (x - \mu)^2 + \frac{(1 - n^2)}{12} \tag{C.3}$$

$$\phi_3(x, \mu, n) = (x - \mu)^3 + \frac{(7 - 3n^2)}{20}(x - \mu) \tag{C.4}$$

$$\phi_4(x, \mu, n) = (x - \mu)^4 + \frac{(13 - 3n^2)}{14}(x - \mu)^2 + \frac{3(n^2 - 1)(n^2 - 9)}{560} \tag{C.5}$$

where μ takes on the value μ_x or μ_y and where n takes on the value n_x or n_y.

By expressing the most complicated polynomial used by the algorithm, simpler polynomials are easily obtained by letting the appropriate coefficients be zero. For each pixel p in the region of interest R, the three coordinates $x(p)$, $y(p)$, $z(p)$ for that pixel are plugged into the following equation to provide one equation for the least-squares system to be solved.

$$\begin{aligned} z(x,y) = {} & c_0\phi_0(x)\phi_0(y) + c_1\phi_1(x,\mu_x)\phi_0(y) + c_2\phi_0(x)\phi_1(y,\mu_y) + \\ & c_3\phi_1(x,\mu_x)\phi_1(y,\mu_y) + c_4\phi_2(x,\mu_x,n_x)\phi_0(y) + c_5\phi_0(x)\phi_2(y,\mu_y,n_y) + \\ & c_6\phi_2(x,\mu_x,n_x)\phi_1(y,\mu_y) + c_7\phi_1(x,\mu_x)\phi_2(y,\mu_y,n_y) + c_8\phi_3(x,\mu_x,n_x)\phi_0(y) + \\ & c_9\phi_0(x)\phi_3(y,\mu_y,n_y) + c_{10}\phi_3(x,\mu_x,n_x)\phi_1(y,\mu_y) + \\ & c_{11}\phi_2(x,\mu_x,n_x)\phi_2(y,\mu_y,n_y) + c_{12}\phi_1(x,\mu_x)\phi_3(y,\mu_y,n_y) + \\ & c_{13}\phi_4(x,\mu_x,n_x)\phi_0(y) + c_{14}\phi_0(x)\phi_4(y,\mu_y,n_y). \end{aligned} \tag{C.6}$$

The number of equations for a region is the number of pixels $N_p = |R|$ in that

region. Let us denote the 15 x 1 vector of coefficients as $\vec{c}$, the N_p x 1 vector of z values as $\vec{z}$, and the N_p x 15 matrix of $\phi_i \phi_j$ values as the matrix Φ_{zy}, then all N_p equations may be written down as a single matrix equation:

$$\vec{z} = \Phi_{zy} \vec{c} \tag{C.7}$$

The so-called normal equations are formed by multiplying through by the transpose of the Φ_{zy} matrix:

$$\Phi_{zy}^T \Phi_{zy} \vec{c} = \Phi_{zy}^T \vec{z} \tag{C.8}$$

The normal equations represent a square 15 x 15 matrix equation. When the region R is nearly rectangular, the square normal matrix is nearly diagonal. Even when R is not rectangular, the diagonal terms are still generally the dominant terms in the matrix. The coefficient vector is solved for analytically as follows:

$$\vec{c} = (\Phi_{zy}^T \Phi_{zy})^{-1} \Phi_{zy}^T \vec{z} \tag{C.9}$$

If the square matrix of the normal equations is not invertible, the (unique) Moore-Penrose pseudo-inverse can be used. Once the coefficient vector $\vec{c}$ has been solved for, the root mean square error ϵ_{rms} can be computed:

$$\begin{aligned} \epsilon_{rms} &= \frac{1}{\sqrt{N_p}} \mid \vec{z} - \Phi_{zy} \vec{c} \mid \\ &= \sqrt{\vec{z}^T (\vec{z} - \Phi_{zy} \vec{c})/N_p} \end{aligned} \tag{C.10}$$

In practice, numerical methods are used to solve the least-squares equation. The normal equations do not need to be formed to solve for $\vec{c}$ as in Golub [1965]. Singular value decomposition methods can also be used [Lawson and Hanson 1974]. In the iterative variable-order surface-fitting algorithm, the use of the QR decomposition method allows the least squares problem to be solved incrementally by dropping and adding points to a

region to create the next region description for the next surface fit, or by adding columns to the matrix for the next higher-order surface fit [Golub and van Loan 1983]. The use of the orthogonal polynomials tends to make the problem very well conditioned so that several different numerical methods could be used to solve these equations without difficulty.

Once the coefficient vector $\vec{c}$ and the fit error ϵ_{rms} have been computed, equation (C.6) can be used to compute the value of z for any given (x, y) pair. For completeness, a new set of coefficients $\vec{a}$ has been computed so that the approximating polynomial may be expressed without involving the evaluation of the discrete orthogonal polynomials. That is, the equation (C.6) above is recast in the form of

$$z = g(\vec{a}, x, y) = a_0 + a_1 x + a_2 y + a_3 xy + a_4 x^2 + a_5 y^2 + \quad \text{(C.11)}$$
$$a_6 x^2 y + a_7 x y^2 + a_8 x^3 + a_9 y^3 + a_{10} x^3 y + a_{11} x^2 y^2 + a_{12} x y^3 +$$
$$a_{13} x^4 + a_{14} y^4 .$$

The a_i coefficients are linear combinations of the c_i coefficients. Although this transformation is numerically unstable, the $\vec{c} \rightarrow \vec{a}$ transformation is included here to indicate the mathematical equivalence of equations (C.6) and (C.11). First, four constants that occur in the expressions below are defined.

$$n_3^x = \frac{7 - 3n_x^2}{20} \qquad n_3^y = \frac{7 - 3n_y^2}{20} \qquad \text{(C.12)}$$
$$n_4^x = \frac{13 - 3n_x^2}{14} \qquad n_4^y = \frac{13 - 3n_y^2}{14}$$

The first term is obtained by setting $x = y = 0$:

$$a_0 = c_0 - (c_1 \mu_x + c_2 \mu_y) + c_3 \mu_x \mu_y + c_4 \phi_2(0, \mu_x, n_x) + c_5 \phi_2(0, \mu_y, n_y) +$$

$$- \left(c_6\mu_y\,\phi_2(0,\mu_x,n_x) + c_7\mu_x\,\phi_2(0,\mu_y,n_y) \right) + c_8\phi_3(0,\mu_x,n_x) + c_9\phi_3(0,\mu_y,n_y) +$$

$$- \left(c_{10}\phi_3(0,\mu_x,n_x)\mu_y + c_{12}\mu_x\,\phi_3(0,\mu_y,n_y) \right) + c_{11}\phi_2(0,\mu_x,n_x)\phi_2(0,\mu_y,n_y) +$$

$$c_{13}\phi_4(0,\mu_x,n_x) + c_{14}\phi_4(0,\mu_y,n_y) .$$

The next two terms are obtained by taking derivatives of both sides and by setting $x = y = 0$:

$$a_1 = c_1 - (c_3\mu_y + 2c_4\mu_x) + c_7\phi_2(0,\mu_y,n_y) + c_8(3\mu_x{}^2 + n_3^x) + (2c_6\mu_x\mu_y)$$
$$- c_{10}\mu_y(3\mu_x{}^2 + n_3^x) - 2c_{11}\mu_x\,\phi_2(0,\mu_y,n_y) + c_{12}\phi_3(0,\mu_y,n_y) - c_{13}(4\mu_x{}^3 + 2\mu_x\,n_4^x)$$

$$a_2 = c_2 - (c_3\mu_x + 2c_5\mu_y) + c_6\phi_2(0,\mu_x,n_x) + c_9(3\mu_y{}^2 + n_3^y) + (2c_7\mu_x\mu_y)$$
$$+ c_{10}\phi_3(0,\mu_x,n_x) - 2c_{11}\mu_y\,\phi_2(0,\mu_x,n_x) - c_{12}(\mu_x(3\mu_y{}^2 + n_3^y)) - c_{14}(4\mu_y{}^3 + 2\mu_y\,n_4^y)$$

The remaining terms are computed similarly:

$$a_3 = c_3 - (2c_6\mu_x + 2c_7\mu_y) + 4c_{11}\mu_x\mu_y + c_{10}(3\mu_x{}^2 + n_3^x) + c_{12}(3\mu_y{}^2 + n_3^y)$$
$$a_4 = c_4 - (c_6\mu_y + 3c_8\mu_x) + 3c_{10}\mu_x\mu_y + c_{11}\phi_2(0,\mu_y,n_y) + c_{13}(6\mu_x{}^2 + n_4^x)$$
$$a_5 = c_5 - (c_7\mu_x + 3c_9\mu_y) + 3c_{12}\mu_x\mu_y + c_{11}\phi_2(0,\mu_x,n_x) + c_{14}(6\mu_y{}^2 + n_4^y)$$
$$a_6 = c_6 - (2c_{11}\mu_y + 3c_{10}\mu_x) \qquad a_7 = c_7 - (2c_{11}\mu_x + 3c_{12}\mu_y)$$
$$a_8 = c_8 - (4\mu_x c_{13} + c_{10}\mu_y) \qquad a_9 = c_9 - (4\mu_y c_{14} + c_{12}\mu_x)$$
$$a_{10} = c_{10} \qquad a_{11} = c_{11} \qquad a_{12} = c_{12} \qquad a_{13} = c_{13} \qquad a_{14} = c_{14}$$

In summary, a list of (x,y,z) coordinates is used to form the least-squares matrix equation, and the equation is solved numerically to create the vector $\vec{c}$. The vector $\vec{c}$ is mathematically equivalent to the vector $\vec{a}$ as stated in the conversion equations above.

In the notation used in the text of Chapter 4, the least squares fitting procedure $L_g^{\cdot}(g, R_i^{(k)}, m_F)$ accepts the original image g, a region $R_i^{(k)}$, and a function type m_F. The original image defines a large set of (x,y,z) coordinates, and the region definition

selects a certain subset of this data for the fit. As stated above, the biquartic case includes all lower order cases. For a planar surface, only c_0, c_1, c_2 are allowed to be non-zero. This significantly reduces the matrix size (N_p x 3) and the computation required for the Φ_{xy} matrix. For the biquadratic surface, c_3, c_4, c_5 coefficients are also allowed to be non-zero and the appropriate terms of the data matrix must be included to solve the N_p x 6 problem.

APPENDIX D

EQUAL ANGLE INCREMENT SAMPLING

Not all range sensors produce rectangular Cartesian coordinates on a regularly-spaced grid directly as output. That is, ideal equally-spaced orthographic-projection range images are not always available. In this appendix, two types of perspective projection coordinate systems are treated that are necessary to describe several real range sensors that use equal-angle-increment (EAI) sampling. Even though orthographic depth measurements are obtainable from these sensors, the (x,y) locations of pixels are not regularly spaced. One reason for special projection mechanisms is that it is often very convenient to use stepper motors to control the direction of the range sensing operation. The use of stepper motors typically implies that depth samples are obtained at equal-angle increments, not equal-distance increments. For example, the laser range finder developed by the Environmental Research Institute of Michigan (ERIM) uses two mirrors that rotate in equal-angle increments around orthogonal axes. Most of the real range images discussed in this thesis were acquired using the ERIM range finder. Other range finders (e.g. [Milgram and Bjorklund 1980]) measure slant range as a function of azimuth and elevation. This type of device also normally uses equal-angle increments in azimuth and elevation.

Since the methods proposed in this thesis are designed primarily for orthographic projection range images, it is important that the nature of perspective projection measurements is understood in detail since range finders are often encountered that do not fit the equal-distance increment orthographic projection model. Most range finders fall into one of the two categories of perspective models discussed here.

Equal-angle-increment (EAI) range finders may be divided into two types: the two orthogonal-axis mirror (TOAM) range finder and the azimuth-elevation (AZEL) range finder. Two-way (forward and inverse) mathematical transformations are developed to model the geometry of these two types of range finders. The two types are compared to each other and to the Cartesian orthographic projection model.

D.1 Two Orthogonal-Axis Mirror (TOAM) Range Finder

The TOAM range finder measures range r to points in a scene as a function of horizontal angle θ and vertical angle ϕ. As shown in Figure D.1, the x direction is associated with the angle θ and the y direction is associated with the angle ϕ. The z direction is the direction that is pointed to when $\theta = \phi = 0$. This direction is typically the direction corresponding to the center pixel in a range image. The range finder is assumed to scan from θ_a to θ_b in the horizontal direction and from ϕ_a to ϕ_b in the vertical direction. Let $(x, y, z) = (0,0,0)$ be the effective point that the range finder pivots around. Now suppose that the range finder is measuring the range r to some point (x, y, z) while the horizontal mirror is set to position $\theta \in [\theta_a, \theta_b]$ and the vertical

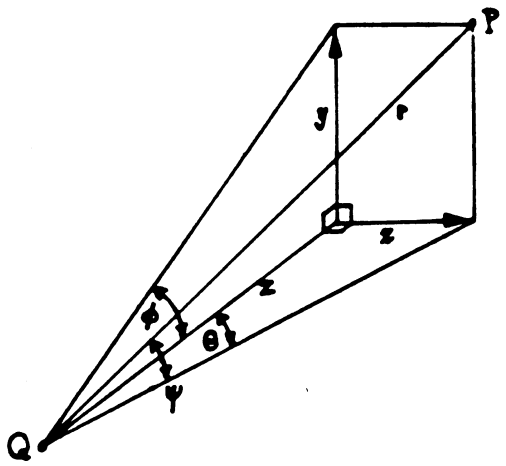


Figure D.1. Perspective Angles and Distances

mirror is set to position $\phi \in [\phi_a, \phi_b]$. Let us formulate the (x, y, z) coordinates in terms of the (r, θ, ϕ) coordinates and vice versa.

No matter what angular model is chosen, it is certain that

$$r^2 = x^2 + y^2 + z^2 \tag{D.1}$$

is one condition that must hold true. From the diagram in Figure D.1, it is clear that

$$x = z \tan\theta \qquad y = z \tan\phi \,. \tag{D.2}$$

Therefore, the range to the point may be expressed as

$$r^2 = z^2 (1 + \tan^2\theta + \tan^2\phi) \tag{D.3}$$

This analysis yields the transformation from (r, θ, ϕ) coordinates to (x, y, z) coordinates:

$$x(r, \theta, \phi) = \frac{r \tan\theta}{\sqrt{1 + \tan^2\theta + \tan^2\phi}} \tag{D.4}$$

$$y(r, \theta, \phi) = \frac{r \tan\phi}{\sqrt{1 + \tan^2\theta + \tan^2\phi}}$$

$$z(r, \theta, \phi) = \frac{r}{\sqrt{1 + \tan^2\theta + \tan^2\phi}}$$

Note the symmetry between the horizontal and vertical angles. This symmetry will not be present in the AZEL model. The inverse transformation is given by:

$$r(x, y, z) = \sqrt{x^2 + y^2 + z^2} \tag{D.5}$$

$$\theta(x, z) = \tan^{-1}(x/z)$$

$$\phi(y, z) = \tan^{-1}(y/z)$$

These equations completely specify the TOAM range finder model.

Let us consider the behavior of range samples from the plane specified by $z = h$. As a point moves in the x direction, θ changes as follows:

$$\delta\theta = \left(\frac{h}{h^2 + x^2} \right) \delta x \tag{D.6}$$

Also,

$$\delta x = (h \sec^2\theta)\, \delta\theta \tag{D.7}$$

Thus, it is clear that incremental changes in one variable do not yield linear incremental changes in the other variable. The same relationships hold for y and ϕ. Moreover, the explicit functional relationships for z and r are the following:

$$\begin{aligned} z(x,y) &= h \qquad\qquad (\textit{Orthographic Projection}) \\ r(x,y) &= \sqrt{h^2 + x^2 + y^2} \\ r(\theta,\phi) &= h\sqrt{1 + \tan^2\theta + \tan^2\phi} \end{aligned} \tag{D.8}$$

Since $\tan\theta \approx \theta$ to within 3% up to 17 degrees of arc, it is clear that the two perspective range images of the plane $r(x,y)$ and $r(\theta,\phi)$ will be very similar within a 34^o x 34^o field of view.

In Figure D.2, two surface plots for a plane at depth $z = h$ are displayed. In the first plot, the expected constant depth surface is shown under the standard Cartesian projection. In the second plot, the exact same surface is shown under the TOAM orthographic projection where the two angular coordinates are considered equivalent to the two (x,y) Euclidean coordinates. A 90 degree field of view was assumed. Note the severe warping of the surface at the larger angles.

D.2 Azimuth-Elevation (AZEL) Range Finder

The AZEL model is slightly different than the TOAM model. In the AZEL model, the horizontal rotation angle, known as the azimuth, is denoted as θ because it is practically identical to the θ in the TOAM model. The elevation angle ψ is not the same as the vertical rotation angle ϕ in the TOAM model. The range r and x,y,z have the

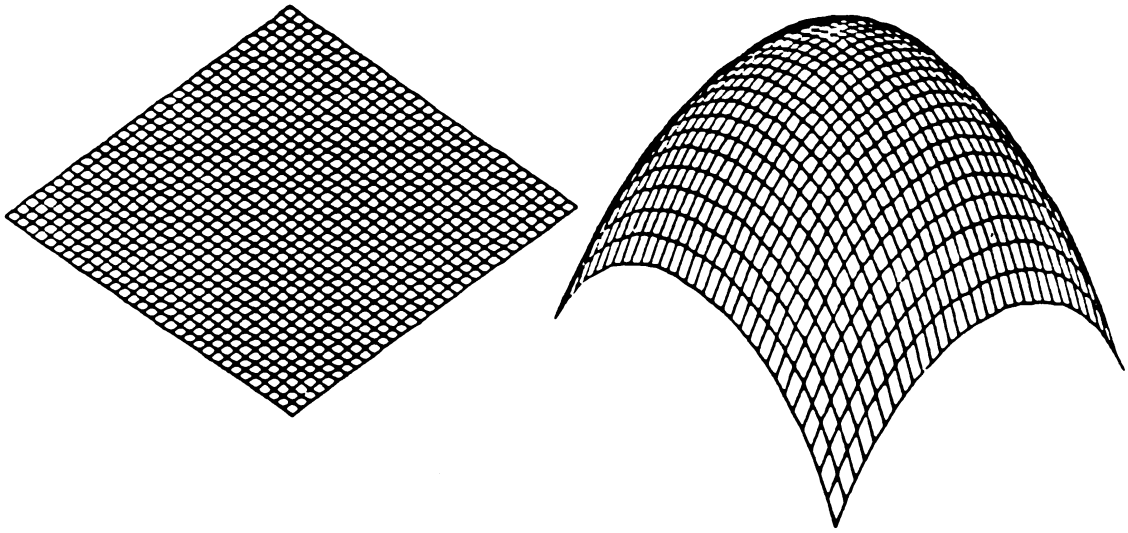


Figure D.2. Orthographic Cartesian Projection and TOAM Projection

same meaning as above. The (r,θ,ψ) system is a true spherical coordinate system where θ and ψ may be interpreted as longitude and latitude. The key difference in the AZEL model is that, although $r(x,y,z)$ and $\theta(x,y,z)$ are defined as above, the ψ angle is defined as follows:

$$\psi(x,y,z) = \tan^{-1}\left(\frac{y}{\sqrt{x^2+z^2}}\right) \tag{D.9}$$

This defines the transformation from (x,y,z) to (r,θ,ψ) coordinates. Given (r,θ,ψ), it is easy to see that

$$\begin{aligned} y(r,\psi) &= r\sin\psi \\ x(r,\theta,\psi) &= r\cos\psi\sin\theta \\ z(r,\theta,\psi) &= r\cos\psi\cos\theta \end{aligned} \tag{D.10}$$

Note that $z(r,0,0) = r$ as expected. Note also that ψ depends on all three x,y,z coordinates whereas ϕ (TOAM model) depended only y and z, and that y depends only on r and ψ in the AZEL model whereas it depended on all three r,θ,ϕ coordinates in

the TOAM model. Thus, the x and y axes are not symmetrical in the AZEL model as they are in the TOAM model owing to the coupling between the azimuth and elevation angles.

Suppose again that a range image on the plane $z = h$ is acquired using an AZEL type range finder. This means $z(x,y) = h$ and $r(x,y) = \sqrt{h^2 + x^2 + y^2}$, but the function $r(\theta,\psi)$ is slightly different in that it contains an extra term that couples the θ and ψ directions:

$$r(\theta,\psi) = h\sqrt{1 + \tan^2\theta + \tan^2\psi + \tan^2\theta\,\tan^2\psi} \qquad \text{(D.11)}$$

In this case, $r(x,y)$ is still approximately the same as $r(\theta,\psi)$ for small angles because the coupling term is a product of two small numbers yielding a much smaller number. However, the distortion near the limits of θ and ψ is worse for an AZEL-type range finder than for a TOAM-type range finder.

D.3 Summary

Two types of range finders (TOAM and AZEL) have been analyzed. Six types of range image projections have arisen implicitly or explicitly in the above discussion and are summarized here:

(1) Cartesian Orthographic Projection: $z(x,y)$

(2) Cartesian Perspective Projection: $r(x,y)$

(3) TOAM Orthographic Projection: $z(\theta,\phi)$

(4) TOAM Perspective Projection: $r(\theta,\phi)$

(5) AZEL Orthographic Projection: $z(\theta,\psi)$

(6) AZEL Perspective Projection: $r(\theta,\psi)$

Note that each Perspective Projection allows direct recovery of z from the Orthographic Projection through an appropriate transformation:

$$z(x,y) = \sqrt{(r(x,y))^2 - (x^2 + y^2)} \tag{D.12}$$

$$z(\theta,\phi) = \frac{r(\theta,\phi)}{\sqrt{1 + \tan^2\theta + \tan^2\phi}} \tag{D.13}$$

$$z(\theta,\psi) = r(\theta,\psi)\cos\theta\cos\psi \tag{D.14}$$

The problem with the Orthographic TOAM and AZEL projections is that the digital surfaces corresponding to the graph surfaces $z(\theta,\phi)$ and $z(\theta,\psi)$ are still slightly warped due to the EAI sampling of the angular coordinates. To get a range image on an (x,y) grid, all three x,y,z coordinates should be computed for each angle pair.

$$x_{TOAM}(\theta,\phi) = z(\theta,\phi)\tan\theta \qquad x_{AZEL}(\theta,\psi) = z(\theta,\psi)\tan\theta \tag{D.15}$$

$$y_{TOAM}(\theta,\phi) = z(\theta,\phi)\tan\phi \qquad y_{AZEL}(\theta,\psi) = \sqrt{z^2(\theta,\psi) + x^2_{AZEL}(\theta,\psi)}\,\tan\psi$$

Then an interpolation scheme could be used to resample the interpolated surfaces to obtain the desired equally-spaced sampled Cartesian Orthographic projection $z(x,y)$. Since most of the ERIM range images of objects use a small field of view, the range images have been processed directly without resampling and the results have been satisfactory. If an object surface matching algorithm were to use this data directly, it would have to account for a slight warping in the sensed surfaces using the current segmentation algorithm.

APPENDIX E

A METRIC SPACE OF IMAGE REGIONS

In this appendix, it is proved that the area of the symmetric difference of two connected finite-area regions in an image is a metric on the space of all connected regions of finite area in an image. The set of connected regions is denoted C^I as in Chapter 4. This region set along with the symmetric-difference size metric form a metric space. In such a mathematical framework, it is meaningful to discuss the "distance" between two region descriptions. A region distance metric allows one to quantitatively measure the quality of a region approximation just as the quality of a functional approximation is measured.

A metric space is a set X and a mapping $d : X \times X \rightarrow \mathbf{R}^+$ that maps pairs of elements in X to non-negative reals and satisfies the following conditions for any $A, B, C \in X$:

$$\begin{aligned} & d(A,B) \geq 0 && (Non\text{-}Negative) \\ & d(A,B) = 0 \longleftrightarrow A = B && (Positive\ Definite) \\ & d(A,B) = d(B,A) && (Symmetric) \\ & d(A,B) \leq d(A,C) + d(C,B) && (Triangle\ Inequality)\,. \end{aligned} \tag{E.1}$$

By defining a mapping d for an existing set and proving that these four conditions hold, the analysis of that set may use the mathematics of metric spaces.

The region metric is defined using a set measure m. Let I be the universal set. Let C^I be the set of connected finite-area regions of interest. Let A be a subset of I and an element of the space C^I. $m(A)$ is defined to be the area of the set A. In a continuous domain I, $m(A) = \int_A dxdy$. In a discrete image domain I,

$m(A) = |A| =$ the number of pixels in the region A. Hence, $m(A) = 0$ iff A is a set of measure zero. Sets of measure zero are logically equated with the null set ϕ since curves and points have no area. Hence, $m(A) > 0$ for any non-null connected set A in C^I. Also, $m(A \bigcup B) = m(A) + m(B) - m(A \bigcap B)$. The symmetric difference of two subsets A, B of I is as defined in the text: $A \Delta B = (A - B) \bigcup (B - A)$.

THEOREM E.1: The set C^I and the mapping $d(A,B) = m(A \Delta B)$ form a metric space.

Proof: The first condition is always satisfied because the symmetric difference of two sets is another set and all sets have non-negative area. The third condition is also satisfied due to the symmetry of the symmetric difference operator. The second condition is proved as follows: If $A = B$, then $A \Delta B = A - B \bigcup B - A = \phi$, which implies that $m(A \Delta B) = 0$. If $m(A \Delta B) = m(A - B) + m(B - A) = 0$, then non-negativity requires that $m(A - B) = 0$ and $m(B - A) = 0$ separately. This implies $A - B = \phi$ and $B - A = \phi$. Therefore, $A \subseteq B$ and $B \subseteq A$, which implies $A = B$.

The proof of the triangle inequality is more involved. Henceforth, $A \bigcup B$ is written as $A + B$ and $A \bigcap B$ as AB. The useful property that $A = AB + (A - B)$ yields

$$m(A) = m(AB) + m(A - B), \tag{E.2}$$

which expresses that the area of A consists of the part of A in B and the part of A not in B; this is needed for the proof. It must be shown that, for any set C,

$$m(A \Delta B) \leq m(A \Delta C) + m(C \Delta B). \tag{E.3}$$

An exhaustive expansion approach is used.

$$\begin{aligned} m(A \Delta B) &= m(A - B) + m(B - A) \\ &= m(A) + m(B) - 2m(AB) \\ &\stackrel{?}{\leq} m(A \Delta C) + m(C \Delta B) \\ &= m(A - C) + m(C - A) + m(B - C) + m(C - B) \\ &= m(A) + m(B) + 2m(C) - (2m(AC) + 2m(BC)) \end{aligned} \tag{E.4}$$

After canceling and rearranging the terms, the question becomes

$$m(AC) + m(BC) \stackrel{?}{\leq} m(C) + m(AB) . \tag{E.5}$$

Each set is further decomposed into disjoint subsets. The set C is considered with the following disjoint decomposition:

$$C = (C - (A + B)) + (BC - A) + (AC - B) + ABC . \tag{E.6}$$

The preliminary lemma (E.2) mentioned above is used on the other sets.

$$m(AC - B) + 2m(ABC) + m(BC - A) \leq \tag{E.7}$$
$$m(C - (A + B)) + m(BC - A) + m(AC - B) + 2m(ABC) + m(AB - C).$$

Everything on the left-hand side cancels out leaving the following statement, which is obviously true, by the definition of the measure of area:

$$0 \leq m(C - (A + B)) + m(AB - C). \tag{E.8}$$

This proves the triangle inequality and completes the proof that the set C' and the metric $m(A \Delta B)$ form a metric space.

Note that if the binary field {0,1} is used with the metric space above, the measure $m(A)$ is also a norm. Let 1 times a set be the set and let 0 times the set be the null set.

$$m(A) \geq 0 \qquad (Non\text{-}negative) \qquad \text{(E.9a)}$$

$$m(A + B) \leq m(A) + m(B) \qquad (Triangle\ Inequality) \qquad \text{(E.9b)}$$

$$m(\alpha A) = |\alpha|\, m(A) \qquad (Homogeneity) \qquad \text{(E.9c)}$$

$$m(A) = 0 \longleftrightarrow A = \phi \qquad (Positive\ Definite) \qquad \text{(E.9d)}$$

All of the properties are trivially satisfied by definition of $m(A)$ and the conventions for the sets in C^I. Note also that

$$m(A + B) = m(A) + m(B) - m(AB) \leq m(A) + m(B)\,. \qquad \text{(E.10)}$$

The notions of norm and metric are useful for formulating questions about convergence, approximation quality, and segmentation quality. Although all convergence and segmentation quality issues have not been resolved, a mathematical framework for discussing such issues has been explicitly established. It is interesting to note that when the regions A and B are represented as bit strings (bitmaps), the distance metric above is the Hamming distance between the bit strings.

APPENDIX F

SOFTWARE IMPLEMENTATION

The experimental results were obtained using a range image segmentation system and a software support environment developed by the author. The complete software package consists of over 37,000 lines of commented C source code. The three primary programs of the segmentation system are the (1) surface characterization program, (2) surface refinement program (region-growing segmentation), and (3) surface merging program with edge description option. These three programs represent over 15,000 lines of application specific code out of the total above. The purpose of this appendix is to briefly describe several key procedures needed to implement these programs. The reason for doing this is to stress that most of the required computations for the segmentation system can be performed using general-purpose low-level image processing modules or math subroutines. An interesting aspect of this approach is that only a relatively small number of algorithm-specific functions are required. Because these general-purpose modules have been mentioned in the text without any explanatory discussion, they are listed here as brief algorithm statements and descriptions.

BINARY IMAGE CONTRACTION (EROSION)

The binary image contraction (erosion) operation for 3x3 windows maps a binary image into another binary image in which regions of the output image are smaller than the corresponding regions in the input image.

Algorithm Statement:

For each ON pixel in the input binary image, test each of the eight neighbors of that

pixel in the input image. If any neighbor is OFF, turn that pixel OFF in the output binary image. For each OFF pixel in the input binary image, leave that pixel OFF in the output binary image.

Contraction can be achieved very quickly on appropriate image processing hardware. It is useful for making a region smaller while maintaining most of the shape properties of the region.

BINARY IMAGE DILATION (EXPANSION)

The binary image dilation (expansion) operation for 3x3 windows maps a binary image into another binary image in which regions of the output image are larger than the corresponding regions in the input image.

Algorithm Statement:

For each OFF pixel in the input binary image, test each of the eight neighbors of that pixel in the input image. If any neighbor is ON, turn that pixel ON in the output binary image. For each ON pixel in the input binary image, leave that pixel ON in the output binary image.

Dilation is also achieved very quickly on appropriate image processing hardware. It is useful for making a region larger while maintaining most of the shape properties of the region.

BINARY IMAGE CONNECTED COMPONENT ANALYZER

The binary image connected component analysis operation maps an input binary image into an output list of regions in which each region consists of a set of pixels that are connected to each other in some sense. Four-connected regions have the property that any two pixels in a four-connected region can be connected by a string of four-

connected pixels all belonging to the same region. One pixel p is four-connected to another adjacent pixel q if $|x(p) - x(q)| = 1$ and $|y(p) - y(q)| = 0$, or if $|x(p) - x(q)| = 0$ and $|y(p) - y(q)| = 1$. Only four-connected regions are of interest for surface-based algorithms.

Algorithm Statement [Rosenfeld and Kak 1982][Ballard and Brown 1982]:

Set label value to zero. Initialize label buffer the same size as the image with label value. For each ON pixel in the input binary image as one scans the image left-to-right top-to-bottom, check the upper and left (four-connected) neighbors of that pixel.

(1) If the upper neighbor is ON and the left neighbor is OFF, give that pixel the label of the upper neighbor.

(2) If the upper neighbor is OFF and the left neighbor is ON, give that pixel the label of the left neighbor.

(3) If the upper neighbor is OFF and the left neighbor is OFF, increment the current label value and give the new label value to that pixel.

(4) If the upper neighbor is ON and the left neighbor is ON, give that pixel the label of the upper neighbor and, if the labels of the two neighbor pixels are different, record the logical equivalence of the two different labels in an equivalence table if not already recorded.

After all input image pixels have been checked, use the equivalence table to create a lookup table that maps old non-unique labels into a new set of unique labels with one label per four-connected region. Then, for each pixel in the label buffer map the old value to the new value using the lookup table. The number of unique non-zero labels is the number of regions in the binary image. Each region can be extracted by isolating all pixels of a given label.

Note that two sequential passes over the image pixels are required in this standard four-connected component algorithm. This operation can be performed very quickly on special hardware. The connected component analysis algorithm is executed many, many times during the execution of the region growing algorithm. It is a key limiting factor to the speed of the current implementation on a general-purpose digital computer.

SEPARABLE WINDOW CONVOLUTION

Window convolutions are required extensively for smoothing and differentiation in both the surface and edge characterization software. A single set of subroutines is able to perform both tasks. Because of the one-dimensional nature of general-purpose digital computer memory, two separate subroutines are required to do separable convolutions. The row convolution subroutine accepts (1) an image or a vector of data, (2) the window row vector of the separable window, and (3) the appropriate size constants. The column convolution subroutine accepts (1) an image, (2) the window column vector of the separable window, and (3) the appropriate size constants. Smoothing or differentiation is achieved using two separate subroutines that generate the appropriate window vectors. With four subroutines, one can do arbitrary window size convolutions on arbitrary size rectangular images or edge vectors to accomplish smoothing or differentiation.

LEAST SQUARES FITTING

As mentioned in Appendix C, several types of least squares fitting techniques are possible. Once a basic least squares solver is chosen, it is possible to have a single subroutine to do all surface fitting for any order and another subroutine to do all edge fitting for any order. Basic polynomial evaluation subroutines are of course required for either program.

3 x 3 WINDOW MAPPING

Edge refinement and region refinement operations on binary images can both be formulated in terms of a single mathematical operation. Every 3x3 window in a binary image can be represented as an integer between 1 and 512. A simple lookup table can then be used to define a mapping between the 3x3 windows that exist in the data and the 3x3 windows that you, the human interpreter, would like to see. An array of 512 elements (short integers) can be defined for every type of operation to be performed. Generally, only a few values in the lookup table need to be altered from their original index value. Separate lookup tables have been used for notch filling, burr trimming, and edge refinement.

IMAGE BOOLEANS AND IMAGE ARITHMETIC

No software package for image processing would be complete without a set of image boolean operators to do AND's, OR's, NOT's, and XOR's, and a set of image arithmetic operators to do addition, subtraction, and scaling. An image copy subroutine, a set-an-image-to-a-value subroutine, and an image lookup-table-mapping subroutine should also be included in this set.

BASIC FILE INPUT AND OUTPUT, DISPLAYS, AND CONVERSIONS

Separate routines are needed to get an image from disk, write an image to disk, and to display an image on a special device. Interactive, almost instantaneous displays were a critical part in the development of the algorithm in this thesis. By visually examining each step of the algorithm as a picture, debugging and algorithm refinement can take place rapidly. Floating point images, one-byte-per-pixel images, and bitmaps are used by the algorithm for different purposes. It is necessary to have at least four different

conversion subroutines in order to do the appropriate conversions.

SUMMARY

This appendix was included to give the reader an idea of the basic software structure required to implement the surface-based segmentation algorithm. The basic, general-purpose software environment that enables one to build such an algorithm has been described. Once these basic tools are available, the algorithm descriptions in this thesis can be converted to working software.

REFERENCES

REFERENCES

ABELSON, H., AND DISESSA, A.A. 1980. *Turtle Geometry.* MIT Press, Cambridge, Mass.

AGIN, G.J., AND BINFORD, T.O. 1973. Computer description of curved objects. In *Proceedings of 3rd International Joint Conference on* Artificial Intelligence (Stanford, Calif., Aug. 20-23). pp. 629-640.

ANDERSON, R.L., AND HOUSEMAN, E.E. 1942. *Tables of Orthogonal Polynomial Values Extended to N=104.* Research Bulletin 297, Iowa State College of Agriculture and Mechanic Arts, Ames, Iowa. (Apr.).

ASADA, H., AND BRADY, M. 1986. The curvature primal sketch. *IEEE Trans. Patt. Anal. Mach. Intell.* PAMI-8, 1 (Jan.), 2-15.

BALLARD, D.H. AND BROWN, C.M. 1982. *Computer Vision.* Prentice-Hall, Englewood Cliffs, N.J.

BARROW, H.G., AND POPPLESTONE, R.J. 1971. Relational descriptions in picture processing. In *Machine Intelligence VI* (B. Meltzer and D. Michie, Eds.), American Elsevier, New York, pp. 377-396.

BARROW, H.G., AND TENENBAUM, J.M. 1978. Recovering Intrinsic Scene Characteristics from Images. Technical Note 157, SRI International (April).

BARROW, H.G., AND TENENBAUM, J.M. 1981. Computational vision. *Proc. IEEE* **69,** 5 (May), 572-595.

BARTELS, R.H., AND JEZIORANSKI, J.J. 1985. Least-squares fitting using orthogonal multinomials. *ACM Trans. Mathematical Software* **11,** 3 (Sept.), 201-217.

BEAUDET, P.R. 1978. Rotationally invariant image operators. In *Proceedings of 4th International Conference Pattern Recognition* (Kyoto, Japan, Nov. 7-10). pp. 579-583.

BESL, P.J., AND JAIN, R.C. 1986. Invariant surface characteristics for three-dimensional object recognition in range images. *Computer Vision, Graphics, Image Processing* **33,** 1 (January), 33-80.

BESL, P.J., AND JAIN, R.C. 1985. Three-dimensional object recognition. *ACM Computing Surveys* **17,** 1 (March), 75-145.

BESL, P.J., DELP, E.J., AND JAIN, R.C. 1985. Automatic visual solder joint inspection. *IEEE J. Robotics and Automation* **1,** 1 (May), 42-56.

BHANU, B. 1984. Representation and shape matching of 3-D objects. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-6, 3 (May), 340-350.

BOLLE, R.M., AND COOPER, D.B. 1984. Bayesian recognition of local 3-D shape by approximating image intensity functions with quadric polynomials. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-6, 4 (July), 418-429.

BOLLES, R.C., AND FISCHLER, M.A. 1981. A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *Proceedings of 7th International Joint Conference on* Artificial Intelligence (Vancouver, B.C., Canada,

Aug. 24-28). pp. 637-643.

BOLLES, R.C., HORAUD, P., AND HANNAH, M.J. 1983. 3DPO: A three-dimensional part orientation system. In *Proceedings of 8th International Joint Conference on* Artificial Intelligence (Karlsruhe, West Germany, Aug. 8-12). pp. 1116-1120; Also In *Proceedings of 2nd International Symposium on Robotics Research,* (Hanafusa, H. and Inoue, H. Eds.), MIT Press, Cambridge, Mass, pp. 413-424.

BRADY, M. 1982. Computational approaches to image understanding. *ACM Computing Surveys* **14,** 1 (Mar.), 3-71.

BRADY, M., PONCE, J., YUILLE, A., AND ASADA, H. 1985. Describing surfaces. In *Proceedings of 2nd International Symposium on Robotics Research,* (Hanafusa, H. and Inoue, H. Eds.), MIT Press, Cambridge, Mass.

BRICE, C., AND FENNEMA, C. 1970. Scene Analysis Using Regions. *Artificial Intelligence* **1,** 205-226.

BROOKS, R.A. 1981. Symbolic reasoning among 3-D models and 2-D images. *Artificial Intell.* **17,** (Aug.), 285-348.

BROOKS, R.A. 1982. Representing possible realities for vision and manipulation. In *Proceedings of Pattern Recognition and Image Processing Conference* (Las Vegas, Nevada, June 14-17). pp. 587-592.

BROOKS, R.A. 1983. Model-based three-dimensional interpretations of two-dimensional images. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-5, 2 (Mar.), 140-149.

CASASENT, D., VIJAYA-KUMAR, B.V.K., AND SHARMA, V. 1982. Synthetic discriminant functions for three-dimensional object recognition. In *Proceedings of The Society for Photo-Optical Instrumentation Engineers Conference on Robotics and Industrial Inspection,* vol. 360, (San Diego, Calif., Aug. 24-27). SPIE, Bellingham, Wash., pp. 136-142.

CHERN, S.S. 1957. A proof of the uniqueness of Minkowski's problem for convex surfaces. *Am. J. Math.* **79,** 949-950.

COLEMAN, E.N., AND JAIN, R. 1982. Obtaining shape of textured and specular surfaces using four-source photometry. *Comput. Graphics Image Processing* **18,** 4 (Apr.), 309-328.

CONNOLLY, C.I. 1985. The determination of next best views. In *Proceedings of International Conference on Robotics and Automation* (St. Louis, Mo., Mar. 25-28). IEEE-CS, New York, pp. 432-435.

DAHLQUIST, G., AND BJORK, A. 1974. *Numerical Methods.* Prentice-Hall, Englewood Cliffs, N.J. (Translated by N. Anderson).

DANE C. 1982. An object-centered three-dimensional model builder. Ph.D. dissertation, Comp. and Info. Sci. Dept., Moore School of Electr. Eng., Univ. of Penn., Philadelphia, Pa.

DANIEL, W. 1978. *Applied Nonparametric Statistics.* Houghton-Mifflin, Boston, Mass.

DAVIS, L.S. 1975. A survey of edge detection techniques. *Computer Graphics Image Processing* **4,** 248-270.

DIZENZO, S. 1983. Advances in image segmentation. *Image and Vision Computing* **1**, 4 (November), 196-210.

DOUGLASS, R.M. 1981. Interpreting 3-D scenes: a model-building approach. *Comput. Graphics Image Processing* **17**, 2 (Oct.), 91-113.

DRESCHLER, L., AND NAGEL, H.H. 1981. Volumetric model and 3D-trajectory of a moving car derived from monocular TV-frame sequences of a street scene. In *Proceedings of 7th International Joint Conference on Artificial Intelligence* (Vancouver, B.C., Canada, Aug. 24-28), pp. 692-697.

DUDA, R.O., AND HART, P.E. 1972. The use of Hough transform to detect lines and curves in pictures. *Comm. ACM* **15**, 11-15.

DUDA, R.O., NITZAN, D., AND BARRETT, P. 1979. Use of range and reflectance data to find planar surface regions. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-1, 3 (July), 254-271.

EICHEL, P.H. 1985. Sequential Detection of Linear Features in Two-Dimensional Random Fields. Ph.D. dissertation, EECS Dept., Univ. of Mich., Ann Arbor.

FANG, T.J., HUANG, Z.H., KANAL, L.N., LAMBIRD, B., LAVINE, D., STOCKMAN, G., AND XIONG, F.L. 1982. Three-dimensional object recognition using a transformation clustering technique. In *Proceedings of 6th International Conference Pattern Recognition* (Munich, West Germany, Oct. 19-22). pp. 678-681.

FAUGERAS, O.D. 1984. New steps toward a flexible 3-D vision system for robotics. In *Proceedings of 7th International Conference Pattern Recognition* (Montreal, Canada, July 30-Aug.2). pp. 796-805.

FAUGERAS, O.D., HEBERT, M., PAUCHON, E., AND PONCE, J. 1985. Object Representation, Identification, and Positioning from Range Data. In *Proceedings of 2nd International Symposium on Robotics Research,* (Hanafusa, H. and Inoue, H. Eds.), MIT Press, Cambridge, Mass., pp. 425-446.

FAUGERAS, O.D., HEBERT, M., AND PAUCHON, E. 1983. Segmentation of range data into planar and quadric patches. In *Proceedings of 3rd Computer Vision and Pattern Recognition Conference* (Arlington, Va.), pp. 8-13.

FAUX, I.D., AND PRATT, M.J. 1979. *Computational Geometry for Design and Manufacture.* Ellis Horwood, Chichester, U.K.

FELDMAN, J.A., AND YAKIMOVSKY, Y. 1974. Decision theory and artificial intelligence. I. A semantics-based region analyzer. *Artificial Intelligence* **5**, 349-371.

FOLEY, J.D., AND VAN DAM, A. 1982. *Fundamentals of Interactive Computer Graphics.* Addison-Wesley, Reading, Mass.

FU, K.S., AND MUI, J.K. 1981. A survey on image segmentation. *Pattern Recognition* **13**, 3-16.

GENNERY, D.B. 1979. Object detection and measurement using stereo vision. In *Proceedings of 6th International Joint Conference on Artificial Intelligence* (Tokyo, Japan, Aug. 20-23). pp. 320-327.

GEOMOD User Manual and Reference Manual 1983. Structural Dynamics Research

Corporation (SDRC), Cincinnati, Ohio.

GIL, B., MITICHE, A., AND AGGARWAL, J.K. 1983. Experiments in combining intensity and range edge maps. *Comput. Vision, Graphics, Image Processing* **21,** (Mar.), 395-411.

GILBARG, D., AND TRUDINGER, N. 1983. *Elliptic Partial Differential Equations of Second Order.* Springer-Verlag, Berlin, West Germany.

GOLUB, G.H. 1965. Numerical methods for solving least squares problems. Numerische Mathematik **7,** 3, 206-216

GOLUB, G.H., AND VAN LOAN, C.F. 1983. *Matrix Computations.* Johns Hopkins Univ. Press, Baltimore, Md.

GRIMSON, W.E.L., AND PAVLIDIS, T. 1985. Discontinuity detection for visual surface reconstruction. *Computer Vision, Graphics, and Image Processing* **30,** 316-330.

GROGAN, T.A., AND MITCHELL, O.R. 1983. 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,* (January), pp. ThA19-1:ThA19-4,

GUISTI, E. 1978. On the equation of surfaces of prescribed mean curvature: existence and uniqueness without boundary conditions. *Inventiones Mathematicae* **46,** 111-137.

GUZMAN, A. 1968. Computer recognition of three-dimensional objects in a visual scene. MAC-TR-59 (Ph.D. dissertation), Project MAC, MIT, Cambridge, Mass.

HALL, E.L., TIO, J.B.K., MCPHERSON, C.A., AND SADJADI, F.A. 1982. Measuring curved surfaces for robot vision. *Computer* **15,** 12 (Dec.), 42-54.

HANSON, A.R., RISEMAN, E.M., AND NAGIN, P. 1975. Region growing in textured outdoor scenes. In *Proceedings of 3rd Milwaukee Symposium on Automated Computation and Control,* pp. 407-417.

HARALICK, R.M. 1984. Digital step edges from zero-crossings of second directional derivatives. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-6, 1 (Jan.), 58-68.

HARALICK, R.M., AND SHAPIRO, L.G. 1985. Image Segmentation Techniques. *Computer Vision, Graphics, Image Processing* **29,** 100-132.

HARALICK, R.M., AND WATSON, L. 1981. A facet model for image data. *Computer Graphics Image Processing* **15,** 113-129.

HARALICK, R.M., WATSON, L.T., AND LAFFEY, T.J. 1983. The topographic primal sketch. *Int. J. Robotics Res.* **2,** 1 (Spring) 50-72.

HEBERT, M., AND KANADE, T. 1985. The 3-D profile method for object recognition. In *Proceedings of Computer Vision and Pattern Recognition Conference* (San Francisco, Calif., June 9-13), IEEE-CS, New York, pp. 458-463.

HEBERT, M., AND PONCE, J. 1982. A new method for segmenting 3-D scenes into primitives. In *Proceedings of 6th International Conference Pattern Recognition* (Munich, West Germany, Oct. 19-22). pp. 836-838.

HENDERSON, T.C. 1983. Efficient 3-D object representations for industrial vision systems. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-5, 6 (Nov.), 609-617.

HENDERSON, T.C., AND BHANU, B. 1982. Three-point seed method for the extraction of planar faces from range data. In *Proceedings of Workshop on Industrial Applications of Machine Vision* (Research Triangle Park, N.C., May). IEEE, New York, pp. 181-186.

HERMAN, M. 1985. Generating detailed scene descriptions from range images. In *Proceedings of International Conference on Robotics and Automation* (St. Louis, Mo., Mar. 25-28). IEEE-CS, New York, pp. 426-431.

HORAUD, P., AND BOLLES, R.C. 1984. 3DPO's strategy for matching three-dimensional objects in range data. In *Proceedings of the International Conference Robotics.* (Atlanta, Ga., Mar. 13-15). IEEE-CS, New York, pp. 78-85.

HORN, B.K.P. 1977. Understanding image intensities. *Artificial Intell.* **8,** 2 (Apr.), 201-231.

HORN, B.K.P. 1984. Extended Gaussian images. *Proc. IEEE* **72,** 12 (Dec.) 1656-1678.

HOROWITZ, S.L., AND PAVLIDIS, T. 1974. Picture segmentation by a directed split-and-merge procedure. *Proceedings 2nd International Joint Conference Pattern Recognition,* pp. 424-433.

HSIUNG, C.C. 1981. *A first course in differential geometry.* Wiley-Interscience, New York.

HUECKEL, M. 1973. A local operator which recognizes edges and lines. *J. Assoc. Comp. Mach.* **20,** 634-647.

IKEUCHI, K., AND HORN, B.K.P. 1981. Numerical shape from shading and occluding boundaries. *Artificial Intell.* **17,** (Aug.), 141-184.

IKEUCHI, K., HORN, B.K.P., NAGATA, S., CALLAHAN, T., AND FEIMGOLD, O. 1983. Picking up an object from a pile of objects. MIT Artificial Intelligence Lab Memo 726. Cambridge, Mass.

IKEUCHI, K. 1981. Recognition of 3-D objects using the extended Gaussian image. In *Proceedings of 7th International Joint Conference on Artificial Intelligence* (Vancouver, B.C., Canada, Aug. 24-28). pp. 595-600.

INOKUCHI, S., NITA, T., MATSUDAY, F., AND SAKURAI, Y. 1982. A three-dimensional edge-region operator for range pictures. In *Proceedings of 6th International Conference Pattern Recognition* (Munich, West Germany, Oct. 19-22). pp. 918-920.

INOKUCHI, S., AND NEVATIA, R. 1980. Boundary detection in range pictures. In *Proceedings of 5th International Conference Pattern Recognition* (Miami, Fla., Dec. 1-4). pp. 1031-1035.

ITTNER, D.J., AND JAIN, A.K. 1985. 3-D surface discrimination from local curvature measures. In *Proceedings of Computer Vision and Pattern Recognition Conference* (San Francisco, Calif., June 9-13), IEEE-CS, New York, pp. 119-123.

KANADE, T. 1981. Recovery of the three-dimensional shape of an object from a single view. *Artificial Intell.* **17,** (Aug.), 409-460.

KANADE, T. 1980. Survey: Region Segmentation: Signal vs. Semantics. *Computer Graphics Image Processing* **13**, 279-297.

KIM, H.S., JAIN, R.C., AND VOLZ, R.A. 1985. Object recognition using multiple views. In *Proceedings of International Conference on Robotics and Automation* (St. Louis, Mo., Mar. 25-28). IEEE-CS, New York, pp. 28-33.

KNOLL, T.K., AND JAIN, R.C. 1985. Recognizing Partially Visible Objects using feature-indexed hypotheses. RSD-TR-10-85, EECS Dept, Robot Sys. Div., Univ. of Mich., Ann Arbor (July).

KUAN, D.T., AND DRAZOVICH, R.J. 1984. Model-based interpretation of range imagery. In *Proceedings of the National Conference on Artificial Intelligence* (Austin, Tex., Aug. 6-10). American Association for Artificial Intelligence, pp. 210-215.

LANGRIDGE, D.J. 1984. Detection of discontinuities in the first derivatives of surfaces. *Comput. Vision, Graphics, Image Processing* **27**, 3 (Sept.), 291-308.

LAWSON, C.L., AND HANSON, R.J. 1974. *Solving least squares problems*. Prentice-Hall, Englewood Cliffs, N.J.

LEVINE, M.D., AND NAZIF, A.M. 1985. Dynamic measurement of computer generated image segmentations. *IEEE Trans. Patt. Anal. Mach. Intell.* PAMI-7, 2 (March), 155-164.

LIN, C., AND PERRY, M.J. 1982. Shape description using surface triangularization. In *Proceedings of Workshop on Computer Vision: Representation and Control* (Rindge, N.H., Aug. 23-25). IEEE-CS, New York, pp. 38-43.

LIPSCHUTZ, M.M. 1969. *Differential Geometry*. Mc-Graw Hill, New York.

LITTLE, J.J. 1983. An iterative method for reconstructing convex polyhedra from extended Gaussian images. In *Proceedings of the National Conference on Artificial Intelligence* (Washington, D.C., Aug. 22-26). American Association for Artificial Intelligence, pp. 247-250.

LUENBERGER, D.G. 1984. *Linear and Non-linear Programming (2nd.Ed)*. Addison-Wesley, Reading, Mass.

LYNCH, D.K. 1981. Range enhancement via one-dimensional spatial filtering. *Comput. Graphics Image Processing* **15**, 2 (Feb.), 194-200.

MARIMONT, D.H. 1984. A representation for image curves. In *Proceedings of the National Conference on Artificial Intelligence* (Austin, Tex., Aug. 6-10). American Association for Artificial Intelligence, pp. 237-242.

MARR, D. 1976. Early processing of visual information. *Phil. Trans. Royal Soc. Lond. B 275*, 483-524.

MARR, D. 1982. *Vision*. Freeman, New York.

MEDIONI, G., AND NEVATIA, R. 1984. Description of 3-D surfaces using curvature properties. In *Proceedings of the Image Understanding Workshop* (New Orleans, La., Oct. 3-4). DARPA, pp. 291-299.

MILGRIM, D.L. 1979. Region extraction using convergent evidence. *Computer Graphics Image Processing* **11**, 1-12.

MILGRIM, D.L., AND BJORKLUND, C.M. 1980. Range image processing: planar surface extraction. In *Proceedings of 5th International Conference Pattern Recognition* (Miami, Fla., Dec. 1-4). pp. 912-919.

MINKOWSKI, H. 1897. Allgemeine lehrsatze uber die konvexen polyeder. Nachrichten von der Koniglichen Gesellschaft der Wissenschaften, Mathematisch-Physikalische Klasse, Gottingen, pp. 198-219.

MISNER, C.W., THORNE, K.S., AND WHEELER, J.A. 1973. *Gravitation.* W.H. Freeman, San Francisco, Calif. (Box 14.1, Item 7).

MITICHE A., AND AGGARWAL, J.K. 1985. Image segmentation by conventional and information-integrating techniques: a synopsis. *Image and Vision Computing* **3**, 2 (May), 50-62.

MITICHE, A., AND AGGARWAL, J.K. 1983. Detection of edges using range information. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-5, 2 (Mar.), 174-178.

MOKHTARIAN, F., AND MACKWORTH, A. 1986. Scale-based description and recognition of planar curves and two-dimensional shapes. *IEEE Trans. Patt. Anal. Mach. Intell.* PAMI-8, 1 (Jan.) 34-43.

MUERLE, J.L., AND ALLEN, D.C. 1968. Experimental evaluation of techniques for automatic segmentation of objects in a complex scene. In *Pictorial Pattern Recognition* (Cheng et al., Eds.), Thompson, Washington, pp. 3-13.

NACKMAN, L.R. 1984. Two-dimensional critical point configuration graphs. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-6, 4 (July), 442-449.

NACKMAN, L.R. 1982. Three-dimensional shape description using the symmetric axis transform. Ph.D. dissertation, Comp. Sci. Dept., Univ. of N.C., Chapel Hill, N.C.

NEVATIA, R., AND BINFORD, T.O. 1973. Structured descriptions of complex objects. In *Proceedings of 3rd International Joint Conference on Artificial Intelligence* (Stanford, Calif., Aug. 20-23). pp. 641-647.

NEVATIA, R., AND BINFORD, T.O. 1977. Description and recognition of curved objects. *Artificial Intell.* **8**, 1, 77-98.

NEWMAN, W.M., AND SPROULL, R.F. 1979. *Principles of Interactive Computer Graphics, 2d Ed.* McGraw-Hill, New York.

NITZAN, D., BRAIN, A.E., AND DUDA, R.O. 1977. The measurement and use of registered reflectance and range data in scene analysis. *Proc. IEEE* **65**, (Feb.), 206-220.

OHLANDER, R. 1975. Analysis of natural scenes. Ph.D. dissertation, Dept. of Comp. Sci., Carnegie-Mellon Univ., Pittsburgh, Pa.

O'NEILL, B. 1966. *Elementary Differential Geometry.* Academic Press, New York.

OSHIMA, M., AND SHIRAI, Y. 1983. Object recognition using three-dimensional information. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-5, 4 (July), 353-361.

PALMER, S. 1983. The psychology of perceptual organization: a transformational approach. *Human and Machine Vision* (Beck et al., Eds.), Academic Press, New York, pp. 269-340.

PAVLIDIS, T. 1972. Segmentation of pictures and maps through functional approximation. *Computer Graphics Image Processing* **1,** 360-372.

PEET, F.G. AND SAHOTA, T.S. 1985. Surface curvature as a measure of image texture. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-7, 6 (November), 734-738.

PONCE, J., AND BRADY, M. 1985. Toward a surface primal sketch. In *Proceedings of International Conference on Robotics and Automation* (St. Louis, Mo., Mar. 25-28). IEEE-CS, New York, pp. 420-425.

PONG, T.C., SHAPIRO, L.G., WATSON, L.T., AND HARALICK, R.M. 1984. Experiments in segmentation using a facet model region grower. *Computer Vision, Graphics, and Image Processing* **25,** 1-23.

POPPLESTONE, R.J., BROWN, C.M., AMBLER, A.P., AND CRAWFORD, G.F. 1975. Forming models of plane-and-cylinder faceted bodies from light stripes. In *Proceedings of 4th International Joint Conference on Artificial Intelligence* (Tbilisi, Georgia, USSR, Sept.). pp. 664-668.

POTMESIL, M. 1982. Generating three-dimensional surface models of solid objects from multiple projections. IPL-TR-033, Ph.D. dissertation, Image Proc. Lab, RPI, Troy, NY.

POTMESIL, M. 1983. Generating models of solid objects by matching 3D surface segments. In *Proceedings of 8th International Joint Conference on Artificial Intelligence* (Karlsruhe, West Germany, Aug. 8-12). pp. 1089-1093.

PREWITT, J. 1970. Object enhancement and extraction. In *Picture Processing and Psychopictorics,* B. Lipkin and A. Rosenfeld, Eds., Academic Press, New York, pp 75-149.

PREWITT, J.S.M., AND MENDELSOHN, M.L. 1966. The analysis of cell images. *Ann. N.Y. Acad. Sci.* **128,** 1035-1053.

REEVES, A.P., PROKOP, R.J., AND TAYLOR, R.W. 1985. Shape analysis of 3-D objects using range information. In *Proceedings of Computer Vision and Pattern Recognition Conference* (San Francisco, Calif., June 19-23), IEEE-CS, New York, pp. 452-457.

RISEMAN, E.M., AND ARBIB, M.A. 1977. Computational techniques in the visual segmentation of static scenes. *Computer Graphics and Image Processing,* **6,** 221-276.

ROBERTS, L.G. 1965. Machine perception of three-dimensional solids. *Optical and Electro-Optical Information Processing.* J.T. Tippett et al., Eds., MIT Press, Cambridge, Mass. pp. 159-197.

ROSENFELD, A. 1978. Iterative methods in image analysis. *Pattern Recognition* **10,** 181-187.

ROSENFELD, A., AND DAVIS, L.S. 1979. Image segmentation and image models. *Proc. IEEE* **67,** 5 (May), 764-772.

ROSENFELD, A., AND KAK A. 1982. *Digital Picture Processing,* vols. 1 and 2. Academic Press, New York. (1st Ed. 1976)

ROSENFELD, A., AND THURSTON, M. 1971. Edge and curve detection for visual

scene analysis. *IEEE Trans. Computers* **C-20,** 562-569.

ROSENFELD, A., HUMMEL, R.A., AND ZUCKER, S.W. 1976. Scene labeling by relaxation operations. *IEEE Trans. Systems, Man, Cybernetics* **6,** 6, 420-433.

SADJADI, F.A., AND HALL, E.L. 1980. Three-dimensional moment invariants. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-2, 2 (Mar.), 127-136.

SATO, Y., AND HONDA, I. 1983. Pseudodistance measures for recognition of curved objects. *IEEE Trans. Pattern Anal. Machine Intell.* PAMI-5, 4 (July), 362-373.

SETHI, I.K., AND JAYARAMAMURTHY, S.N. 1984. Surface classification using characteristic contours. In *Proceedings of 7th International Conference Pattern Recognition* (Montreal, Canada, July 30-Aug.2). pp. 438-440.

SHAHRARAY, B. AND ANDERSON, D.J. 1985. Uniform Resampling of Digitized Contours. IEEE Trans. Pattern Anal. Machine Intell. PAMI-7, 6 (November), 674-681.

SHIRAI, Y., AND SUWA, M. 1971. Recognition of polyhedra with a range finder. In *Proceedings of 2nd International Joint Conference on Artificial Intelligence* (London, U.K., Aug.). pp. 80-87.

SMITH, D.R., AND KANADE, T. 1984. Autonomous scene description with range imagery. In *Proceedings of the Image Understanding Workshop* (New Orleans, La., Oct. 3-4). DARPA, pp. 282-290.

SNYDER, W., AND BILBRO, G. 1985. Segmentation of Three-Dimensional Images. In *Proceedings of International Conference on Robotics and Automation* (St. Louis, Mo., Mar. 25-28). IEEE-CS, New York, pp. 396-403.

SUGIHARA, K. 1979. Range-data analysis guided by junction dictionary. *Artificial Intell.* **12,** 41-69.

SVETKOFF, D.J., LEONARD, P.F., SAMPSON, R.E., AND JAIN, R.C. 1984. Techniques for real-time 3D feature extraction using range information. In *Proceedings of The Society for Photo-Optical Instrumentation Engineers Conference on Intelligent Robotics and Computer Vision,* vol. 521, (Cambridge, Mass., Nov. 5-8).

TENNEBAUM, J.M., AND BARROW, H.G. 1976. IGS: a paradigm for integrating image segmentation and interpretation. In *Proceedings 3rd International Joint Conference Pattern Recognition,* pp. 504-513.

TERZOPOLOUS, D. 1985. Computing visible surface representations. AI Memo No. 800, MIT Artif. Intell. Laboratory, Cambridge, Mass. (March)

TERZOPOULOS, D. 1983. Multilevel computational processes for visual surface reconstruction. *Comput. Vision, Graphics, Image Processing* 24, 52-96.

TOMITA, F., AND KANADE, T. 1984. A 3D vision system: generating and matching shape descriptions in range images. In *Proceedings of the International Conference Robotics.* (Atlanta, Ga., Mar. 13-15). IEEE-CS, New York, pp. 186-191.

TSUJI, S., AND TOMITA, F. 1973. A structural analyzer for a class of textures. *Comput. Graphics Image Processing* **2,** 216-231.

TURNEY, J.L., MUDGE, T.N., AND VOLZ, R.A. 1985. Recognizing Partially Occluded Parts. *IEEE Trans. Pattern Anal. Mach. Intell.* PAMI-7, 4 (July), 410-421.

WALLACE, T.P., AND WINTZ, P.A. 1980. An efficient three-dimensional aircraft recognition algorithm using normalized Fourier descriptors. *Comput. Graphics Image Processing* **13**, 96-126.

WALTZ, D.L. 1972. Generating semantic descriptions from drawings of scenes with shadows. AI-TR-271, MIT Artificial Intelligence Lab, Cambridge, Mass., (Nov.).

WATSON, L.T., LAFFEY, T.J., AND HARALICK, R.M. 1985. Topographic classification of digital image intensity surfaces using generalized splines and the discrete cosine transformation. *Comput. Vision, Graphics, Image Processing* **29**, 143-167.

WESZKA, J.S. 1978. A survey of threshold selection techniques. *Comput. Graphics Image Processing* **7**, 259-265.

WITKIN, A.P. 1981. Recovering surface shape and orientation from texture. *Artificial Intell.* **17**, (Aug.), 17-45.

WITKIN, A.P., AND TENNEBAUM, J. 1983. The role of structure in vision. *Human and Machine Vision.* (Beck et al., Eds.), Academic Press, New York, pp. 481-543.

WONG, R.Y., AND HAYREPETIAN, K. 1982. Image processing with intensity and range data. In *Proceedings of Pattern Recognition and Image Processing Conference* (Las Vegas, Nevada, June 14-17). pp. 518-520.

WOODHAM, R.J. 1981. Analysing images of curved surfaces. *Artificial Intell.* **17**, (Aug.), 117-140.

WOODHAM, R.J. 1977. A cooperative algorithm for determining surface orientation from a single view. In *Proceedings of 6rd International Joint Conference on Artificial Intelligence,* pp. 635-641.

YASNOFF, W.A, MUI, J.K., AND BACUS, J.W. 1977. Error measures for scene segmentation. *Pattern Recognition* **9**, 217-233.

YOUNG, R.A. 1985. The Gaussian derivative theory of spatial vision: analysis of cortical cell receptive field line-weighting profiles. GMR-4920, General Motors Research, Warren, Mich.

ZUCKER, S.W. 1976. Region growing: Childhood and Adolescence. *Computer Graphics Image Processing* **5**, 382-399.

ZUCKER, S.W., AND HUMMEL, R.A. 1981. A three-dimensional edge operator. *IEEE Trans. Pattern Anal. Mach. Intell.* PAMI-3, 3, 324-331.