205400-8-F

FINAL REPORT

ON THREE DIMENSIONAL OBJECT RECOGNITION AND POSE-DETERMINATION: AN ABSTRACTION BASED APPROACH

KOK HOW FRANCIS QUEK

JUNE 1990

Under NASA Grant:
NAGW-1198

SpARC
Space Automation & Robotics Center
(Formerly CAMRSS)

NASA
Sponsored

P.O. Box 8618
Ann Arbor, MI 48107-8618

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE June 1990	3. REPORT TYPE AND DATES COVERED Dissertation (Final) June 1990

4. TITLE AND SUBTITLE

On Three-Dimensional Object Recognition and Pose-Determination: An Abstraction Based Approach

5. FUNDING NUMBERS

NAGW-1198

6. AUTHOR(S)

Kok How Francis Quek

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

ERIM/SpARC (Formerly CAMRSS)
P.O. Box 8618
Ann Arbor, MI 48107-8618

8. PERFORMING ORGANIZATION REPORT NUMBER

205400-8-F

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

NASA Office of Commercial Programs
NASA Headquarters
Washington, DC 20546

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report is a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Computer Science in Engineering) in The University of Michigan. The work received partial support from the Space Automation and Robotics Center (formerly CAMRSS). (See attached Abstract)

14. SUBJECT TERMS	15. NUMBER OF PAGES 244
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

ABSTRACT

A method of computing reliable Gaussian and mean curvature sign-map descriptors from the polynomial approximations of surfaces was demonstrated. Such descriptors which are invariant under perspective variation are suitable for hypothesis generation.

A means for determining the pose of constructed geometric forms whose algebraic surface descriptions are non-linear in terms of their orienting parameters was developed. This was done by means of linear functions which are capable of approximating non-linear forms and determining their parameters. It was shown that biquadratic surfaces are suitable *companion-linear forms* for cylinder approximation and parameter estimation. The estimates provided the initial parametric approximations necessary for a non-linear regression stage to fine tune the estimates by fitting the actual non-linear form to the data.

A hypothesis-based split-merge algorithm for extraction and pose determination of cylinders and planes which merge smoothly into other surfaces was developed. It was shown that all split-merge algorithms are hypothesis-based.

A finite-state algorithm for the extraction of the boundaries of run-length regions was developed. The computation takes advantage of the run list topology and boundary direction constraints implicit in the run-length encoding.

ACKNOWLEDGEMENTS

In the course of my doctoral work, Prof. Ramesh Jain has been a wonderful mentor and thesis advisor. While he allowed me the freedom to develop my own ideas and to investigate my curiosities, I always had his insight and interaction so that I was confident that I would not go too far wrong. Beyond being my thesis advisor, he is also a friend to whom I am truly grateful.

I also appreciate the many discussions I had with Dr. Brian Mitchell of the Environmental Research Institute of Michigan throughout the work. Through interactions with Dr. Charles Jacobus of Cybernet Systems Corp. (then director of the NASA Center of Autonomous and Man-Controlled Robotic and Sensing Systems) concerning robotic needs, the more concrete ideas of this thesis germinated. Discussions with Prof. Terry Weymouth on abstraction hierarchies were thought provoking and led to the formulation of the abstraction paradigm.

Financially, I am happy to acknowledge the support of the Environmental Institute of Michigan, NASA/CAMRSS and the University of Michigan.

Finally, my gratitude goes to my wife Lee-Lian for her patience and support at the most trying times, and to the members of my church for their prayers and needed fellowship and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

PRECEDING PAGE BLANK NOT FILMED

LIST OF TABLES

Table

CHAPTER I

INTRODUCTION

Image understanding is a field rich in prospect, approaches and methodologies. This fact is attested to by the many journals, conferences, books and other publications on the subject. There are myriads of papers describing such aspects of computer vision as feature detection, image segmentation, image acquisition technologies, stereo vision, interest operators of all kinds, planar surface fitting, curved surface recognition, object modeling, matching methodologies, object recognition schemes etc. There does not seem to be a want of research attempts. Yet, computer vision systems developed to date remain fragile and the oft mentioned integration of the various research areas into a cogent whole remains ellusive. A simple task was used to focus the research of this thesis. A robot equipped with a laser range scanner is required to recognize an object from a library of objects and to determine its three-dimensional pose to a degree of accuracy which permits a hole to be drilled in the object to specification (e.g. a perpendicular hole through the side of the cylindrical portion of a machine part an inch from the end of the cylinder). No system to date is able to perform this task.

This author believes that much of the problem facing computer vision lies in the paucity of techniques to integrate the various components. Broadly speaking, all of computer vision research may be divided into three classes. First, there are the bottom-up image-based (low-level) operations; second, there are the top-down knowledge intensive (high-level vision) processes; and, third, there is the representation of the data in a manner suitable for the computer vision task. Most computer vision object recognition systems thus far developed emphasize one of these three aspects. This has proven insufficient. Even systems which incorporate all the necessary components are invariably dominated by one of the components. Knowledge-driven systems depend a lot on top-down processes to drive the interpretation of features generated by the bottom-up processes. Such 'complete' systems are scarce, and they too, have met with limited success. Left to itself, the deficiencies of the top-down component and the inability to exploit the strengths of bottom-up processes ultimately overwhelm the strengths of the top-down component.

When one examines the threefold division of computer vision as stated, one finds that the edges of demarcation between high and low level vision want clear definition. It is easy to talk about the detection of edges, for example, as low-level vision and the matching of extracted silhouettes of objects to model silhouettes as high-level vision, but the distinction blurs when one discusses such primitives or features as generalized-cylinders [46, 48, 49, 51], extended Gaussian images [12, 100, 52, 102, 103, 126], extended Hough's transforms on planar faces [12, 64, 68], smoothed local symmetries [44] etc. For these features to be computed, knowledge of various degrees about the scene is required. Even with the computation of edges, the application of scale space edge detectors [18, 61], direction dependent edge detectors [53, 54] and multi-scale boundary classification [7, 71] require scene knowledge as well. The distinction is not inconsequential. The process of extracting features from images is one of abstracting information from the pixel data. The lack of a means of organizing this process of abstraction contributes to the binary high-low level vision architecture and the scarcity of techniques to build hierarchical visual reasoning systems.

This dissertation details an *abstraction-based* paradigm in which a hierarchical process of successive refinement may be implemented. While the paradigm outlines a significantly different approach to field of object recognition than those contained in the literature, a narrow goal was chosen to focus the research and to flesh out the paradigm. The goal was to develop technologies which will allow a robot equipped with a laser range imaging device to perform the recognition and pose determination task outlined in the beginning of this section. An abstraction hierarchy is advanced for range image understanding along with techniques for making refinements to traverse the hierarchy.

1.1 What is Abstraction?

A fundamental task in bridging the chasm between sensed data and an understanding of the scene is that of coalescing massive quantities of numeric data into symbolic entities or features. These features provide a semantic description of the objects or scene contained in the sensed data. Under the bipartite regimen in which the low-level system components generate features for interpretation by the high-level system components, it is not uncommon that a single step separates the feature detection from object recognition. A framework is necessary for multi-levelled reasoning with the data.

The venerated solution to problems of this nature in computer science is to define a hierarchy into which the processes and data may be arranged. Within such a hierarchy, the operative task is to provide, in higher layers of the hierarchy, organization to subsets of data in lower layers of the hierarchy. Such a hierarchy of abstraction may be applied to model the implicit hierarchies of most computer vision systems.

> Abstraction is a condensation of data into units of increasing semantic significance by the application of certain assumptions. The degree of abstraction is measured by the specificity of the assumptions necessary for its computation.

By this definition, the more specific the assumption made to arrive at the abstraction of a particular unit (feature or object), the more abstract is that unit. The assumptions made to perform a particular abstraction is called its *abstracting assumptions*. As an illustration of such hierarchical abstraction, consider the example of the computation of shape from contours [15, 42, 43, 98, 111].

First, edge points may be computed in an intensity image by the application of an operator that extracts points of discontinuity in the intensity values. The abstraction in this case is based upon the general assumptions that true edge points of a three-dimensional object coincide with such discontinuities, that the object surface is lambertian and often, that the illumination is approximated by a distant point light source. The generation of closed contours by edge point linking is based upon a stronger assumption that the edge points are correct (sometimes by examining of edge bright-to-dark directions) and that the contours must be closed and is thus higher in the hierarchy. The computation of the surface shape from the contours is based upon the even stronger assumption that the closed contour belongs to a surface, [often] that the surface is planar and that the surface shape is regular.

1.2 Models and Abstraction

The notion of models is very closely tied to abstraction. Models are overarching units into which data are coalesced. They possess both the capacity to account for data and the ability to impute meaningful interpretation upon the data accounted for. Models may take many forms. They may be mathematical descriptions (e.g. polynomial patches), relation descriptions (e.g. graph-based descriptions of surfaces in an object or syntactic pattern strings), property vector descriptions (e.g. feature vectors used in statistical pattern recognition) etc.

Since data may be coalesced into models which account for and impute meaningful interpretation upon data, a hierarchy of abstraction can be implemented as a hierarchy of models.

We shall now contrast the *abstraction-based* paradigm against the usual formulation of object recognition systems which we shall call the *feature-model* paradigm.

1.2.1 The Feature-Model Paradigm

For the sake of discussion we shall call the system described in figure 1.1 the *feature-model* paradigm which divides computer vision into three major components: feature extraction, model

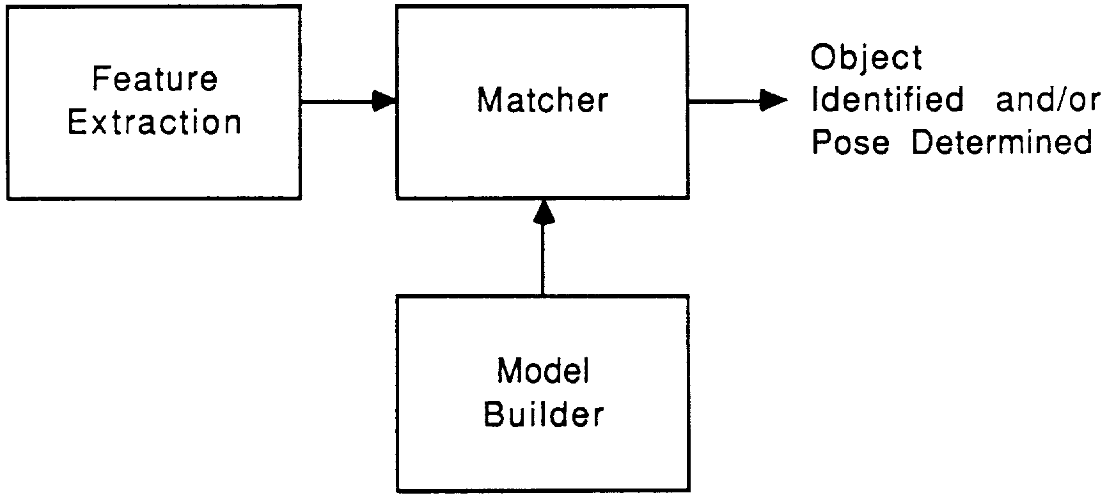


Figure 1.1: Block diagram of a basic model-based image understanding system.

building or representation, and model matching.

Feature extraction addresses the issue of deriving features pertinent to the recognition process from sensed data. It encompasses the problem of the selection of appropriate features and the computation necessary to glean these features from the sensed imagery. *Model building or representation* is the task of describing the objects to be recognized in a manner which facilitates their recognition. *Model matching* is the process in which the features extracted from sensed data are mapped into the set of object models so as to determine the presence of objects and their pose.

The three major components are insufficient in themselves to solve the image understanding problem. The interaction among the components is of utmost importance.

The problems with this basic system are twofold. First, the system is open-looped so that the object model is not able to guide the feature extraction process. Second, in the task of three-dimensional image understanding, features tend to deform with varying viewpoint owing to occlusion and perspective change.

Detailed discussions on model-based object recognition systems can be found in [19, 32, 60].

Feature Extraction in the *Feature-Model* Approach

The feature extraction process of the *feature-model* configuration shown in Figure 1.1 operates autonomously from the rest of the system. A set of features with a corresponding set of detection operators is chosen to generate input to the recognition system which tries to match these features to those in a set of models. Examples of such features are straight edges, surface curvature, polyhedra, planar surfaces, Gaussian sphere characteristics [9, 41, 52, 58, 59, 75, 100, 102, 117, 118, 126, 130, 133] etc. Features are often chosen for their ability to discriminate among objects which the system is designed to recognize. The designated feature detection operator is run over the input images to obtain feature points.

The problem with the feature extraction operations in the context of the *feature-model* system of Figure 1.1 is that they fail to utilize the knowledge contained in the system's model base. This failure to exploit the model level knowledge is paid for in terms of the imprecision of the detected features and the high cost of exhaustive feature extraction. All feature detectors are sensitive to noise and aberrations in the illumination and sensing environment. The application of a general purpose edge detector, for example, will generate many 'false edges' in addition to the portions of edges truly present. False edges will then have to be removed and true edges completed by some process which links the fragmented edge segments and fills in missing segments. Finally, the still vast quantity of data in the incompletely segmented edge image is passed to the matcher to be matched with the object models. During matching, attempts are sometimes made to separate the sheep of actual edges from the goats of extraneous edges (actually the matcher often has to breed new sheep by completing disconnected true edges). These are potentially combinatorially explosive undertakings. Furthermore, in the event of erroneous segmentation, the system has no way to direct the feature extraction module to produce an alternate segmentation and no avenue to guide that resegmentation process. The lack of guidance (what to look for and where to look) also precludes the use of specialized feature detectors.

Object Modeling and Representation

The second problem with the image understanding model of Figure 1.1 resides in the object modeling and representation module labelled *Model Builder.* In the task of three-dimensional image understanding, features tend to deform with varying viewpoint owing to occlusion and perspective projection. A possible solution to this problem is the use of multiview representations. This representation has been termed variously as aspect graphs, visual potentials, characteristic views etc. [56, 57, 76, 77, 95, 136]. The impetus behind these approaches is that they simplify the matching process. Each characteristic view can be thought of as a different model, and the matching problem is reduced to finding a characteristic view model which satisfies the features found in the sensed data. The problem with such approaches is that the storage requirements increase dramatically with complexity of the objects being modeled as more characteristic views need to be maintained. The subsequent increase in the number of views stored also increases the combinatorics of the matching process. A good discussion of multiview representations can be found in [121]. Another approach is that embodied in ACRONYM. ACRONYM makes use of a volumetric representation in which all objects are decomposed into a set of generalized cones. These cones are described by a length, an orientation, a cross-sectional shape and a sweep rule which determines the variation in the cross-sectional area (of the same shape) along the length of the cone [46, 48, 49, 51]. Since the perspective projections of cones yield generalized ribbons, ACRONYM is able to 'predict' the perspective view of the object on-the-fly. As will be discussed later, ACRONYM's modeling scheme has its shortcomings as well.

Matching

Most *feature-model* systems represent the objects to be recognized as graphs or as feature vectors. In graph representation the nodes usually correspond to features of the object and the arcs indicate the spatial relationships between these features. The object recognition problem is thus reduced to one of matching of pairs of graphs. Examples of such systems are described in [37, 38, 39, 46, 48, 49, 51, 58, 59, 80, 152, 153]. Of these, ACRONYM [46, 48, 49, 51] is the most significant attempt at a complete image understanding system. In pattern classification techniques, feature vectors are made up of numeric evaluations of features. Recognition is performed using a similarity measure between the observed feature vector computed from the data and the model feature vectors.

A major problem with traditional matching techniques (with the acception of global statistical feature vector methods which will be discussed later) is that they attempt to match all available models with the detected features sequentially. The matching complexity is thus given by the product of the number of models and the number of features generated by the feature extraction process. Given the large number of real and false features detected, this can prove very expensive. Another problem with the system of Figure 1.1 is that without interaction between the knowledge contained in the model base and the feature extraction module, the matching process distances itself too rapidly from the image data. The matcher knows only of the symbolic data on the graphs to be matched and any error in the construction of either graph is irreversible.

1.2.2 The Abstraction-Based Paradigm

Figure 1.2 is the conceptual block diagram of an abstraction based recognition configuration. We shall call this the *abstraction-based* recognition paradigm. Instead of a spontaneous feature extractor which operates indiscriminately upon the data, features are dealt with as abstractions of the data which, upon verification, become data (albeit more abstract than the data it models) upon which the system makes further abstractions. This allows the system to use more specific feature models which are not generally applicable to all data. This specificity-generality contrast and the trade-offs between them are discussed in the ensuing section.

The *abstraction-based* paradigm employs a hypothesis-test strategy to distill abstractions

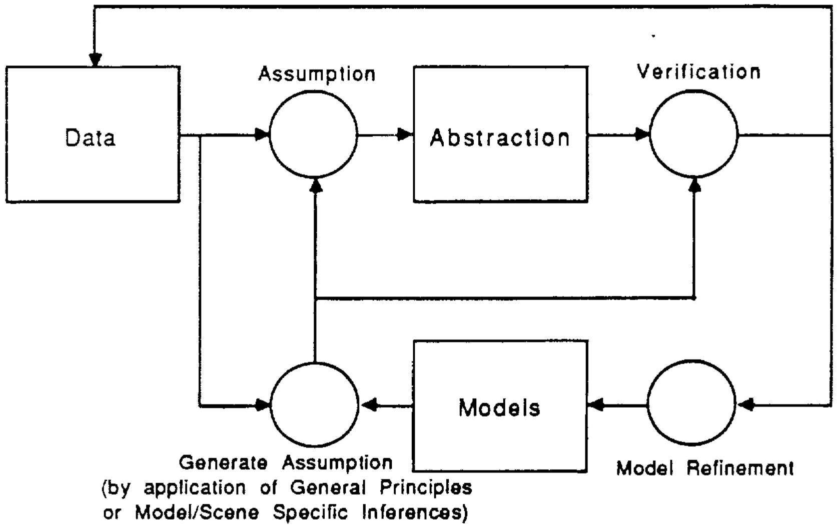


Figure 1.2: Block diagram of the *abstraction-based* paradigm.

out of the data. The object models and the scene cooperate to generate assumptions which translate to particular interpretations (or abstractions) of subsets of the data. These abstractions are instantiated and tested to see if the assumption applies. The resulting accepted abstractions are both incorporated into the data-set and used to refine the model-base with greater context information (e.g. pose information, presence of particular objects or sub-parts of objects serve to constrain what objects are likely).

Assumptions may be generated either by the application of general mathematical and physical principles (e.g. three-dimensional edges occur at discontinuities in the first derivatives in an image) or by model-scene specific inferences (e.g. by indexing or matching). Initially, when there is no information to make specific inferences as to what is in the scene, general assumptions are applied. The features thus generated provide new scene specific information which allows stronger assumptions to be drawn.

The difference between the *abstraction-based* paradigm from traditional hypothesis-test techniques operating under the *feature-model* approach is that the *feature-model* approach reasons with precomputed features while the *abstraction-based* approach reasons about the features which need to be computed.

The *feature-model* approach attempts to impose organization on precomputed features to see if they match models. The hypothesis-test strategy when applied in a *feature-model* system performs recognition by forming hypotheses to explain how the precomputed features may be organized to conform to the models and testing the hypotheses. The *abstraction-based* paradigm makes explicit assumptions upon which features are computed. Upon computation of the hypothesized features, the *abstraction-based* paradigm tests to see if the hypotheses were correct and uses the resulting information to make further assumptions on the data. The key difference is thus that the *abstraction-based* paradigm is able to work with the directly image data throughout the abstraction process, computing features directly on the data. In the *feature-model* paradigm, the feature extractor computes all the features for which it has been programmed 'spontaneously', leaving the interpretation of this data to the 'high-level' system. The *feature-model* system is thus unable to compute features which are relevent only to specific portions of an image and under specific circumstances.

Figure 1.2 serves only as a conceptual block diagram The purpose of this thesis is to provide the processing tools and concepts for making the necessary abstractions from range data, not to design a control architecture to implement such a system.

1.3 Generality vs. Specificity

Up to this point, nothing has been said about the models which constitute the units into which data is coalesced. The models to which data may be matched may be global, logical and relational or numeric in nature (see figure 1.3).

Global matching is usually framed as the matching of feature vectors describing an object to a set of model feature vectors. Such features include two dimensional eccentricity, aspect ratio, Fourier descriptors, area, perimeter length etc. computed globally across the entire object. Statistical pattern recognition techniques are by and large global matching techniques.

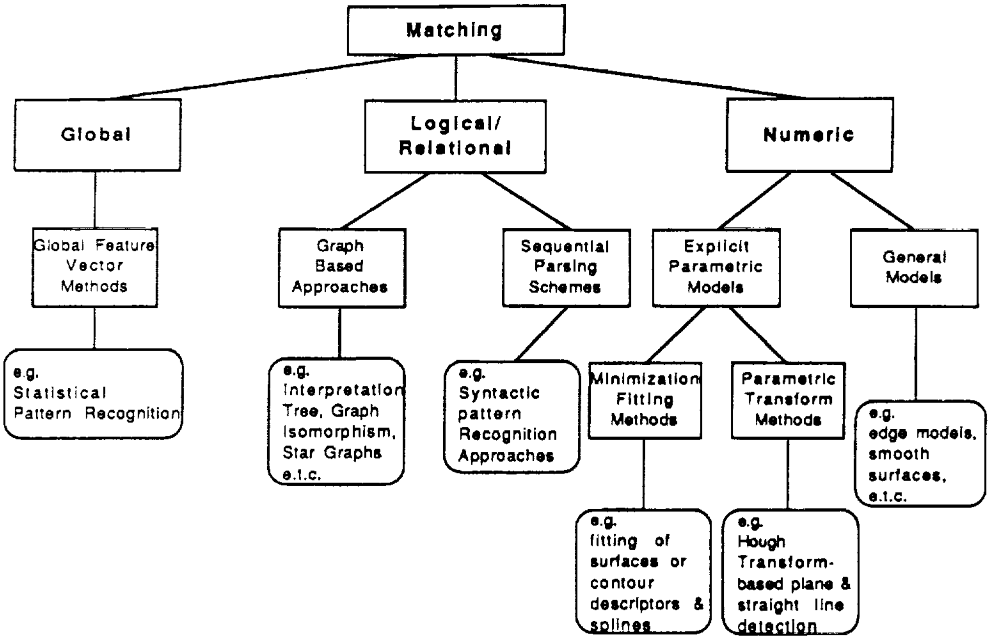

Matching
Global
Logical/ Relational
Numeric
Global Feature Vector Methods
Graph Based Approaches
Sequential Parsing Schemes
Explicit Parametric Models
General Models
e.g. Statistical Pattern Recognition
e.g. Interpretation Tree, Graph Isomorphism, Star Graphs e.t.c.
e.g. Syntactic pattern Recognition Approaches
Minimization Fitting Methods
Parametric Transform Methods
e.g. edge models, smooth surfaces, e.t.c.
e.g. fitting of surfaces or contour descriptors & splines
e.g. Hough Transform-based plane & straight line detection

Figure 1.3: Classes of matching processes in computer vision

Logical/relational matching considers iconic objects (or features) detected in a scene and the relations, both spatial (e.g. above, below, parallel to etc.) and logical (e.g. member-of, is-a etc.) among them. It attempts to find a corresponding set of icons and relations in a model which matches those found in a scene. Examples of this are graph matching techniques such as graph isomorphism, star graph and interpretation tree approaches and sequential parsing schemes such as in syntactic pattern recognition.

Numerical model matching takes one of two forms. The first is an explicit application of some numeric description of an object or feature to the data to obtain the parameters in the description which would account for the data. Examples of this are Hough transforms to extract straight line segments (the straight line equation is applied for the transformation), explicit fitting (usually least-squares) of surface or contour descriptors like bi-variate polynomials, splines, quadrics etc. The second is the application of some general model to extract image properties without explicit fitting or determination of the model parameters. Examples of this are the application of various edge models which describe edges as discontinuities in the intensity distribution of an image, morphological filtering techniques to extract smooth regions etc.

The models to which data are matched may be general in the sense that they can be applied to a broad category of objects or signals. Edge models, polynomial surfaces and surface curvature computations fall into this category. Such models have substantial matching capability in that they are capable of matching large variety of data, but, they have little discriminating power. Specific models, on the other hand, are adept at discriminating data arising from different objects being viewed, but they are incapable of matching a variety of objects in the data. Examples of specific models are cylindrical surfaces, planes, circles, straight lines etc. The most specific model that can be matched to data representing an object is the model of the object itself.

There needs to be a trade off between general and specific models in a computer vision system. It may be observed that general models apply general assumptions for their computation and matching while specific models rely on more particular assumptions. This trade off can be achieved within an abstraction hierarchy. Successive refinement or abstraction permits a system to deal with data in each layer of the hierarchy in a modular fashion while maintaining the relevance of the processing and permitting increasing semantic attribution to collections of data. Each layer 'understands' what lower levels of processing presents to it and presents its results to higher levels of processing, and the input data is maintained at each layer either for forwarding to higher layers or for reprocessing. Each layer also 'understands' what it is asking of lower processing levels.

1.4 Hypotheses in Image Understanding

Specificity is the objective of the recognition. What is desired is the model which is related by identity to the object being viewed. By our earlier discussion, however, the abstraction of highly specific entities from data requires very particular assumptions about the scene. Apart from the trivial case of matching a small number models to a scene which may contain no other objects (where the specifying assumption is precisely that only one of a small set of objects may be in view), it is impossible to match all combinations of data to all combinations of models.

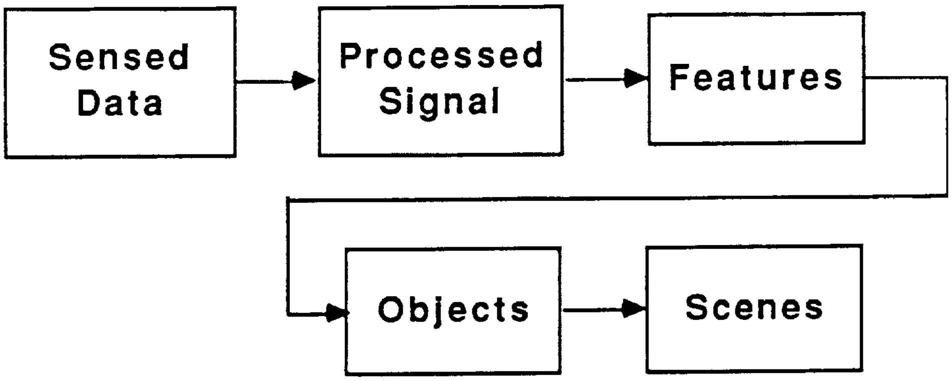


Figure 1.4: The general sequence of computation in computer vision

A logical solution would be a process of assumption (or abstraction) refinement to bridge the progression from the general to the specific.

The key element for the introduction of greater specificity is greater understanding of the context (information about the scene at large). A hypothesis-test strategy would facilitate such context enriching interaction among the layers for computer vision.

Within the hypothesis-test paradigm, data at lower levels of the hierarchy provide evidence for the forming of higher level overarching structures, while higher level structures serve as hypotheses for the organization of sub-sets of lower level data which may be tested at the lower levels of the hierarchy. The former problem (hypothesis generation by evidence) is that of indexing [83, 84, 85, 116, 117, 118] and the latter is that of hypothesis verification.

Figure 1.4 shows the general sequence of computation in computer vision. One begins with the data (in image format), processes it (still in image format), extracts features from the processed data, and then attempts to recognize objects in the scene by manipulating the features. Objects could then be manipulated to construct the scene. This progression is one of abstraction.

This thesis proposes that the 'feature layer' ought to be divided into two portions. The first is *context free* in its computation while the second is *context sensitive.*

Context free features are features whose computation is based upon generally applicable assumptions of physics, illumination and/or mathematics. Such assumptions include contiguity of surfaces, step edges at intensity/range discontinuities, etc. Examples of such features are spline surfaces, polynomial surfaces, curvature-based classifiers, edge points, edge segments etc.

Context sensitive features are features or groups of features whose computation or detection is dependent upon the content of the scene. This strict dependence upon specific hypothesis-driven assumptions is necessary in order to overcome the combinatorics of matching (fitting) all data to all possible features and to overcome the complexities introduced by perspective variation. Examples of such features are constructed forms (e.g. planes, cylinders, cones, etc.), perspective sensitive forms (e.g. silhouettes, oriented surfaces etc.), feature relationships (not all possible relationships are entertained) etc.

A hypothesis-test strategy thus provides a platform for the computation or detection of any context sensitive feature or object.

1.5 Thesis Statement

In a nutshell, this thesis advances an *abstraction-based* paradigm which makes explicit the process of imposing assumptions on data. This *general to specific* refinement process provides a mechanism to proceed gracefully from low level to high level vision processes and vice versa. The task of specifying what is in a scene becomes one of making stronger and stronger assumptions about what is in the image. This process of assumption and abstraction furnishes a path between symbolic descriptors of objects in the scene to their numeric specification. The lack of this interaction has been a major hindrance to the development of robust vision systems.

Other major contributions of this thesis are: the demonstration of a robust method of computing reliable surface curvature descriptors; the development of a means of determining the pose of constructed geometric forms whose algebraic surface descriptions are non-linear in terms of their orienting parameters; the analysis, proof and demonstration that biquadratic surfaces are capable of approximating cylinders and estimating their three-dimensional pose; and the development of a hypothesis-based split-merge algorithm (it was shown that all split-merge algorithms are hypothesis-based) for extraction and pose determination of cylinders and planes which merge smoothly into other surfaces.

1.6 Thesis Layout

The focus of this chapter is the introduction of the theme of abstraction as an ordering principle for hierarchical computer vision and the discussion of the interaction between abstraction and hypothesis generation and test strategies. Chapter II overviews previous work in three-dimensional image understanding; chapter III lays out the underlying strategy employed in the work of this thesis for three-dimensional image understanding; chapter IV reviews and discusses the necessary background material (mathematics and algorithms) for the techniques employed in this thesis; chapter V details the split-merge segmentation technique employed; chapter VI describes the process of estimating and fitting cylindrical surfaces; chapter VII discusses a fast algorithm for extracting boundaries from run-length region descriptions; chapter VIII describes the experimental work performed in the course of the research; and, conclusions are drawn in chapter IX.

CHAPTER II

REVIEW OF PREVIOUS WORK

The *image understanding* problem can be defined as the interpretation of sensed data to yield useful information about the environment being sensed. Such information should be sufficient to describe what is being seen and how the things seen are related to each other in space. The concept of usefulness immediately casts the problem within the framework of task performance (a system to recognize and understand paint defects on an automobile may not be able to solve the problem of the orbital docking of spacecraft). In three-dimensional image understanding, the prevailing task is to obtain the description of a scene in three-dimensional space. This description comprises the identity of objects and the determination of their poses in space.

The identification of objects is ultimately the determination that a sub-set of the sensed data is accounted for by certain *models* of prototypes or classes of objects with the consequent inference that the data arises from (is the result of) the presence in the scene of an article which is an instance of the model. The pose of an object is the location and orientation of that object in space.

The term *model-based image understanding* needs to be better defined. In reality, all recognition systems make use of models in some way[60]. Statistical pattern recognition systems, for example, generally employ object models described as vectors of global features (eg. area, perimeter, principle axes etc.) which are matched against corresponding vectors obtained from the image being processed [79, 113, 161, 170]. Even if only a single critical feature were used in a recognition process, there is still the notion that the critical feature selected is a model of the object to be recognized.

Model-based image understanding is a recognition paradigm which is driven by the models of the objects to be recognized. These models are explicitly declared, and they guide the system in its interpretation of the sensed data. Examples of such systems will be discussed later in this chapter.

2.1 From Whence 3D Data

Three-dimensional data can be obtained in two ways. The first is from explicit range encoding sensors. These sensors produce dense image arrays, each pixel of which is a measurement of the distance of the corresponding surface point in three dimensional space to the sensor's coordinate system. Examples of such sensors are laser range finders and structured light laser range finders [33, 108, 141, 146, 171, 172], triangulation sensors [16, 142], light stripe sensors [6, 8, 37, 39, 95, 97] etc. A comprehensive discussion of explicit range encoding sensors is found in [27].

The second method of obtaining range data is to extract distance information from image data which does not explicitly contain range information. Each pixel of this image contains information (e.g. intensity) about the corresponding point in three-dimensional space. Since information about the three-dimensional configuration of the scene is extracted by the application of certain cues, such techniques are often called *shape-from-x* techniques (where *x* is the cue applied). Examples of these techniques are: shape from shading [67, 101, 135] shape from texture [2, 114, 167], shape from binocular stereo [11, 14, 40, 127] shape from contour, [15, 42, 43, 98, 111], etc. Extensive discussions on these techniques can be found in [1, 19].

It is not our purpose, here, to survey the various techniques involved in generating range or depth data. We do need, however, to have a cursorial understanding of what such data are before we can proceed to determine the processes necessary to recognize objects and their poses from the data. Owing to the availability of range information from the data, one may easily be lulled into believing that the bulk of the problem in three-dimensional image understanding is solved. We do, after all, have information about where everything is. This is far from the truth. The only information explicitly available in the data is how far particular *points* are from the sensor's coordinate system. There is no structure or shape in the data. For this reason, *shape-from-x* is a misnomer. What these techniques provide are either depth points (usually sparse) or estimates of local surface normals at, or intrinsic to, each point in the image. One thinks of shape as such extrinsic descriptors as circular arc, straight line etc. in two-dimensions and cylinders, spheres, planes etc in three-dimensions. At best, *shape-from-x* techniques generate data equivalent to that available from explicit range sensors. To emphasize this lack of structure in range imagery, Barrow and Tenenbaum called such data *intrinsic images*.

What is lacking is organizational structure other than pixel adjacency. According to our abstraction-based model, we need to frame sub-groupings of this range information into semantically significant units.

2.2 Reconstruction and Recognition

Schemes for representing of data for three-dimensional image understanding may be divided into three classes. The first deals with the projection of the three-dimensional form onto two dimensions; the second concentrates on the visible surfaces in range imagery; and, the third emphasizes the volumetric aspects of the objects present in the data. The work of this thesis employs surface representations. In this section, all three representation schemes will first be given cursorial treatment to see how they fit within the scheme of things in three-dimensional image understanding. Surface representation will then be discussed in greater extent with an emphasis on making a distinction between the reconstruction of surfaces and their application to recognition.

2.2.1 Representation Schemes

Two-dimensional representations of three-dimensional data usually take the form of edge segments and/or regions (points grouped using some measure of likelihood that they belong to the same surface) within the image. Global or local recognition schemes may be applied to match these features with models. Global techniques usually extract *feature vectors* from regions which are expected to represent whole objects and match these to similar model feature vectors. The feature elements may comprise such measurements as area, ratio of area to perimeter, moments, centroids, aspect ratio etc. In local techniques, edge segments and regions are often related to each other in the form of graphs which are matched with models. The literature abounds with papers on two-dimensional object representation and recognition schemes. While it is not our purpose to review such techniques, it is appropriate to state how these two-dimension schemes may be (and have been) applied to three-dimensional image understanding. *Multiview* representations [56, 57, 76, 77, 95, 136] are means of mapping three-dimensional objects into multiple two-dimensional representations. These representations are termed variously as aspect graphs, visual potentials, characteristic views etc. The idea is to store each view of the object as a separate two-dimensional model and to perform the matching using these models. Sripradisvarakul and Jain [154] discuss how these representations may be generated for curved objects.

The task of organizing data in three-dimensional image understanding is often viewed in terms of representing the visible surfaces and edges (discontinuities) in the range imagery. This is the most natural representation for range data in the sense that it operates directly on the data and characterizes the pixels in the range images. This representation is favoured by this author and is the one applied in this thesis, and will be discussed in the next section.

Volumetric representations of data and models for recognition attempt to describe objects as three-dimensional entities. Representations include generalized cylinders used in ACRONYM (to be discussed later), constructive solid geometry, voxel and octree representations. While such representations have been proposed, the only significant work done in object recognition with volumetric representation is ACRONYM.

2.2.2 Surface Representations

Range imagery is data-rich across object surfaces in the images. Much work has been done on the extraction and characterization of these surfaces.

First of all, what is a surface? A reasonable first approximation may be that a surface is a smooth three-dimensional manifold where smoothness is defined as the lack of discontinuities in the normals to the surface. This, is however, not always the case. The hood of an automobile is often considered a single surface although there may be an ornamental ridge-like protrusion down the middle of the hood. Two planes melding at an obtuse angle via a smooth chamfer may not possess any abrupt surface discontinuities, but we have just said that they are two planar surfaces. A more general definition has to take into account the kind of surface one *expects* to find. This will be dealt with in greater detail later in this thesis. This being said, smooth surfaces as defined before are one type of surfaces one might have great interest in finding. It is this kind of surface with which much of the literature deals.

Surface characterization schemes may be divided into two classes. The first represents the exact location of the surface and the second details how the surface bends in space. While the latter curvature-based representation has many attractive qualities such as rotation, scale and translation invariance, little real work has been done using the representation for object recognition. The main reason for this is that noise in digital data renders the computation of first and second partial derivatives of the surface, on which curvature depends, unstable [25]. The theoretical work on curvature-based surface characterization will be discussed later in this thesis when surface curvature is discussed. In this section, we shall concentrate on representations which describe the location of surfaces in three-dimensional space.

The task of describing the location of a surface in three-dimensional space is often framed as one of fitting range data to a surface description. These surfaces may be bivariate polynomials (planes being the most prominent class), splines, quadrics, finite element grids, etc.

Besl and Jain [26, 22, 25] fitted bivariate polynomials of the form:

$$z = \sum_{k=0}^{\mathcal{O}} \left(\sum_{n=0}^{k} c_i x^n y^{k-n} \right)$$

where $\mathcal{O}$ is the order of the polynomial and $\{c_i\}$ are the polynomial coefficients. A *variable order*, region-growing strategy was employed to obtain regions of the lowest order which will fit smooth regions. This system was used in the course of this thesis and will be treated in greater detail in a later chapter when the background material (mathematics and algorithms) of this thesis is discussed.

Terzopoulos [157, 158, 159, 160] applied a *finite element grid* technique to perform surface reconstruction. The reconstruction is performed at multiple resolutions as a process of cooperative relaxation processes which minimize a tension measurement between the reconstructed point and the data. The work which can be applied to sparse range data can be viewed as one of smoothing range imagery.

Duda et.al. [66] did some of the early work on incorporating range and reflectance data from a laser range finder in scene analysis. Their system extracts planar surfaces from registered reflectance and range imagery under a paradigm in which easy to extract surfaces are obtained before harder ones are extracted, and where reliable regions are extracted before questionable regions are analyzed.

The system begins by removing all pixels where jump boundaries are detected in the image. These are segmented into connected regions. From these the set of *starting planes* are determined. Such planes are defined by their unit normal and distance from the origin of the coordinate system. The easiest and most reliable to compute of these are horizontal planes whose normal vector is completely defined leaving the distance from the origin (the z value of the plane) to be determined.

Horizontal planes are detected by analyzing the histogram of the z values in the depth image. Vertical planes have one component of the orientation (incline) of their normal vectors defined, leaving the determination of the distance from the origin and the horizontal component of the normal vector. A Hough transform is used to detect these vertical planes and their unknown parameters. For arbitrarily oriented planes, it is assumed that the surface reflectance is constant, and histogram analysis of the reflectance image is used to detect these planes and least squares fitting is applied to determine its parameters.

Once the starting planes have been determined, a refinement process is initiated, beginning with the 'most reliable' of these starting planes. First, a planar 'sandwich' or band is computed around some tolerance of the estimate of the plane. All pixels within this three-dimensional 'sandwich' are collected and grouped into connected regions. These regions are then refined by the application of three tests: the size filter (small regions are rejected); a plane fitted to the region must be reasonably close to the original estimate; and, a planarity test based on the mean and standard deviation of the distance of the points from the best fitted plane is applied.

Bolles and Fischler [37] describe the application of *random sample consensus* (RANSAC), a robust statistical fitting strategy, to the recognition and pose determination of cylinders in structured light imagery. The RANSAC strategy begins with the initiation of a model using a minimum number of points, followed by an iterative least-squares process to improve the estimates while recruiting more points that fit the model.

For the recognition of cylinders the fitting of data to a non-linear combination of parameters is avoided by applying the observed that the planar biquadratic function $(a + bx + cy + dxy + ex^2 + fx^2 = 0)$ (which is linear) is capable of fitting conic sections. Since the illuminated contour in a light stripe image is a planar cross-section of imaged objects, the quadratic function is capable of estimating the ellipses which constitute oblique sections of cylinders. The ellipses which are extracted applying RANSAC to the planar biquadratic fit to the data are then used to estimate the pose of the cylinders. This time, RANSAC is again used in the non-linear least-squares combination of the ellipses.

This system performs entirely in the context-sensitive domain. The objects imaged are cylinders of known radius and the fitting locates these cylinders. The application of the biquadratic fit to estimate the ellipses and the combination of the ellipses apply only to cylinders.

Henderson and Bhanu [93, 94, 29, 30] describe a three-point seed method for extracting planar surfaces from range data. The method begins by extracting a set of three-dimensional (x,y,z) points and organizing them in a *k-d* tree, using the *x, y, z* values of each point as the *k* keys. Sets of three non-collinear points close to each other are selected and to define seed planes. These seed planes are then grown by recruiting sets of points from the *k-d* tree. Since the system handles convex polyhedra, convexity is applied as a constraint on the three-point seed candidates.

Potmesil [137, 138] fitted rectangular bicubic parametric patches to range data. The work concentrates on the integration of such patches computed in surfaces from different viewpoints. One may think of each such patch as a bicubic tile. These tiles are merged using a quadtree structure. Each level in the quadtree contains nodes which represent patches at a quarter of the resolution of the next lower level. Curvature maxima are used to match patches from different viewpoints in the integration process.

Fan et. al [74, 73] apply quadric surface patches of the form:

$$ax^2 + by^2 + cz^2 + dxy + eyz + fzx + gx + hy + iz = 0$$

in their work on three-dimensional object recognition. The segmentation of the range image into different regions is performed by boundary detection. Boundaries are detected by first computing the ‘surface curvature’ across the entire image and then looking for zero-crossings and extrema in the curvature map. The boundaries and patches are used in a graph matching scheme (boundaries are arcs and patches are nodes) which will be described later in this chapter when recognition systems are reviewed. At this point, it is sufficient to note that the principal curvatures of the quadric patches are computed to determine if the patch is planar.

2.2.3 Comments

These techniques lead to representations of the surfaces to a degree at which they can be reconstructed. The papers listed are replete with impressive reconstructed images. The problem is that little has been done with such surfaces in object recognition apart from Fan et. al. and the work on polyhedra. The reason is that the representations can be thought of as yet another form of data to be analyzed. The parameters describing these surface representations are often a collection of numbers which in many cases are well nigh incomprehensible. The only semantic information immediately available is that the surfaces are smooth. The reason for this is that these surfaces estimate the pointwise location of the surface, and not what the surface really is. The exceptions are in the cases of the work on polyhedra and the quadric-based work of Fan et. al.

The reason polyhedra-based recognition systems are viable is that polyhedra are made up of planar surfaces. The plane is an interesting degenerate member of polynomial surfaces in that they often represent what the surface really is (e.g. a side of a file cabinet). Systems for recognition of polyhedra exploit the fact that only polyhedral objects are expected. One may think of plane detectors as specific feature extractors.

The reasons that quadrics are applicable for object recognition in the work by Fan et. al. are twofold. First, the quadric patches served only to label regions as 'smooth-surface' which served as attributed nodes for their graph structure. Second, they exploited the fact that quadrics fitted well to cylinders (although this was not explicitly stated as the reason for using quadrics) and computed the orientations of cylinders as the directions of least curvature of quadric surfaces. It appears that they refitted planes to patches with zero principal curvatures to determine the orientation of planes. Again, quadrics map specifically (and narrowly) into the class of real surfaces in the object set to be recognized. The restriction of the object set makes this possible. If the object set included such objects as toroids and bending pipes which are not amenable to quadric description, the system will not work.

2.3 Recognition Systems

We shall now discuss a sampling of three-dimensional recognition systems based upon approaches which include the use of extended Gaussian images, Hough transforms, extended Hough transforms, graph matching, hypothesis and test and blackboard architectures.

2.3.1 Some Recognition Systems

Andress and Kak 1988

Andress and Kak [4] describe an approach to interpret three-dimensional scenes when the two-dimensional projection (orthographic or perspective) of the structures in the scene can be generated. The two-panel, six-level blackboard system utilizes Dempster-Shafer formalism to perform inexact reasoning in a hierarchical space. All objects must be polyhedral as the system employs only linear line segments and edges as its basic building block for recognition.

The levels of the blackboard contain information in the form of: Vertices, segments, edges, faces, objects and scenes in the *data panel* and vertices, edges, faces, objects and scenes in the *model panel.* Three knowledge sources (KS) operate on the data. The *data-reduction KS* cleans up the line segments generated by the edge detectors, the *grouper KS* groups lower level data into data elements at increasingly higher levels, and the *labeler KS* performs element labeling and confidence estimation for different hypotheses generated by the grouper KS.

The basic primitive features utilized by the system are the line segments provided by edge detectors. The abstracting principle assumption applied is simply that discontinuities in intensity values are significant. Linear edges (the system operates only with linear edges) are generated by grouping collinear line segments which are close together. Collinearity and edge contiguity are again independent of the scene. Vertices are the ends of line segments and linear edges. This abstraction, based upon basic geometry, which is context free by our definition. The grouper KS basically performs evidenced-based hypothesis generation building higher abstraction structures from lower.

Faces and objects can be recognized in the system by context sensitive assumptions. To generate faces, the labeler KS associates model panel structures with the higher level data panel structures. Faces in the model panel elicit faces in the corresponding layer in the data panel by hypothesizing the mapping between model panel edges and data panel edges. Clearly, such a process is context sensitive and driven by the model, otherwise, the possible orderings of fragmented edges to form faces is combinatorially explosive. The object layer structures coalesced from the face and edge layers are similarly context sensitive. Object poses are determined at the object and scene layers.

Archibald and Rioux 1986

The WITNESS system [5] recognizes polyhedral objects in range imagery by matching objects for which 'view models' are available. These 'surface adjacency graph' (region adjacency graph of planar surface regions) models are sensor-tuned in that they are generated 'showing' the system a prototype in stable resting positions. Each view model is rotation independent within the plane of support for the object. A two stage *hypothesis generate-and-test* strategy was employed. A set of possible models are first identified using 'generic information'. The resulting hypothesized surface adjacency graph models are then matched in turn.

The basic feature applied in this work are planar patches. First, the surface gradient at each point of the image is computed using a gradient operator which performs implicit Gaussian smoothing. The resulting gradient-space images are then segmented in regions of similar surface normals – these constitute planes. Such an operation for the identification of planes is context free, depending on the mathematical properties of planes for its computation. (This is not strictly true if surfaces of other types were allowed since curved surfaces of large curvature and perspective warping which result from inadequate calibration can make planes extremely hard to identify reliably.)

Both the generation of the model sub-set and the matching of the surface adjacency graphs are extremely context sensitive. Generic view model properties (e.g. the number of faces, the total projected surface area, and the maximum height in a view) are utilized in the first stage to trim the number of candidate view models. In the testing of the hypotheses, surfaces in the data are matched to model surfaces following a constraint-guided protocol which determines the order of matching to reduce the combinations. The *reliability constraint* determines that surfaces with large areas, low magnitude of slope and high compactness are matched first. The *structural constraint* dictates that surfaces in the models are bound to data surfaces by order of adjacency and the *relational constraint* tests the matching by examining the differences in such geometric properties as surface area and slope. Obviously, one must know the set of objects being considered before either of these can be perfomed.

The emphasis of this system is on the high level processing. Such a system will quickly be overwhelmed as the complexity and number of objects to be recognized increase. It will also be sensitive to variance in the context by such artifacts as noise, occlusion and calibration error.

Archibald and Merritt 1989

The system performs pose determination for a block and a robotic grapple mechanism, each of which has clearly defined straight edges of surface discontinuity (as opposed to occlusion edges as would occur when one scans off the rounded edge of a cylinder – in which case there would be an 'edge' in the data where there is none on the object). Range data is acquired using the NRCC *range profile* sensor which takes data along a 'stripe' mounted on a robot manipulator wrist.

The system makes use of discontinuities in the average inclination of the surface to the sensor origin for successive readings along the scan line to extract valley edge points. Cliff edge points are detected as 'horizon' points. When the change in average inclination exceeds some threshold, the cliff edge detector kicks in to find the last point visible from some point at the 'base' of the cliff. The abstracting assumptions applied are general, exploiting the basic properties of the geometry of triangles in the computation based on surface inclination and exploiting the characteristics of the scanner (laser light travels in straight lines) in cliff edge detection.

The models maintained by the system are sets of parametric lines defined on and object-centered coordinate system. The edge points computed by the context-free computations form the input to the subsequent hypothesis-test based context-sensitive computation. An edge point is selected and an assumption is made to map it onto one of the model's edges. A second point is picked, and using the distance from the first point as a criterion, a set of candidate mappings of the point to the set of lines in the model is computed. A third point is selected, and its possible mappings to the model's lines are computed in like manner using the hypothesized second points in the model. The hypotheses which cannot 'close the triangle' (with all three points resting on lines in the model) are rejected. The accepted three-point sets yield transformation hypotheses to map the sensed points to the model's edges. If more than 80% of the points are close to a model's edge after transformation, the hypothesis is accepted.

One may think of the three-point sets as triangle features subtended at the corners by edges of the wire frame of the object. The possible three-point combinations would become too large if the model were not used to constrain the hypotheses. These triangles are thus context-sensitive based upon the assumption of the wireframe and suggested by the evidence provided by the edge points.

Asada et. al. 1988

Asada et. al. [8] present a system to extract planar surfaces and their orientations and singly curved surfaces from light stripe images. Surface normals (instead of range) are computed directly from the scene images with a light pattern projected into the viewing space (actually from the edges of the light-to-dark transitions). These normals are then plotted in gradient-space and the clustering of the data points are examined to determine the types of surfaces present and their orientations. Planar surfaces result in data clustered around a point in gradient-space while cylindrical surfaces appear as points clustered around a line. The location of these cluster points and the orientation of these lines determine the pose of these surfaces.

This technique is sensitive to errors introduced by the assumption of orthographic projection (the gradient-space plots do not cluster so well under perspective projection), accuracy of the stripe locations, camera lens calibration, non-parallelism of the light stripes, the non-uniformity of the camera pixels and the inaccuracy of the light-to-dark shadow-edge separation owing to surface reflectance and angle. The system is incapable of providing depth information and parallel planes become easily confused.

No discussion is made of using the features for object recognition. Because the assumptions made in the segmentation are based solely on the clustering of surface points in gradient-space, the method is context free. However, context is implicitly imported in that all surfaces are assumed to be planes or singly curved (cylinders).

Augusteijn and Dyer 1986

Augusteijn and Dyer [9] present a model-based system which performs three-dimensional recognition and pose determination of planar point patterns or polygons. The algorithm computes the correct surface orientation as well as the correspondence between the set of model features and the set of image features. The system, which assumes orthographic projection, employs an iterative algorithm which simultaneously determines the correspondence between the model and data points and computes the surface orientation of the point pattern. The principle behind the algorithm is that if two lines in an image under orthographic projection are mapped to two corresponding lines in the model object plane (all objects are planes), the rotational transform (tilt and slant) between the image plane and the object planes can be determined.

The system identifies the points of interest (usually high curvature points like corners) and computes the angles they make with the image plane coordinate axes with the centroid of the object as origin (the images are thresholded to obtain a binary pattern). Four of these are paired with four points in the model (with four corresponding angles with the object plane axes centered at the centroid of the model). This yields two equations with the angles of tilt and slant as the two unknowns which converge iteratively to a solution if one exists. If there is no convergence, the match fails and other points are selected.

Although Augusteijn and Dyer did not describe the pre-processing in the system (the binary images appear to have been synthetically generated and the points of interest were picked by hand), one may imagine that some histogram-based thresholding and corner detection processing may be performed. These are to a large degree independent of the object being viewed (assuming constant surface reflectance, illumination etc.)

The system is almost completely context sensitive, depending on the spatial distribution of model points to determine correspondence and orientation.

Owing to the high context sensitivity of the system (jumping directly from points to recognition, correspondence and orientation), the system is expected to be fragile. It would be sensitive to inaccuracies in the computation of the centroid (which in turn depends on the thresholding to obtain the binary image). Because it mixes both the need for a global feature (centroid), and local features (corners), it will suffer from the weakness of both approaches. For example, centroid computations are extremely difficult under occlusion.

Ballard and Sabbah 1983

Ballard and Sabbah [12] describe an extended Hough transform-based approach using *constraint tables* to detect the presence of a known object and to compute its pose (orientation, translation and scale from its canonical description). The class of objects considered are polyhedra. The problem is framed as one of determining the transformation between the *object-centered* frame and the *viewer-centered* frame. The data is assumed to be orthographic projection and it is assumed that the image has been segmented into edge components for the two-dimensional case and into planar components for the three-dimensional case.

The algorithm decouples the interdependence among scale, orientation and translation. In orthographic projection, scale is dependent only on the *depth map*. For the computation of orientation, the extended Gaussian map (basically an attributed edge or surface normal plot for two and three dimensions respectively) is used.

For the two-dimensional problem, the length of the edges is used to attribute the normal directions of the edges in the Gaussian map. The offset between the extended Gaussian map of the image and that of the object centered model is the rotation offset of the object in the image. The Hough accumulator array is one of possible orientations. Edge lengths are used to match image edges to model edges and the difference between the orientations of the model edge and the data edge are entered into the array. The maximum in the accumulator array corresponds to the object orientation. For translation in two-dimensions, (the image and model are assumed to be of the same orientation and scale) the accumulator array is a two-dimensional array in x and y. For each data edge, the x and y offsets of the model edge with the same orientation and length from the object frame origin are summed with the x and y locations of the data edge respectively to provide an index into the accumulator array element which is incremented. When all the edges have been considered, the maxima in the accumulator array corresponds to the offset of the viewer frame from the object frame.

Orientation determination in three dimensions is more complicated although the principle is similar to that in two dimensions. The normal of each plane defines a planar locus for the orientation of the viewer frame from the object frame. Nine parameters (three in each dimension) need to be determined by the extended Hough transform. For each match between a model normal and data normal, the loci of three sets of three direction cosines may be ascertained. The Hough accumulator entries along this entire loci are incremented (there are three accumulators corresponding to the three sets). The three dimensional translation Hough transform is identical to that in two dimensions but for the added dimension.

The context free features in this work are the edges and planar surfaces which can be computed using general assumptions of surface discontinuities and surface normal coherence.

The entire extended Hough transform technique is clearly context sensitive (defined for particular object models).

Bastuscheck et. al. 1986

Bastuscheck et. al. [16] presented work on the recognition of general three-dimensional space curves using registered range and intensity imagery obtained using a structured light optical triangulation sensor. The system matches an observed curve with a model curve, recovering the displacement (in terms of arclength) between the curves for which the curves are 'closest' (in terms least squares difference). The algorithm attempts a match for each unit of displacement, picking the best match. It was also shown that the least squares equation for the match can be reduced to three summations and a trace computation of a convolution (which is computed via fast Fourier transform) for each possible displacement.

The system did not attempt to solve the problem of obtaining good range information at three-dimensional edges. The experiments reported recognized 'edges' of construction paper figures glued to the surface of terra-cotta flower pots.

Apart from the computation of edge points, this system is completely context sensitive attempting to match edge data directly to model curves. Given the space of possible boundary configurations, this system is expected to be fragile and not scalable to larger object sets.

Bhanu 1982, 1984

Bhanu [29, 30] presents a system to perform three-dimensional scene analysis on laser range data, yielding the identity and orientation of the object in the scene. Objects are represented as polyhedra and the features applied in the matching are planar surfaces extracted by a three-point seed algorithm. Models are generated from images taken of a physical prototype, and the planar faces detected in each view are arranged in 'neighbor' tables. There is no indication in the papers that the sets of faces in the different views are unified.

Recognition task is framed as one of obtaining a consistent labelling between faces extracted from image data from an unknown view and model faces. A stochastic labelling technique applied. The compatibility of a data faces to model faces were computed in two stages – the first considering the matches pairwise (two model faces and two data faces, one model-data pair being the ones compared and the other pair for context) and then in sets of threes (one model-data pair being the ones compared and the other two model and data faces for context). The function, which determines the compatibility among the two (first stage) and three (second stage) neighbouring data faces, computes the transformations (scale, translation, orientation, and rotation) which minimize the difference between data and model faces. In the first stage, two transformations are computed – one for each model-data pair, and in the second stage three such transformation are required. In the experiments, the 29 'best' surfaces extracted from the data are matched to model surfaces.

To the degree that the planes are computed applying the mathematical properties of coplanar points (the three-point seed method), the computation of the planes is context free. However, the a priori knowledge that all the surfaces in the images are planar makes this somewhat context sensitive.

The matching and computation of compatibility is clearly context sensitive, being based completely on the models.

Bhanu also describes a similar two-dimensional system which operates with linear edge segments instead of planar faces [28].

Boissonnat and Faugeras 1981

Boissonnat and Faugeras [34, 35] describe a triangulation technique to approximate range data as polyhedra. The approach is graph based where each graph node in the graph is a range data point. Applying Gaussian curvature the data are segmented into sub-groups of saddle and cup-shaped points (the method will not work on parabolic surfaces). These sub-groups are then triangulated applying a graph splitting operation to yield a triangle-facet description of the range data.

Bolle and Cooper 1984

Bolle and Cooper [36] presented work on the mapping of intensity image data to three-dimension planar, cylindrical and spherical shapes. The image is partitioned into a grid of windows into which quadric polynomials are fitted over the intensity values. The relationship between the three-dimensional shapes in question and the quadric representation of the intensity surfaces under orthographic projection and pure Lambertian reflectance to a point light source was investigated. Assuming that each window views a piece of only one surface, or at most two surfaces, each window is labelled as 'planar', 'cylindrical', 'spherical' or 'unknown' by applying this relationship to the quadrics fitted to the window. Upon the assumption that the noise in the intensity images obey a zero-mean Gaussian distribution, a Bayesian model is applied to reparameterize the quadric to yield a set of three constrained quadrics corresponding to the three-dimensional forms. By comparing the unconstrained quadric fit with the each of the constrained fits in each window, a likelihood measure is obtained.

This work is a context sensitive feature computation in that the types of surfaces to be labelled are known a priori (otherwise the number of constrained fits would be prohibitive). While the assumptions (eg. windows mapping into only one surface, pure Lambertian surfaces, point light source, Gaussian noise distribution, independence for Bayesian probability) imposed make it unlikely that the technique will be applicable under real-world conditions, the argument for the utility of feature sensitive intermediate feature computations to identify objects is sound.

Bolles et.al. 1984

Bolles et. al. [39, 97] describe 3DPO which performs three-dimensional part orientation (hence its name) on range imagery obtained from a *plane-of-light* triangulation sensor. The approach, basically a three-dimensional rendition of the *local-feature-focus* method [38] which operated on two-dimensional imagery, bases the generation of hypotheses on a small number of features or feature clusters. 3DPO partitions the recognition process into primitive feature detection, feature cluster formation, hypothesis generation, hypothesis verification, and parameter refinement. The primitive features extracted by the system are edge-based. An example of such a feature is a coplanar sequence of edge points which make up a circular arc of a particular radius. Object hypotheses are generated by examining all pairwise combinations of these features with model features. These hypotheses are in turn used to grow the feature clusters (adding a feature at a time) around the focus feature by applying the information contained in the models.

The context free features in 3DPO are edge points which are computed by applying two assumptions. Jump edges are extracted upon the assumption that there is a discontinuity in the light stripe at such edges. Other edges are obtained by the conventional technique of finding zero-crossings in the first derivative of the filtered intensity image. Another context free feature is the linear edge segment.

Virtually all other processing in 3DPO is context sensitive. Specific ellipse detection is required, for example, to find cylinders. In fact, this is the basic tenet of 3DPO – that context information provided by the object models be used to constrain the search and matching. The generation of focus feature hypotheses and the building of the object graphs around these focus features are dependent upon the models of the objects expected, as is the recognition of the object by graph matching.

ACRONYM – Brooks et.al. 1979–1983

ACRONYM [46, 47, 48, 49, 50, 51] employs a volumetric representation in which all objects are decomposed into a set of generalized cones. These cones are described by a length, an orientation, a cross-sectional shape and a sweep rule which determines the variation in the cross-sectional area (of the same shape) along the length of the cone. This representation scheme is capable of handling hierarchies of detail by applying combinations of generalized cones (e.g. conjuncts, intersections etc.) of increasing detail deeper in the hierarchy. Since the perspective projection of cones yield generalized ribbons, ACRONYM is able to 'predict' the perspective view of the object on-the-fly.

In the matching process, ACRONYM's low-level processes extract the features to be matched and organize them as a 'picture graph'. The prediction/planning process generates perspective projection called the 'observability graph' from the volumetric model stored in the form of an 'object graph'. The matcher then generates an 'interpretation graph' which determines the identity of the object being viewed. Although there are lines of control or data (it is not clear which) from the interpretation graph to the low-level vision processes in the block diagram of ACRONYM in a 1979 paper [46], no such control is described in that or subsequent papers. ACRONYM can thus be assumed to run open-loop. The matching process thus becomes one of matching the picture graph (from sensed data) to the observability graph (derived from the object model).

The only context free features in ACRONYM are the edge points used as boundaries of the generalized cones.

Because generalized cone representation is ambiguous (many generalized cone configurations may describe the same object), the computation of the cones must be guided by the models. The graph matching activity is clearly dependent on the object models.

Several things can be said about ACRONYM. By the admission of the designers of the system, ACRONYM suffers from inadequate low-level vision. This has been blamed for much of ACRONYM's ills; but, this researcher believes that the problem is more fundamental. Low-level vision is generally fragile and extremely sensitive to environmental and sensing variations. There is no reason to believe that a low-level feature extractor (edge detectors in the case of ACRONYM) will be developed which will solve the problem. The model representation scheme which is a key strength of the system also constitutes one of its greatest weaknesses. As the objects to be represented increase in complexity and number the generalized cone representation scheme quickly becomes overwhelmed. This approach to model representation may just not be rich enough to handle an unconstrained object space. As a consequence, ACRONYM is known to have 'understood' only one image from one perspective.

We have already seen how the system under discussion abstracts its visual data by a means of feature extraction into a *picture graph* to be matched with an *observability graph.* ACRONYM, thus, quickly distances itself from the visual data. This is perhaps the greatest weakness of the system. It does not exploit the knowledge contained in the image models to guide its low-level processes.

Brou 1984

Brou [52] discussed the application of the *extended Gaussian image* (EGI) to determine the three-dimensional orientation of objects. The EGI is based upon Minkowski's theorem which states that the 2-tuples of pointwise Gaussian curvature of a surface and unit surface normals uniquely describe convex forms. This principle has also been used by Ikeuchi and Horn [100, 102, 103] to object recognition algorithms based on the EGI. The space of all normals of a smooth convex form maps into a sphere known as the Gaussian sphere.

Since it is impossible to represent and match the surface normals across the Gaussian sphere in a continuous fashion, Brou discusses methods to partition this sphere for sampling. The sphere is approximated by an icosahedron, each triangular face of which is again subdivided into more triangles which constitute geodesic domes for each icosahedron face. The EGI image is then smoothed with a Gaussian filter for matching.

This is basically a histograming approach in the space of surface normals, and is a pointwise matching scheme. While the computation of the extended Gaussian image is context free, the matching is dependent on the model. Each model has to be tried until one matches. An oft cited problem with the approach is that EGI does not preserve spatial connectivity.

Chakravarty and Freeman 1982

Chakravarty and Freeman [56, 77] presented a multi-view modelling approach based on *characteristic views*. The space of all possible perspective projections of an object is factored into a set of characteristic views each of which defines a set of topologically identical projections related by a linear transformation. This effectively maps the three-dimensional object into a set of two-dimensional models for matching purposes. Experiments in which the silhouette of a polyhedral object is matched to its characteristic view edge models was presented. Fine pose is obtained within the characteristic view by computing the linear transformation of the labelled edges in the data to the model edges.

The computation in which edge points are extracted and grouped as chain-coded a object boundary is dependent only upon assumptions of surface discontinuity and connectivity of edge points forming the boundary. The labeling of the data edges (matching) and the computation of the perspective transformation within the characteristic view are context sensitive.

Chen et. al. 1980

Chen et. al. [57] describe a system for estimating the pose of a workpiece using the *feature points method.* A stereo algorithm is applied to obtain the three-dimensional position of such features as corners and small holes. The feature point which is accorded the highest confidence value by the feature extraction process is matched to all model points of the same type. Each of these possible matches was tested for consistency with other data points in relation to the object model (the tests applied for such consistency test are Euclidean distance and the presence of edges linking the feature points). Of those matches which had sufficient support, three points were selected to obtain a workpiece to model transformation hypothesis. Each hypothesis is then tested to see if the computed transformation accounts for the other points in the workpiece feature set.

The context free features are the corners, small holes and edges and their three-dimensional positions which are computed applying general image processing principles and binocular stereo. The feature clusters formed with the aid of the models are in essence context dependent higher level features.

Coleman and Sampson 1985

Coleman and Sampson [63] presented a mathematical morphology feature-based method for locating grip points of objects for bin-picking. The algorithm operates on range imagery and determines with the application of a morphological structuring element locations where the robot gripper would fit. Only the position and orientation of the grip points are determined. No attempt is made in recognizing the parts.

The operation is context sensitive in that it look for positions on prespecified objects jumbled in a bin which 'fit' a particular robot gripper model. The system is thus extremely sensitive to context. Any change in grippers or objects in the bin will require major changes to the system.

Dhome and Kasvand 1986

Dhome and Kasvand[64] applied a generalized Hough transform method to recognize polyhedral objects in range imagery. Pairs of adjacent planar surfaces are used as primitives in the generalized Hough transform. Each such pair provides a hypothesis comprising the orientation of the model view axis, the model orientation around the view axis and the model center position. These parameters form a three layered hierarchy. The Hough spaces of each of these are computed in turn. A clustering program is used to determine the 'winner' of the Hough transform for every layer.

To the extent that polyhedra are expected in the scene, the extraction of planar surfaces depends only on the general principle governing their surface normals. This and the detection of adjacent planes are therefore context free. The generalized Hough transform is strongly dependent upon the object models. This is thus very context sensitive.

Douglas 1981

Douglas [65] presented a 'model-building' approach for three-dimensional scene interpretation. The EYE system integrates depth and semantic information to form a three-dimensional model of a scene from colour pictures of outdoor scenes.

The preprocessing stage of the system performs a set of image-to-image transforms to yield a stack of arrays corresponding to colour, averaged images, texture, edges and angles. This stack, known as the 'recognition cone', is further processed to obtain a set of 'visually similar' regions. A model of the scene is then built in a four step algorithm. First, a set of label hypotheses is assigned to the regions; second, depth cues (shading) are used to estimate the regions three-dimensional structure; third, the initial depth estimates are corrected by an iterative relaxation process; and fourth, the region labels are finalized to constitute consistent objects. These operations are performed by three routines which operate within a blackboard representation of the scene. The *placement procedure* performs the first two operations; the *adjustment routine* performs the third; and, the *object formation routine* does the fourth.

The placement routine uses two kinds of information to estimate the depth of a region. The cues employed are: occlusion, expected size of the object, ground plane hypothesis (grass patches are expected on the ground plane), texture gradients, shadows and highlights and linear perspectives. Context free (e.g. occlusion, texture gradients) and context sensitive (e.g. object size, ground plane hypothesis) cues are thus applied at the same time. The adjustment routine relaxes the labels of the regions and their depth estimates by introducing relational information among the regions. The object formation procedure matches clusters of regions to object models which take the form of associative nets. These nets comprise PART-OF and ISA links.

Engelbrecht and Friedrich 1988

Engelbrecht and Friedrich [68] describe a Hough transform-based method for recognition of polyhedral objects in three-dimensional space. The standard Hough transform is applied to edge points detected in the intensity image to obtain the set of straight lines. The Hough space is further analyzed to determine such higher level features as parallel lines (vertically aligned cluster points in Hough space), n-line intersections at a vertex (colinear clusters in Hough space), two vertices linked by one line (two point colinearities sharing one cluster point), and n-colinear vertices (n cluster point colinearities sharing one cluster point). These lines are organized into an attributed graph (nodes are vertices, arcs attributed by length, slope etc.) and matched to model graphs. To avoid explosive combinatorics, the attributes of the arcs are used as a key to detect isomorphic sub-graphs.

The only experiments reported were on synthetic imagery.

Ettinger 1987

Ettinger [71, 72] presented the SAPPHIRE system which performs hierarchical object recognition using libraries of parameterized model sub-parts. The two-dimensional recognition system operates in the domain of object boundaries extracted by means of an edge detector (Canny's edge detector). Objects are modeled both in terms of a sub-part and scale hierarchy. Both of these are facilitated by the application of features derived from Asada and Brady's Curvature Primal Sketch [7]. These features, derived by observing the results of convolutions of the first and second derivatives of the Gaussian filter at various scales on the region boundary, are 'smooth joint, corner, crank, bump, end and dent.' Such features are inherently multi-scale and SAPPHIRE records the largest scale at which each feature appears to derive a scale hierarchy. The sub-part segmentation is obtained by a set of rules which governs points on the boundary at which to make breaks. An example of such a rule is that 'two crank features, close to each other, facing opposite directions with one or more other features on the boundary separating them is a good place to make a break' – this determines a constriction point where two portions of the same boundary come close together.

During recognition, an indexing step is first performed using a variant of the Hough transform to obtain a set of probable interpretations. The transform counts the 'votes' for each type of feature in the scene to hypothesize sub-parts. Next, a model-constrained interpretation tree search is applied in which each probable model is tested to see if its nodes can be populated by detected features at a consistent pose and scale.

The context free features used in the system are the edge points and the curvature primal sketch features. The sub-part segmentation and matching are dependent on the model set and is thus context sensitive.

Fan et. al. 1989

Fan et. al [74, 73] describe a graph-based object recognition system which operates with dense range data. The nodes of the graph represent the surfaces in the image and the arcs detail the inter-node relations. Objects models are stored in multi-view format (a model for each view). The problem is posed as one of decomposing the graph representing the scene into a set of subgraphs which correspond to different objects in the scene.

The scene is segmented into surfaces by first detecting the boundaries which separate them. These boundaries are characterized as *jump boundaries*, *limbs* and *creases*. The computation of the boundaries is curvature-based. The entire scene is treated as a single sheet and the local surface curvatures are computed across the entire scene. Discontinuities in range readings which correspond to zero-crossings in surface curvature are labelled as *jump boundaries*. Discontinuities in surface orientation which correspond to extrema in surface curvature are labelled as *creases*. Jump boundaries at which the surface normals of one of the surfaces gradually become perpendicular to the viewing direction (e.g. at the side of a cylinder) are labelled as *limbs*. *Creases* are further labelled as *convex* or *concave* creases depending on whether they are maxima or minima of surface curvature. Quadric patches are fitted to the regions formed by the boundaries. A graph structure is generated in which the nodes correspond to the surface patches and the arcs represent the common boundaries among the nodes. These adjacency relationships are the first cue as to whether surfaces belong to the same object. Surfaces separated by convex creases and concave creases are assigned the probability of 1.0 and 0.75 respectively that they belong to the same object. Jump boundary and limb separated nodes are assigned same-object probabilities of 0 to 0.5 depending on the distance of the jump between the surfaces. The nodes are further attributed by their surface area, orientation (surface normal for planes, axis for cylinders and direction of least curvature for all other surfaces), average of principal curvatures, estimated occlusion ratio (a measure dependent on the boundary type and length of shared boundary), centroid, type of adjacency (what kind of boundary lies between nodes) and the probability that adjacent nodes belong to the same object.

The matching takes place in three steps. First a *screener* selects an ordered list of object model candidates using a set of heuristics. The heuristics are that the ratio of visible area of largest node (surface), number of planar nodes and the number of surfaces in the image graph to the model graph must exceed some threshold. Second, a *graph matcher* which attempts to detect sub-graph isomorphism between the image graph and the object graphs applying the following steps:

1. *Pair-wise compatibility*: All possible pairings between scene and model nodes are tested for compatibility. The measure of compatibility is based upon the attributes of the nodes described earlier.

2. Sets of 4 scene-model node pairs are grouped applying such constraints as the difference in patch orientations, distances between centroids, consistency of the types of the dividing

boundaries etc.

3. The best four-pair set are chosen for expansion to obtain the complete match between the scene graph and model graph.
4. The actual two-dimensional geometric transform (in the viewing plane) is computed to register the model with the surfaces in the scene.
5. A goodness of match is computed to determine the viability of a match and to determine if alternate model candidates ought to be tried.

Third, an *analyzer* is employed to prune the graph. This is a heuristic driven process which splits the scene graph into object sub-graphs and merges originally disconnected nodes (nodes for which the probability of their belonging to the same object was originally zero).

The computation of the boundaries and the original segmentation into regions are context free since they are dependent upon general mathematical assumptions. The computation of the quadric surfaces is dependent on the assumption that all regions within a boundary can be approximated by quadric surfaces and that the boundaries exist. The computation of most of the attributes of each node (area, centroid, principal curvatures etc.) is also dependent upon the same assumptions. All graph manipulation processes are context sensitive, depending on the contents of the scene and the models.

Jain and Hoffman 1988

Jain and Hoffman [106] describe a three stage three-dimensional object recognition system for range images.

At the lowest level, the range images are segmented to yield morphological, surface patch and boundary information. Morphological information characterizes the entire image in terms of background/nonbackground pixels, connected components of background pixels, and the 'number of background components within the convex hull formed on the set of nonbackground pixels'. Surface patches, computed by clustering surface normals, are classified as planar, convex or concave using 'the eigenvalues of the covariance matrix computed on the patch pixels, the variation of unit surface normals over the patch and outcomes of applying a nonparametric statistical trend test to the patch.' Boundary information details the relationships between pairs of patches which include the classification of the boundary edges (crease or jump edges). This constitutes the *initial representation* which is oversegmented.

At the next level, the oversegmented surface patches are merged by applying object-specific likelihood metrics which determine if adjacent patches should be merged. An attempt is made to coerce the patches to conform to the description for each object within some bounds. An alternate representation (or interpretation of the surfaces) is thus generated for each object in the domain. This is are called the *modified representations.*

Finally, an *evidence-based* recognition procedure is applied. The rule-based system comprises a set of evidence conditions which support particular object models. Each rule comprises a set features involved in the rule, bounds for the numeric values for each feature, the minimum and maximum number of occurrences of that feature and a set of evidence weights (one weight for each object – 1.0, 0.5, 0.0, -0.5, -1.0 for 'strongly supports, tends to support, gives no information, tends to refute and strongly refutes' respectively) which specifies the amount of support the satisfaction of that evidence condition gives each object. These rules generate a set of *similarity measures* relating each alternate scene representation (one for each object) to the object. A further constraint is laid on the recognition process by requiring that a *major evidence condition* for an object must be satisfied as a necessary condition for the recognition of that object.

In the first stage of the system, the computation of the initial representation is context free. All subsequent computation, including the generation of the modified representations in the second step are context sensitive. This system does not provide pose information although the authors suggest that such determination up to the gross orientation of the multiview models may be obtained by tagging each model with this orientation information. Since the system makes use of a feature vector-like discriminator, the explicit correspondence of the features to model features are not easily accessible.

2.3.2 Comments

In the midst of all the diversity among the systems reviewed, the general theme of the *feature-model* paradigm prevails. Each has a general purpose (or *context-free* in our parlance) feature detector or image segmenter, followed by a process of manipulating these features either as graphs or as numeric feature vectors. This process is dependent on the objects expected and the scene contents. The latter process may often be construed to be a course of interpreting the features in terms of scene contents and expectations. The idea of applying increasing context information to refine the detected features does not arise except for the work of [106]. In that work, the refinement is performed via a 'shotgun approach' in which an attempt is made to refine the feature representation for each object model in the domain.

In the final analysis, though, it should be realized that even in the early processing, domain knowledge has been already implicitly incorporated. The determination of the feature set, the inevitable *magic number* parameters and thresholds, and even the selection of the sensing modality are dependent on the domain or context. The cost of not making the context-based assumptions explicit is the fragility of computer vision systems to modifications in the domain, sensor and scene and the failure to exploit context information in the detection of features. One should not attempt so much to build a general purpose system with no magic numbers and domain-specific *hacks* as to endeavour to make intelligent decisions for the choice of such numbers and hacks (and to know when these have been employed).

CHAPTER III

THE UNDERLYING STRATEGY

Objects to be recognized in the industrial environment are usually a composite of different structures. This is especially true of machine parts and objects designed on computer-aided design systems. It is therefore reasonable to attempt to recognize and determine the orientations of these constructed structural components. The fundamental structures which make up most parts or which are at least present in a predominant majority of parts are planes, cylinders, cones and spheres. In the course of this work, planes and cylinders have been chosen as a representative sub-set (they are by far the most common).

This chapter provides an extended outline of the research described in the rest of this thesis. Lest the wood of purpose be missed for the trees of methods, mathematics and detail, we shall lay a 'bread crumb trail' of reason behind the computations performed at each stage of the processing.

3.1 Exploiting the Data

In working with laser range data, it is prudent to exploit the strengths of the technology and data. Such data are rich in information on smooth surfaces and are poor at discontinuities. The reasons for this are:

- There are far more data points (samples) on continuous surfaces than at surface discontinuities.

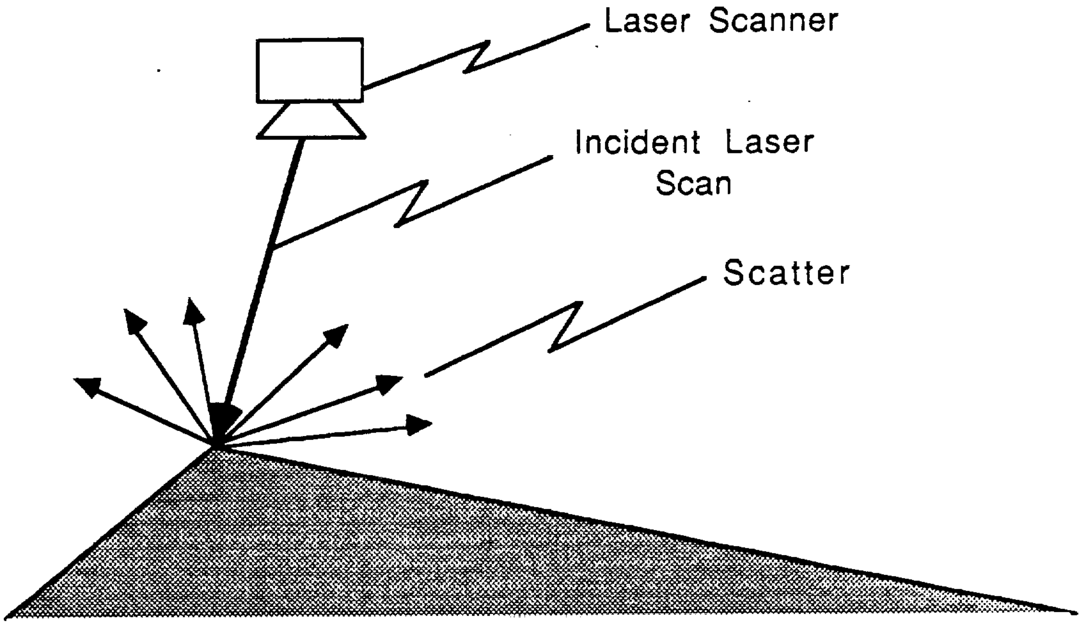


Figure 3.1: Laser range reading inaccuracies at surface discontinuities owing to scatter.

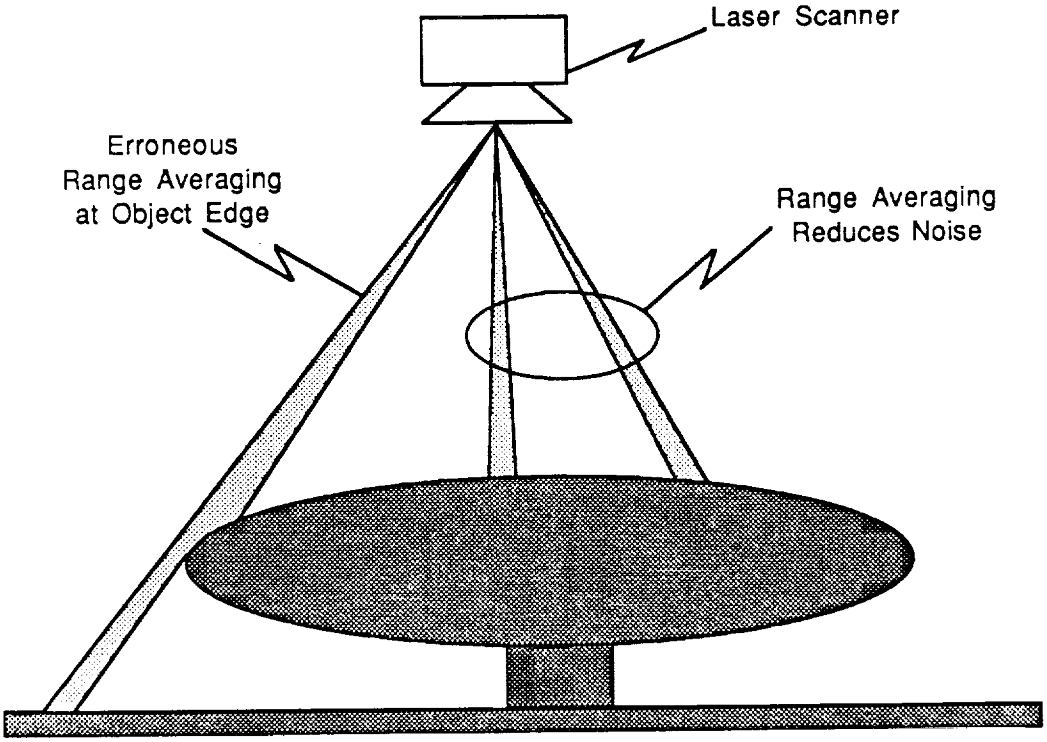

Laser Scanner
Erroneous Range Averaging at Object Edge
Range Averaging Reduces Noise

Figure 3.2: Laser range reading inaccuracies at surface discontinuities owing to range averaging over the laser spot-size

- On smooth surfaces the point range readings are supported by those of neighbouring points when some continuous surface function is fitted to the points.
- The active laser scanning beam scatters at points of rapid surface discontinuity (sharp edges) (see Figure 3.1.) This leads to inaccuracies at such edges.
- Since scanning lasers are not perfectly coherent, the spot-size of the laser on the object surface varies with range (See Figure 3.2). As the spot gets larger, the reading become inaccurate. This reasons for this are:
  - To reduce reading errors, scanners often take the average of multiple readings at each location (In the scanner used for this work two readings are added and the least significant bit is dropped). On smooth surfaces, such averaging makes sense, but at edge discontinuities (especially at jump edges), this averaging leads to meaningless range readings.
  - Most laser range scanners make use of a sensing mechanism which detects phase shift of an amplitude modulated laser beam reflected from a target surface. At jump edges, two return signals of the same frequency at differing phase shifts are seen by the scanner. These signals may interfere with each other at the phase detector.
  - Owing to the rapid reading changes at jump edges, the sensor electronics (operational amplifiers etc.) may not be able to respond quickly enough.

For laser range data, therefore, it is advantageous to operate on data taken from contiguous surfaces. Points of discontinuity in the image or edges may be useful for determining the boundaries of these surfaces; but, the range readings to these edges should not be used for the pose determination.

3.2 Extraction of Structure from Scenes

To determine whether a collection of points subtend (and are due to) a particular physical constructed geometric form of the surface of the target object, the form could be fitted to the data. This would yield a validation of the hypothesis (by examining the *goodness of fit* or fitting error) and the position of the structure in space (by observing the parameters of the fit). As will be discussed later, for linear surfaces such as planes (where the coefficients determining the relation of the depth z reading to the x and y terms are linear), such fitting is simple. Linear parameter estimation methods like QR minimization, normal equations and Singular Value Decomposition can be applied directly to the data. For structures like cylinders and cones, however, the surface functions are non-linear with respect to their orienting parameters. Fitting techniques for non-linear parameter estimation like the Levenberg-Marquardt method require relatively accurate estimates of the parameters; otherwise, the parameters will not converge or they may converge to local maxima in the parameter space. Furthermore, the set of possible structures to fit can be arbitrarily large, the possible regions to attempt the fits are virtually infinite, and the method of fit depends on the structure being fitted. Clearly evidence sufficient to make structural decomposition hypotheses and parameter estimates are needed. Within the paradigm of successive refinement/abstraction laid out previously, such structural features are context sensitive.

3.2.1 Smooth Contiguous Regions

Before attempting the fitting of constructed geometric forms therefore, one needs to form a hypothesis of the structures present and the regions they occupy in the sensed scene. It is clear that such an enterprise has to be general and must apply abstracting assumptions based solely upon physics, the sensing methodology and/or mathematics.

The first assumption applied to the range data is that contiguous smooth surfaces can be approximated by high-order polynomials. This was shown by Besl and Jain [25, 26]. Applying the algorithms and programmes described in [22] one can obtain fairly robust segmentations of a laser-range image into smooth three-dimensional patches. Such an operation is clearly context independent (does not depend on contents of the scene). This will be discussed in greater detail later.

While high order polynomial surfaces are excellent for the identification of smooth surfaces and for surface reconstruction, the coefficients of the polynomials carry little semantic content for surface recognition. They are still just an impregnable collection of numbers. Further processing is needed to extract semantic content other than surface contiguity from the data.

3.2.2 Surface Curvature-Based Labelling

Surface curvature computations provide a method for assigning semantic interpretation to regions in the scene. By examining the signs of the *Gaussian* and *mean* curvature one can determine if surface regions are planes, ridges, valleys, peaks, saddles etc. We shall call images in which surfaces have been labelled this way *curvature sign maps*. Since such symbolic labelling is perspective independent, it is well suited for manipulation by high-level vision interpretation processes. There is, however, a serious impediment to the computation of such features from the image data using digital image differentiation techniques. Digital quantization noise makes reliable computation of such features virtually impossible because the surface derivatives computed by such digital differentiation techniques are inaccurate [25]. In the work of this thesis, these features are computed analytically from the close-form polynomial descriptions generated by the high-order polynomial fits. This yields much better results because these surfaces describe quite accurately the surface subtended by the data points and provide an implicit interpolation between the digitally sampled points. The high-order polynomial surface description also performs noise suppression in that it smooths the surface (in accordance to the surface contiguity assumption) and rejects high frequency noisy image regions. In this process, the abstracting assumption applied is the nature of surfaces defined by differential geometry. It provides a higher level of abstraction than the high-order polynomial fitting and is more context dependent in that it assumes that the correctness of the segmentation of the high-order polynomial patches. The mathematics and processing of these surface curvature features will be discussed in greater detail later.

Curvature sign maps offer semantic perspective independent description of the various contiguous surfaces in a range data scene and provide a coarse two-dimensional viewer-centered symbolic description of the scene which is useful for matching to view-centered models. Herein lies the weakness of such curvature maps. The features describe the surfaces only in terms of type and not in degree. A larger cylindrical pipe further away would be labelled exactly as a smaller cylindrical pipe. Both sections of cones and cylinders appear as ridges and there is no way of telling the directions of maximal or constant curvature. It would be impossible, in a curvature sign map, for example, to determine where two lengths of straight piping joined at an obtuse angle should be separated, or when a conduit of circular cross section ceases being a cylinder and tapers into a cone. These limitations also render features derived from curvature sign maps incapable of three-dimensional pose determination (except at the coarseness of the viewer-centered models). Curvature sign maps coupled with view-centered models of the object set to be recognized and located do provide adequate evidence to hypothesize the constructed geometric forms which give rise to the surfaces detected in the scene.

3.3 Linear Form Estimation

We are still, however, not ready to fit the constructed geometric forms to the range data because such fitting requires three-dimensional parameter estimates (a cylinder appears as a *ridge region* in curvature sign maps irrespective of the angle its axis makes with view-plane). Here, we introduce the notion of segmentation and parameter estimation by the application of *companion*

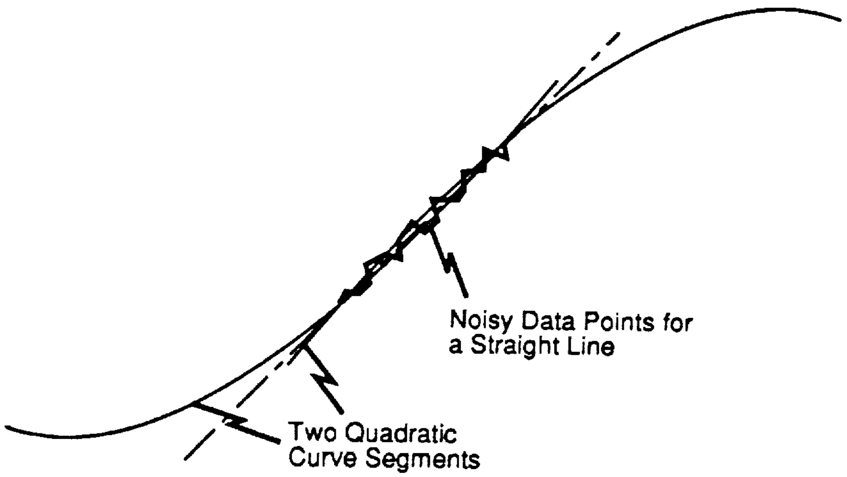

Noisy Data Points for a Straight Line
Two Quadratic Curve Segments

Figure 3.3: Fitting of minimal order functions is more appropriate for parameter estimation although higher order fits may yield lower fit errors.

linear forms to the non-linear form to describe constructed surfaces. Pending formal definition of such forms in the next chapter, companion-linear forms are linear parametric functions which will fit to data arising from the presence of non-linear surfaces (e.g. cylinders and cones which are non-linear with respect to their orientation and scale parameters) and which contain information necessary to estimate the pose of the non-linear surface.

Since the purpose of fitting companion linear forms to the data is to estimate the parameters of the underlying constructed structure, a ground rule is that the minimal order surface function should be used. Taking the two dimensional analogy (see figure 3.3), a quadratic equation can often fit a noisy straight line better than a straight line fit, but the straight line fit is more powerful for parameter estimation; and a cubic function may yield a fit of lower error to a convex surface than would a quadratic, but the quadratic fit results in less ambiguity in the shape matched (one never knows which bend in the higher order function fitted the data). In the case of planar surfaces the depth (z) value of the image is a linear combination of the x and y terms. The companion linear form for such surfaces is therefore trivially defined as planes. For cones and cylinders, however, the relation of z to x and y is far from linear in the orienting parameters (as will be shown later). In the course of this dissertation, biquadratic surfaces have been found to be an excellent intermediate or companion form for cylinders and cones. A study of the properties of biquadratic surfaces will show that the coefficients derived by fitting these surfaces to cylinders and cones yield accurate approximations of the parameters of orientation of the original constructed forms in space. It should be noted that such companion linear forms are fitted to the range data only when a hypothesis of the existence of the constructed forms are present in the scene. These hypotheses are derived from the curvature sign maps and object models. The process of fitting companion linear forms is therefore very context sensitive.

Once the companion linear forms have been fitted and the parameters of the original constructed forms have been estimated, non-linear fitting can be performed to obtain more accurate estimates.

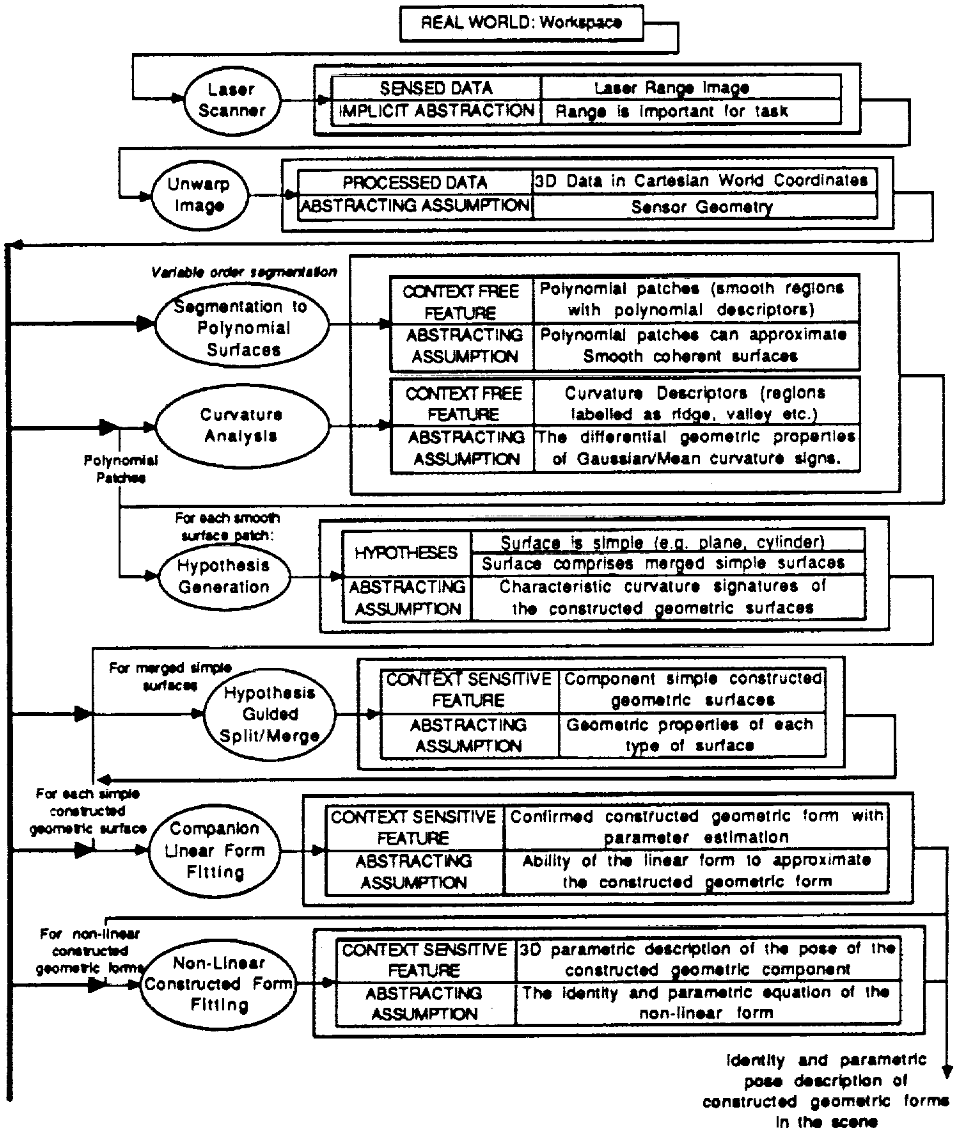
3.4 Overview of the Sequence of Abstraction

Figure 3.4 shows the sequence of abstraction implemented in this thesis along with the processes which traverse the hierarchy.

The processing begins with the transformation of the laser range imagery from the native coordinate system of the laser scanner to real world Cartesian coordinates.

There are two distinct stages in the processing sequence beside the data coordinate transformation. First, there are the *context free* processing stages which does not rely on knowledge about what is actually in the scene. These are the *segmentation to polynomial surfaces* and *curvature analysis*. The subsequent processing makes use of more and more information about the scene and its contents.

It should be noticed that the coordinate-transformed data are used throughout the system. At each stage, the abstracting mechanism makes hypotheses about the interpretation of the unwarped three-dimensional data. Thus, the system remains data-bound as it makes stronger and stronger statements about the way the data ought to be organized.


REAL WORLD: Workspace
Laser Scanner
Unwarp Image
Variable order segmentation
Segmentation to Polynomial Surfaces
Curvature Analysis
Polynomial Patches
For each smooth surface patch:
Hypothesis Generation
For merged simple surfaces
Hypothesis Guided Split/Merge
For each simple constructed geometric surface
Companion Linear Form Fitting
For non-linear constructed geometric forms
Non-Linear Constructed Form Fitting
Identity and parametric pose description of constructed geometric forms in the scene

SENSED DATA	Laser Range Image
IMPLICIT ABSTRACTION	Range is important for task

PROCESSED DATA	3D Data in Cartesian World Coordinates
ABSTRACTING ASSUMPTION	Sensor Geometry

CONTEXT FREE FEATURE	Polynomial patches (smooth regions with polynomial descriptors)
ABSTRACTING ASSUMPTION	Polynomial patches can approximate Smooth coherent surfaces

CONTEXT FREE FEATURE	Curvature Descriptors (regions labelled as ridge, valley etc.)
ABSTRACTING ASSUMPTION	The differential geometric properties of Gaussian/Mean curvature signs.

HYPOTHESES	Surface is simple (e.g. plane, cylinder) Surface comprises merged simple surfaces
ABSTRACTING ASSUMPTION	Characteristic curvature signatures of the constructed geometric surfaces

CONTEXT SENSITIVE FEATURE	Component simple constructed geometric surfaces
ABSTRACTING ASSUMPTION	Geometric properties of each type of surface

CONTEXT SENSITIVE FEATURE	Confirmed constructed geometric form with parameter estimation
ABSTRACTING ASSUMPTION	Ability of the linear form to approximate the constructed geometric form

CONTEXT SENSITIVE FEATURE	3D parametric description of the pose of the constructed geometric component
ABSTRACTING ASSUMPTION	The identity and parametric equation of the non-linear form

Figure 3.4: Block diagram showing the levels of abstraction and processes which traverse the hierarchy

More will be said about this diagram at the conclusion of this thesis after all the labelled processes have been discussed.

CHAPTER IV

MATHEMATICAL AND ALGORITHMIC BASES

In this chapter, we shall discuss the mathematics and algorithms employed in the work of this thesis. In each section, both the primary mathematics and extensions applied to this work are detailed. The chapter is divided into four sections. In the first section, both linear and non-linear coefficient or parameter estimation will be reviewed. In the second section, Besl and Jain's variable order segmentation algorithm for extracting contiguous surfaces from an image as polynomial patches is discussed. In the third section, surface curvature computation is reviewed and a method for computing curvature-based features reliably is advanced. In the fourth section, the properties of the bi-quadratic function and how they may be applied to cylinder estimation are discussed.

Effort has been made to discuss the mathematics and algorithms as they relate to this work and computer vision in general and not in the abstract.

4.1 Review of Coefficient/Parameter Estimation

Since coefficient/parameter estimation underlies all of the work here discussed, we shall begin with an overview of the principles, assumptions and techniques of linear and non-linear parameter estimation.

Fitting, in general, is the modeling of data with some overarching mathematical description. This is done by assigning to the fitting mathematical description or function an appropriate set of parameters. In this work, a surface description is desired for a set of range measurements which yield three dimensional points:

$$(x_i, y_i, z_i) \qquad (0 \leq i \leq N)$$

where N is the number of measurements taken.

Fitting can then be defined as follows:

> Given a fitting function
>
> $$z(x, y) = f(x, y; a_1 \ldots a_M)$$
>
> where M is the number of adjustable parameters of f, compute a set of values for $[A] = (a_1 \ldots a_M)$ such that some *likelihood criterion* is maximized.

Two constraints need to be satisfied for this operation to make sense. First, the data must be describable by a bivariate function and second, the fitting function must be capable of describing the data.

For a set of data to be describable by a bivariate function $f(x, y)$ on an orthogonal x-y basis, the coordinate patch C_k on that space must be one-to-one (i.e. there must be only one z_i value for each pair of x_i and y_i).

4.1.1 The Data

For a C_k coordinate patch to be one-to-one, a function $\mathbf{x} : \Re \rightarrow \mathbf{R}^3$ describing C_k for some $k \geq 1$, where $\Re$ is an open subset of $\mathbf{R}^2$ with coodinates u^1 and u^2 and $\frac{\partial \mathbf{x}}{\partial u^1} \times \frac{\partial \mathbf{x}}{\partial u^2} \neq \mathbf{0}$ on $\Re$ [131].

A subset $\Re$ of $\mathbf{R}^2$ is *open* if for every point $P \in \Re$ there is an Euclidean ϵ-neighbourhood around $P \in \Re$.[131]. In other words, for any point on the surface, there is a disk around the point which is completely contained in the surface.

The constraint, therefore, on the data is that the surface be continuous and have non-zero first partial derivatives (otherwise, the surface is no longer a function on x-y since at points of zero first partial derivatives, the surface will experience a one-to-many mapping of x, y onto z). Such surfaces or patches are called *Monge patches*.

Laser range data are constrained by the sensing physics to be a one-to-one mapping between the array indices and range readings (i.e. each location in the sensed image can have one and only one range value). The problem occurs in such images at edge and occlusion discontinuities where the first derivative approaches infinity. If step edge changes occur in a region being fitted, the fit would be poor and the disparity between the data and the fitted surface would be large. If, however, the entire range image is first segmented to obtain separate smooth regions, each of these regions would be a *Monge patch*.

4.1.2 The Fitting Function

A fitting function must be capable of describing the surface for a reasonable fit to take place. For example, if the planar function $z = a + bx + cy$ were to be fitted over a large portion of data representing a cylindrical surface, the fit would invariable be subject to large fitting errors because there is no combination of parameter values (a, b, c) which can *bend* the function into the required surface.

In general, there are two kinds of surface functions which can be fitted to the data. First, the function may describe precisely the form of the data being fitted. Fitting a function describing a cylinder to data of a sensed cylinder, a spherical function to data describing a sphere or a planar surface function to data of a planar surface fall into this first category. Second, a general function may be used to fit the data. The purpose of such fitting is often to represent the surface for reconstruction and display. The bivariate polynomial surface function is an example of such a function. For any arbitrarily complex surface (many smooth bends), an arbitrarily high-ordered polynomial can be selected to fit the surface.

Unfortunately, for most interesting structural forms like cylinders and cones, the constructing functions are non-linear in terms of their orienting parameters. This poses a problem since it is more convenient to fit linear functions. As will be explained later many problems are attendant to non-linear fitting. Unless the original surface were linear (as with planes), therefore, approximating functions are generally used for such fitting.

4.1.3 Fitting as a Minimization Process

The task of fitting a function to a set of data points may be framed as one of maximizing some *goodness-of-fit* metric if a closed-form description of such a metric exists. The maximizing of this metric would constitute the minimizing of fitting errors.

The fitting of a bivariate function

$$z(x, y) = f(x, y; a_1 \ldots a_M) \tag{4.1}$$

(M being the number of adjustable parameters of f) to a set of data

$$z_i \in \mathbf{Z} \mid 1 \leq i \leq N$$

where N is the number of sample points may be stated as follows:

$$\text{maximize over } a_1 \ldots a_M : \qquad \sum_{i=1}^{N} G(z_i, x_i, y_i) \tag{4.2}$$

(where G is the goodness-of-fit metric of $f(x_i, y_i)$ to z_i).

If we assume that the sampling and quantization errors in the data are independently and normally distributed with a standard deviation σ_i and zero mean, the probability of error at each point is the Gaussian probability:

$$P_i = \Delta z \, e^{-\frac{1}{2}(\frac{z_i - f(x_i, y_i)}{\sigma_i})^2} \tag{4.3}$$

where Δz is the mean interval between z readings.

Owing to the independence assumption the probability of the data set is the product of the probabilities of the points:

$$P = \prod_{i=1}^{N} \left\{ \Delta z \, e^{-\frac{1}{2}\left(\frac{z_i - f(x_i, y_i)}{\sigma_i}\right)^2} \right\} \tag{4.4}$$

The maximization of equation 4.4 may be obtained by minimizing the logarithm of its reciprocal:

$$\left[\sum_{i=1}^{N} \frac{(z_i - f(x_i, y_i))^2}{2\sigma_i{}^2} \right] - N log \Delta z \tag{4.5}$$

Since $N$and Δz are constants, if the standard deviation σ_i is also constant for all i, this is equivalent to minimizing

$$(z_i - f(x_i, y_i))^2 \tag{4.6}$$

This is the familiar least-squares fitting criterion.

If, however, the standard deviation σ_i varies across the range image, the minimization of equation 4.4 is equivalent to reducing the χ^2 function given by:

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{z_i - f(x_i, y_i)}{\sigma_i} \right)^2 \tag{4.7}$$

4.1.4 Review of Linear Coefficient/Parameter Estimation

The linear function to be fit to the data may be written:

$$z(x, y) = \sum_{k=1}^{M} a_k X_k(x, y) \tag{4.8}$$

where $X_1(x, y), X_2(x, y), \cdots X_M(x, y)$ are the basis functions

$a_1, a_2, \cdots a_M$ are the function parameters

or

$$\mathbf{z} = \mathbf{a} \cdot \mathbf{X} \tag{4.9}$$

Equation 4.7 then becomes:

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{z_i - \sum_{k=1}^{M} a_k X_k(x_i, y_i)}{\sigma_i} \right)^2 \tag{4.10}$$

For convenience in the ensuing discussion, we define two matrices **A** and **b** such that

$$\begin{aligned} A_{ij} &= \frac{X_j(x_i, y_i)}{\sigma_i} \\ b_i &= \frac{z_i}{\sigma_i} \end{aligned}$$

A is known as the design matrix.

Solution by Normal Equations

The χ^2 expression of equation 4.10 is at its minimum value when its first derivatives with respect to all $a_1, a_2, \ldots a_M$ are zero. The minimization of χ^2 thus becomes the solution of:

$$\sum_{i=1}^{N} \frac{1}{\sigma_i{}^2} \left[z_i - \sum_{j=1}^{M} a_j X_j(x_i, y_i) \right] X_k(x_i, y_i) \qquad k = 1, 2, \ldots M \tag{4.11}$$

This yields:

$$\sum_{i=1}^{N} \frac{z_i X_k(x_i, y_i)}{\sigma_i{}^2} = \sum_{j=1}^{M} \left[\sum_{i=1}^{N} \frac{X_j(x_i, y_i) X_k(x_i, y_i)}{\sigma_i{}^2} \right] a_j \qquad k = 1, 2, \ldots M \tag{4.12}$$

Let

$$\alpha_{kj} = \sum_{i=1}^{N} \frac{X_j(x_i, y_i) X_k(x_i, y_i)}{\sigma_i{}^2} \tag{4.13}$$

$$\beta_j = \sum_{i=1}^{N} \frac{y_i X_k(x_i, y_i)}{\sigma_i{}^2} \tag{4.14}$$

By inspection, $[\alpha] = \mathbf{A}^T \cdot \mathbf{A}$ and $[\beta] = \mathbf{A}^T \cdot \mathbf{b}$. Equation 4.12 can now be rewritten as:

$$\sum_{j=1}^{M} \alpha_{kj} a_j = \beta_k \qquad k = 1, 2, \ldots M \tag{4.15}$$

or in matrix notation:

$$\begin{aligned} [\alpha] \cdot \mathbf{a} &= [\beta] \\ \mathbf{A}^T \cdot \mathbf{A} \cdot \mathbf{a} &= \mathbf{A}^T \cdot \mathbf{b} \end{aligned} \tag{4.16}$$

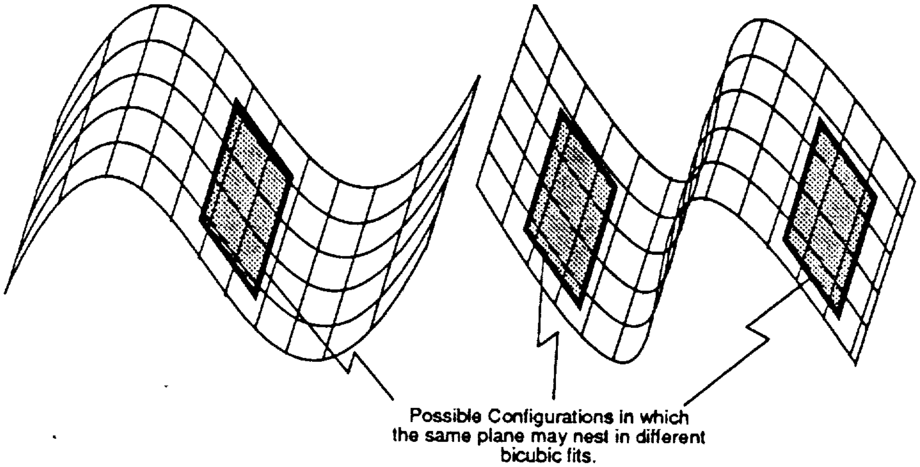

Possible Configurations in which the same plane may nest in different bicubic fits.

Figure 4.1: Multiple portions of a bicubic surface may fit the same plane.

Equation 4.12 and equation 4.15 are known as *normal equations*. The solution of these equations for a will yield the χ^2 minimized result. Such minimization may be accomplished by the standard linear algebra techniques of LU-decomposition with backsubstitution or by Gauss-Jordan elimination or by QR minimization of the design matrix. As discussed in [140], the latter method is preferable because direct solution of the normal equations by the first two methods is susceptible to round off errors.

Solution by Singular Value Decomposition

A serious problem dogs the normal equations methods of χ^2 minimization. When the normal equations are singular or close to singular, the technique will fail. Should the equations be singular, one of the pivots in the solution matrix becomes zero and all the linear equation solution algorithms mentioned previously will fail. When one of the pivots becomes very small, the typical result is that the linear solution becomes unstable, yielding very large parameters which cancel each other out when the resulting function is evaluated. While such a singular solution is not normally expected in grossly overdetermined (much more data than unknown variables) fitting problems, they can occur in surface fitting when more than one set of parameters can fit the data. For example, if a bicubic surface function were fitted to small patch of data which represents a plane, it could occur that multiple combinations of coefficients will fit the data. Since bicubic surfaces are capable of producing 'S-shaped' sections, one can see intuitively that there are multiple portions of the 'S' which might fit the data (see figure 4.1). This results in an ambiguity in the solution. For this reason, singular value decomposition (SVD)[140, 81, 155] was the algorithm of choice in the work reported here.

The minimization of the χ^2 expression in equation 4.10 can be rewritten as to the minimization of

$$\chi^2 = |\mathbf{A} \cdot \mathbf{a} - \mathbf{b}|^2 \tag{4.17}$$

In traditional linear algebra, it has been shown [140, 81, 155] that any $M \times N$ matrix $\mathbf{A}$ can be decomposed as follows:

$$\mathbf{A} = \mathbf{U} \cdot \mathbf{w} \cdot \mathbf{V}^T \tag{4.18}$$

where $\mathbf{U}$ and $\mathbf{V}$ are orthogonal matrices (i.e. $\mathbf{U} \cdot \mathbf{U}^T = \mathbf{I}$ and $\mathbf{V} \cdot \mathbf{V}^T = \mathbf{I}$, $\mathbf{I}$ being the identity matrix). $\mathbf{U}$ is an $M \times N$ matrix and the dimension of $\mathbf{V}$ is $N \times N$. The matrix $\mathbf{w}$ is an $N \times N$ diagonal matrix with non negative diagonal elements.

Since $\mathbf{U}$ and $\mathbf{V}$ are orthogonal, $\mathbf{U}^{-1} = \mathbf{U}^T$ and $\mathbf{V}^{-1} = \mathbf{V}^T$. If $M = N$, it follows then that the inverse of $\mathbf{A}$ is given by:

$$\mathbf{A}^{-1} = \mathbf{V} \cdot [\mathrm{diag}\frac{1}{w_{ii}}] \cdot \mathbf{U}^T \tag{4.19}$$

At this point, the algorithm applies a very important relaxation. The chief question to be answered is whether $\mathbf{b}$ lies in the range of $\mathbf{A}$. If it does, a solution exists for $\mathbf{a}$. While it seems counter-intuitive, Golub et. al. [81] showed that the solution is attainable by simply replacing all occurrences of $\frac{1}{w_{ii}}$ with zero in the new diagonal matrix whenever w_{ii} in **w** is zero. The reason for this is that within the null space, any corresponding column of **V** can be added to **a** in any combination. The SVD solution then becomes:

$$\mathbf{a} = \mathbf{V} \cdot \mathbf{w} \cdot (\mathbf{U} \cdot \mathbf{b}) \tag{4.20}$$

where **w** is the modified $[\text{diag}\frac{1}{w_{ii}}]$.

'$\mathbf{V} \cdot \mathbf{w} \cdot \mathbf{U}$' is known as the pseudoinverse of **A** and Golub et. al. showed that this solution works for overspecified systems where $(N > M)$ as well.

The algorithm used in this work is a modified form of the routines due to the *Numerical Recipes Library* by Press et. al[140]. In the experimentations, it was found that the ϵ threshold of w_{ii} at which to set the inverse to zero is critical in the fitting. For single precision, the value given in the library is 10^{-5} and this seems to work well. At double precision, this threshold should be 10^{-6} to 10^{-12}. Within the range of acceptable values for the double precision fit, there is little deviance in the result, but there seems to be a threshold (approximately 10^{-5}) at which the fit becomes unstable.

4.1.5 Review of Non-Linear Coefficient/Parameter Estimation

Non-linear parameter estimation involves the fitting of a set of data to a function which varies nonlinearly with respect to the parameters being estimated.

Consider the χ^2 function $\chi^2(\mathbf{a})$, where **a** is a multivariate coordinate system in M dimensions (M being the number of parameters being estimated). By Maclaurin series, this is:

$$\begin{aligned}\chi^2(\mathbf{a}) &= \chi^2(\mathbf{a}_0) + \sum_{i=1}^{M} \frac{\partial \chi^2}{\partial a_i}(\mathbf{a}_0) a_i + \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M} \frac{\partial^2 \chi^2}{\partial a_i \partial a_j}(\mathbf{a}_0) a_i a_j + \cdots \\ &\approx \gamma - \mathbf{d} \cdot \mathbf{a} + \frac{1}{2}\mathbf{a} \cdot \mathbf{D} \cdot \mathbf{a}\end{aligned} \tag{4.21}$$

where $\mathbf{a}_0$ is the origin of the coordinate system $\mathbf{a}$

$\gamma \equiv \chi^2(\mathbf{a}_0)$

$\mathbf{d}$ is the dimension M gradient vector: $\mathbf{d} \equiv -\nabla\chi^2(\mathbf{a})$

$\mathbf{D}$ is the $M \times M$ Hessian matrix: $D_{ij} \equiv \frac{\partial^2\chi^2}{\partial a_i \partial a_j}(\mathbf{a}_0)$

Therefore, the gradient of the χ^2 function may be approximated by:

$$\nabla\chi^2 = \mathbf{D}\cdot\mathbf{a} - \mathbf{d} \tag{4.22}$$

Hence, the extremum of the χ^2 function occurs when

$$\mathbf{D}\cdot\mathbf{a}_{min} = \mathbf{d} \tag{4.23}$$

and, at some approximation point $\mathbf{a}_{approx}$,

$$\mathbf{D}\cdot\mathbf{a}_{approx} = \nabla\chi^2(\mathbf{a}_{approx}) + \mathbf{d} \tag{4.24}$$

By subtracting equation 4.24 from equation 4.23 and algebraic manipulation, we have:

$$\mathbf{D}\cdot(\mathbf{a}_{min} - \mathbf{a}_{approx}) = -\nabla\chi^2(\mathbf{a}_{approx}) \tag{4.25}$$

or

$$\mathbf{a}_{min} = \mathbf{a}_{approx} + \mathbf{D}^{-1}\cdot[-\nabla\chi^2(\mathbf{a}_{approx})] \tag{4.26}$$

Thus, it is possible to jump directly from some approximation point to the χ^2 minimum point if equation 4.21 is a good approximation and the inverse of the Hessian matrix is known.

From section 4.1.3, we have the expression for χ^2 (equation 4.7) as:

$$\chi^2(\mathbf{a}) = \sum_{i=1}^{N}\left(\frac{z_i - f(x_i, y_i; \mathbf{a})}{\sigma_i}\right)^2$$

where N is the number of points (x_i, y_i, z_i) in the data set. The gradient ∇f is thus:

$$\frac{\partial\chi^2}{\partial a_k} = -2\sum_{i=1}^{N}\frac{z_i - f(x_i, y_i; \mathbf{a})}{{\sigma_i}^2}\frac{\partial f(x_i, y_i; \mathbf{a})}{\partial a_k} \qquad k = 1, 2, \ldots, M \tag{4.27}$$

where M is the number of parameters being estimated.

The elements of the Hessian matrix D_{kl} are given by:

$$\frac{\partial^2\chi^2}{\partial a_k \partial a_l} = 2\sum_{i=1}^{N}\frac{1}{\sigma_i{}^2}\left[\frac{\partial f(x_i, y_i; \mathbf{a})}{\partial a_k}\frac{\partial f(x_i, y_i; \mathbf{a})}{\partial a_l} - [z_i - f(x_i, y_i; \mathbf{a})]\frac{\partial^2 f(x_i, y_i; \mathbf{a})}{\partial a_k \partial a_l}\right] \qquad (4.28)$$

Equation 4.25 may thus be written as the set of linear equations:

$$\sum_{l=1}^{M} \alpha_{kl}\delta a_l = \beta_k \qquad k = 1, 2, \ldots, M \qquad (4.29)$$

where

$$\beta \equiv -\frac{1}{2}\frac{\partial\chi^2}{\partial a_k} \quad \text{and} \quad \alpha \equiv \frac{1}{2}\frac{\partial^2\chi^2}{\partial a_k \partial a_l}$$

$[\alpha]$ is usually called the *curvature matrix* in the literature.

For each set of parameter approximations $\mathbf{a}_{approx}$, equations 4.29 may be solved for the increments δa_l to be added to $\mathbf{a}_{approx}$ to obtain a better approximation.

Should the iterative solution of equations 4.29 not converge, an alternative solution is the *steepest descent* method which adds a small constant fraction of the the *local downhill gradient* (negation of the gradient vector):

$$\mathbf{a}_{next} = \mathbf{a}_{current} - constant \times \nabla\chi^2(\mathbf{a}_{current}) \qquad (4.30)$$

or

$$\delta a_l = constant \times \beta_k \qquad k = 1, 2, \ldots, M \qquad (4.31)$$

The *Levenberg Marquardt method* applies both the Hessian matrix solution of equation 4.29 and the steepest descent solution of equation 4.31 for non-linear parameter estimation. The steepest descent method operates when the approximation is far from the solution and the Hessian matrix method is activated when the approximation is close to the solution. The Levenberg-Marquardt algorithm in [140] is used in this work.

4.2 Contiguity-based Segmentation

At this juncture, it is *fitting* that we discuss the *variable-order segmentation* work of Besl and Jain[22, 25, 26]. The variable-order algorithm and programs were used in this work to perform the initial segmentation of the range imagery.

4.2.1 The Algorithm

The variable-order segmentation strategy is based on the idea of *spatial coherence* or *surface coherence*. The basic idea is that surfaces that are contiguous (i.e. no step jumps or discontinuities) can be approximated by bivariate polynomials. If a sensed scene is made up of a set of such piecewise smooth patches, then an appropriate segmentation scheme should yield a corresponding set of polynomial surfaces. The Besl-Jain algorithm does this in a region growing and surface-order varying process.

In overview, the algorithm proceeds as follows:

1. Select a set of small *seed* regions $\mathcal{R}_{seed}$ at which there is a high degree of confidence that they are portions of contiguous surfaces.
2. Rank these seed regions by order of confidence
3. For each region r_i in $\mathcal{R}_{seed}$, do the following:
   (a) Fit a polynomial surface function $\mathcal{S}_i$ (where i the polynomial order is initially 1 and $\mathcal{S}_i$ is planar).

   (b) If the *goodness-of-fit* threshold is not exceeded for the $\mathcal{S}_i$ fit, grow the region and perform the $\mathcal{S}_i$ fit again.

   (c) If the *goodness-of-fit* threshold is exceeded, increment i (i.e. $\mathcal{S}_i$ progresses from planar to biquadratic to bicubic to biquartic).

   (d) If i is greater than a set order limit (e.g. $i > 4$ or $\mathcal{S}_i$ is greater than biquartic), proceed to the next seed region (i.e. go to step 3g)

   (e) If there has been no improvement in the *goodness-of-fit* for two successive surface orders, go to step 3g.

(f) Go to step 3a.

(g) If the *goodness-of-fit* of resulting grown region is within the acceptance threshold, the region is collected into the set of *grown surfaces* and remove from $\mathcal{R}_{seed}$ those seed regions which are in the grown region.

Besl/Jain used surfaces computed from differential geometry concepts to obtain the surfaces in step 1. Using these surface regions as the initial segmentation hypothesis, the regions were shrunk to obtain seed regions. The rationale for this was that while the initial segmentation was rather inaccurate, there is good confidence that there is some *kernel* of truth in the segmentation – especially in the middle of the segmented regions.

The increasing orders of surface functions being fit could be construed as a partial order in surface space where *a surface of order i will always yield a better or equal goodness-of-fit to a coordinate patch r than a surface of k if $i > k$. If two successive orders i and $i + 1$ yield the same goodness-of-fit, then the all fits of order $k > i$ will yield the same goodness-of-fit.* This is obviously true for increasing orders of polynomials in fitting of data in two dimensions x and y. By extension, it is also true for bivariate polynomial functions fitting three dimensional data.

4.2.2 Comments on the Variable-Order Segmentation Algorithm

The variable-order region growing strategy has been shown to work impressively on a wide variety of data [22, 25, 26] because its underlying assumption of spatial coherence is very general.

As will be discussed later, the computation of surface curvature descriptors using digital kernel-type partial differentiation of the image is unstable. The algorithm often generated a very fragmented initial segmentation. Since the purpose of this segmentation is only to provide a first guess for variable-order fitting and region growing, the only criterion for this initial segmentation is that it yields hypotheses of contiguous (spatially coherent) regions. A simple smoothing filter which rejects regions of data discontinuity like the *morphological spherical filter*[70] could be

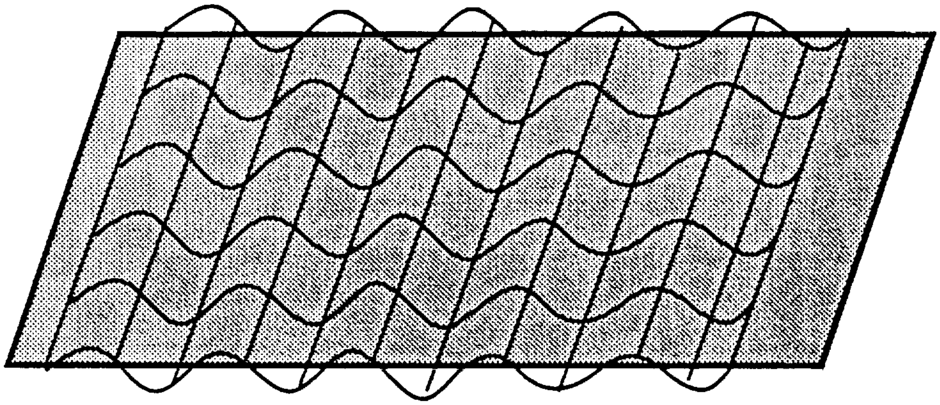

Ringing in a high order polynomial fit
to a plane (effect exaggerated for illustration)

Figure 4.2: Ringing occurs when one fits a higher order function to the data than is necessary.

used to provide more stable initial seed regions. A similar algorithm using variable scale edge detectors is described in [123].

Experiments performed with this algorithm on laser data showed that the variable-order segmentation often (for most regions) bottomed out at the order limit. It fitted the highest allowable order surface (biquartic in the program used) even if the original data was planar. The reason for this is that the higher order surfaces are more compliant to sampling measurement noise, often yielding better goodness-of-fit measures than lower order surfaces. This said, one may be tempted to skip variable order fitting and simply fit high order surfaces to the data from the very beginning. Such an exercise, however, is susceptible to the common illness which infects splining operations. A surface order that is too high suffers from the problem of 'ringing' (see figure 4.2). Variable order fitting will reject higher order fits in which ringing occurs because it terminates when an error threshold is satisfied. This accepts the lowest order fit which fits the data before severe ringing can begin.

The surfaces generated by the algorithm are smooth regions which are described by a surface order and a set of coefficients to the bivariate polynomial function. While this provides a good description of the scene in that one can reconstruct the scene quite accurately by simply applying the bivariate polynomials to each point in the regions, it does not readily lend itself to semantic interpretation of the surfaces.

4.3 Curvature Analysis and Structural Hypothesis

Sign maps of the Gaussian and mean curvatures of a surface provide a means of describing the surface in symbolic terms of semantic significance. These curvatures are treated in the field of classical and modern differential geometry [125, 131]. Besl and Jain discusses the properties of these surface metrics in great detail in [19, 20, 21, 23, 24, 25, 26] and the application of the sign maps of these surface curvatures to object recognition is a major portion of Paul Besl's doctoral dissertation[22]. In our discussion, we shall begin with a review of the mathematics and proceed to treat the extension of Besl's work to yield stable features whose computation is robust.

4.3.1 Differential Geometry Review

In classical differential geometry, the shape of a surface is uniquely determined by the first and second fundamental forms of the surface, **I** and **II** respectively.

As in the previous discussion on surface fitting, the space in which the curvature measures are computed takes the form of C^2 continuous (twice differentiable) *Monge* patches.

$$z = f(x, y)$$

Applying the parameterization: $u = x$, $v = y$ such that $w = g(u, v)$, the surface may be rewritten as the vector:

$$\mathbf{w} = [\begin{array}{ccc} u & v & g(u,v) \end{array}] \qquad (4.32)$$

The First Fundamental Form

The first fundamental form of the surface $\mathbf{w}$ is given by:

$$\begin{aligned} \mathbf{I} &= d\mathbf{w} \cdot d\mathbf{w} \\ &= (\mathbf{w}_u du + \mathbf{w}_v dv) \cdot (\mathbf{w}_u du + \mathbf{w}_v dv) \\ &= (\mathbf{w}_u \cdot \mathbf{w}_u) du^2 + 2(\mathbf{w}_u \cdot \mathbf{w}_v) du\, dv + (\mathbf{w}_v \cdot \mathbf{w}_v) dv^2 \\ &= E\, du^2 + F\, dv^2 + G\, dv^2 \end{aligned} \tag{4.33}$$

where

$$E = \mathbf{w}_u \cdot \mathbf{w}_u \ ; \ F = \mathbf{w}_u \cdot \mathbf{w}_v \ ; \ G = \mathbf{w}_v \cdot \mathbf{w}_v$$

The subscripts denote partial differentiation such that

$$\mathbf{w}_u = \frac{\partial \mathbf{w}}{\partial u} \ ; \ \mathbf{w}_v = \frac{\partial \mathbf{w}}{\partial v}$$

$\mathbf{w}_u$ and $\mathbf{w}_v$ are tangent vectors to the surface at point $(u, v, g(u, v))$ and form the basis of the tangent plane $T(u, v)$ of the surface at that point.

Alternatively, the first fundamental form may be written in matrix form as:

$$\begin{aligned} \mathbf{I}(u, v, du, dv) &= [du\ dv] \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} \\ &= \vec{\mathbf{du}}^T [\mathrm{g}] \vec{\mathbf{du}} \end{aligned} \tag{4.34}$$

where $g_{11} = E = \mathbf{w}_u \cdot \mathbf{w}_u$, $g_{12} = g_{21} = F = \mathbf{w}_u \cdot \mathbf{w}_v$ and $g_{22} = G = \mathbf{w}_v \cdot \mathbf{w}_v$.

The symmetric matrix [g] is known as the *metric coefficients*, the *coefficients of the metric tensor*, the *coefficients of the Riemannian metric* or simply the *metric tensor* or *metric* of the surface. It should be noted that the E, F and G, and thus the first fundamental form, are defined only in term of $\mathbf{w}$ and its first derivatives. $\mathbf{I}$ is therefore an *intrinsic* property of the surface. It measures the variation of the surface $\mathbf{w}$ at point (u, v) on the *Monge* basis as a result of some movement (du, dv) at that point. Some useful properties related to this metric are:

$$g = |\mathbf{g}| = |\mathbf{w}_u \times \mathbf{w}_v|^2$$

$$g^{11} = \frac{g_{22}}{g} \; ; \; g^{12} = g^{21} = -\frac{g_{12}}{g} \; ; \; g^{22} = \frac{g_{11}}{g}$$

where g^{kl} is the (k, l) entry of the inverse matrix of (g_{ij})

$$(4.35)$$

The Second Fundamental Form

The second fundamental form of a surface $\mathbf{w}$ is given by:

$$\begin{aligned} \mathbf{II} &= -d\mathbf{w} \cdot d\vec{\mathbf{N}} \\ &= \mathbf{w}_u \cdot \vec{\mathbf{N}}_u du^2 - (\mathbf{w}_u \cdot \vec{\mathbf{N}}_v + \mathbf{w}_v \cdot \vec{\mathbf{N}}_u) du\, dv - \mathbf{w}_v \cdot \vec{\mathbf{N}}_v dv^2 \\ &= L\, du^2 + 2M\, du\, dv + N\, dv^2 \end{aligned} \tag{4.36}$$

where

$$L = -\mathbf{w}_u \cdot \vec{\mathbf{N}}_u \; ; \; M = -\frac{1}{2}(\mathbf{w}_u \cdot \vec{\mathbf{N}}_v + \mathbf{w}_v \cdot \vec{\mathbf{N}}_u) \; ; \; N = -\mathbf{w}_v \cdot \vec{\mathbf{N}}_v$$

$\vec{\mathbf{N}}$ being the unit normal of the surface given by:

$$\vec{\mathbf{N}} = \frac{\mathbf{w} \times \vec{\mathbf{N}}}{|\mathbf{w} \times \vec{\mathbf{N}}|} \tag{4.37}$$

with differential $d\vec{\mathbf{N}} = \vec{\mathbf{N}}_u du + \vec{\mathbf{N}}_v dv$.

Since the tangent vectors $\mathbf{w}_u$ and $\mathbf{w}_v$ are perpendicular to the normal $\vec{\mathbf{N}}$ for all (u, v),

$$0 = (\mathbf{w}_u \cdot \vec{\mathbf{N}})_u = \mathbf{w}_{uu} \cdot \vec{\mathbf{N}} + \mathbf{w}_u \cdot \vec{\mathbf{N}}_u$$

$$0 = (\mathbf{w}_u \cdot \vec{\mathbf{N}})_v = \mathbf{w}_{uv} \cdot \vec{\mathbf{N}} + \mathbf{w}_u \cdot \vec{\mathbf{N}}_v$$

$$0 = (\mathbf{w}_v \cdot \vec{\mathbf{N}})_u = \mathbf{w}_{vv} \cdot \vec{\mathbf{N}} + \mathbf{w}_v \cdot \vec{\mathbf{N}}_u$$

$$0 = (\mathbf{w}_v \cdot \vec{\mathbf{N}})_v = \mathbf{w}_{vv} \cdot \vec{\mathbf{N}} + \mathbf{w}_v \cdot \vec{\mathbf{N}}_v$$

where the double subscripts denote second partial derivatives with respect to the variables indicated by the subscripts, such that

$$\begin{aligned} \mathbf{w}_{uu} &= \frac{\partial^2 \mathbf{w}}{\partial u^2} \; ; \; \mathbf{w}_{vv} = \frac{\partial^2 \mathbf{w}}{\partial v^2} \\ \mathbf{w}_{uv} &= \mathbf{w}_{vu} = \frac{\partial^2 \mathbf{w}}{\partial u \partial v} \end{aligned}$$

Thus

$$\begin{aligned} \mathbf{w}_{uu} \cdot \vec{\mathbf{N}} &= -\mathbf{w}_u \cdot \vec{\mathbf{N}}_u \\ \mathbf{w}_{uv} \cdot \vec{\mathbf{N}} &= -\mathbf{w}_u \cdot \vec{\mathbf{N}}_v = -\mathbf{w}_v \cdot \vec{\mathbf{N}}_u \\ \mathbf{w}_{vv} \cdot \vec{\mathbf{N}} &= -\mathbf{w}_v \cdot \vec{\mathbf{N}}_v \end{aligned}$$

L, M and N of the second fundamental form can therefore be written as

$$L = \mathbf{w}_{uu} \cdot \vec{\mathbf{N}} \; ; \; M = \mathbf{w}_{uv} \cdot \vec{\mathbf{N}} \; ; \; N = \mathbf{w}_{vv} \cdot \vec{\mathbf{N}} \tag{4.38}$$

Alternatively, the second fundamental form may be written in matrix form as:

$$\begin{aligned} \mathbf{II}(u, v, du, dv) &= [du\, dv] \begin{bmatrix} b_{11} & b_{12} \\ b21 & b_{22} \end{bmatrix} \begin{bmatrix} du \\ dv \end{bmatrix} \\ &= \vec{\mathbf{du}}^T [\mathbf{b}] \vec{\mathbf{du}} \end{aligned} \tag{4.39}$$

where $b_{11} = L = \mathbf{w}_{uu} \cdot \vec{\mathbf{N}}$, $b_{12} = b_{21} = M = \mathbf{w}_{uv} \cdot \vec{\mathbf{N}}$ and $b_{22} = N = \mathbf{w}_{vv} \cdot \vec{\mathbf{N}}$.

The symmetric matrix [**b**] is known as the *second fundamental form matrix.* It should be noted that L, M and N, and thus the second fundamental form, are defined not only in terms of **w** and its partial derivatives, but also the pointwise surface normal $\vec{\mathbf{N}}$. **II** is therefore said to be an *extrinsic* property of the surface. It measures the correlation between the variation of the normal vector $\vec{\mathbf{N}}$ and the variation in surface **w** at point (u, v) on the *Monge* basis as a result of some movement (du, dv) at that point.

The Wiengarten Map

An alternative way to view the first and second fundamental forms is by means of the Wiengarten map **L** which is defined as follows:

> The Wiengarten map **L** is, for each point $P \in \mathbf{w}$, the function $\mathbf{L} : T_p\mathbf{w} \rightarrow \mathbf{R}^3$ given by $\mathbf{L}(\mathbf{w}) = -\mathbf{w} \cdot \mathbf{N}$ where $T_p\mathbf{w}$ is the tangent space of the surface **w** at point P.[131]

This is a linear mapping of vectors in the tangent plane to other vectors in the tangent plane and describes the directional derivative of the normal vector to the surface. **L** can be defined in terms of the matrices of the first and second fundamental forms by:

$$\mathcal{L}^j{}_k = \sum b_{ik} g^{ij} \text{ or } \mathbf{L} = [\mathbf{g}^{-1}][\mathbf{b}] \tag{4.40}$$

where $\mathcal{L}^j{}_k$ is the (j,k) element of the Wiengarten map **L**.

The E, F, G, L, M, N notation of the first and second fundamental forms are more suited for some purposes, while for others, it is more convenient to use the Wiengarten map as an operator.

4.3.2 Differential Geometry-Based Surface Properties

A number of important surface properties can be computed from the first and second fundamental forms and the Wiengarten map, **I**, **II** and **L** respectively.

Normal and Principal Curvatures

Let some space curve s lie in the surface **w** defined on the *Monge* space (u, v). The *normal curvature vector* $\mathbf{k}_n$ of s at point $P = (u, v)$ is the vector projection of the *curvature vector* **k** of s at that point onto the normal **N** of **w**. i.e.

$$\begin{aligned} \mathbf{k}_n &= (\mathbf{k} \cdot \mathbf{N})\mathbf{N} \\ &= \kappa_n \mathbf{N} \end{aligned} \tag{4.41}$$

where

$$\kappa_n = \mathbf{k} \cdot \mathbf{N} \tag{4.42}$$

is known as the *normal curvature*.

Parameterizing the curve in t such that the surface along s is given by $\mathbf{w} = [\ u(t) \quad v(t) \quad g(u(t), v(t))\ ]$, the unit tangent vector $\mathbf{t}$ of s is given by $\mathbf{t} = \frac{d\mathbf{w}}{ds} = \frac{d\mathbf{w}}{dt}/|\frac{d\mathbf{w}}{dt}|$ and the curvature vector $\mathbf{k}$ at P is given by $\mathbf{k} = \frac{d\mathbf{t}}{ds} = \frac{d\mathbf{t}}{dt}/|\frac{d\mathbf{w}}{dt}|$ Applying the fact that $\mathbf{k}$ and $\mathbf{N}$ are perpendicular by definition, equation 4.42 expands to:

$$\begin{aligned} \kappa_n &= -\frac{\left(\mathbf{w}_u \frac{du}{dt} + \mathbf{w}_v \frac{dv}{dt}\right) \cdot \left(\mathbf{N}_u \frac{du}{dt} + \mathbf{N}_v \frac{dv}{dt}\right)}{\left(\mathbf{w}_u \frac{du}{dt} + \mathbf{w}_v \frac{dv}{dt}\right) \cdot \left(\mathbf{w}_u \frac{du}{dt} + \mathbf{w}_v \frac{dv}{dt}\right)} \\ &= \frac{L\frac{du}{dt}^2 + 2M\frac{du}{dt}\frac{dv}{dt} + N\frac{dv}{dt}^2}{E\frac{du}{dt}^2 + 2F\frac{du}{dt}\frac{dv}{dt} + G\frac{dv}{dt}^2} \end{aligned} \tag{4.43}$$

It should be observed that by factoring out $\left(\frac{dv}{dt}\right)^{-2}$ from both the numerator and denominator of equation 4.43, κ_n is dependent only on the ratio of $\frac{du}{dt}$ to $\frac{dv}{dt}$ and not their absolute values. Hence, one may consider the normal curvature at point P in the direction $du : dv$, $du \neq 0$ and $dv \neq 0$, in terms the first and second fundamental forms:

$$\begin{aligned} \kappa_n &= \frac{L\,du^2 + 2M\,du\,dv + N\,dv^2}{E\,du^2 + 2F\,du\,dv + G\,dv^2} \\ &= \frac{\mathbf{II}}{\mathbf{I}} \end{aligned} \tag{4.44}$$

The *principal curvatures* κ_1 and κ_2 of a surface $\mathbf{w}$ are the maximal and minimal normal curvatures of the surface corresponding to the *principal directions*. In terms of the Wiengarten map $\mathbf{L}$, the principal curvatures are the eigen values of $\mathbf{L}$, the corresponding eigen vectors of which define the principal directions. The principal curvatures and directions of a surface at point P can therefore be viewed as the extrema of the Wiengarten map of the surface at that point. Solving the system, κ_1 and κ_2 can be expressed in terms of E, F, G, L, M, N as the roots of:

$$(EG - F^2)\kappa^2 - (EN + GL - 2FM)\kappa + (LN - M^2) = 0 \tag{4.45}$$

or

$$\kappa_{1,2} = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A} \tag{4.46}$$

where $A = (EG - F^2)$; $B = (EN + GL - 2FM)$ and $C = (LN - M^2)$.

The principal directions are defined by the roots of:

$$(FN - GM)\tan^2\phi + (EN - GL)\tan\phi + (EM - FL) = 0 \tag{4.47}$$

or

$$\phi_{1,2} = \tan^{-1}\left(\frac{-B \pm \sqrt{B^2 - 4AC}}{2A}\right) \tag{4.48}$$

where $A = (FN - GM)$; $B = (EN - GL)$ and $C = (EM - FL)$.

Since the principal curvatures of a surface at a point are the maximal and minimal normal curvatures of the surface at that point, principal curvatures and directions are suitable for application to the generation of hypotheses for cylinder parameter estimation. This will be considered later in our discussion of the axis of sweep of cylinders.

Gaussian and Mean Curvatures

The *Gaussian curvature* K and the *mean curvature* H of a surface

$$\mathbf{w} = \ [u \quad v \quad g(u,v)]\ ]$$

are defined as follows:

> The *Gaussian curvature* of w at point P is the determinant of the Wiengarten map $|\mathbf{L}|$ at that point. The *mean curvature* of w at point P is half the trace of the Wiengarten map $\frac{1}{2}$ trace $(\mathbf{L})$ at that point.

Thus,

$$K \;=\; |\mathbf{L}| \;=\; \det\left([\mathbf{g}^{-1}][\mathbf{b}]\right)$$

$$= \det[\mathbf{g}^{-1}]\ \det[\mathbf{b}] \tag{4.49}$$

$$H = \frac{1}{2} \text{trace } [\mathbf{g}^{-1}][\mathbf{b}] \tag{4.50}$$

In the E, F, G, L, M, N notations of the first and second fundamental form, these become:

$$K = \frac{LM - N^2}{EF - F^2} \tag{4.51}$$

$$H = \frac{EN + GL - 2FM}{2(EG - F^2)} \tag{4.52}$$

Applying equation 4.34, equation 4.39, and the relations given in equation 4.35 to equation 4.49 and equation 4.50 respectively, and solving for in terms of $g(u,v)$ (where $\mathbf{w} = [\ u \quad v \quad g(u,v)\ ]$), we have

$$K = \frac{g_{uu}g_{vv} - g_{uv}^2}{(1 + g_u^2 + g_v^2)^2} \tag{4.53}$$

$$H = \frac{g_{uu}(1 + g_v^2) + g_{vv}(1 + g_u^2) - 2g_u g_v g_{uv}}{2\left(\sqrt{1 + g_u^2 + g_v^2}\right)^3} \tag{4.54}$$

where the subscripts of g denote partial differentiation. For example,

$$g_{uv} = \frac{1}{\partial v}\partial\left(\frac{\partial g(u,v)}{\partial u}\right)$$

These curvatures possess the following properties:

- The Gaussian and mean curvatures are invariant with respect to viewing perspective (rotation, translation and scale) and depend solely on the *shape* of the surface.

- The Gaussian curvature K at a point on the *Monge* basis (x, y) is an *intrinsic* property of the surface at that point. This means that it is determined only by the local variations of the surface and not by how that surface is embedded in a higher dimensional space. One may think of this as the *microbe-eye-view* of the surface. If a microbe sits at a point (x, y, z) of

	$K > 0$	$K = 0$	$K < 0$
$H < 0$	peak	ridge	saddle ridge
$H = 0$	*none*	plane	minimal surface
$H > 0$	pit	valley	saddle valley

Table 4.1: Surface interpretation of Gaussian curvature (K) and mean curvature (H) signs

the surface and views its surroundings, taking measurements of the surface around itself, the results it obtains are completely intrinsic to the surface at that point. If the entire surface were bent in some form (but not stretched or creased to form a discontinuity in the surface), the local readings taken by the microbe would remain stable. Under such a bending of the surface, the mean curvature would change, making it an *extrinsic* surface property.

- The Gaussian and mean curvatures are related to the principal curvatures κ_1, κ_2 as follows:

$$H = \frac{(k_1 + k_2)}{2}; \qquad K = \kappa_1 \kappa_2$$

or

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

- Surfaces may be *typed* by observing the signs of the Gaussian and mean curvatures as shown in table 4.1.

The properties of these curvatures and the ease with which they yield surface typing make them a desirable metric for the computation of features for matching. The invariance of the

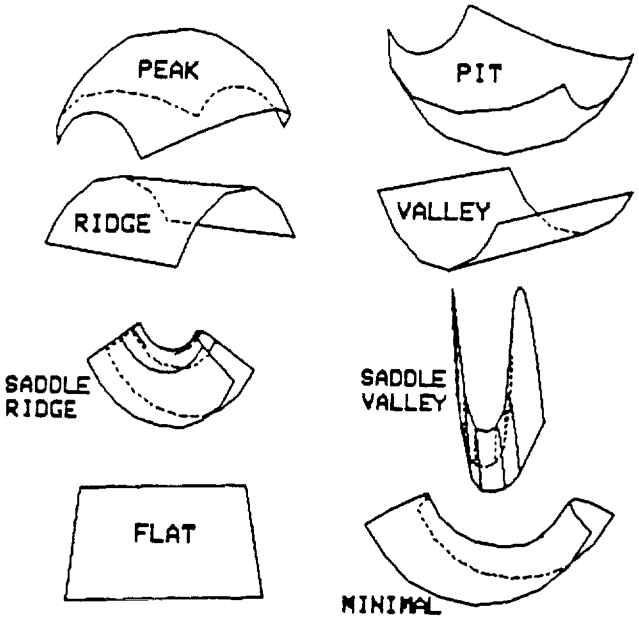


Figure 4.3: The eight surfaces types described by the curvature sign map

curvature sign map to perspective transformation permits one to perform matching of model region adjacency graphs to the regions in the curvature sign map. That these curvatures are local to surface regions make them stable under occlusion. The visible shapes of these features are shown in figure 4.3.

Surface Area

The surface area of the patch **w** can be computed from the first fundamental form. The square root of the determinant of the *metric tensor* ($g = |\mathbf{g}|$) summed over the patch yields the surface area as follows:

$$\sqrt{g} = \sqrt{EG - F^2} = \sqrt{1 + g_u{}^2 + g_v{}^2}$$

$$\text{Surface Area} = \int\int \sqrt{g}\, du\, dv \qquad (4.55)$$

4.3.3 Problems with Digital Computation of Curvature Sign Maps

In practice, however, serious problems exist in the computation of features derived from curvature sign maps. These problems which are discussed in [25] caused Besl to abandon the use of these features *per se* for symbolic feature matching purposes. In his work, he reduced the importance of these features and used them to generate *seed regions* which are then grown in the *variable order segmentation* strategy discussed earlier. These problems are by and large associated with the computation of the curvatures from digital data.

The usual methods for computing surface curvatures in a digital image are the digital kernel-type computation of derivatives employed by Besl in his work and the *facet model* approach of Haralick et. al. [87].

Besl utilized kernel operators comprising the following:

$$f_u = D_u * S * f \qquad f_v = D_v * S * f$$

$$f_{uu} = D_{uu} * S * f \qquad f_{vv} = D_{vv} * S * f$$

$$f_{uv} = D_{uv} * S * f$$

where

$*$ denotes convolution

S is the binomial smoothing window

$D_?$ are the weighted least squares derivative estimation windows

f is the bivariate function describing image

The binomial smoothing window S is given by $S = \vec{s}\,\vec{s}^T$, where

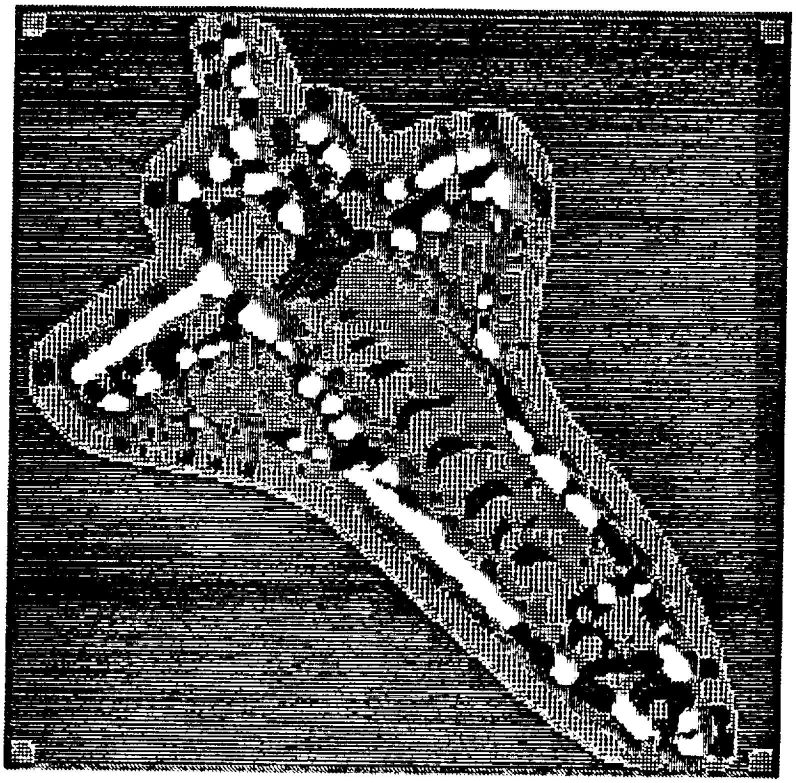
$$\vec{s} = \frac{1}{64}[\ 1 \quad 6 \quad 15 \quad 20 \quad 15 \quad 6 \quad 1\ ]^T$$

The weighted least squares derivative estimation windows are given by:

$$D_u = \vec{d_0}\vec{d_1}^T \qquad D_v = \vec{d_1}\vec{d_0}^T$$

$$D_{uu} = \vec{d_0}\vec{d_2}^T \qquad D_{vv} = \vec{d_2}\vec{d_0}^T$$

$$D_{uv} = \vec{d_1}\vec{d_1}^T$$



N.B. See figure 8.7 for the key to the surface type labels.

Figure 4.4: The curvature sign map computed by the application of kernel-type convolution operators

For a 7 × 7 window (as used by Besl), the column vectors $\vec{d_0}$, $\vec{d_1}$ and $\vec{d_2}$ are given by:

$$\begin{aligned}
\vec{d_0} &= \frac{1}{7}[\ 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1\ ]^T \\
\vec{d_1} &= \frac{1}{28}[\ -2 \quad -3 \quad -1 \quad 0 \quad 1 \quad 2 \quad 3\ ]^T \\
\vec{d_2} &= \frac{1}{84}[\ 5 \quad 0 \quad -3 \quad -4 \quad -3 \quad 0 \quad 5\ ]^T
\end{aligned}$$

An extensive set of such window operators is detailed in [17].

Derivatives generated by convolving these operators with the digital image turn out to be extremely noisy. This is due mainly to digital quantization noise and measurement error. Figure 4.4 shows the Gaussian and mean curvature sign map computed on a 200 × 200 range image of a model of the space shuttle using the kernel convolution operators. Notice how the surface regions are fragmented and bear little resemblance to the surface description expected.

Haralick et. al. [86, 87, 88, 124] computed principal curvature-based features in intensity using the *facet model* approach. A linear surface model such as a bicubic surface is applied to the image. Each $K \times K$ neighbourhood around a point (x, y) in the image is fitted to the facet surface model to obtain the coefficients of the surface fit at that point. If a goodness of fit threshold is exceeded, the point is discarded as the location of surface discontinuity; otherwise, the coefficients are used to compute the curvature values analytically. Haralick et. al. reported reasonable success in computing features in intensity images. The features 'peak, pit, ridge, ravine, saddle, flat and hillside' were computed from the zero-crossover maps of the principal curvatures and these corresponded with the highlighting of edges in the images etc. In the course of this dissertation, the facet model algorithm was replicated and run on range imagery. The resulting feature maps were no better than figure 4.4. Unless the kernels are made so large as to make it encompass more than one contiguous surface most of the time, the digital sampling and measurement noise render the local facet fit meaningless. The reason the facet model (and the kernel-type convolution computations) work better on intensity images is that intensity image sensors such as phosphor-based and CCD cameras are subject to a certain amount of image blooming and cross-pixel diffusion. This implicitly smooths the image sufficiently to make the local fits and digitally computed partial derivatives meaningful. Such smoothing is not as dominant in laser range imagery.

4.3.4 Analytical Curvature Computation from Fitted High Order Surfaces

Until the technology for laser range sensors with sufficient range resolution is available, techniques have to be developed for the computation of surface partial derivatives if these mathematically appealing curvature features are to be used. A solution to the problem of quantization

C-2

noise may be to smooth the images before they are subjected to either kernel convolution or facet model computations. The problem with this approach is that smoothing alters the structure (and curvature) of the image surfaces (even if adaptive filters are used to avoid inadvertent merging of discontinuous surfaces).

A solution developed in the course of this dissertation is to exploit the segmented contiguous surfaces to which high order polynomials have been fitted. These yield a closed form mathematical description of the surfaces. Assuming that a surface order has been selected to represent accurately the surface, surface curvatures can be computed for each point (x, y) in the surface analytically. This process has the following advantages:

- Since the high order surface is fitted to an entire smooth region, the surface description at each point on the surface is based upon a greater amount of data than the points in a local window.

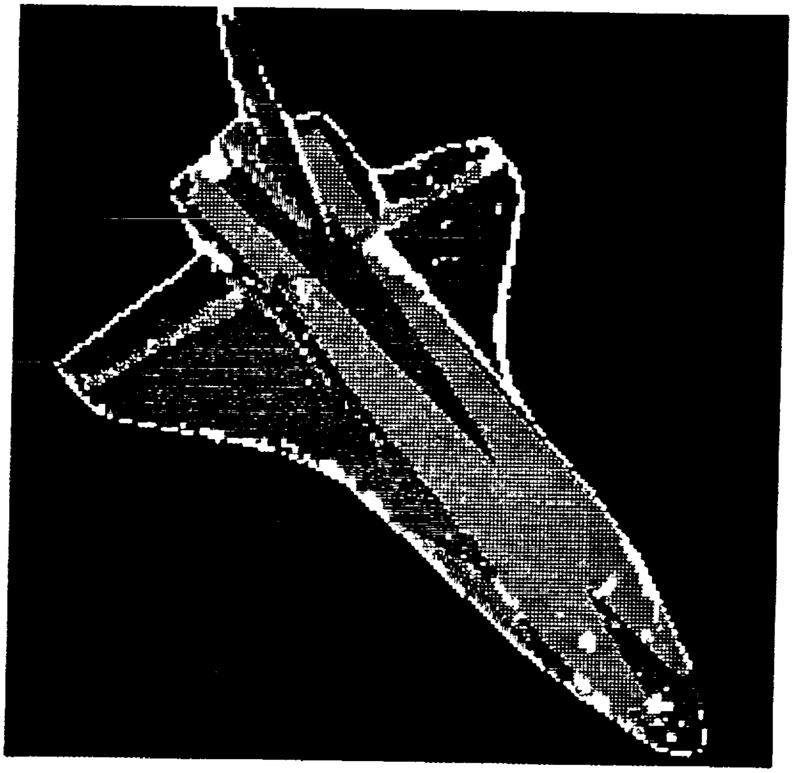
- The curvature computation is valid to the degree that the polynomial surface is faithful to the data.

- Since the contiguous surface is computed by adaptive region growing, it actually *creeps* close to the edge discontinuities without crossing them, permitting curvature computations close to discontinuities in the images.

- The high order surface fit can be thought of as an adaptive filter which smooths the image yielding an analytically continuous surface. It is possible to extrapolate points infinitesimally close to each other in full floating-point glory.

- Assuming that the high order fit is accurate, the smoothing implicit to fitting does not deform the image surfaces significantly.

Figure 4.5 is the Gaussian and mean curvature sign map for the same shuttle image as figure 4.4 computed from the high order surface fits on the contiguous surface regions. Notice



N.B. See figure 8.7 for the key to the surface type labels.

Figure 4.5: The curvature sign map computed analytically from the high order surface fits

how the features computed correspond to intuitive expectations. The wings and tail are planes, valleys fill the gap between the wing and the fuselage and the fuselage and shuttle engines are ridge regions.

4.4 The Properties of Biquadratics applied to Cylinder Estimation

The bi-variate function describing a three-dimensional biquadratic surface is given by:

$$z = a + bx + cy + dxy + ex^2 + fy^2 \tag{4.56}$$

It would be beneficial to discuss the motivation of using this function and to have a intuitive grasp of how this equation can represent cylindrical forms before taking a plunge into derivations and proofs.

First of all, one may make observations concerning the second partial derivatives of the biquadratic form with respect to x and y.

$$\frac{\partial^2 z}{\partial x^2} = e \tag{4.57}$$

$$\frac{\partial^2 z}{\partial y^2} = f \tag{4.58}$$

These equations yield the *partial acceleration vector* [112] which describes the direction of maximal acceleration. This indicates that there would be a constant direction of maximal acceleration that is homogeneous throughout the entire surface. This means that if a biquadratic surface is fitted to a cylindrical surface (either convex or concave), the orientation of the cylinder axis projection onto the *x*-*y* plane can be computed. This projection is perpendicular to the acceleration vector. It will be shown later that the gradient of the cylinder axis is the arctangent of the root of the quotient of *e* and *f*, but the present discussion serves to indicate the promise of biquadratics in fitting cylinders and cones (which also have constant directions of maximal acceleration).

Second, the two-dimensional function

$$0 = a + bx + cy + dxy + ex^2 + fy^2 \tag{4.59}$$

describes the ellipse, parabola, hyperbola (or a limiting form of one of these [115, 166] as will be shown later). These forms correspond to plane sections of a right circular cone, and are therefore known as *conic sections* or *conics* Since for any $z = z'$ in equation 4.56, equation 4.59 can be obtained with $a' = a + z'$, all horizontal cross-sections of biquadratic surfaces are conics.

Finally, a fundamental property of biquadratics is that it is a symmetric function. It will be shown that with an appropriate rotation of coordinate axes, the cross-variable term xy can be eliminated. This holds the promise that all smooth symmetric surfaces can be described to differing degrees of accuracy by biquadratics.

4.4.1 Properties of Biquadratics – A Review.

The bulk of the properties of the three-dimensional biquadratic equation 4.56 can be observed in its limiting two-dimensional form (with $z = 0$) in equation 4.59. The present discussion will therefore begin with a examination of the latter equation. We shall proceed to discuss the three-dimensional form and finally, we shall form an association between the biquadratic function with the surface classes obtained from Gaussian and mean curvature sign maps.

4.4.2 The Limiting Two-Dimensional Function

We shall begin our analysis of equation 4.59 by showing that with an appropriate rotation of the x and y coordinate axes, the xy term can be eliminated, leaving us with a function which is completely separable in x and y. This will aid us in our second exercise in which we shall make some geometric sense of function.

Rotation of coordinate axes

Equation 4.59 can be simplified to yield

$$A + Bu + Cv + Eu^2 + Fv^2 = 0 \qquad (4.60)$$

(with no uv term) by a rotation of axes[115, 166].

Rotating the x-y axes by θ with the formulae

$$\begin{aligned} x &= u\cos\theta - v\sin\theta \\ y &= u\sin\theta + v\cos\theta \end{aligned} \qquad (4.61)$$

equation 4.59 becomes:

$$\begin{aligned} a + u(b\cos\theta + c\sin\theta) + v(c\cos\theta - b\sin\theta) + \\ u^2(e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta) + \\ v^2(e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta) + \\ uv\Big(d(\cos^2\theta - \sin^2\theta) + 2\sin\theta\cos\theta(f - e)\Big) = 0 \end{aligned} \qquad (4.62)$$

By application of trigonometric identities, the coefficient of the uv term of equation 4.62

$$d(\cos^2\theta - \sin^2\theta) + 2\sin\theta\cos\theta(f - e)$$

reduces to

$$d\cos 2\theta - (e - f)\sin 2\theta$$

The coefficient of the uv term becomes zero when

$$\begin{aligned} d\cos 2\theta &= (e - f)\sin 2\theta \\ \tan 2\theta &= \frac{d}{e - f} \end{aligned} \qquad (4.63)$$

Let $\theta' = \tan^{-1}\left(\frac{d}{e-f}\right) \qquad -\frac{\pi}{2} \leq \theta' \leq \frac{\pi}{2}$

$$\begin{aligned} 2\theta &= \ldots\theta' - 2\pi, \theta' - \pi, \theta', \theta' + \pi, \theta' + 2\pi \ldots \\ &= \theta' + n\pi \qquad (n = \ldots - 1, 0, 1, 2, \ldots) \\ \theta &= \frac{\theta' + n\pi}{2} \qquad (n = \ldots - 1, 0, 1, 2, \ldots) \end{aligned} \qquad (4.64)$$

Equation 4.64 is periodic on $\frac{\pi}{2}$. This result is significant, because, as will be evident later, θ defines the principal axes of the conic described by equation 4.59 and its three-dimensional counterpart. What has been shown thus far is that:

For every two-dimensional quadratic equation

$$a + bx + cy + dxy + ex^2 + fy^2 = 0$$

there exists a rotation of the coordinate system with interval $\frac{\pi}{2}$ which will eliminate the cross-variable xy term.

Geometric Descriptions

We shall now make some observations concerning the simplified equation 4.60

$$A + Bu + Cv + Eu^2 + Fv^2 = 0$$

If both E and F are zero, the function degenerates to a planar equation. This degenerate case is of no interest to cylindrical/conical hypothesis generation.

Parabolas

When E is zero, equation 4.60 takes the form:

$$a + bx + cy + fy^2 = 0$$

Performing the translation $x = u + h$, $y = v + k$; we get:

$$(a + bh + ck + fk^2) + bu + v(c + 2fk) + fv^2 = 0 \tag{4.65}$$

When $k = -\frac{c}{2f}$, the v term drops out and equation 4.65 becomes:

$$(a + bh - \frac{c^2}{4f}) + bu + fv^2 = 0 \tag{4.66}$$

We can now remove the constant term with the substitution $h = \frac{1}{b}\left(\frac{c^2}{4f} - a\right)$ and obtain the form:

$$fv^2 + bu = 0 \tag{4.67}$$

If both B and E are zero in equation 4.60, equation 4.66 becomes:

$$(a - \frac{c^2}{4f}) + fv^2 = 0$$

or

$$v^2 = \frac{1}{f}\left(a - \frac{c^2}{4f}\right)$$

If $\frac{1}{f}(a - \frac{c^2}{4f})$ is positive, the function describes two lines parallel to the u-axis; if it is zero, the locus of the function lies along the u-axis; if it is negative, the locus of the function is imaginary.

Similarly when F is zero, equation 4.60 can be simplified by translations of $x = u + h$, $y = v + k$; where

$$h = \frac{b}{2e} \text{ and } k = \frac{1}{c}\left(\frac{b^2}{4e} - a\right)$$

to yield:

$$eu^2 + cv = 0 \tag{4.68}$$

The same analysis applied to B and E being zero also applies to C and F being zero. Both equation 4.67 and equation 4.68 are parabolas. It should be observed that the translations performed shift the coordinate axes to the axes of symmetry of the function and that the coefficients of the second order terms remain the same.

That the parabola is a conic section can be seen in the following analysis[166]. Let OAB be a vertical cross-section of a right-circular cone (figure 4.6). A plane perpendicular to OAB and parallel to OA will intersect the cone producing the section CND. Let the locus of intersection between the two planes be MN as shown in the figure. Let a horizontal plane pass through the cone forming the circular section FPG (where P is a point of intersection between the plane and CND). This plane intersects with planes OAB and CND at point Q. Let a horizontal plane passing through point N intersect the cone producing the circular section of diameter HN. Now, PQ lies on CND and is perpendicular to NM. Assigning a coordinate plane with N as the origin and NM as the x-axis, we have $x = NQ$ and $y = QP$. Since NM is parallel to

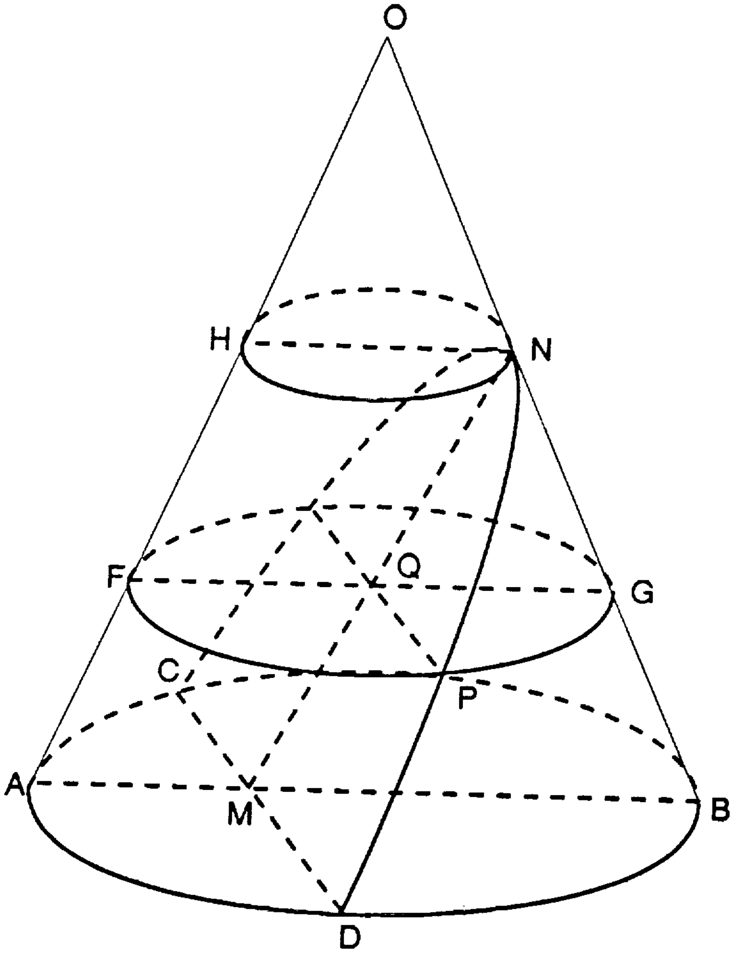


Figure 4.6: The parabola as a conic section

OA, $\frac{QG}{NQ} = \frac{OH}{HN}$ yielding

$$QG = \frac{HN}{OH}x \tag{4.69}$$

Since QP is perpendicular to FG, and FG is the diameter of FPG (see inset of figure 4.6),

$$y^2 = |QP|^2 = FQ \cdot QG \tag{4.70}$$

Since $HNQF$ forms a parallelogram, and $|FQ| = |QP|$, $|HN| = |QP|$. Hence, equation 4.69 and equation 4.70 yield:

$$y^2 = \frac{HN^2}{OH}x \tag{4.71}$$

Since HN and OH are constants for any sectioning plane parallel to OA, equation 4.71 is similar in form to equation 4.67 and equation 4.68.

Ellipses

When E and F in equation 4.60 have the same sign, applying the translations $x = u + h$, $y = v + k$; we get:

$$(a + bh + ck + eh^2 + fk^2) + u(b + 2eh) + v(c + 2fk) + eu^2 + fv^2 = 0 \tag{4.72}$$

When $h = -\frac{b}{2e}$ and $k = -\frac{c}{2f}$, the u and v terms drop out and equation 4.72 becomes:

$$\begin{aligned} (a - \frac{b^2}{4e} - \frac{c^2}{4f}) + eu^2 + fv^2 &= 0 \\ \mathcal{K} + eu^2 + fv^2 &= 0 \end{aligned} \tag{4.73}$$

If $\mathcal{K}$ has the same sign as E and F, the function describes an ellipse; if $\mathcal{K}$ is zero, the function degenerates into a single point and if $\mathcal{K}$ differs in sign from E and F, the locus of the function is imaginary.

That the ellipse is a conic section can be seen in the following analysis[166]. Let OAB be a vertical cross-section of a right-circular cone (figure 4.7). A plane $WXYZ$ perpendicular to OAB at an angle from the horizontal intersecting the cone produces a symmetric section, the axis of which is MN (the locus of intersection). Let C be the midpoint of MN. Passing

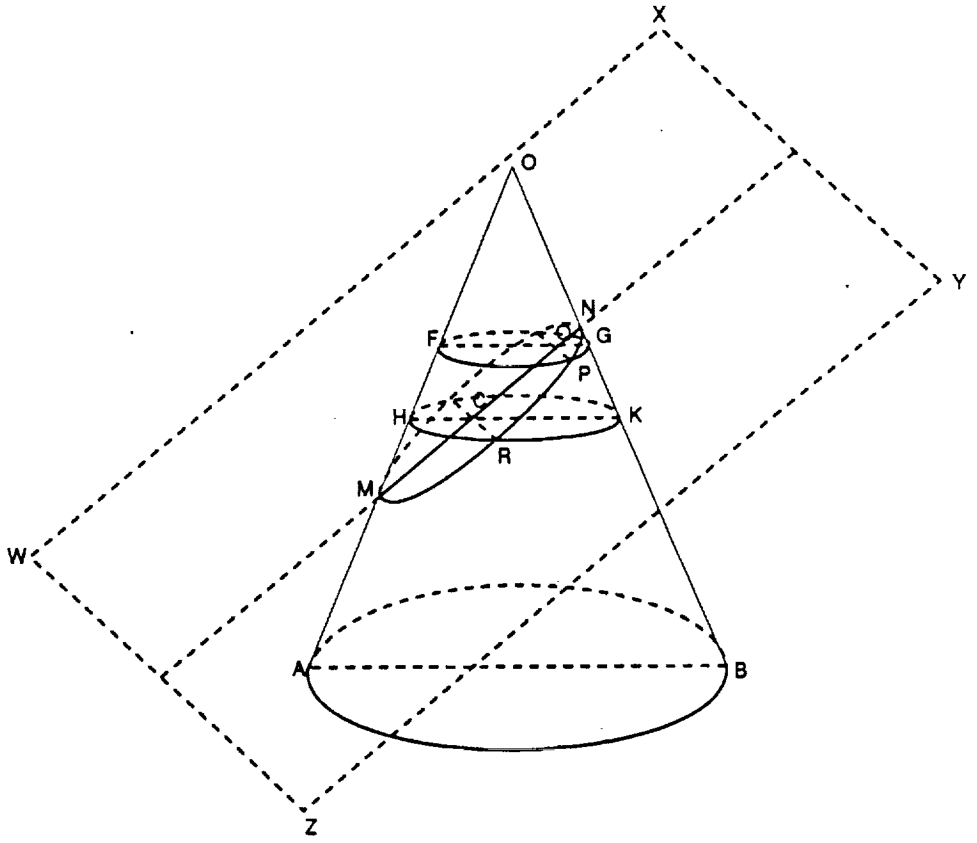


Figure 4.7: The ellipse as a conic section

horizontal planes through C and any other point Q on MN, we get two circular sections of which HK and FG respectively are diameters. The locus of intersection between $WXYZ$ and the horizontal plane through Q forms the line QP which is perpendicular to MN and FG. In circles HRK and FPG, CR and QP are perpendicular to diameters HK and FG respectively. Thus,

$$\begin{aligned} |CR|^2 &= HC \cdot CK \\ |QP|^2 &= FQ \cdot QG \end{aligned} \tag{4.74}$$

Since FQ and HC are parallel, triangles MQF and NCK are similar to triangles MHC and NQG respectively. Thus,

$$\begin{aligned} \frac{FQ}{HC} &= \frac{MQ}{MC} \\ \frac{QG}{CK} &= \frac{QN}{CN} \end{aligned} \tag{4.75}$$

Now, CR and MN are perpendicular in plane $WXYZ$. Taking MN as the x-axis with C as the origin, the coordinates of point P are $x = CQ$ and $y = QP$. Thus equations 4.74 and equations 4.75 yield:

$$\begin{aligned} \frac{|QP|^2}{|CR|^2} &= \frac{FQ \cdot QG}{HC \cdot CK} \\ \frac{y^2}{|CR|^2} &= \frac{MQ}{MC} \cdot \frac{QN}{CN} \end{aligned} \tag{4.76}$$

Since CN and CR are constants for any sectioning plane $WXYZ$, we let $|CN| = |MC| = a$ and $|CR| = b$. Thus $QN = a - x$ and $MQ = a + x$. Equation 4.76 becomes:

$$\frac{y^2}{b^2} = \frac{(a+x)(a-x)}{a^2}$$

yielding:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \tag{4.77}$$

or

$$b^2x^2 + a^2y^2 - a^2b^2 = 0 \tag{4.78}$$

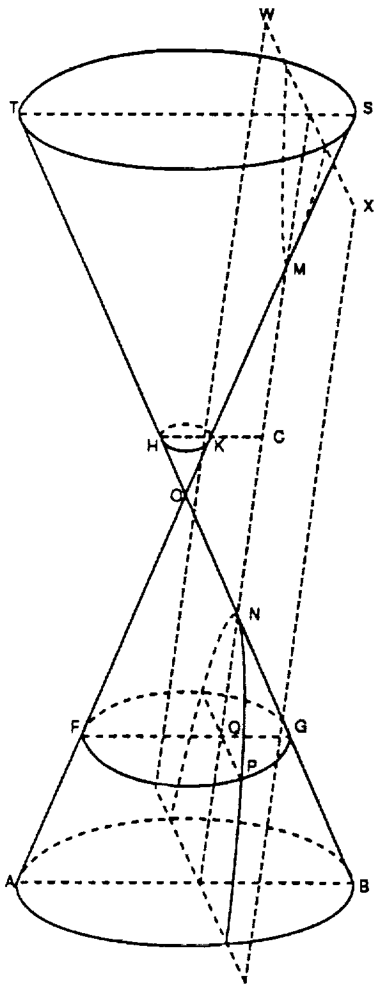


Figure 4.8: The hyperbola as a conic section

Equation 4.78 is isomorphic with equation 4.73 (E and F having the same sign) with a constant translation.

Hyperbolas

When E and F in equation 4.60 are of different signs, equation 4.73

$$\mathcal{K} + eu^2 + fv^2 = 0$$

describes a hyperbola unless $\mathcal{K}$ is zero. If $\mathcal{K}$ is zero, the locus equation 4.73 is a pair of lines intersecting at the $u - v$ origin.

That the hyperbola is a conic section can be seen in the following analysis[166]. Let OAB and OST be the vertical cross-sections of right-circular cones as shown in figure 4.8. A plane

$WXYZ$ perpendicular to OAB (and OST) at an angle from the horizontal intersecting the cones produces a section on each cone. Let the intersection of the plane $WXYZ$ and OS and OB be M and N respectively, and let C be the midpoint of MN. Passing horizontal planes through C and any other point Q on MN extended into the lower section, we get two circular sections of which HK and FG respectively are diameters. The locus of intersection between $WXYZ$ and the horizontal plane through Q forms the line QP which is perpendicular to MN and FG. In FPG, QP is perpendicular to diameter FG. Thus,

$$|QP|^2 = FQ \cdot QG \tag{4.79}$$

Since FQ and KC are parallel, triangles MFQ and NQG are similar to triangles MKC and HCN respectively. Thus,

$$\begin{aligned} \frac{FQ}{KC} &= \frac{MQ}{MC} \\ \frac{QG}{HC} &= \frac{NQ}{CN} \end{aligned} \tag{4.80}$$

Now, CR and MN are perpendicular in plane $WXYZ$. Taking MN as the x-axis with C as the origin, the coordinates of point P are $x = CQ$ and $y = QP$. Thus equations 4.79 and equations 4.80 yield:

$$|y|^2 = KC\left(\frac{MQ}{MC}\right) \cdot HC\left(\frac{NQ}{CN}\right) \tag{4.81}$$

Since CN and KC are constants for any sectioning plane $WXYZ$, we let $|CN| = |MC| = a$ and $|KC| \cdot |HC| = b^2$. Thus $MQ = a + x$ and $NQ = a - x$. Equation 4.81 becomes:

$$y^2 = \frac{b^2}{a^2}(a+x)(a-x)$$

yielding:

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \tag{4.82}$$

or

$$b^2x^2 - a^2y^2 - a^2b^2 = 0 \tag{4.83}$$

Equation 4.83 is isomorphic with equation 4.73 (E and F having different signs) with a constant translation.

The Characteristic of the Two-Dimensional Quadratic Form

In the two-dimensional quadratic form, the quantity

$$d^2 - 4ef \tag{4.84}$$

known as the characteristic of the quadratic form is invariant through rotation and translation of the coordinate axis.

When the function is rotated through angle θ the first three terms of the rotated form are:

$$\begin{aligned} (e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta) \quad u^2 \quad + \\ (e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta) \quad v^2 \quad + \\ \Big(d(\cos^2\theta - \sin^2\theta) + 2\sin\theta\cos\theta(f-e)\Big) \quad uv \end{aligned}$$

Letting the corresponding coefficients of the rotated equation be d', e', f' such that

$$\begin{aligned} e' &= (e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta) \\ f' &= (e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta) \\ d' &= \Big(d(\cos^2\theta - \sin^2\theta) + 2\sin\theta\cos\theta(f-e)\Big) \end{aligned}$$

the characteristic of the rotated form

$$d' - 4e'f'$$

becomes:

$$\begin{aligned} d^2\cos^4\theta + d^2\sin^4\theta - 8ef\sin^2\theta\cos^2\theta - 4ef\sin^2\theta \\ + 4ef\cos^4\theta + 2d^2\sin^2\theta\cos^2\theta \\ = (d^2 - 4ef)(\cos^4\theta + 2\sin^2\theta\cos^2\theta + \sin^4\theta \end{aligned}$$

$d^2 - 4ef < 0$	Ellipse
$d^2 - 4ef = 0$	Parabola
$d^2 - 4ef > 0$	Hyperbola

Table 4.2: Summary of $a + bx + cy + dxy + ex^2 + fy^2 = 0$ forms

$$= (d^2 - 4ef)(\cos^2\theta + 2\sin^2\theta)^2$$

$$= d^2 - 4ef$$

By the application of similar trigonometric manipulation, it can also be shown that with any translation $x = u + h$, $y = v + k$, the characteristic remains constant.

The characteristic of the two-dimensional biquadratic form determines the shape of the curve as shown in table 4.2.

The Limiting Two-Dimensional Form – A Summary

In summary, it has been shown that

- The equation

  $$a + bx + cy + dxy + ex^2 + fy^2 = 0$$

  can be reduced by some rotation θ of the coordinate axes to yield the form

  $$A + Bu + Cv + Eu^2 + Fv^2 = 0$$

- θ is given by:

  $$\theta = \frac{\theta' + n\pi}{2} \qquad (n = \ldots -1, 0, 1, 2, \ldots)$$

  where $\theta' = \tan^{-1}\left(\frac{d}{e-f}\right)$, and is periodic on $\frac{\pi}{2}$.

E and F	Form	Degenerate Forms
Either one zero	Parabola	$E = 0$ & $B = 0$: pair of lines parallel to u-axis
		$F = 0$ & $C = 0$: pair of lines parallel to v-axis
		The solution may be imaginary
Same Sign	Ellipse ($\mathcal{K}^{\dagger}$, E same sign)	($\mathcal{K}$, E different signs) imaginary locus.
		($\mathcal{K} = 0$): a point
Different Signs	Hyperbola	($\mathcal{K} = 0$) : pair of lines intersecting at the u-v origin.

† : Where $\mathcal{K} = A - \frac{B^2}{4E} - \frac{C^2}{4F}$

Table 4.3: Summary of $A + Bu + Cv + Eu^2 + Fv^2 = 0$ forms

- The equation $A + Bu + Cv + Eu^2 + Fv^2 = 0$ (and thus the full quadratic form from which it is derived) always defines a conic section or one of its degenerate forms as shown in table 4.3
- The characteristic $d^2 - 4ef$ of the quadratic form is invariant to rotation and translation of the coordinate axes and describes the shape of the surface as shown in table 4.2.

4.4.3 The Three-Dimensional Biquadratic Function

One may think of the three-dimensional biquadratic function (equation 4.56):

$$z = a + bx + cy + dxy + ex^2 + fy^2$$

for any particular $z = z'$ as the limiting two-dimensional function (equation 4.60) with a translation along the z-axis of $-z'$.

Rotation of Coordinate Axes

By performing the same rotation (equation 4.61) and the same analysis in the previous section, equation 4.56 becomes:

$$\begin{aligned} a + u(b\cos\theta + c\sin\theta) + v(c\cos\theta - b\sin\theta) &+ \\ u^2(e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta) &+ \\ v^2(e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta) &+ \\ uv\Big(d(\cos^2\theta - \sin^2\theta) + 2\sin\theta\cos\theta(f - e)\Big) &= z \end{aligned} \qquad (4.85)$$

As before, the coefficient of the uv term becomes zero when

$$\tan 2\theta = \frac{d}{e - f} \qquad \text{(equation 4.63)}$$

and

$$\theta = \frac{\theta' + n\pi}{2} \qquad (n = \ldots - 1, 0, 1, 2, \ldots)$$

where $\theta' = \tan^{-1}\left(\frac{d}{e-f}\right)$, and is periodic on $\frac{\pi}{2}$ (equation 4.64).

Equation 4.56 therefore reduces by a rotation of axes by angle θ to:

$$\begin{aligned} a + u(b\cos\theta + c\sin\theta) + v(c\cos\theta - b\sin\theta) + \\ u^2(e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta) + v^2(e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta) \\ = z \end{aligned} \qquad (4.86)$$

or

$$A + Bu + Cv + Eu^2 + Fv^2 = z \qquad (4.87)$$

This analysis shows that:

- For every three-dimensional biquadratic equation

  $$a + bx + cy + dxy + ex^2 + fy^2 = z$$

  there exists a rotation of the coordinate system with interval $\frac{\pi}{2}$ which will eliminate the cross-variable xy term.

- Since the cross-variable xy term can be removed without altering the shape of the surface described, a biquadratic surface is necessarily a smooth symmetric surface where the axes of symmetry are perpendicular.

- The axes of symmetry must be straight lines. This is important for the segmentation of smooth surfaces (like cylinders and cones merging into planes or other cylinders and cones) into straight conic units.

Geometric Descriptions

The analysis performed on the limiting case where $z = 0$ applies also for the general biquadratic function for all values $z = z'$. The x-y plane may be shifted by letting $a' = a - z'$

yielding:

$$a' + bx + cy + dxy + ex^2 + fy^2 = 0$$

The rotated limiting two-dimensional quadratic function (equation 4.62) becomes:

$$\begin{aligned} &a' + u(b\cos\theta + c\sin\theta) + v(c\cos\theta - b\sin\theta) + \\ u^2(e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta) &+ v^2(e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta) \\ &= 0 \end{aligned}$$

or

$$A' + Bu + Cv + Eu^2 + Fv^2 = 0$$

where $A' = A + z'$.

In our earlier discussion of the geometric form of the limiting two-dimensional function, we determined that this function always defines a conic section. Furthermore, we found that the shape traced by the locus of the function is determined chiefly by the coefficients of the second degree terms. This means that the summary rendered in table 4.3 applies to the full three-dimensional biquadratic function. Notice that in the table A affects only the value of $\mathcal{K}$ and thus affects only the degenerate cases. In fact, in three dimensions, it is far easier to understand such cases. The limiting two-dimensional function with $z = 0$ reduces to imaginary roots when the surface described by the full three-dimensional form does not intersect the x-y plane. The locus becomes a single point when the three-dimensional form is a ellipsoid, one tangent plane of which is coplanar with the x-y plane. The locus becomes a pair of parallel lines when the three-dimensional form is a linearly swept parabola and where the sweep axis lies on a plane parallel to the x-y plane.

	$sgn(K)$	3-D Form	2-D Form
$d^2 - 4ef < 0$	$+$	peak/pit	ellipse
$d^2 - 4ef = 0$	0	ridge/valley	parabola
$d^2 - 4ef > 0$	$-$	saddle	hyperbola

Table 4.4: Comparison between two-dimensional quadratic interpretation and three-dimensional interpretation of Gaussian curvature (K) signs

4.4.4 Comparison with Curvature evaluation

Having seen the properties of the limiting two-dimensional quadratic form and the three-dimensional biquadratic surface function, we can relate some of these properties to curvature analysis to derive a more intuitive feel of what the functions are doing, and how they relate to each other.

The sign of the Gaussian curvature K for a three-dimensional biquadratic surface is constant throughout the surface. From equation 4.54, we have

$$K = \frac{g_{uu}g_{vv} - g_{uv}^2}{(1 + g_u^2 + g_v^2)^2}$$

Since the first partial derivatives of the function cannot be imaginary, the the denominator of the function is always positive. Hence, the sign of Gaussian curvature depends solely on the the numerator $g_{uu}g_{vv} - g_{uv}^2$. This evaluates to:

$$sgn(4ef - d^2) \tag{4.88}$$

Equation 4.88 is clearly the negation of the characteristic of the limiting two-dimensional form in equation 4.84. Table 4.4 cross-references the curvature map table 4.1 and quadratic characteristic table 4.2.

CHAPTER V

A SPLIT-MERGE STRATEGY FOR STRUCTURAL COMPONENT EXTRACTION FROM SMOOTH SURFACES

Consider the instance of constructed forms which merge smoothly into other forms in such a way that there is no local discontinuity anywhere along the seam. Examples of these are bending pipes elbows, cylinders chamfering into planes, constructed forms meeting in chamfers, etc. In these cases, local image operators like edge detectors will not separate the components, and neither will smooth surface finders like the variable order segmentation algorithm described earlier. There is nothing intrinsic to the data that warrants the segmentation of such merging forms. The information needed to make such separation lies in the observers 'understanding' and expectation of the forms.

Let us assume that a range image has been segmented into smooth surfaces which are described by bivariate polynomials as described previously. Assume also that a hypothesis has been generated by matching the shape descriptors generated from curvature sign maps to object models. This hypothesis says that a particular smooth surface represents two or more structural forms merging smoothly at the seams. The task is to separate these forms and locate them in $\mathbf{R}^3$ space.

The algorithm described here employs a split-merge paradigm. Since it has been concluded that nothing intrinsic in the surface warrants any specific partitioning, this split-merge operation is *hypothesis-guided.* The splitting operation segments the smooth-regions into sub-regions by the application of some criterion which is consistent with the hypothesis of the constructed form which occupies the region. These sub-regions are then merged using *companion linear forms* which either approximate or are identical to the hypothesized constructed forms.

In this chapter, the general split-merge paradigm and algorithm is set forth. The discussion will be in the following order:

- *Hypothesis guided region splitting* where the criterion to be satisfied for region splitting is discussed;
- *Region adjacency graphs* which describes the basic data structures employed in region merging;
- *Hypothesis guided region merging* where the merging paradigm is set forth;
- *Companion linear forms and the merging predicate* where the criterion for merging regions is established; and
- *The hypothesis-guided split-merge algorithm* where the overall algorithm is discussed.

The particular operations for the split-merge extraction of planes, cylinders and cones will be set forth in the next section.

5.1 Hypothesis Guided Region Splitting

Let a smooth region described by a surface function $\mathcal{S}(u, v)$ on a *Monge* basis (u, v) be a representation of one or more smoothly merging constructed surfaces (e.g. cylinders, cones, planes etc.). The purpose of splitting the surface into sub-regions is to obtain a set of sub-regions each member of which is in one and only one constructed surface.

Let $(c_i \in \mathcal{C})$ be the set of N constructed surfaces represented by an overarching surface $\mathcal{S}(u, v)$. $\mathcal{C}$ is by definition a disjoint set such that $\{(c_i \bigcap c_j = \emptyset) \mid \forall i \neq j\}$. Let $(r_j \in \mathcal{R})$ be the

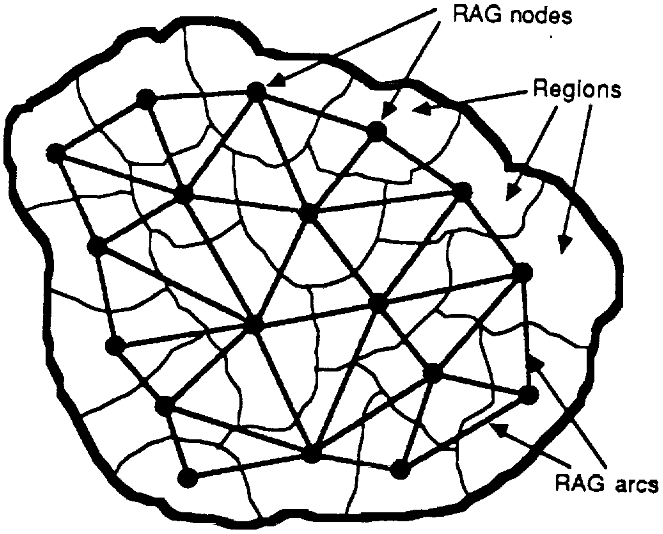


Figure 5.1: A *Region Adjacency Graph* where the nodes represent regions and the undirected arcs represent the adjacency relationship among regions.

set of M sub-regions obtained by the splitting the surface $\mathcal{S}$. Again, $\mathcal{R}$ is a disjoint set such that $\{(r_i \bigcap r_j = \emptyset) \mid \forall i \neq j\}$. It is desired that

$$(r_i \subset c_j) \text{ and } (r_i \subset c_k) \, \textit{iff} \, (j = k) \, \forall (1 \leq i \leq M) \text{ and } (1 \leq j, k \leq N) \tag{5.1}$$

We shall call equation 5.1 the *splitting criterion.*

We have already seen that nothing intrinsic to the data warrants such splitting. To perform such a split, one must operate on a hypothesis of the forms that are present in the image.

It is further argued that all splitting operations are hypothesis based. If one applies the split-merge to extract regions of similar brightness from an intensity image, one would split the image differently than if the intent were to obtain regions of similar brightness gradient. The *abstaction-based* regimen which serves as the underpinning strategy for this work simply makes this explicit.

5.2 Region Adjacency Graphs

The basic structure employed in the region merging process is the *region adjacency graph* which was first introduced by Brice and Fennema[45]. A *region adjacency graph* is an undirected graph description of a set of regions in an image. The nodes of the graph represent the regions and the undirected arcs between nodes represent adjacency relationships of the nodes.

The adjacency relationship is usually determined in terms of 4-connected or 8-connected neighbourhoods. Thus regions r_i and r_j in the graph (here we use the notation for regions and nodes interchangeably to simplify discussion) are adjacent if and only if some subset of $\mathcal{P}_i$ is 4-connected or 8-connected with $\mathcal{P}_j$ (where $\mathcal{P}_i$ and $\mathcal{P}_j$ are the sets of pixels constituting r_i and r_j respectively). We shall denote this such a relationship r_i *adj* r_j.

A representation of such a graph is shown in figure 5.1.

5.3 Hypothesis Guided Region Merging

Let $(r_j \in \mathcal{R})$ be the set of M sub-regions obtained by a splitting operation which satisfies the *splitting criterion* of equation 5.1:

$$(r_i \subset c_j) \text{ and } (r_i \subset c_k) \textit{ iff } (j = k)\ \forall (1 \leq i \leq M) \text{ and } (1 \leq j, k \leq N)$$

The purpose of region merging is to recover the desired $\{c_i \in \mathcal{C} \mid (1 \leq i \leq N)\}$ (N being the number of desired surfaces constituted of $\mathcal{R}$). In the context of this work, $\mathcal{C}$ is the set of constructed surfaces within a smooth surface represented by some high order surface function.

Define *form-set* $\{\mathcal{O}_i \in \ \mid (1 \leq i \leq K)\}$ as the set of K forms or classifications into which the splitted regions may be grouped. Define the *merging predicate* $\mathcal{M}_i$ which determines if a sub-set of adjacent splitted regions $[r_j]$ belongs to the **same instance** of class $\mathcal{O}_i$, i.e.,

$$\mathcal{M}_i([r_j]) \rightarrow ([r_j] \subseteq \mathcal{F}_k) \tag{5.2}$$

where $\mathcal{F}_k$ is an instance of class $\mathcal{O}_i$. The requirement of adjacency may be relaxed to allow the reconstruction of surfaces which are fragmented by occlusion, but such a relaxation is usually employed to merge the larger regions which result from the coalescing of all adjacent splitted regions.

It is obvious that the selection of the merging predicate $\mathcal{M}_i$ in any coalescing operation to uncover constructed surfaces has to be hypothesis guided. Later, we shall consider a *form set* which comprises planes and cylinders. For now, we shall consider the general notion of *companion linear forms* as operators in the merging predicate.

5.4 Companion Linear Forms and the Merging Predicate

A *companion linear form* to a constructed surface form is one which either approximates or is identical to the constructed form and is linear in terms of its orienting parameters. In other words, if a set of pixels $\mathcal{P}$ in an image is due to constructed form $(c_i \in \mathcal{C})$ (in accordance to notation introduced in section 5.1), and if $(\lambda_i \in \Lambda)$ is the *companion linear form* to c_i, then $\mathcal{P}$ fits to form λ_i within some criterion based upon the *goodness-of-fit*. The motivation for *companion linear forms* is that they are easier to fit to the data than the original forms which may be non-linear with respect to their orienting parameters.

Companion linear forms are applied in the merging process as an operator in the merging predicate $\mathcal{M}_k$ to determine if the regions in some set of adjacent splitted sub-regions $[r_j]$ should be merged.

Since the decision whether to merge the regions is based upon how well they fit the companion linear form, the *goodness-of-fit* measure is critical to the process. In the section 4.1.3 discussion of parameter estimation, it was shown that if a Gaussian noise distribution is assumed for the data, the χ^2 measure:

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{z_i - f(x_i, y_i)}{\sigma_i} \right)^2$$

(equation 4.7) is a good measure of how well a bivariate function $f(x, y)$ fits a set of N data points. Thus the χ^2 measure may be used as a merging predicate operator. When the data points $\bigcup_{i=1}^{M} \mathcal{P}_i$ of the set of sub-regions $[r_i]$ (where $\mathcal{P}_i$ is the set of points in sub-region r_i) are fitted by the *companion linear form* to the hypothesized constructed form within some χ^2 tolerance, the sub-regions may be said to be compatible under the imposed hypothesis. Since the area of the region being fitted varies, the χ^2 value should be normalized by the area of the region. The merging predicate operator may be written as:

$$\left(\frac{1}{N}\mathbf{X}(\bigcup_{i=1}^{M}\mathcal{P}_i, f, \mathbf{a}) < \epsilon_{\chi-sq}\right) \rightarrow ([r_i] \subseteq \mathcal{F}_k) \tag{5.3}$$

where M is the number of regions being merged, $\mathcal{F}_k$ is a surface of the constructed form corresponding to the *companion linear form* described by the bivariate function f, $\mathbf{a}$ is the set of parameters of f obtained by fitting the f to $\bigcup_{i=1}^{M}\mathcal{P}_i$, N is the total number of pixels in regions being merged, $\epsilon_{\chi-sq}$ is the χ^2 error bound, and $\mathbf{X}$ is the function for computing the χ^2 given by equation 4.7.

Equation 5.3 leaves something to be desired in the region merging paradigm. Let some set of sub-regions $\Re$ be determined to belong to the same constructed surface. The merging algorithm now considers merging some sub-region r_k to $\Re$. The fitting of $(\Re \cup \{r_k\})$ to the *companion linear form* yields the function $f(x, y)$, and equation 5.3 is satisfied in that the χ^2 value for the new region falls within the tolerance bound. The problem is that r_k is much smaller than the regions encompassed by $\Re$ and the χ^2 expression (equation 4.7) is heavily weighted by $\Re$. As sufficient erroneous sub-regions are incorporated, the χ^2 tolerance bound will eventually be exceeded. We are therefore faced with the *slippery slope* problem of determining at what point the error began. It is impossible to set $\epsilon_{\chi-sq}$ to be so sensitive as to reject such spurious region recruitment without making the system unstable.

Clearly equation 5.3 has to be augmented to solve the problem. This may be done by obtaining the χ^2 errors (after $\mathbf{a}$ is obtained by fitting of $(\Re \cup \{r_k\})$) of the fit of f to each of $(\Re \cup \{r_k\})$, $\Re$, and r_k. If the χ^2 error bound is exceeded for any one of the three, the merge is abandoned. Thus equation 5.3 may be modified to yield the merging predicate given by:

$$\mathit{CONNECTED}([r_i])$$

$$\begin{aligned} &\bigwedge \left(\frac{1}{N} \mathbf{X}(\bigcup_{i=1}^{M} \mathcal{P}_i, f, \mathbf{a}) < \epsilon_{\chi-sq} \right) \\ &\bigwedge \left\{ \left(\frac{1}{area(\mathcal{P}_i)} \mathbf{X}(\mathcal{P}_i, f, \mathbf{a}) < \epsilon_{\chi-sq} \right) \ \mid \forall \, (1 \leq i \leq M) \right\} \\ &\rightarrow ([r_i] \subseteq \mathcal{F}_k) \end{aligned} \tag{5.4}$$

This is the *merging predicate* applied to the merging algorithm.

Notice that the parameters of the *companion linear form* are estimated directly from the original data. This keeps the algorithm *stimulus bound* in the merging operation while it operates upon the higher level hypothesis. Another comment to be made about *merging predicate* is that the use of the χ^2 measure as a threshold in the merging algorithm is not as dependent upon the Gaussian assumption as is the fitting algorithm because the χ^2 measure is used only as a *comparison* of fitting accuracies for purposes of hypothesis verification. All this process needs to claim is that a poorer χ^2 value signifies a poorer fit for a particular signal to noise ratio in the data. It will be shown later how *companion linear forms* are capable of parameter estimation of the constructed forms as well.

5.5 The Hypothesis-Guided Split-Merge Algorithm

Assume that the hypothesis generation process has produced a hypothesis (by matching models to the Gaussian-mean curvature sign map and the smooth surface segmentation with high order polynomials) that N constructed surfaces $\{c_1 \cdots c_N\}$ are present in an image, and that these constructed surfaces are embedded in K smooth surfaces $\{s_1 \cdots s_K\}$. Assume, too, that the hypothesis yields an approximate mapping from $\{c_1 \cdots c_N\}$ to $\{s_1 \cdots s_K\}$ (i.e. the subset of smooth surfaces which encompasses each particular constructed surface is contained in the hypothesis). Normally the subset of smooth surfaces which encompasses a particular constructed surface has only one member since constructed surfaces are usually wholly contained in one smooth surface, but this is not always the case. Figure 5.2 is an example of such a mapping hypothesis. While cylinders may be modeled with one high order polynomial the torus in the

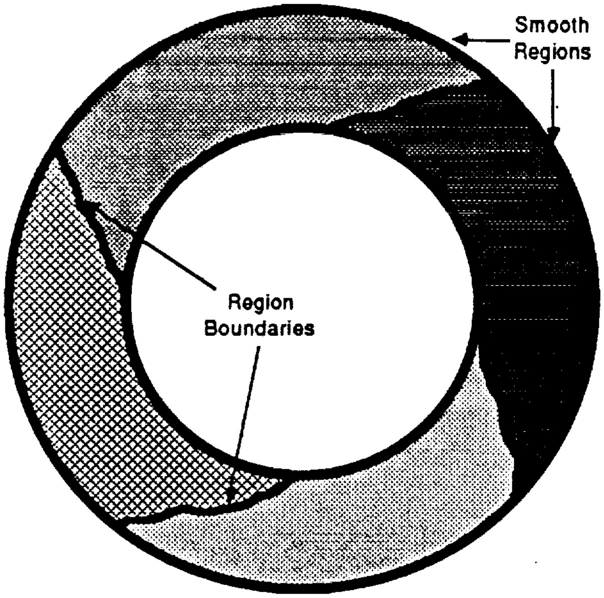


Figure 5.2: The entire smooth surface of a torus is split into four smooth polynomial surfaces. The region boundaries cannot be predicted.

figure encompasses four such surfaces.

The algorithm is as follows:

1. Pick a hypothesized constructed surface c_i which has not yet been processed through the algorithm and let the hypothesized set of smooth surfaces which is encompassed by c_i be $\mathcal{S}_i$.

2. Split each member of $\mathcal{S}_i$ into sub-regions $\{r_j \in \mathcal{R}_i \mid 1 \leq j \leq M\}$ in accordance to the *splitting criterion* in equation 5.1 (The splitting operator is determined by the type of surface of the hypothesized constructed surface c_i).

3. Each subregion r_j is assigned a *split-parent pointer* to the original smooth surface out of which it is split and each smooth surface is assigned a set of *split-children pointers* to the set of sub-regions generated. The pointers themselves are grouped by labels indicating the splitting operation employed.

4. Organize $\mathcal{R}_i$ into a *region adjacency graph* (RAG).

5. Let the bivariate function f describe the *companion linear form* of c_i.

(a) Pick a *seed node* r_s in the RAG which is most likely inside c_i. (This is done by estimating the center of the hypothesized surface and finding the member of $\mathcal{R}$ which contains it or whose centroid is closest to it.)

(b) Fit function f to the set of data points $\mathcal{P}_s$ in r_s and obtain the set of parameters to f, **a**.

(c) Compute the normalized χ^2 error of the fit (equation 5.3). If this error exceeds the χ^2 error bound $\epsilon_{\chi-sq}$, remove r_s from the RAG† and goto step 5a.

(d) If there are no adjacent nodes of the RAG to r_s, assign c_i the region occupancy of r_s and goto step 6.

(e) Pick a sub-region r_k which is adjacent to r_s and fit function f to the set of data points $\mathcal{P}_s$ in $\{r_s, r_k\}$ and obtain the set of parameters to f, **a**.

(f) Apply the *merging predicate* in equation 5.4 to the set $\{r_s, r_k\}$. If equation 5.4 resolves to *TRUE*, fuse† r_s and r_k in the RAG, set $r_s := \{r_s, r_k\}$, and goto step 5e; else, remove r_k from the RAG and goto step 5e.

6. Reduce the region occupancy of each member of S_i to reflect the removal of r_s and goto step 1.

† : To fuse nodes $[r_i]$ in a *region adjacency graph*,

1. Create a new node r_k
2. Set the region occupancy to the total occupancy of the merged nodes.
3. For each node in $[r_i]$
   - Remove all adjacency arcs which terminate at nodes inside $[r_i]$.
   - For each adjacency arc leading to some node r_m outside $[r_i]$, redirect the arc to link r_m to r_k.

- Include the node in the *constituency list* of r_k. (The new node will therefore have a pointer to the original smooth surface regions with which it intersects).

‡ : To remove a node r_k from the RAG, remove all adjacency arcs joining r_k with other nodes in the RAG. Note that this may split the RAG into two RAGs, but the seed region r_s would maintain its adjacency arcs to the RAG to which it is connected.

5.6 Specifics on the Split-Merge Extraction of Planes and Cylinders

Now that the general framework for the split-merge operation has been set forth, we shall proceed to the specifics of the splitting operators and *companion linear forms* for the constructed forms of choice in this work: planes and cylinders.

In overview, the split-merge operation for planes and cylinders is accomplished using the following concepts:

Planes : Three concepts are applied to the extraction of planar regions from a smooth surface:

- The surface normals of a plane is constant throughout the plane.
- Planes occur at *FLAT* regions in the Gaussian-mean curvature sign map (where (abs(H) $< \epsilon_H$) and (abs(K) $< \epsilon_K$)).
- Planar surfaces are already linear in form and so, the companion linear form is identical to the constructed form.

Cylinders : Three concepts may be applied in the extraction of regions which represent cylindrical/conical forms from a smooth surface:

- The traces of maximal surface acceleration or curvature run along the circular cross-section (perpendicular to the axis of rotation of sweep) of cylindrical (or conical) surfaces.

- Regions which represent cylinders (or cones in general) contain *RIDGE* regions in the Gaussian-mean curvature sign map (where $(H < -\epsilon_H)$ and $(\text{abs}(K) < \epsilon_K)$).
- Biquadratic surfaces are symmetric around the principal axes of the conic section generated by a horizontal sectioning plane.

5.7 Specifics of Hypothesis Guided Smooth Surface Splitting for Planes and Cylinders

In this section, the operators necessary to perform the hypothesis guided splitting of smooth surface regions in the algorithm outlined in section 5.5 are discussed. Recall that the hypothesis says that a particular constructed surface c_i is embedded in the set of smooth surfaces $\mathcal{S}_i$. Each smooth surface in $\mathcal{S}_i$ is described by its image region and a high order bivariate polynomial function.

The goal is to split each member of the smooth surface $\mathcal{S}_i$ into the set of sub-regions $\{r_j \in \mathcal{R}_i \mid 1 \leq j \leq M\}$ in accordance to the *splitting criterion* in equation 5.1. We shall now describe the splitting operators which apply to hypothesized plane and cylinder surfaces.

5.8 Normal Vector Segmentation

In the case of planes, the split can be based upon the Gaussian and mean curvature sign maps. One needs to be careful in this. While all planes will map to $(\text{abs}(H) < \epsilon_H)$ and $(\text{abs}(K) < \epsilon_K)$ regions in the curvature sign map, not all such regions represent true planes. Consider, for example, regions at the edge of cylinders where the scanning angle is oblique (see figure 5.3). Quantization and measurement noise in the range sensing and the surface approximation by high order polynomials make such regions appear planar. It is therefore necessary to have a hypothesis that a plane actually exists before labeling the region as a constructed plane.

Consider a situation where a planar region chamfers smoothly into a cylindrical region as shown in figure 5.4a. The Gaussian-mean curvature sign map would label sub-regions within

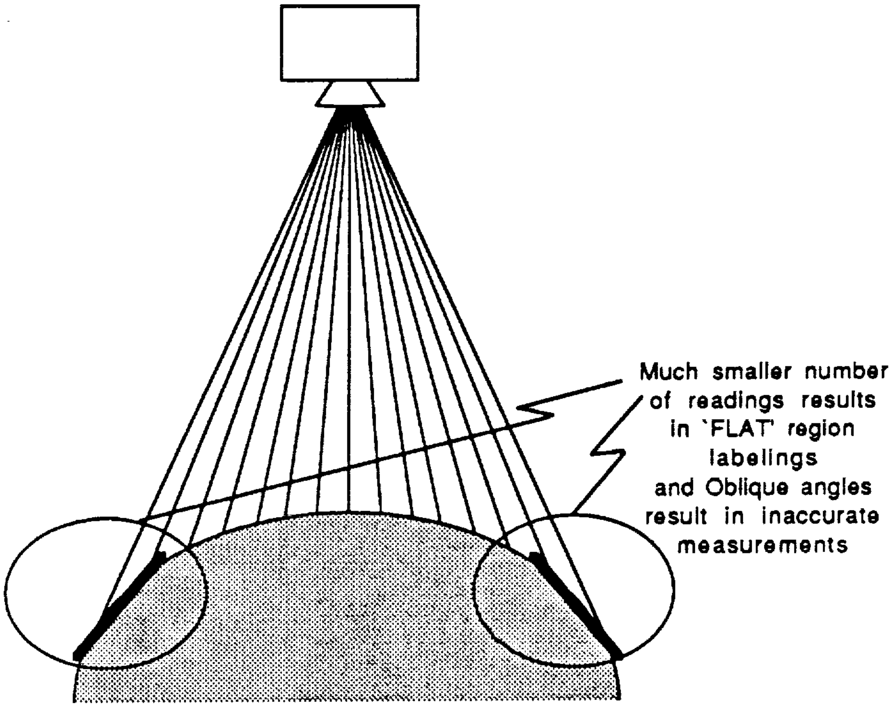


Figure 5.3: The cross-section of a cylindrical surface showing that at the oblique cylinder surfaces, the surface appears planar.

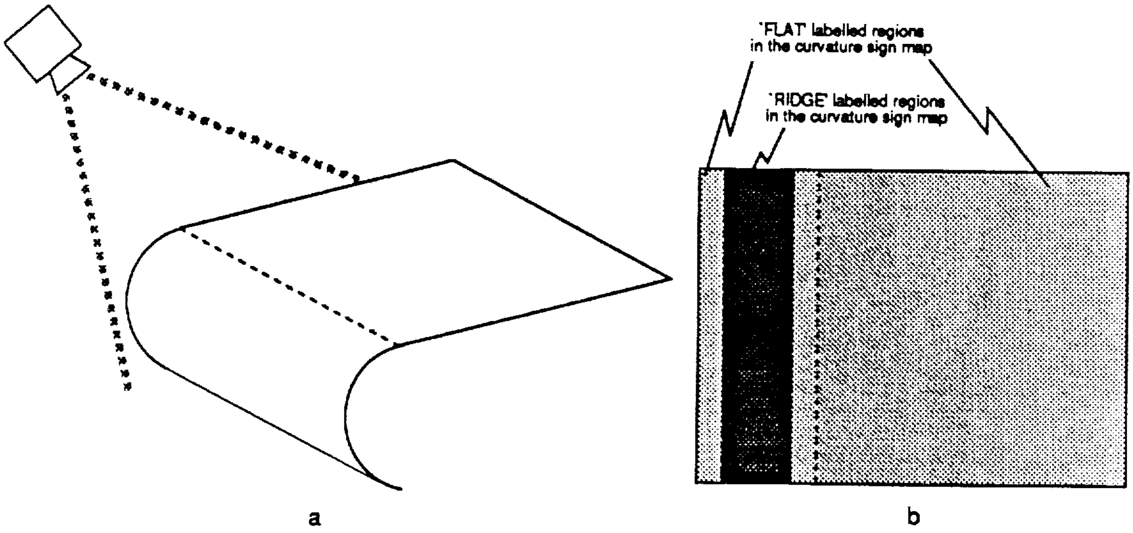


Figure 5.4: a: A plane chamfering into a cylinder. b: The Gaussian-mean curvature sign map of the smooth region.

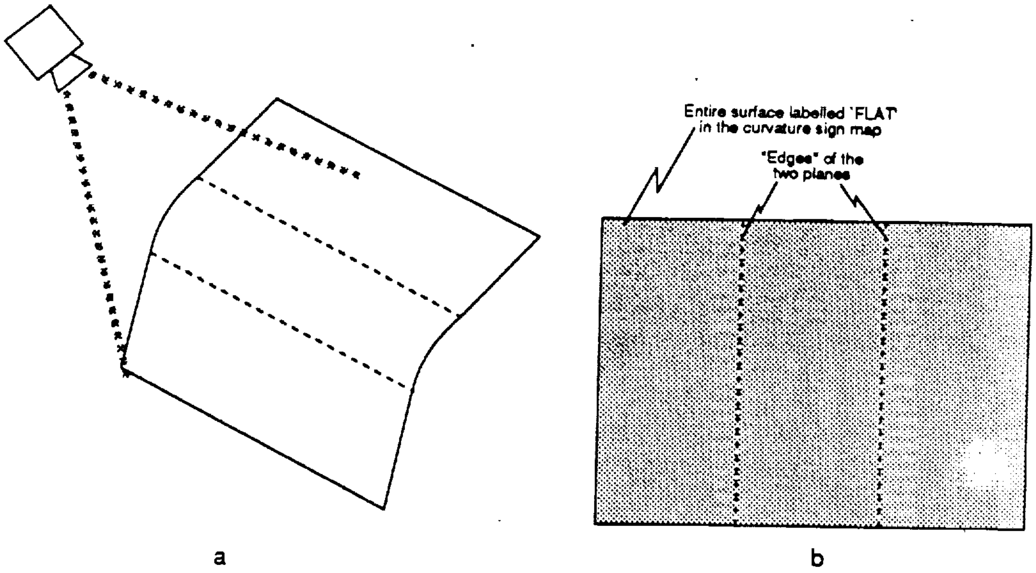


Figure 5.5: a: Two smoothly merging planes. b: The Gaussian-mean curvature sign map of the smooth region.

the parent smooth region variously as *FLAT* and *RIDGE* as shown in figure 5.4b. Clearly, the curvature sign map which participates in generating the initial hypothesis is incapable of determining the edge of partition between the planar and cylindrical regions. The same problem exists in the partitioning of two smoothly merging planes in figure 5.5. The entire smooth region resolves to one *FLAT*-labelled region in the Gaussian-mean curvature sign map because the bend is 'soft' enough that no *VALLEY* regions show (i.e. the mean curvature H does not go below $-\epsilon_H$ for all points in the region).

The key to the extraction of the planar surfaces in figure 5.4 and figure 5.5 lies in an examination of the inclination or surface normal of the regions and not in the rate of change of the inclination or surface normals (which yields the curvature).

If we had a scalar metric which measures the likelihood that pixels within a region belong on the same plane, we could split the parent region into sub-regions in such a way as to satisfy the *splitting criterion* of equation 5.1. Introduce the idea of *like-normal neigbourhoods*. Suppose we computed the unit normals across the entire smooth region (analytically by computing the partial derivatives of the region) which is given by:

$$\begin{aligned} \mathbf{N} &= \frac{-g_u\vec{u} - g_v\vec{v} + \vec{w}}{\sqrt{{g_u}^2 + {g_v}^2 + 1}} \\ &= \frac{-g_u\vec{u} - g_v\vec{v} + \vec{w}}{|n|} \\ &= n^u\vec{u} + n^v\vec{v} + n^w\vec{w} \qquad (5.5) \end{aligned}$$

where $|n| = \sqrt{{g_u}^2 + {g_v}^2 + 1}$ is the magnitude of the surface normal.

Since ($\mathbf{N} = 1$), ($\text{abs}(n^u) \leq 1.0$), ($\text{abs}(n^v) \leq 1.0$), and ($\text{abs}(n^w) \leq 1.0$). Hence, define the *like-normal neighbourhood map* $\mathcal{N}$ as one whose elements are:

$$\begin{aligned} \mathcal{N}(u,v) &= (K(1.0 + n^u(u,v))mod8) + \\ & \quad 8\,(K(1.0 + n^v(u,v))mod8) + \\ & \quad 64\,(Kn^w(u,v)mod4) \qquad (5.6) \end{aligned}$$

where *mod* is the modulus operator and K is the number of values into which to divide the circle in each dimension. For example, if ($K = 128$), the range of variation of the $\vec{u}$ and $\vec{v}$ components is quantized into 256 values and the range of the $\vec{w}$ component is quantized into 128 values. The n^w component is scaled at half the n^u and n^v components because the original surface is a *Monge* patch and the $\vec{w}$ component cannot become negative.

This yields a scalar map with values ranging from 0 to 256 in which each value corresponds to a solid angle of approximately $\frac{4\pi}{K^3}$ steradian wrapped around at intervals of approximately $\frac{4\pi}{8\text{x}8\text{x}4}$ steradian. The *like-normal neighbourhood map* is then segmented into connected components of similar values. Each component, then comprises points which vary in surface normal by no more than $\frac{8\pi}{K^3}$ steradian. The wrap around does not affect the connected component analysis since no two regions with normals $\frac{4\pi}{8\text{x}8\text{x}4}$ steradian apart can be adjacent without triggering segmentation in the Gaussian-mean curvature sign map. The sub-regions obtained from the *like-normal neighbourhood map* therefore conform to the *splitting criterion* of equation 5.1.

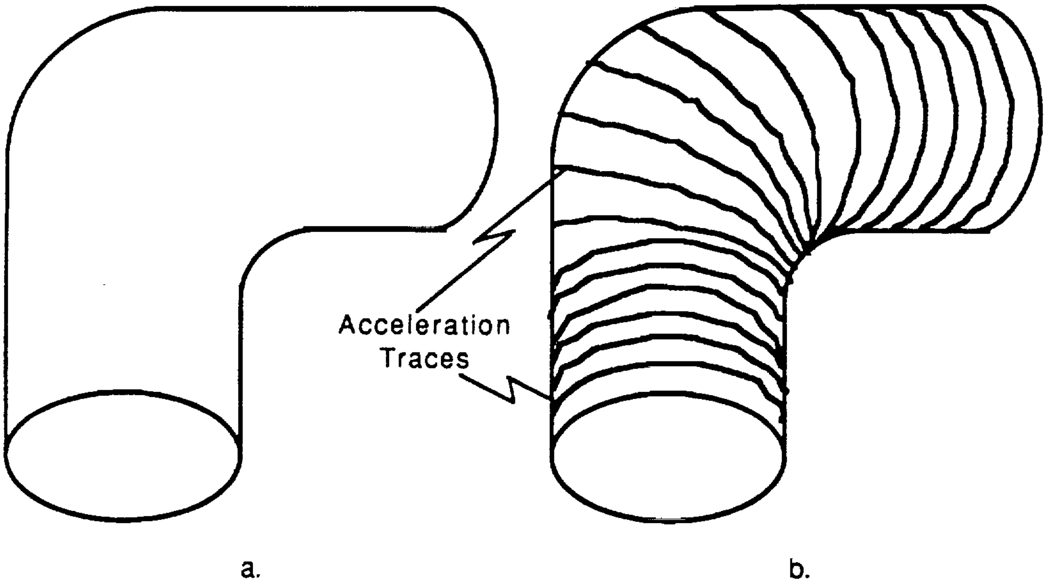


Figure 5.6: a: L-pipe junction which comprises two cylinders merging seamlessly; b: Maximal acceleration traces along the surface of an L-pipe junction

5.9 Acceleration Band Separation

In the case of cylinders (or cones), it is observed that the maximal acceleration or curvature vector at each point on that surface will be in the plane of the circular cross-section of the cylinder or cone at that point. If one were to trace the loci of such vectors at some specified intervals along the edge of the region, these loci will 'slice' the region into bands.

To make the discussion more concrete, we shall consider the segmentation of a L-pipe junction shown in figure 5.6a which comprises two cylinders merging seamlessly (chamfering smoothly at the intersection). The entire junction is represented as a single smooth surface in the initial segmentation. Figure 5.6b shows the desired result of the operation to split this surface into sub-regions which satisfy the *splitting criterion* of equation 5.1. We shall call the segmented sub-region *acceleration bands*.

Let the unit maximal acceleration or curvature vector at some point (u, v) on a *Monge* surface be $\mathrm{a}(u, v) = \alpha\vec{u} + \beta\vec{v}$. The algorithm to obtain the *acceleration bands* is as follows:

1. Extract the boundary of the smooth surface.

2. Obtain the set of boundary interval points $\mathcal{B}$ at regular intervals along boundary.

3. For each point $\left((\vec{b}_i = b_{iu}\vec{u} + b_{iv}\vec{v}) \in \mathcal{B}\right)$,

   (a) Let $(\vec{\gamma} = \vec{b})$; the trace unit t be a point (floor(γ_u, γ_v)) (denoted floor$(\vec{\gamma})$); the trace set $\mathcal{T}_i$ associated with the i^{th} member of B be a set with t as its sole member, and counter $k = 1$

   (b) Set $(\vec{\gamma} := \vec{\gamma} + 0.5\vec{a}(\gamma_u, \gamma_v))$ and increment k by 1.

   (c) If (floor$(\vec{\gamma}) \neq t$), set $t :=$ floor$(\vec{\gamma})$ and add t to $\mathcal{T}_i$

   (d) If t intersects the region boundary (and $\left(t \neq \text{floor}(\vec{b}_i)\right)$), or k is greater than some preset threshold, goto 3 else goto 3b.

4. Segment the surface $\mathcal{S}$ into bands separated by the set of acceleration traces $[\mathcal{T}_i]$.

The computation of the maximal acceleration of curvature vector can be accomplished in three ways:

Principal Directions : We have already observed that the principal curvatures of a surface are the extrema of the normal curvatures of that surface. The corresponding principal directions can therefore be applied to the generation of maximal acceleration vectors. Recall that the principal directions are given by

$$\phi_{1,2} = \tan^{-1}\left(\frac{-B \pm \sqrt{B^2 - 4AC}}{2A}\right)$$

where $A = (FN - GM)$; $B = (EN - GL)$ $C = (EM - FL)$ (equation 4.48) From equation 4.33 and equation 4.38, we can express E, F, G, L, M, N for surface **w** in terms of the derivatives of $g(u, v)$ as follows:

$$E = \mathbf{w}_u \cdot \mathbf{w}_u = 1 + {g_u}^2 \qquad F = \mathbf{w}_u \cdot \mathbf{w}_v = 1 + g_u g_v$$

$$G = \mathbf{w}_v \cdot \mathbf{w}_v = 1 + {g_v}^2$$

$$\begin{aligned}
\mathbf{N} &= \frac{-g_u\vec{u} - g_v\vec{v} + \vec{w}}{\sqrt{{g_u}^2 + {g_v}^2 + 1}} \\
L = \mathbf{w}_{uu} \cdot \mathbf{N} &= \frac{g_{uu}}{|n|} \qquad M = \mathbf{w}_{uv} \cdot \mathbf{N} = \frac{g_{uv}}{|n|} \\
N = \mathbf{w}_{vv} \cdot \mathbf{N} &= \frac{g_{vv}}{|n|}
\end{aligned} \tag{5.7}$$

where $|n| = \sqrt{{g_u}^2 + {g_v}^2 + 1}$ is the magnitude of the surface normal. By algebraic manipulation, equation 4.48 becomes:

$$\phi_{1,2} = \tan^{-1}\left(\frac{-\beta \pm \sqrt{\beta^2 - 4\alpha\gamma}}{2\alpha}\right) \tag{5.8}$$

where

$$\begin{aligned}
\alpha &= |n|(FN - GM) = g_{vv}(1 + g_u g_v) - g_{uv}(1 + {g_v}^2) \\
\beta &= |n|(EN - GL) = g_{vv}(1 + {g_u}^2) - g_{uu}(1 + {g_v}^2) \\
\gamma &= |n|(EM - FL) = g_{uv}(1 + {g_u}^2) - g_{uu}(1 + g_u g_v)
\end{aligned}$$

For some angle ψ, the normal curvature of a surface in the direction of that angle can be computed by applying equation 4.44:

$$\kappa_n = \frac{L\,du^2 + 2M\,du\,dv + N\,dv^2}{E\,du^2 + 2F\,du\,dv + G\,dv^2}$$

where $du = \cos(\psi)$ and $dv = \sin(\psi)$. The principal curvatures corresponding to the principal directions can thus be computed and the direction corresponding to the maximal curvature obtained.

Derivatives of Surface Normal : Another way to look at the direction of maximal surface acceleration or curvature in terms of the variation in the unit surface normal.

$$\begin{aligned}
\mathbf{N} &= \frac{-g_u\vec{u} - g_v\vec{v} + \vec{w}}{\sqrt{{g_u}^2 + {g_v}^2 + 1}} \\
&= \frac{-g_u\vec{u} - g_v\vec{v} + \vec{w}}{|n|}
\end{aligned}$$

$$
\begin{aligned}
\frac{d\mathbf{N}}{du} &= \frac{1}{|n|^3}\left[g_u(g_u g_{uu} + g_v g_{uv}) - g_{uu}|n|^2\right]\vec{u} + \\
&\quad \frac{1}{|n|^3}\left[g_v(g_u g_{uu} + g_v g_{uv}) - g_{uv}|n|^2\right]\vec{v} + \\
&\quad \frac{g_u g_{uu} + g_v g_{uv}}{|n|}\vec{w} \\
\frac{d\mathbf{N}}{dv} &= \frac{1}{|n|^3}\left[g_u(g_v g_{vv} + g_u g_{uv}) - g_{uv}|n|^2\right]\vec{u} + \\
&\quad \frac{1}{|n|^3}\left[g_v(g_v g_{vv} + g_u g_{uv}) - g_{vv}|n|^2\right]\vec{v} + \\
&\quad \frac{g_v g_{vv} + g_u g_{uv}}{|n|}\vec{w} \\
\frac{d\mathbf{N}}{du} + \frac{d\mathbf{N}}{dv} &= \frac{1}{|n|^3}\left[g_u\left(g_u g_{uu} + g_v g_{uv} + g_v g_{vv} + g_u g_{uv}\right) - |n|^2\left(g_{uu} + g_{uv}\right)\right]\vec{u} + \\
&\quad \frac{1}{|n|^3}\left[g_v\left(g_u g_{uu} + g_v g_{uv} + g_v g_{vv} + g_u g_{uv}\right) - |n|^2\left(g_{vv} + g_{uv}\right)\right]\vec{v} + \\
&\quad \frac{g_u g_{uu} + g_v g_{uv} + g_v g_{vv} + g_u g_{uv}}{|n|}\vec{w} \\
&= \frac{1}{|n|^3}\left[g_u\eta - |n|^2\left(g_{uu} + g_{uv}\right)\right]\vec{u} + \\
&\quad \frac{1}{|n|^3}\left[g_v\eta - |n|^2\left(g_{vv} + g_{uv}\right)\right]\vec{v} + \\
&\quad \frac{\eta}{|n|}\vec{w} \qquad (5.9)
\end{aligned}
$$

where $\eta = g_u g_{uu} + g_v g_{uv} + g_v g_{vv} + g_u g_{uv}$.

The direction of maximal change in the unit surface normal in the plane of the *Monge* basis is therefore given by:

$$\phi = \tan^{-1}\left[\frac{g_v\eta - |n|^2\left(g_{vv} + g_{uv}\right)}{g_u\eta - |n|^2\left(g_{uu} + g_{uv}\right)}\right] \qquad (5.10)$$

This computation has an advantage over the principal directions computation in that it does not require the computation of both principal directions and the extra step to test for the direction corresponding to the larger of the principal curvatures.

Derivatives of Gradient : Finally, the direction of maximal surface acceleration or curvature may be approximated by taking the gradient of the gradient of the surface:

$$\mathcal{A}(u,v) \;=\; \nabla(\nabla g(u,v))$$

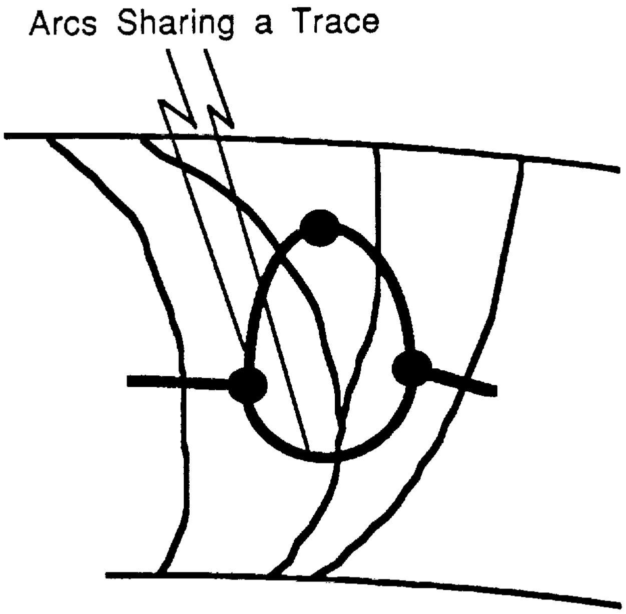


Figure 5.7: More than one arc may map to the same acceleration trace in the modified *region adjacency graph*

$$= g_{uu}\vec{u} + g_{vv}\vec{v} \tag{5.11}$$

This is not strictly correct in terms of surface acceleration or curvature since it is only the gradient of change of the maximal velocity vector of the surface, but experiments have shown that it is a useful approximation.

5.10 Modifications to the Region Adjacency Graph

Modifications are necessary in the adjacency relationship when *region adjacency graphs* are applied to acceleration bands. The 4-connected or 8-connected neighbourhood criterion discussed in section 5.2 does not apply because all the sub-regions in the split are separated by the *acceleration traces* (see the example for the segmented L-pipe in figure 5.6).

Define the *trace-neighbours* of an *acceleration band* to be set of *acceleration traces* which are 4 or 8 connected with the band; and, define the *band-neighbours* of an *acceleration trace* to be the set of *acceleration bands* similarly connected to the trace. We now have a modified *region adjacency graph* in which the nodes are the *acceleration bands* and the arcs are the *acceleration traces*. Note that more than one arc in the graph may map to the same *acceleration trace* as shown in figure 5.7. Theoretically, this should not happen since the acceleration traces may get very close to each other, but never actually intersect. In practice, however, traces are rounded into the pixels they occupy – and such trace mergings occur.

5.11 Specifics of Hypothesis Guided Region Merging for Planes and Cylinders

At this point, the operators necessary to perform the hypothesis guided merging of sub-regions generated by the splitting operations of section 5.6 into constructed surface regions are discussed. These operators are utilized in step 5 of the algorithm outlined in section 5.5.

Recall that the algorithm is based upon the fitting of *companion linear forms* of the hypothesized constructed surface to the splitted sub-regions and observing the *goodness-of-fit* to determine whether a sub-region ought to be recruited into the constructed surface. We shall now discuss the *companion linear forms* for planes, cones and cylinders.

Later it will be shown how these *companion linear forms* are capable of parameter estimation of the constructed forms as well.

5.11.1 Companion Linear Form for Planar Regions

Since the planar equation given by:

$$a + bx + cy = z \tag{5.12}$$

is a linear function in the form of equation 4.8, the *companion linear form* is identical to the constructed form (i.e. it is the plane function).

5.11.2 Companion Linear Form for Cylindrical Regions

The depth function $z = f(x, y)$ for cylinders, however, is a non-linear function in terms of the angles of inclination and offsets of the cylinder axes. An approximating linear function for the surface is required – such a form is the biquadratic surface (equation 4.56):

$$z = a + bx + cy + dxy + ex^2 + fy^2$$

which was analyzed in sections 4.4.2 and 4.4.3

In brief review of our analysis of the biquadratic function, it has been shown that:

- Any horizontal section of the biquadratic function yields a conic section
- The function has two perpendicular principal axes in the x-y plane.
- The function is symmetric around the principal axes.
- When $d^2 - 4ef = 0$ (known as the *characteristic* of the biquadratic form) the function is a *RIDGE* region everywhere (therefore it is capable of fitting cylinders).
- The characteristic is invariant under translation and rotation.

It will be shown later, in our discussion of cylinder parameter estimation, that this function is an approximation (rather than an exact fit) to cylindrical surfaces. It will be shown, however, that the ratio of the fitting error to the area of the fitted region remains constant and is independent of the length of the cylinder being fit.

CHAPTER VI

DETERMINING THE POSE OF CYLINDERS

Once a region in a scene is known to be a cylindrical surface, the question remains as to how the pose of that cylinder may be extracted. In this chapter, we shall discuss how the pose parameters may be estimated using the coefficients of the biquadratic surface fitted to the region and how this may be refined by fitting the cylindrical function to the data.

6.1 Biquadratic Properties-Based Cylinder Parameter Estimation

In section 4.4, the properties of the biquadratic function:

$$z = a + bx + cy + dxy + ex^2 + fy^2 \qquad \text{equation 4.56}$$

has been discussed in detail. In the previous section, we have seen how this function is applicable as a *companion linear function* for hypothesis-based region merging to cylinder regions in a range image. In this section, we shall see the utility of this function for parameter estimation of cylinders.

Assume that from the previous matching and split-merge operations, the region in the range image occupied by a cylinder is known. Assume, too, that the radius of the cylinder is known from the model. The task now is to place the cylinder in $\mathbf{R}^3$ space.

A cylinder in $\mathbf{R}^3$ space can be wholly determined by the equation of its axis and the radius of its circular cross section. The task is then to acquire the equation of the cylinder axis.

6.1.1 Axis Projection on the *x*-*y* Plane

The equation of the projection of the axis in the *x*-*y* plane may be obtained through the ensuing analysis.

The partial derivatives of the biquadratic function are:

$$\begin{aligned}\frac{\partial z}{\partial x} &= b + dy + 2ex \\ \frac{\partial z}{\partial y} &= c + dx + 2fy\end{aligned}$$

By chain rule, we obtain:

$$\frac{\partial y}{\partial x} = \frac{b + dy + 2ex}{c + dx + 2fy} \tag{6.1}$$

In the space of lines parallel to the cylinder axis which lie on the cylinder surface, we enforce the relation

$$\frac{\mathrm{d}y}{\mathrm{d}x} = m = constant \tag{6.2}$$

as a constraint on equation 6.1.

These lines are thus defined from equation 6.1 and equation 6.2, we have:

$$m(c + dx + 2fy) = b + dy + 2ex \tag{6.3}$$

Differentiating expression 6.3 with respect to x, we have:

$$c\frac{\partial m}{\partial x} + dm + 2fy\frac{\partial m}{\partial x} + 2fm\frac{\mathrm{d}y}{\mathrm{d}x} = d\frac{\mathrm{d}y}{\mathrm{d}x} + 2e \tag{6.4}$$

But, as $m \rightarrow constant$, $\frac{\partial m}{\partial x} \rightarrow 0$. Therefore, applying the constraint of equation 6.2 to equation 6.4, we have:

$$\begin{aligned}dm + 2fm^2 &= dm + 2e \\ m &= \pm\sqrt{\frac{e}{f}} = \tan\theta\end{aligned} \tag{6.5}$$

where θ is the angle between the projection of the cylinder axis on the *x*-*y* plane and the x-axis.

6.1.2 Test of 'Cylinderness'

From equation 4.63, the principal axes of the biquadratic function are given by:

$$\tan 2\theta = \frac{d}{e-f}$$

Applying the trigonometric identity $\tan 2\theta = \frac{2\tan\theta}{1+\tan^2\theta}$ to equation 4.63 and equation 6.5, we have

$$d = -2\sqrt{ef} \tag{6.6}$$

Comparing equation 6.7 with the characteristic of the quadratic form given in equation 4.84 and table 4.4, it is evident that this relationship indicates that horizontal ($z = constant$) section of the function is a parabola and the three-dimensional form is a *RIDGE* or *VALLEY* region everywhere. Equation 6.5 can thus be thought of as a *parabolic* constraint, forcing the parabolic characteristic ($d^2 - 4ef = 0$) on the biquadratic function.

Thus, from equation 6.6 we have

$$\text{abs}\left[1 - \text{abs}\left(\frac{d}{-2\sqrt{ef}}\right)\right] < \epsilon_{cylinder} \tag{6.7}$$

which constitutes the test if a set of data, fitted by a biquadratic function, forms a cylindrical surface.

6.1.3 Biquadratics are Only Approximations of Cylinders

Let the coordinate axes be rotated by an angle θ to coincide with the principal axes. From equation 4.86, the surface function is described by:

$$A + Bu + Cv + Eu^2 + Fv^2 = z$$

where

$$E = e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta \tag{6.8}$$

$$F = e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta \tag{6.9}$$

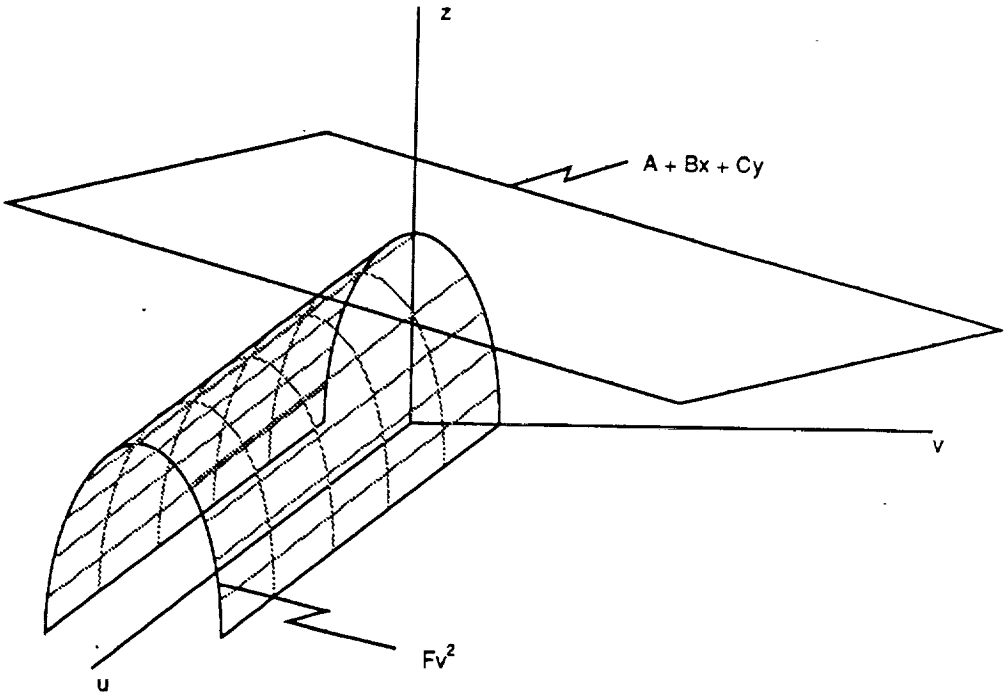


Figure 6.1: Decomposition of the rotated function $A + Bu + Cv + Fv^2 = z$

Applying the relation $\tan\theta = \pm\sqrt{\frac{e}{f}}$ (equation 6.5), and by trigonometric manipulation, the right hand side of equation 6.8 reduces to zero. F resolves to

$$F = \frac{e^2 + f^2 + 4ef}{e + f} \tag{6.10}$$

The surface function in the rotated coordinate system thus becomes:

$$A + Bu + Cv + Fv^2 = z \tag{6.11}$$

The first three terms of equation 6.11 describes a plane. Equation 6.11, therefore, describes the function $Fv^2 = z'$ summed with a planar surface (see figure 6.1).

By the preceding analysis, it can be seen that the biquadratic surface is only an approximation of a cylinder, the cross-section of which may be described by $z^2 = v^2 + r^2$, where r is the radius of the cylinder (see figure 6.2).

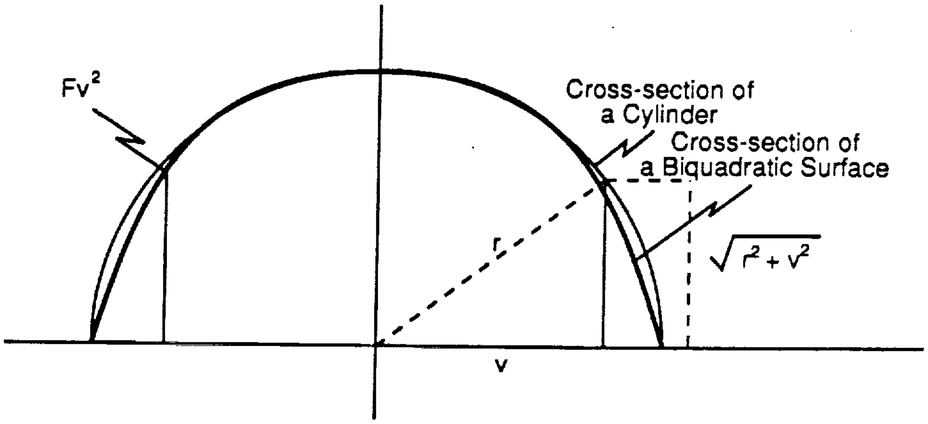

Fv^2
Cross-section of a Cylinder
Cross-section of a Biquadratic Surface
r
$\sqrt{r^2+v^2}$
v

Figure 6.2: The biquadratic surface is only an approximation of a cylindrical surface.

This approximation is appropriate, firstly, because the *error per unit length* of the cylinder to the fitted biquadratic surface is constant. Hence, the normalized χ^2 metric of equation 5.3 holds. Secondly, the error distribution is symmetric around the cylinder axis. This ensures that the principal axis of the biquadratic function lies on the cylinder axis. Furthermore, in our discussion of the chapter III and section 5.8, it has been shown that the data is more reliable at the top of the cylinder (close to the projection of the axis onto the x-y plane). At the edges where the surface is oblique to the range scanner, the data is less reliable and even appears planar.

6.1.4 Geometric Description

It should be observed from expression 6.5 that a non-imaginary tangent of the cylinder axis exists only if the coefficients of the square terms of the biquadratic equations have the same sign. If this is so, then the sign of F in equation 6.11 takes the same sign as e and f (see equation 6.10). Thus, if e and f are positive, the function describes a trough (the concave inside of a cylinder) and if they are negative, the function describes a ridge (convex top of a cylinder) – see figure 6.3.

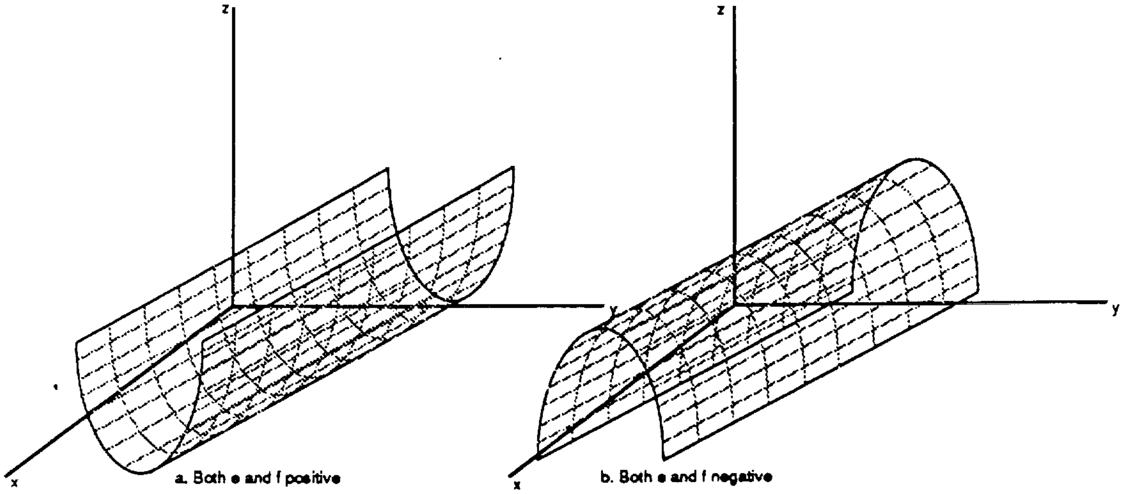


Figure 6.3: a. Positive values of e and f: The function describes a trough (the concave inside of a cylinder); b. Negative values of e and f: The function describes a ridge (convex top of a cylinder)

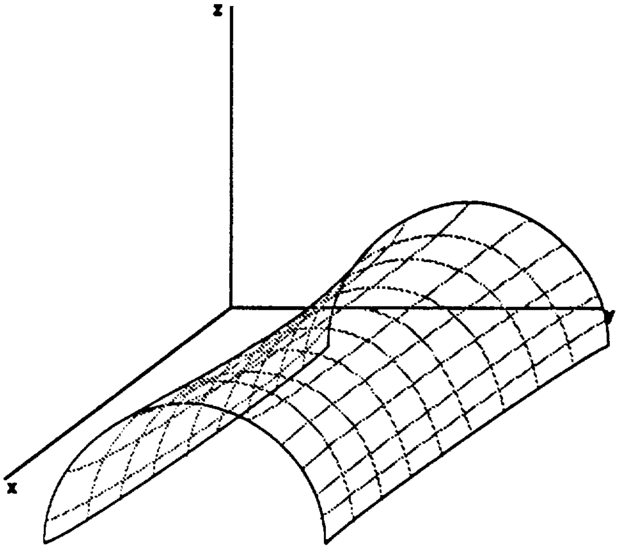


Figure 6.4: When the axis of the cylinder is parallel with one of the coordinate axes and fitting errors may give rise to a 'bowing' of the fitted surface away from the cylinder axis yielding a 'saddle'.

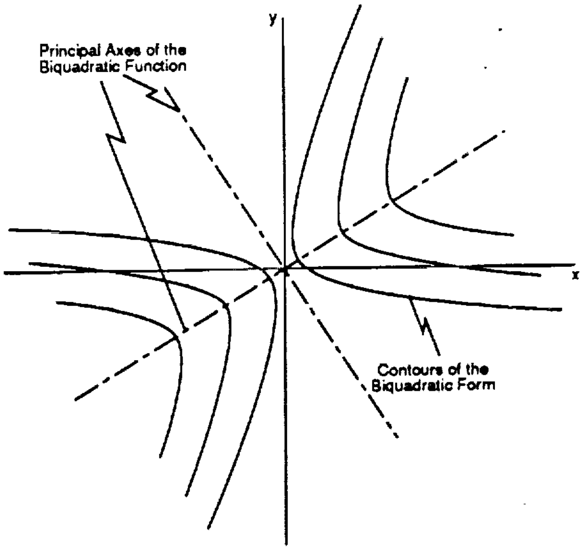

Principal Axes of the Biquadratic Function
Contours of the Biquadratic Form
y
x

Figure 6.5: The principal axes in a saddle biquadratic form

When the signs are different, the limiting two-dimensional form becomes a hyperbola and the three-dimensional form is a saddle (see table 4.3 and table 4.4). This happens only when the axis of the cylinder is parallel with one of the coordinate axes and fitting errors give rise to a 'bowing' of the fitted surface away from the cylinder axis (see figure 6.4). Even in such situations, the function is symmetrical around principal axes which are perpendicular to each other (see figure 6.5). The principal axes can still be computed by taking the absolute values of e and f before applying equation 6.5. Experimentation has shown this to be a very rare occurrence (it was never observed).

6.1.5 A Solution to the Quadrant Ambiguity

We have already observed that the computation of the principal axes by equation 4.63 and equation 4.64 that the computation is periodic on $\frac{\pi}{2}$. The computation of the axis of minimal curvature by equation 6.5 is also ambiguous on an interval of $\frac{\pi}{2}$. A resolution of this ambiguity is needed. One could compute the normal curvatures of the biquadratic surface for both angles $(\frac{\pi}{2})$ apart and determine the direction of minimal curvature. Fortunately, this is not necessary.

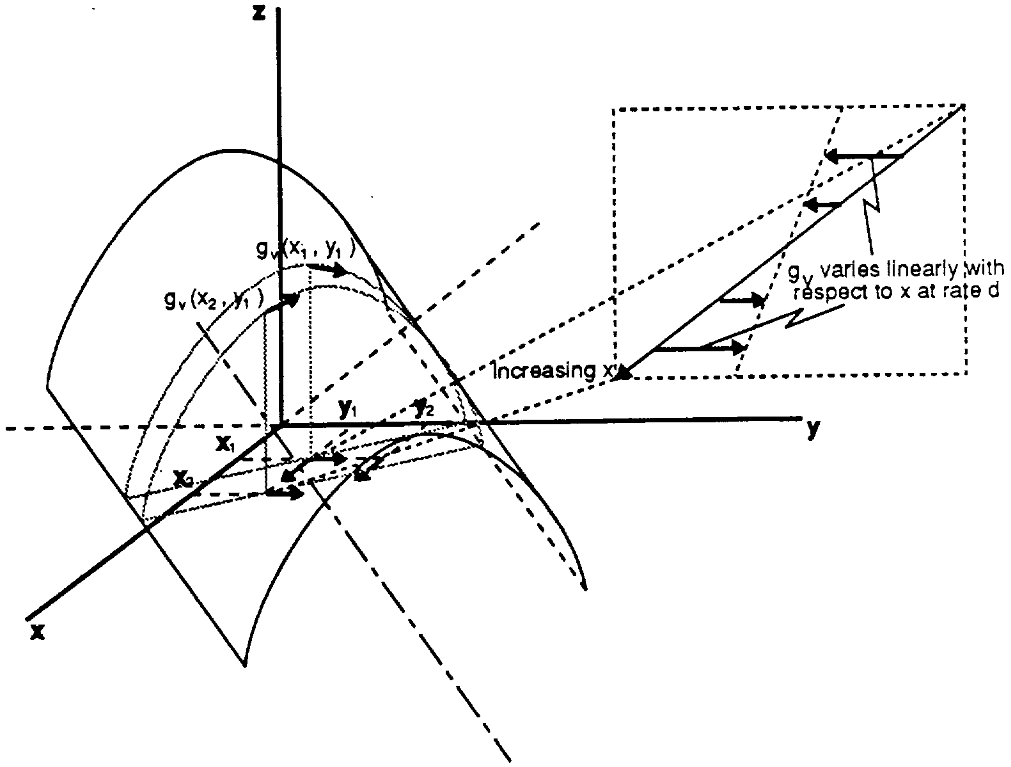


Figure 6.6: Convex biquadratic function with axis of symmetry passing through the first and third quadrants

The necessary information can be obtained by simply observing the signs of the coefficients of the biquadratic function.

Let us first make an observation concerning convex surfaces which satisfy the test of equation 6.7. We observe that

$$g_{uv} = g_{vu} = d \tag{6.12}$$

This says that the rate of change of the u-tangent with respect to v is equal to the rate of change of the v-tangent with respect to u; and that these rates are a constant equal to the coefficient of the cross-variable xy term, d.

Consider figure 6.6a of a convex biquadratic function (both e and f are negative). The u-

	$(e < 0)\&(f < 0)$ Convex Surface	$(e > 0)\&(f > 0)$ Concave Surface
$d < 0$	Quadrants 2,4 $\tan\theta < 0$	Quadrants 1,3 $\tan\theta > 0$
$d > 0$	Quadrants 1,3 $\tan\theta > 0$	Quadrants 2,4 $\tan\theta < 0$

Table 6.1: Summary of the quadrant/gradient analysis for biquadratic functions modeling cylinders and cones.

tangent vectors at points (x_1, y_1), (x_1, y_2) and (x_1, y_3) are $\vec{g_u}(x_2, y_1)$, $\vec{g_u}(x_2, y_2)$ and $\vec{g_u}(x_2, y_3)$ respectively. By observation, it is evident that

$$\vec{g_u}(x_2, y_1) \; < \; \vec{g_u}(x_2, y_2) \; < \; \vec{g_u}(x_2, y_3)$$

It is also clear that g_u increases monotonicly with increasing y. By the same process of observation, it is evident that the v-tangent vectors $\vec{g_v}(x_1, y_2)$, $\vec{g_v}(x_2, y_2)$ and $\vec{g_v}(x_3, y_2)$ are also a increasing progression in terms of magnitude. Thus $g_{uv} = g_{vu}$ for this surface configuration with the cylinder axis in the first and third quadrants is positive; or, the coefficient of the cross-variable xy term is greater than zero $(d > 0)$.

By the same analysis, it can be shown for figure 6.6b that a convex biquadratic function modeling a cylinder with axis inclined in the second and fourth quadrants has negative $g_{uv} = g_{vu}$.

Similarly, for a concave biquadratic function describing the inside surface of a cylinder, (both e and f are positive). The u-tangent and v-tangent vectors decrease with increasing y and x respectively when the cylinder axis is inclined in the first and third quadrants; and, the gradient of change of the u-tangent and v-tangent vectors is positive with respect to y and x respectively.

Table 6.1 summarizes the preceding analysis which applies for cones as well.

6.1.6 The Algorithm for Axis Determination

Up to this point, we have only discussed how the gradient of the projection of the axis of the cylinder axis on the x-y plane is estimated. To define a cylinder, the entire line equation of the axis in three-dimensional space is required along with the radius of the cylinder. As we have done all along, a hypothesis for the radius of the cylinder is assumed to be available from the object model. What is required then is the axis description. Since a line in three-dimensions can be described by the pair of gradient-intercept equations,

$$\begin{aligned} m_z x + c_z &= z \\ m_y x + c_y &= y \end{aligned} \tag{6.13}$$

These equations describe the projections of the axis on the x-z and x-y planes respectively with respect to x (see figure 6.7).

What we have to work with are the original data describing the cylinder, an estimate for m_y (and $\theta = \tan^{-1} m_y$) and the set of coefficients of the biquadratic function which fits the data.

We make the following observations:

- The axis of the cylinder and the 'top' of the cylinder are colinear in the image data space.
- The derivative of the function describing the cross-section of a cylindrical surface is zero at the top of the cylinder.
- The biquadratic fit is most reliable at the centroid of the data region being fitted.

The algorithm, based upon these observations, for computing the axis equations is as follows:

1. Compute the centroid (x_c, y_c) of the connected region which has been determined to belong to the cylinder surface.
2. Rotate the coordinate axes u-v by the inclination of the axis of the cylinder (the major axis

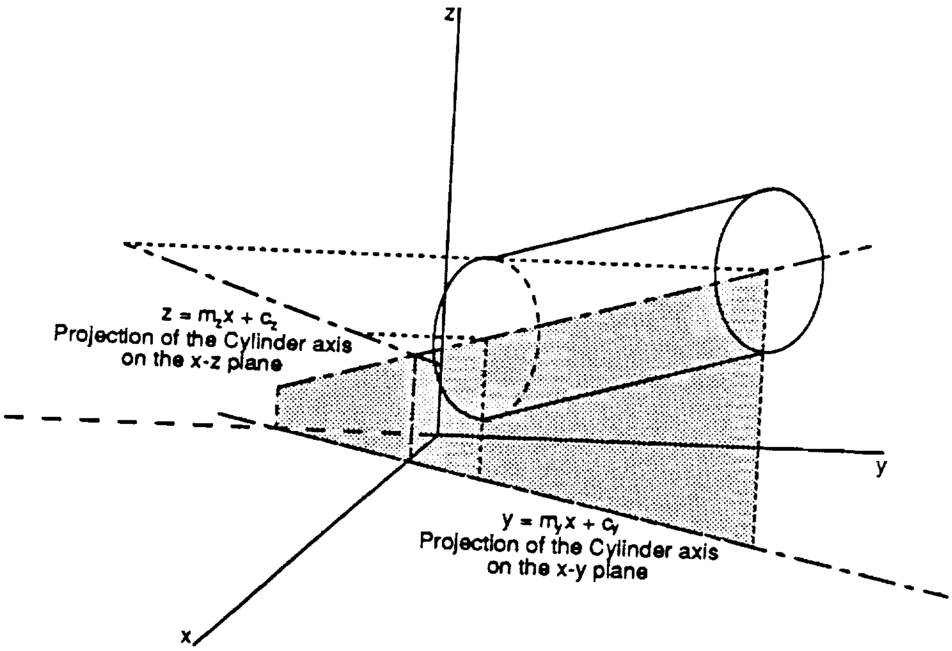


Figure 6.7: The three-dimensional description of a cylinder.

of the biquadratic function or principal axis of maximal curvature) θ and compute point (u_c, v_c) corresponding to (x_c, y_c) (see figure 6.8).

3. Obtain the function $z = f(u, v)$ for the section of the biquadratic surface generated by a vertical sectioning plane in the direction of the minor axis (principal axis of minimal curvature perpendicular to the major axis) passing through the (u_c, v_c).

4. Compute the maximal (or minimal) point (u_p, v_p) for the sectional function from its first derivatives.

5. Compute the corresponding maximal/minimal point (x_p, y_p) to (u_p, v_p).

6. Compute the y-intercept (c_y of equation 6.13) from (x_p, y_p) and m_y.

7. Compute the tangent (m_z of equation 6.13) of the biquadratic function in the vertical plane described by $y = m_y x + c_y$.

8. Compute the peak z value, z_p, of the biquadratic function at point (x_p, y_p) and compute the z-intercept of the line on the top of the cylinder $c_z{}'$ from (x_p, z_p) and m_z.

9. Compute the intercept of the cylinder axis from $c_z{}'$, $\phi = \tan^{-1} m_z$ and the radius of the hypothesized cylinder.

6.1.7 Axis Rotation and y-Intercept Computation

Let the centroid of the region occupied by the cylinder in the range image be (x_c, y_c). By algebraic manipulation of the rotation formulae (equation 4.61 the centroid of the region in a coordinate system rotated by θ from the x-y axis is:

$$\begin{aligned} u_c &= x_c \cos\theta + y_c \sin\theta \\ v_c &= -x_c \sin\theta + y_c \cos\theta \end{aligned} \tag{6.14}$$

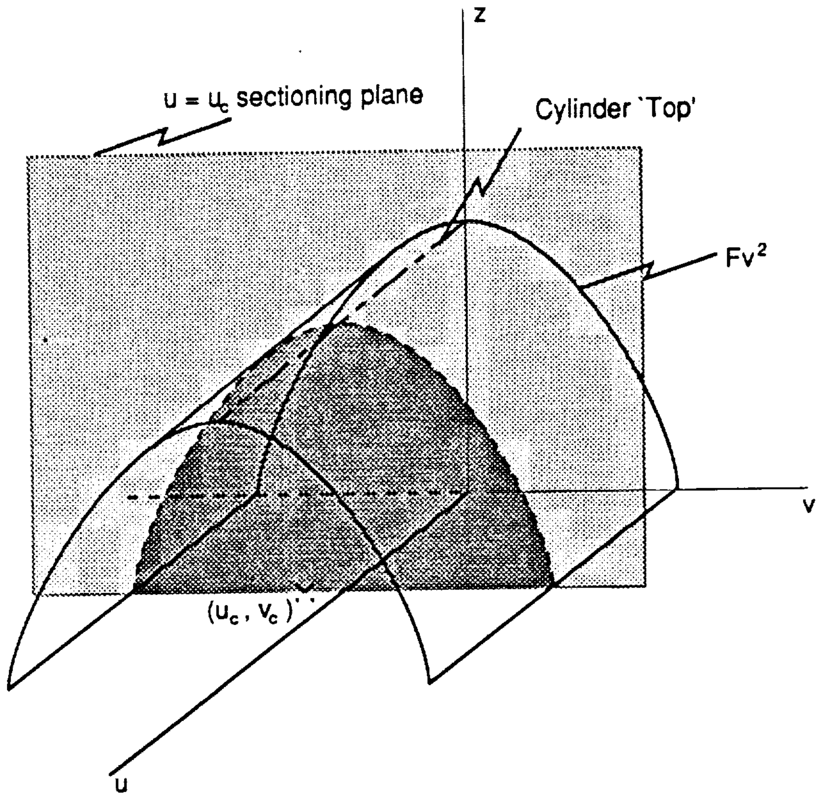


Figure 6.8: The cross-sectional function for computing the top of the cylinder.

It has been determined in section 4.4.3 that when the coordinate axes are rotated to correspond with a principal axis of a biquadratic function, the cross-variable drops out and we have equation 4.87:

$$A + Bu + Cv + Eu^2 + Fv^2 = z$$

where

$$\begin{aligned} A &= a \\ B &= b\cos\theta + c\sin\theta \\ C &= c\cos\theta - b\sin\theta \\ E &= e\cos^2\theta + f\sin^2\theta + d\sin\theta\cos\theta \\ F &= e\sin^2\theta + f\cos^2\theta - d\sin\theta\cos\theta \end{aligned}$$

Along the plane $u = u_c$, the biquadratic function traces the locus:

$$z = (A + Bu_c + E{u_c}^2) + Cv + Fv^2 \tag{6.15}$$

At the extremal point of expression 6.15,

$$\frac{\mathrm{d}z}{\mathrm{d}v} = C + 2Fv = 0 \tag{6.16}$$

Thus, the peak or trough point of the section occurs at

$$u_p = u_c \qquad v_p = -\frac{C}{2F} \tag{6.17}$$

Notice that, v_p is independent of u_c (equation 6.16). This is to be expected since we have already shown in section 6.1.3 that the cylinder axis runs parallel to the v-axis.

Having obtained (u_p, v_p), one can easily compute the corresponding point in x-y space from the rotation formulae (equation 4.61).

The y-intercept of the cylinder axis is thus given by:

$$c_y = y_p - x_p\tan\theta \tag{6.18}$$

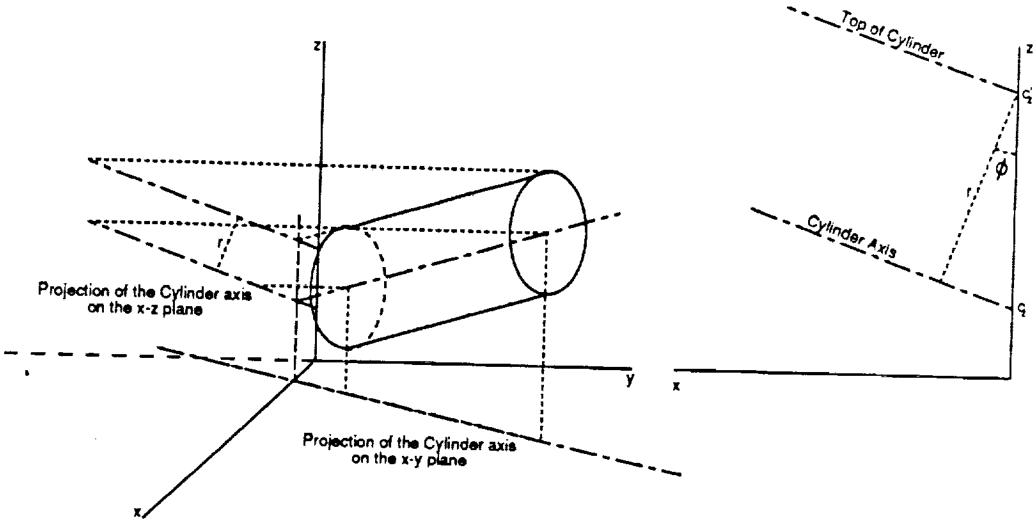

z
Projection of the Cylinder axis on the x-z plane
r
y
x
Projection of the Cylinder axis on the x-y plane
Top of Cylinder
z
c_z'
ϕ
r
Cylinder Axis
c_z
x

Figure 6.9: Estimation of the projection of the axis onto the *x*-*z* plane

6.1.8 Estimation of the Axis Projection Onto *x*-*z* Plane

Once the projection of the axis onto the *x*-*y* plane has been computed, the task remaining is to compute the half of the three-dimensional axis description (equation 6.13) which describes the projection of the axis in the *x*-*z* plane. This can be obtained by first computing the equation, in the *x*-*z* plane, of the line at the 'top' of the cylinder (which is parallel to and colinear in the image data space with the cylinder axis).

This, in turn, can be accomplished from the locus of intersection between the $m_y x + c_y = y$ plane with the biquadratic function. This locus is:

$$\begin{aligned} z &= (a + c\,c_y + f\,c_y) + \\ &\quad (b + c\,m_y + d\,c_y + 2f\,m_y\,c_y)x + \\ &\quad (dm_y + e + fm_y)x^2 \qquad (6.19) \end{aligned}$$

The gradient of z with respect to x may be computed by taking the first derivative of equation 6.19, giving:

$$\frac{\mathrm{d}z}{\mathrm{d}x} = b + c\, m_y + d\, c_y + 2f\, m_y\, c_y + (d\, m_y + e + f\, {m_y}^2)x \tag{6.20}$$

In accordance with the assumption made in section 6.1.6 that the biquadratic fit is most reliable at the centroid of the data region begin fitted, the gradient m_z is computed at x_p (obtained from (u_c, v_c)). Thus

$$m_z = b + c\, m_y + d\, c_y + 2f\, m_y\, c_y + (d\, m_y + e + f\, {m_y}^2)x_p \tag{6.21}$$

At point (x_p, y_p), the correspond z_p can be ascertained directly from the biquadratic function. Therefore the z-intercept of the 'top' of the cylinder may be computed as follows:

$${c_z}' = z_p - m_z x_p \tag{6.22}$$

As shown in figure 6.9, z-intercept of the cylinder axis can be computed from ${c_z}'$ by:

$$c_z = {c_z}' - \frac{r}{\cos\phi} \tag{6.23}$$

where the angle of incline of the axis is $\phi = \tan^{-1} m_y$, and r is the radius of the cylinder.

While this constitutes a second order estimation (it depends on the accuracy of the estimation of the projection of the axis onto the *x*-*y* plane), experiments have demonstrated that there is sufficient accuracy to permit convergence of the non-linear function fitting of the actual cylinder function to the data.

6.2 Non-Linear Cylinder Fitting

In section 4.1.5, non-linear parameter estimation was discussed. In this section, the cylinder will be used as an example of a constructed form for which non-linear fitting is required. In order to apply the *Levenberg-Marquardt* algorithm, the analytic expression for the data in the form: $z = f(x, y; \mathbf{a})$ and its first derivatives with respect to the pose parameters are required.

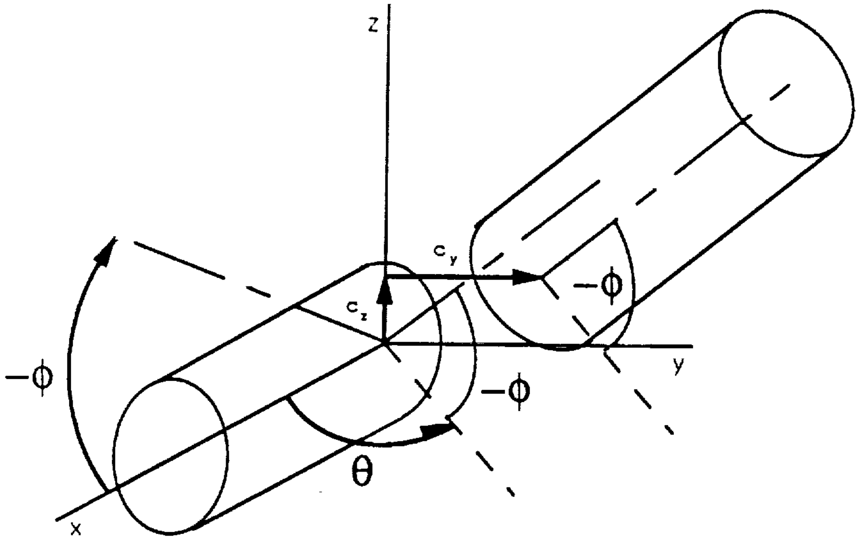


Figure 6.10: Rotation of a cylinder with axis colinear to the x-axis.

Consider figure 6.10 of a cylinder with its axis colinear with the x-axis. This cylinder is first rotated by an angle $-\phi$ from the x-axis in the x-z plane around the y-axis then by an angle θ around the z-axis; and then it is translated so that the axis projections on the x-y and x-z planes intersect the x-axis at c_y (call this the *y-intercept*) and c_z (call this the *z-intercept*) respectively.

The transformations are therefore:

$$[x \;\; y \;\; z \;\; 1] \begin{bmatrix} \cos(\phi) & 0 & \sin(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & c_y & c_z & 1 \end{bmatrix} \tag{6.24}$$

These evaluate to the single transformation matrix $\mathcal{T}$:

$$\mathcal{T}^T = \begin{bmatrix} (\cos\theta(x\cos\phi - z\sin\phi) - y\sin\theta) \\ (\sin\theta(x\cos\phi - z\sin\phi) + y\cos\theta + c_y) \\ (x\sin\phi + z\cos\phi + c_z) \\ 1 \end{bmatrix} \tag{6.25}$$

The equation of a cylinder with axis on the x-axis is:

$$z = \sqrt{r^2 - y^2} \tag{6.26}$$

Applying the parameterization to the coordinate system u, v, w, the transformation yields the system of four equations:

$$\begin{aligned} x &= \cos\theta(u\cos\phi - w\sin\phi) - v\sin\theta \\ y &= \sin\theta(u\cos\phi - w\sin\phi) + v\cos\theta + c_y \\ z &= u\sin\phi + w\cos\phi + c_z \\ w &= \sqrt{r^2 - v^2} \end{aligned} \tag{6.27}$$

Eliminating u, v and w from the system, we are left with the form:

$$\begin{aligned} z = {} & x\frac{\tan\phi}{\cos\theta} + \frac{\tan\phi\tan\theta}{\cos\theta}\left(\frac{y - x\tan\theta - c_y}{\tan^2\theta + 1}\right) + \\ & \cos\phi(\tan^2\phi + 1)\sqrt{r^2 + \left(\frac{y - x\tan\theta - c_y}{\cos\theta(\tan^2\theta + 1)}\right)^2} + c_z \end{aligned} \tag{6.28}$$

Let

$$\begin{aligned} \lambda &= y - x\tan\theta - c_y \\ \gamma &= \tan^2\theta + 1 \\ \beta &= \frac{\lambda}{\gamma} \\ \alpha &= \tan^2\phi + 1 \end{aligned}$$

Then equation 6.28 becomes

$$z = x\frac{\tan\phi}{\cos\theta} + \frac{\tan\phi\tan\theta}{\cos\theta}\beta + \alpha\cos\phi\sqrt{r^2 - \left(\frac{\beta}{cos\theta}\right)^2} + c_z \tag{6.29}$$

To compute the Hessian, the first derivatives of equation 6.29 with respect to the parameters of pose are required. They are given by:

$$\begin{aligned}
\frac{\partial z}{\partial \theta} &= x\frac{\tan\phi\tan\theta}{\cos\theta} + \frac{\tan\phi}{\gamma\cos^3\theta}\left[\lambda(1+\sin^2\theta) + \tan\theta(-x - 2\beta\tan\theta)\right] + \\
&\quad \frac{\alpha\beta\cos\phi(x + 2\beta\tan\theta - \lambda\sin\theta\cos\theta)}{\gamma\cos^4\theta\sqrt{r^2 - \left(\frac{\beta}{cos\theta}\right)^2}} \\
\frac{\partial z}{\partial \phi} &= \frac{x + \beta\tan\theta}{\cos\theta\cos^2\phi} + \left(\frac{2\tan\phi}{\cos\phi} - \alpha\sin\phi\right)\sqrt{r^2 - \left(\frac{\beta}{cos\theta}\right)^2} \\
\frac{\partial z}{\partial c_y} &= \frac{1}{\gamma\cos\theta}\left[\frac{\alpha\beta\cos\phi}{\cos\theta\sqrt{r^2 - \left(\frac{\beta}{cos\theta}\right)^2}} - \tan\phi\tan\theta\right] \\
\frac{\partial z}{\partial c_z} &= 1 \\
\frac{\partial z}{\partial r} &= \frac{r\alpha\cos\theta}{\sqrt{r^2 - \left(\frac{\beta}{cos\theta}\right)^2}}
\end{aligned} \tag{6.30}$$

In the previous sections, the axis of the cylinder was estimated in terms of the gradients of z and y with respect to x, m_z and m_y respectively. These estimates can be converted to ϕ and θ as follows (see figure 6.11):

$$\begin{aligned}
\theta &= \tan^{-1}(m_y) \\
\phi &= \tan^{-1}\left(\frac{m_z}{\sqrt{1 + {m_y}^2}}\right)
\end{aligned} \tag{6.31}$$

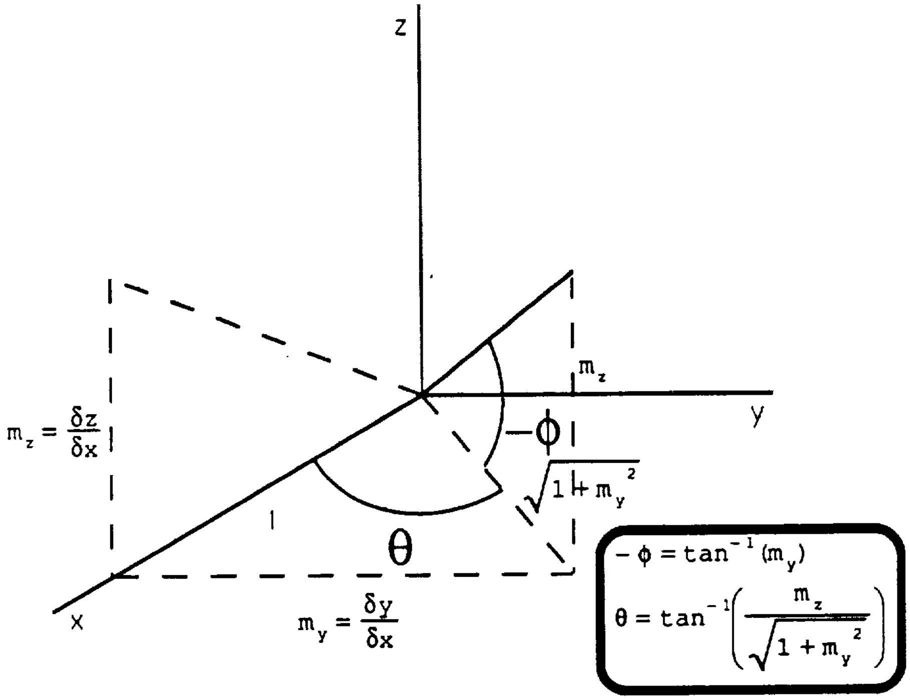

z
y
x
m_z
$m_z = \frac{\delta z}{\delta x}$
$-\phi$
$\sqrt{1+m_y^2}$
1
θ
$m_y = \frac{\delta y}{\delta x}$
$-\phi = \tan^{-1}(m_y)$
$\theta = \tan^{-1}\left(\frac{m_z}{\sqrt{1+m_y^2}}\right)$

Figure 6.11: Conversion between gradient notation and ϕ-θ notation.

CHAPTER VII

ON THE EXTRACTION OF EDGES FROM RUN-LENGTH REPRESENTATIONS

7.1 Run Length Regions

In the major portion of the work, data are organized as regions which occupy space in the imagery. The data format to represent this occupancy is the *run-length region* (see figure 7.1). Each region is assigned a list of *runs*, each of which is a three-tuple: $(y_i, x1_i, x2_i)$ where y_i is the y coordinate (line number in the image) and $x1_i$ and $x2_i$ are the start (lowest column index) and the end (highest column index) of x coordinate values in line y_i. We shall designate the k^{th} region $\Re_k$ and the i^{th} run $r_i = (y_i, x1_i, x2_i)$. Thus r_i in region $\Re_k$ is the set of pixels:

$$\{(x, y_i) \in \Re_k | \forall x1_i \leq x \leq x2_i\} \tag{7.1}$$

If region $\Re_k$ is made up of N runs, then

$$\bigcup_{i=1}^{N} r_i = \Re_k \text{ and } r_i \cap r_j \neq \emptyset \iff (i = j) \tag{7.2}$$

7.2 Scanning Algorithms for Extracting Region Boundaries

Given a region, the task of computing its boundary may be simply accomplished by generating a labelled image representation of the region where all points in the image belongs to the region if and only if its pixel value bears some labelling value. A walking algorithm may then be applied to trace the boundary of the region. Using the scanning template in figure 7.2, to obtain

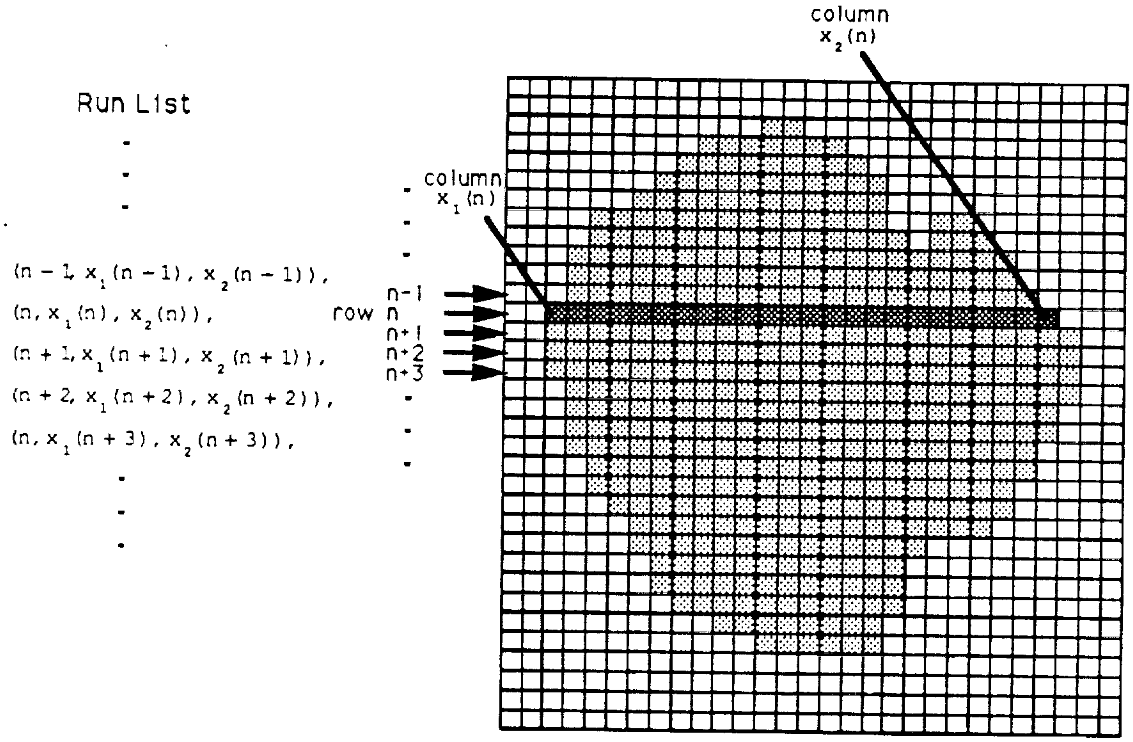

column
$x_2(n)$
Run List
(n − 1, x_1(n − 1), x_2(n − 1)),
(n, x_1(n), x_2(n)),
(n + 1, x_1(n + 1), x_2(n + 1)),
(n + 2, x_1(n + 2), x_2(n + 2)),
(n, x_1(n + 3), x_2(n + 3)),
column
$x_1(n)$
n-1
row n
n+1
n+2
n+3

Figure 7.1: A run-length region representation of spatial occupancy in an image.

3	2	1
4	p_{curr}	8
5	6	7

Figure 7.2: Scanning template to generate a positively directed boundary

a positively directed boundary (where the region is always on the left if one walks along the directed boundary generated), one may begin with any pixel p_s at the boundary of the region and do the following:

1. Set the current boundary point p_{curr} to p_s. With p_{curr} at the center of the template, scan the pixel's neighbours in the order $1, 2, \ldots 8$ (call this the scanning index) until there is a *transition* in the scanned pixel from outside the region to inside the region. (Let s be the scanning index at the transition point.)

2. Tag p_{curr} with a unique boundary label b.

3. Set the p_{curr} to the new boundary position (at the transition point *inside* the region). Set $s := (s + 5)$ *mod* 8 (where *mod* is the modulus operator).

4. While the scanned point is a region point, set $s := (s + 1)$ *mod* 8. (The desired point is the transition from non-region to region.)

5. While the scanned point is **not** a region point and p_s has not been reached, set $s := (s + 1)$ *mod* 8.

6. If p_s has been reached,

   (a) Scan the image for an untagged boundary point.

   (b) If a point is found, Set p_s to this point (This will take care of holes in the region) and goto step 2; else exit the boundary tracing algorithm.

7. Goto step 2

This algorithm may be extended to operate directly on the list of runs by ordering the runs by the y index and considering only the runs ±1 line from the current boundary point. A lot of extra bookkeeping would have to be implemented to label traversed pixels and the runs will have to be searched for each pixel being scanned by the scanning template.

7.3 Finite State Machine Approach to Boundary Extraction

In the scanning algorithm in which an image array is used, for each region, the image has to be labelled (and cleared if labels are to be reused), for each pixel in the boundary, the average scanning performed is 4 pixels, and for each hole, an average of half the region has to be scanned to find the hole. Thus for a region of M pixels, K boundary points and H holes, the number of operations is $O(4K + 2M + H\frac{M}{2})$. In the modified algorithm which does not use an image array, the array needs not be updated. However, if there is an average of R runs per line the number of operations becomes $O(4KR + H\frac{M}{2})$ excluding the considerable bookkeeping necessary. Neither of these algorithm exploits the information contained in the topology of the run list and the constraints introduced by direction of the boundary.

In this section, a finite state machine based algorithm which takes advantage of the run list topology and the boundary direction constraints is presented.

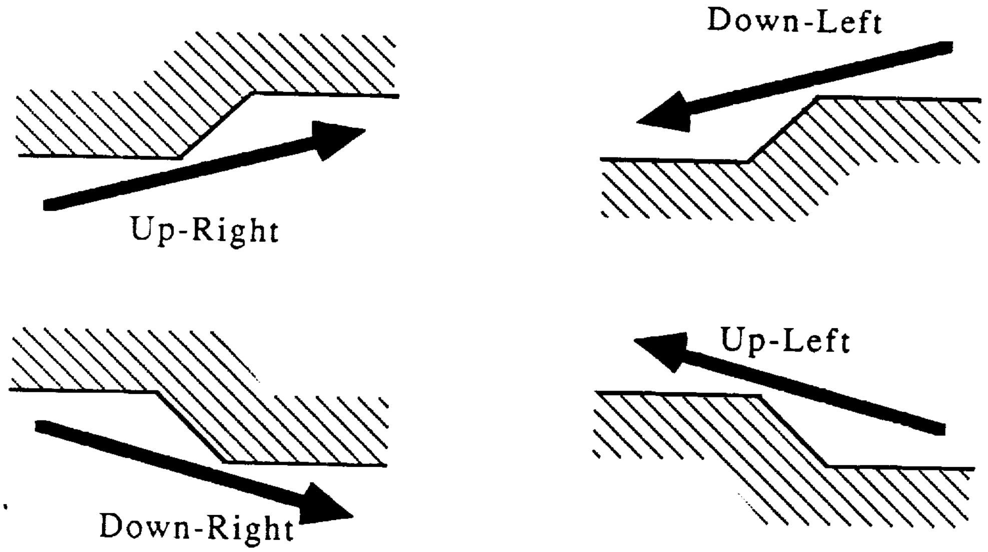


Figure 7.3: Boundary segment labels for a positively directed boundary.

7.3.1 The States

Observe that for positively directed region boundary, segments can be grouped into four forms (see figure 7.3):

Up-Right : Where the boundary is moving up and/or right with the region above and/or to the left of the undirected boundary points.

Down-Left : Where the boundary is moving down and/or left with the region beneath and/or to the right of the undirected boundary points.

Down-Right : Where the boundary is moving down and/or right with the region above and/or to the right of the undirected boundary points.

Up-Left : Where the boundary is moving up and/or left with the region beneath and/or to the left of the undirected boundary points.

These segment types constitute the states of our finite state machine. Let the positively directed region boundary comprise the ordered boundary segments $\{b_1, b_2, \ldots, b_N\}$. State transitions occur for each move from one segment to the next consecutive segment.

7.3.2 State Transitions

In the computation of the boundary of some region $\Re_k$, let p_{curr} be the current point on the boundary; R_{curr} be the current run; and R_{next} be the next run in a positively directed 'trek' around the boundary.

Let

$\mathcal{R}(n)$ denote the set of runs in $\Re_k$ on line n

$\mathcal{R}(n)_{l(p)}$, $\mathcal{R}(n)_{r(p)}$ denote the subsets of $\mathcal{R}(n)$ to the left and right of point p respectively.

$\mathcal{R}(n)_{l(Rc)}$, $\mathcal{R}(n)_{r(Rc)}$ denote the subset of $\mathcal{R}(n)$ completely to the left and right of run Rc respectively.

r_{left}, r_{right} denote the leftmost and rightmost points of run r respectively.

$x(p)$ denote the x (column) value of point p.

$line(r)$ denote the y (line or row) value of run r.

Define, also, the following predicates:

$IN(p, \mathcal{R})$ which searches a set of runs $\mathcal{R}$ for a run r such that $x(r_{left}) \leq x(p) \leq x(r_{right})$. The function returns r if it is found, else NIL is returned.

$\mathcal{I}_{left}(\mathcal{R}, r)$ which searches a set of runs $\mathcal{R}$ for a leftmost run λ which satisfies the condition: $x(r_{left}) \leq x(\lambda_{left}) \leq x(r_{right})$. The function returns λ if it is found, else NIL is returned.

$\mathcal{I}_{right}(\mathcal{R}, r)$ which searches a set of runs $\mathcal{R}$ for a rightmost run λ which satisfies the condition: $x(r_{left}) \leq x(\lambda_{right}) \leq x(r_{right})$. The function returns λ if it is found, else NIL is returned.

Finally, we define the 'walking' function:

$$WALK(n, p, r, end)$$

which adds points to the boundary list moving from point p toward the specified *end* point of run r (*end* may be either *left* or *right* designating r_{left} or r_{right} respectively.) such that:

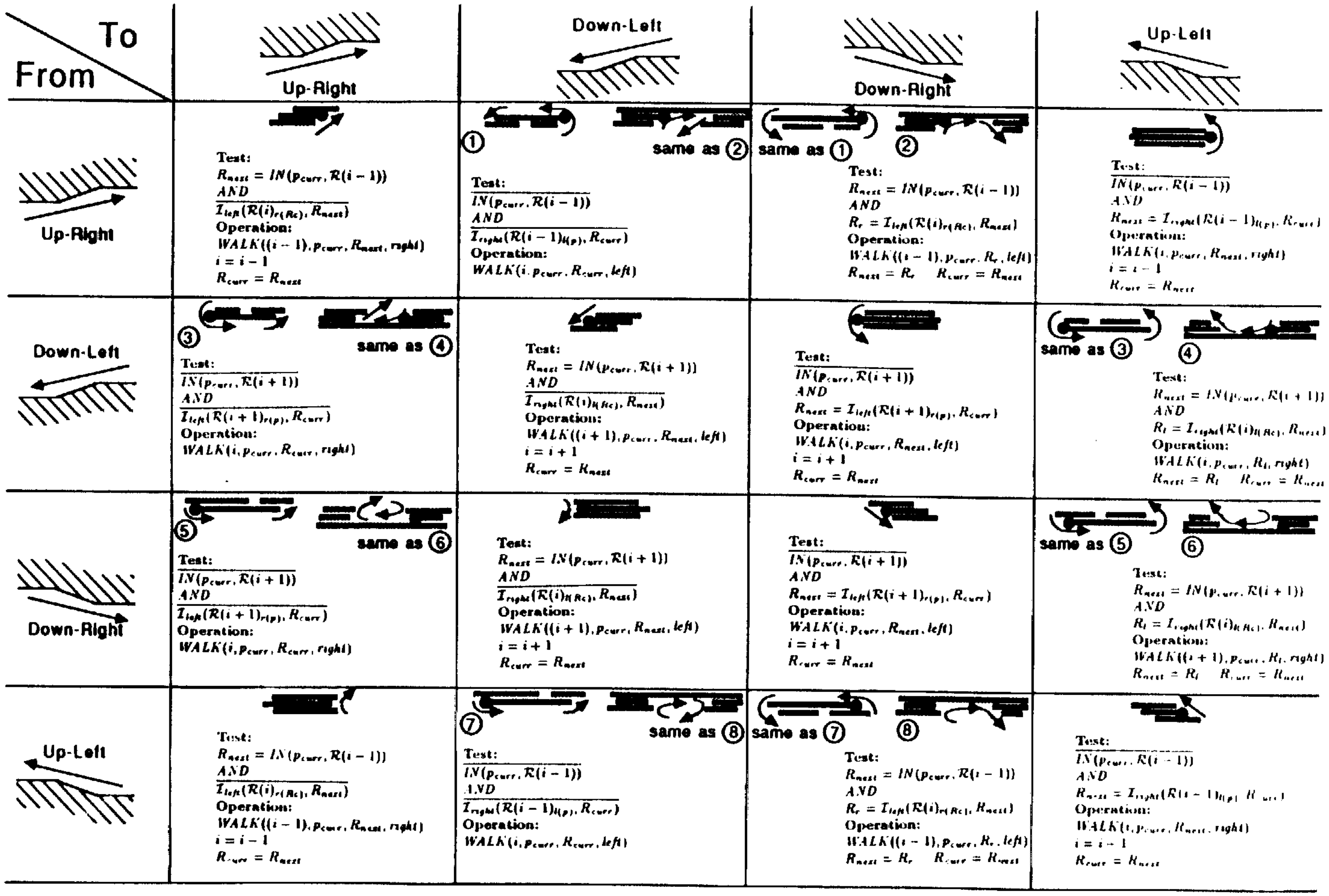
- All the points are on line n except for the final point which will be the specified end point of run r. Run r will always be ±1 line from n.
- If p is on line n, the set of points added to the boundary includes point p as the first point, else p is excluded (it would have been added in the previous state transition).
- If r is not on line n, r_{end} (i.e. r_{left} or r_{right}) is appended to the boundary list.

At the end of the *WALK*, r is tagged as being **`<left-used>`** or **`<right-used>`** with *end* being *left* or *right* respectively. Note that the direction of walk (left or right) is not specified. The walk is leftward (decreasing column index) if $(x(p) > r_{end})$ and rightward (increasing column index) if $(x(p) < r_{end})$.

For example, $WALK(i, p_{curr}, R_{next}, right)$ does the following:

1. Add point $(x(p_{curr}), i)$ to the boundary list if p_{curr} is not on line i.
2. Add points
$$(x(p_{curr}) + 1, i), (x(p_{curr}) + 2, i), \cdots, (x((R_{next})_{right}), i)$$
to the boundary list.
3. Tag R_{next} as being **`<right-used>`**.
4. Set $p_{curr} := (R_{next})_{right}$

Figure 7.4 is the transition table making use of these operations to generate a closed boundary (for readability, R_{curr} is written Rc and p_{curr} is written simply as p when they appear as subscripts). The rows of the transition table represent the *FROM* states while the columns are the target *TO* states. Each transition box in the table has one or more figures showing the run



To / From	Up-Right	Down-Left	Down-Right	Up-Left
Up-Right	Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i-1))$ AND $\overline{\mathcal{I}_{left}(\mathcal{R}(i)_{r(Rc)}, R_{next})}$ Operation: $WALK((i-1), p_{curr}, R_{next}, right)$ $i = i - 1$ $R_{curr} = R_{next}$	① same as ② Test: $\overline{IN(p_{curr}, \mathcal{R}(i-1))}$ AND $\overline{\mathcal{I}_{right}(\mathcal{R}(i-1)_{l(p)}, R_{curr})}$ Operation: $WALK(i, p_{curr}, R_{curr}, left)$	same as ① ② Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i-1))$ AND $R_r = \mathcal{I}_{left}(\mathcal{R}(i)_{r(Rc)}, R_{next})$ Operation: $WALK((i-1), p_{curr}, R_r, left)$ $R_{next} = R_r \quad R_{curr} = R_{next}$	Test: $\overline{IN(p_{curr}, R(i-1))}$ AND $R_{next} = \mathcal{I}_{right}(R(i-1)_{l(p)}, R_{curr})$ Operation: $WALK(i, p_{curr}, R_{next}, right)$ $i = i - 1$ $R_{curr} = R_{next}$
Down-Left	③ same as ④ Test: $\overline{IN(p_{curr}, \mathcal{R}(i+1))}$ AND $\overline{\mathcal{I}_{left}(\mathcal{R}(i+1)_{r(p)}, R_{curr})}$ Operation: $WALK(i, p_{curr}, R_{curr}, right)$	Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i+1))$ AND $\overline{\mathcal{I}_{right}(\mathcal{R}(i)_{l(Rc)}, R_{next})}$ Operation: $WALK((i+1), p_{curr}, R_{next}, left)$ $i = i + 1$ $R_{curr} = R_{next}$	Test: $\overline{IN(p_{curr}, \mathcal{R}(i+1))}$ AND $R_{next} = \mathcal{I}_{left}(\mathcal{R}(i+1)_{r(p)}, R_{curr})$ Operation: $WALK(i, p_{curr}, R_{next}, left)$ $i = i + 1$ $R_{curr} = R_{next}$	same as ③ ④ Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i+1))$ AND $R_l = \mathcal{I}_{right}(\mathcal{R}(i)_{l(Rc)}, R_{next})$ Operation: $WALK(i, p_{curr}, R_l, right)$ $R_{next} = R_l \quad R_{curr} = R_{next}$
Down-Right	⑤ same as ⑥ Test: $\overline{IN(p_{curr}, \mathcal{R}(i+1))}$ AND $\overline{\mathcal{I}_{left}(\mathcal{R}(i+1)_{r(p)}, R_{curr})}$ Operation: $WALK(i, p_{curr}, R_{curr}, right)$	Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i+1))$ AND $\overline{\mathcal{I}_{right}(\mathcal{R}(i)_{l(Rc)}, R_{next})}$ Operation: $WALK((i+1), p_{curr}, R_{next}, left)$ $i = i + 1$ $R_{curr} = R_{next}$	Test: $\overline{IN(p_{curr}, \mathcal{R}(i+1))}$ AND $R_{next} = \mathcal{I}_{left}(\mathcal{R}(i+1)_{r(p)}, R_{curr})$ Operation: $WALK(i, p_{curr}, R_{next}, left)$ $i = i + 1$ $R_{curr} = R_{next}$	same as ⑤ ⑥ Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i+1))$ AND $R_l = \mathcal{I}_{right}(\mathcal{R}(i)_{l(Rc)}, R_{next})$ Operation: $WALK((i+1), p_{curr}, R_l, right)$ $R_{next} = R_l \quad R_{curr} = R_{next}$
Up-Left	Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i-1))$ AND $\overline{\mathcal{I}_{left}(\mathcal{R}(i)_{r(Rc)}, R_{next})}$ Operation: $WALK((i-1), p_{curr}, R_{next}, right)$ $i = i - 1$ $R_{curr} = R_{next}$	⑦ same as ⑧ Test: $\overline{IN(p_{curr}, \mathcal{R}(i-1))}$ AND $\overline{\mathcal{I}_{right}(\mathcal{R}(i-1)_{l(p)}, R_{curr})}$ Operation: $WALK(i, p_{curr}, R_{curr}, left)$	same as ⑦ ⑧ Test: $R_{next} = IN(p_{curr}, \mathcal{R}(i-1))$ AND $R_r = \mathcal{I}_{left}(\mathcal{R}(i)_{r(Rc)}, R_{next})$ Operation: $WALK((i-1), p_{curr}, R_r, left)$ $R_{next} = R_r \quad R_{curr} = R_{next}$	Test: $\overline{IN(p_{curr}, R(i-1))}$ AND $R_{next} = \mathcal{I}_{right}(R(i-1)_{l(p)}, R_{curr})$ Operation: $WALK(i, p_{curr}, R_{next}, right)$ $i = i - 1$ $R_{curr} = R_{next}$

Figure 7.4: Transition table for the Finite State Machine to compute region boundaries

configuration. The horizontal lines in these figures represent the runs in the region with the solid black run being the current run R_{curr}. The current point *before* the transition is marked by the '•' on R_{curr}. Each transition box also contains one or two sets of '**Test-Operation**' pairs. Each Test contains two predicates which are performed sequentially to specify a decision tree. For example, from state **Up-Right**, if the test $IN(p_{curr}, \mathcal{R}(i-1))$ evaluates to *TRUE* (returning a run R_{next}), **Down-Left** and **Up-Left** are eliminated as possible target states. The second test: $\mathcal{I}_{left}(\mathcal{R}(i)_{r(Rc)}, R_{next})$ selects the *TO* state from between **Up-Right** and **Down-Right**. It will also be noticed that some tests lead to ambiguous *TO* states. This is not a problem because the rows 1 and 4 are identical in **Test** and **Operation** as are rows 2 and 3. The ambiguities in rows 1 and 4 are in columns 2 and 3 and the ambiguities of row 2 and 3 are in columns 1 and 4. The ambiguity is resolved in subsequent transitions. The transition table, therefore specifies a finite state machine with *ϵ-moves*[96, 120]. For example, from state **Up-Right**, satisfying the test of condition 2 yields two possible target states: **Down-Left** and **Down-Right**. From states **Down-Left** and **Down-Right**, however, all the tests and operations are identical and there is no ambiguity between **Down-Left** and **Down-Right** as *TO* states – thereby resolving the ambiguity.

7.3.3 The Algorithm

Given a starting point, the finite state machine in figure 7.4 will generate a closed positively directed boundary. This has to be repeated for all the holes which may exist in the region. Holes are located after each closed boundary is found by searching for *unused* ends of runs. The algorithm follows:

1. Organize the runs into an array of run lists indexed by line number and ordered from left to right.

2. **Set up for external boundary:** Set *STATE* := **DOWN-LEFT**; p_{curr} to the left end of the top left run; and $p_{start} = p_{curr}$. Tag the left end of the top left run as **<used>**.

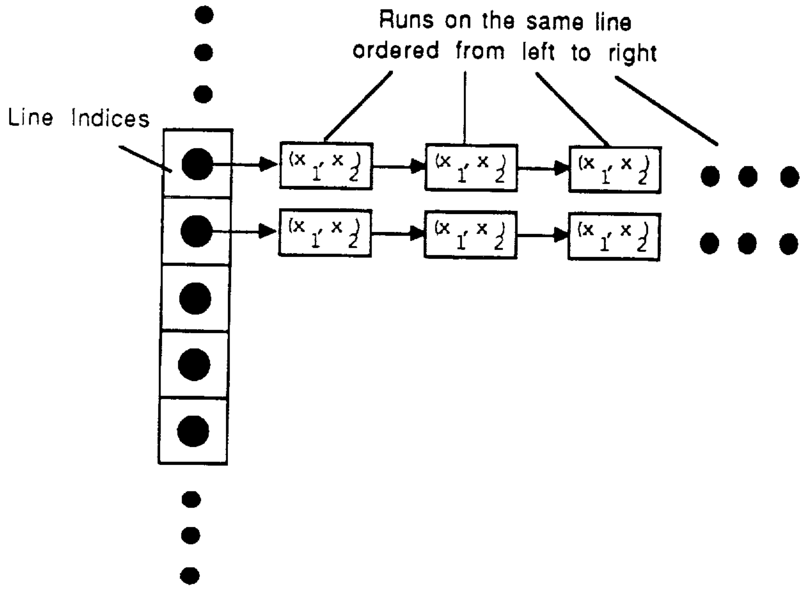


Figure 7.5: Organization of the region run lists for boundary extraction

```
3. Extract the external boundary:

   Loop

      Apply the finite state machine for one state transition.

   UNTIL $p_{curr} = p_{start}$

4. Proceed to operate on holes:

   Loop while there are <unused> ends

      (a) Set up for each hole: Set $STATE$ := UP-RIGHT; $p_{curr}$ to the top left
          <unused> end (of some run); and $p_{start} = p_{curr}$. Tag the top-left <unused>
          end as <used>.

      (b)

          Loop

             Apply the finite state machine for one state transition.

          UNTIL $p_{curr} = p_{start}$

   Loop end
```

CHAPTER VIII

EXPERIMENTAL RESULTS

This chapter sets forth the experiments performed in the course of this work. The discussion is arranged in three sections. The first details the apparatus employed; the second profiles the experimental procedure; and, the third describes the results of the experiments performed.

8.1 The Experimental Apparatus

The experimental apparatus comprises range sensors, computing hardware and the software tools. Each of these will be discussed in turn.

8.1.1 Sensor Hardware

Data obtained from two laser range sensors developed by the Environmental Research Institute of Michigan are analysed in the course of this work. The first is the *Intelligent Task Automation* sensor[156] (henceforth referred to as the ERIM/ITA scanner). The data available from this scanner is in the angle-angle-range format and the software does not exist to extract true Cartesian (x-y-z) readings from it. Data from the ERIM/ITA scanner are used to illustrate the robustness and generality of the curvature-based analysis. The rest of the data are obtained using the second sensor developed for the United States Postal Service[70, 141, 146] (henceforth referred to as the ERIM/USPS scanner). Software exists to convert the native angle-angle-range readings from this scanner into true Cartesian (x-y-z) form. Most of the experimental work is performed with data obtained from this scanner. The ensuing discussion pertains specifically to the ERIM/USPS scanner although the ERIM/ITA scanner operates under the same principles and possesses similar geometry (albeit with different parameters).

The ERIM/USPS scanner operates under two scanning modes – static and dynamic. In the static mode, the scanner sweeps out a solid angle in the same fashion as the electron gun in a television picture tube. This produces an image in which each pixel is addressed by the angle of sweep (row and column) and the value of the pixel describes the range or distance to a surface in the target image at that point. In dynamic mode, one axis of scan is removed yielding a single line scan. A conveyer belt then draws the objects to be scanned across this scan line in the target area. Since the experiments employ the scanner in the static mode alone, dynamic mode operation will not be discussed.

Sensing Mechanism

The ERIM/USPS scanner is a three-dimensional 'active' laser ranging sensor. A beam of sinusoidally amplitude modulated laser light irradiates each element of a target scene. The laser radiation which originates in a laser diode, is collimated by the *Transmit Optical System* into a narrow beam, and is directed at the target area by a scanning mechanism. The return signal from the laser radiation bouncing off the target is detected by the scanner's *Receiver System* which comprises the "scanner, *Receiver Optical System, Detector Diode, Digital Phase Detector* and the *Logarithmic Amplifier* with its analog-to-digital (A/D) Converter."[70] (see figure 8.1). In the figure, 700 MHz is the operating frequency of the scanner in static mode and in dynamic mode, the modulating frequency is 280 MHz. The *Receiver Optical System* is essentially a telescope whose field of view is synchronized with the scanning beam by the scanning mechanism. It directs the reflected laser radiation onto a photosensitive detector diode, generating a electric signal (at point X of figure 8.1) which is modulated by the amplitude and frequency of the

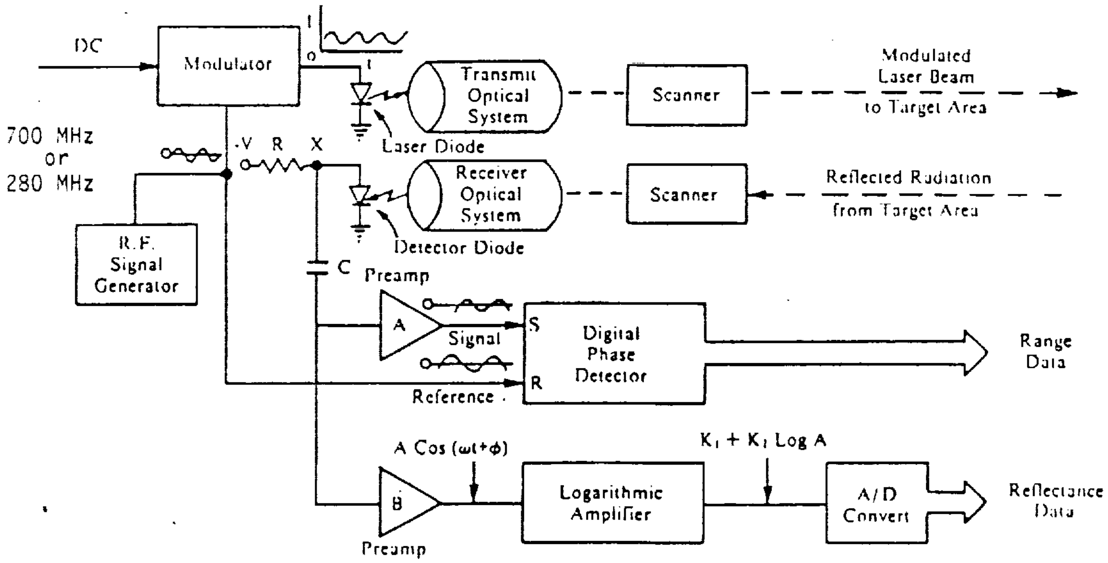


Figure 8.1: Simplified block diagram of the ERIM/USPS 3D scanner (from *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.)

reflected radiation. The phase difference between the sinusoidal component (at 700 MHz in static mode) of the reflected signal and the modulating signal of the incident beam (which serves as the reference signal) is thus a function of the distance covered by the laser beam to the target point. The *Digital Phase Detector* measures the time difference between the positive-going zero-crossing of the reference signal and that of the reference signal, generating a 12-bit digital range measure. The circuitry beginning with the preamplifier B in the figure extracts a measure of the reflectivity of the target surface. This will not be discussed as the reflectance data are not used.

Phase Measurement

The time taken for the scanning beam to make the round trip from the scanner to the target and back is given by:

$$\Delta t = 2\frac{R}{c}secs.$$

where R is the distance between the scanner and the target and c is the speed of light in air (1.1807×10^{10} inches/second at 15° and 76 cm Hg) – see figure 8.2. This yields a phase shift of

$$\Delta\phi = 2\pi f \Delta t = 4\pi\frac{fR}{c}radians \tag{8.1}$$

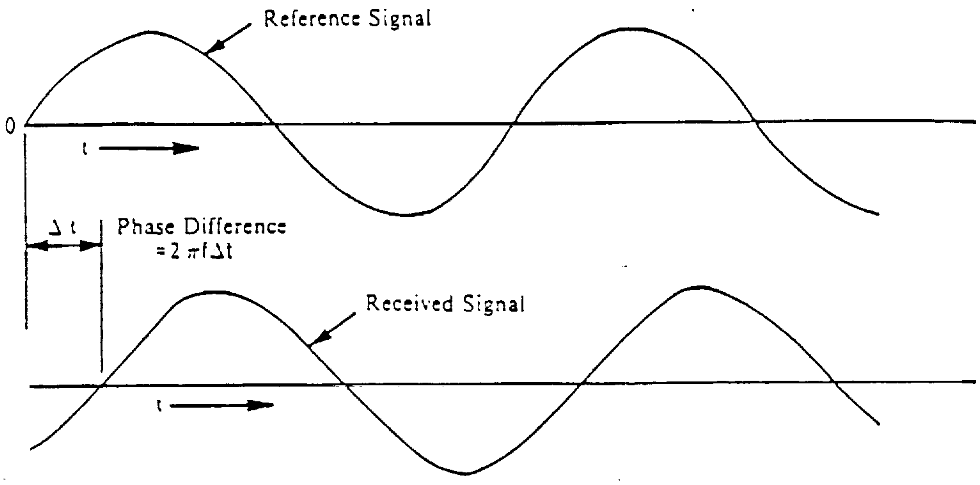


Figure 8.2: Phase relation between the reference signal and the received signal (from *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.)

where f is the modulation frequency in hertz.

Ambiguity Interval

In equation 8.1, $\Delta\phi$ is periodic on 2π. When the phase difference between the reference and reflected laser signals is coincident, the phase shift is zero. This occurs at integral intervals of 2π radians. This constitutes the ambiguity interval of the sensor which is given by substituting $\Delta\phi = 2\pi$ into equation 8.1, yielding:

$$R = \frac{c}{2f} \tag{8.2}$$

At a modulating frequency of 700 MHz, the ambiguity interval is

$$R = \frac{1.1807 \times 10^{10}}{2 \times 700 \times 10^{6}} = 8.434 in.$$

The range readings, hence, wrap around at intervals of 8.434 in. (see figure 8.3).

According to the specification detailed thus far, the sensor should have a range resolution of $8.434/2^{12} = 2.06 \times 10^{-3}$ inches. In practice, though, the instabilities in the radio-frequency circuitry and other noise restrict current capability of the sensor to an effective eight bits of useful range information.

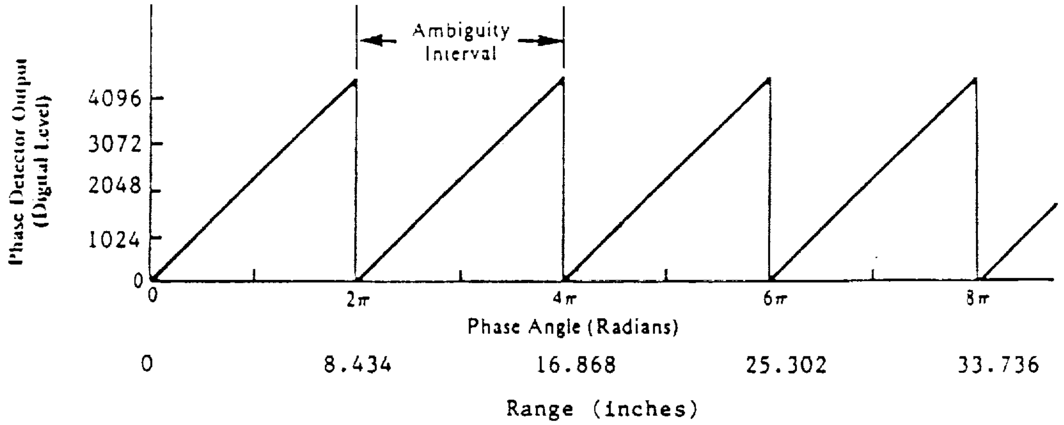


Figure 8.3: The digital phase detector response showing the sensor's range ambiguity interval (from *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.)

Sensor Geometry

In static mode, the ERIM/USPS scanner employs a scanning regimen which performs 320 sweeps (line scans) across the target area. During each sweep 320 readings are taken, yielding a 320×320 range image. This covers volume of 8.434 inches x 35° x 35° at a 32 inch standoff (see figure 8.4). The table detailing the ERIM/USPS scanner's static mode parameters is reproduced in table 8.1.

The horizontal line scan is generated by a rotating 8-sided polygonal mirror and the vertical scan is produced by use of nodding mirror (see figure 8.5).

We now reproduce the equations for calculating Cartesian *x*-*y*-*z* data from the raw range data from [70]. Given the sensed distance from the nodding mirror to the target R, the polygon angle (reflecting beam angle) θ_p, and the nodding mirror angle θ_n (see figure 8.6). The 'true' polygon angle is given by:

$$\theta_p' = \frac{\theta_p + 39.11}{2}$$

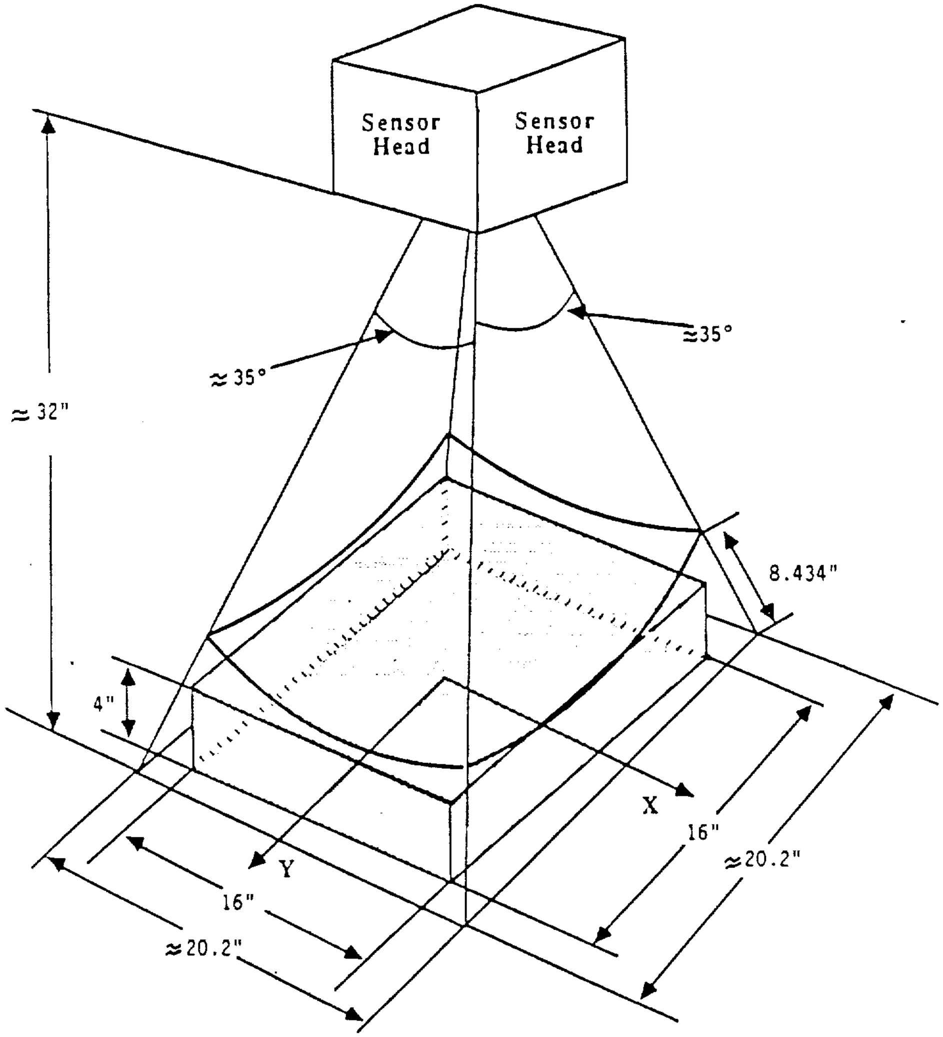


Figure 8.4: The field of view of the ERIM/USPS 3D scanner (from *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.)

Lines per image	320
Pixels per line	320
Beginning nod angle	−17.8797° ± 0.5°
Nodding angle increment	0.100448°
Beginning polygon angle	−14.76° ± 0.5°
Polygon angle increment	0.09°
Height above table	32.3542 ± 0.25*inches*
Inches per range count	8.434/4096
Nadir line	164
Nadir pixel	180
Constant offset	12", 12", 12"

N.B. The height of the sensor is calculated from nadir, where the beam has the shortest distance to the table. Transformed data is still ordered as it was scanned: the pixels in each scanline of increasing polygon angle and the scanlines in each image of increasing nodding mirror angle. Constant offsets were added to insure that all data were positive.

Table 8.1: Parameters of the ERIM/USPS 3D laser range scanner (from *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.)

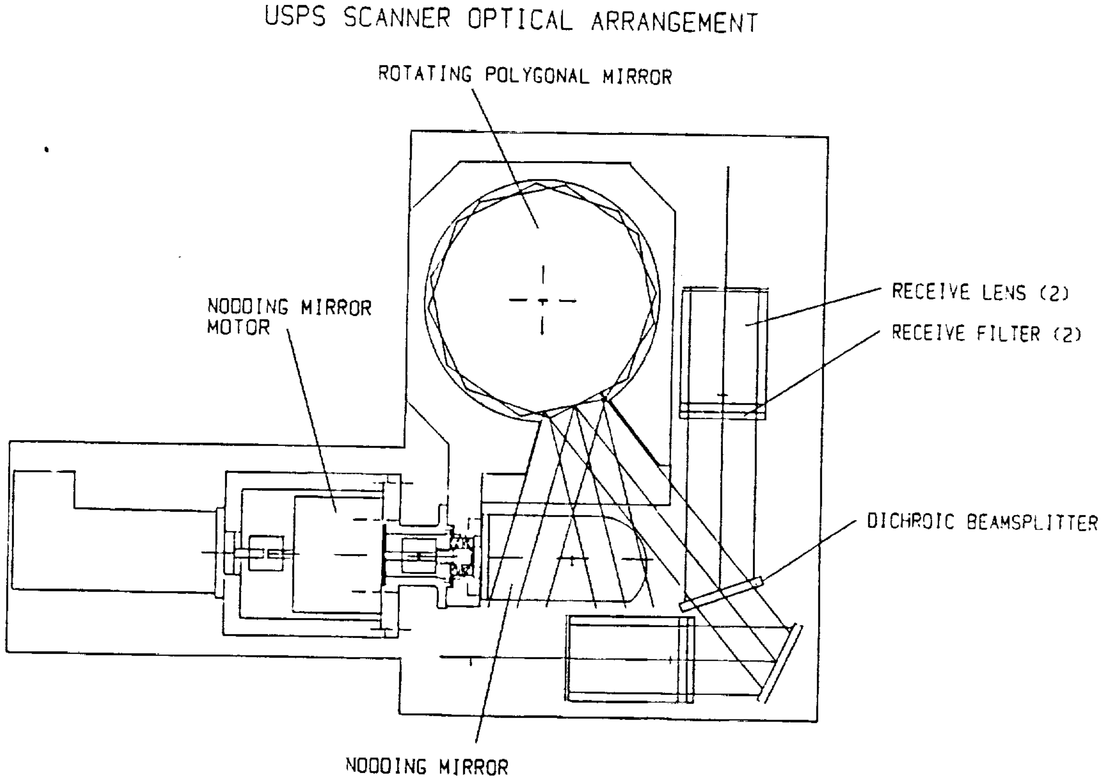


Figure 8.5: The optical arrangement of the ERIM/USPS 3D scanner (from *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.)

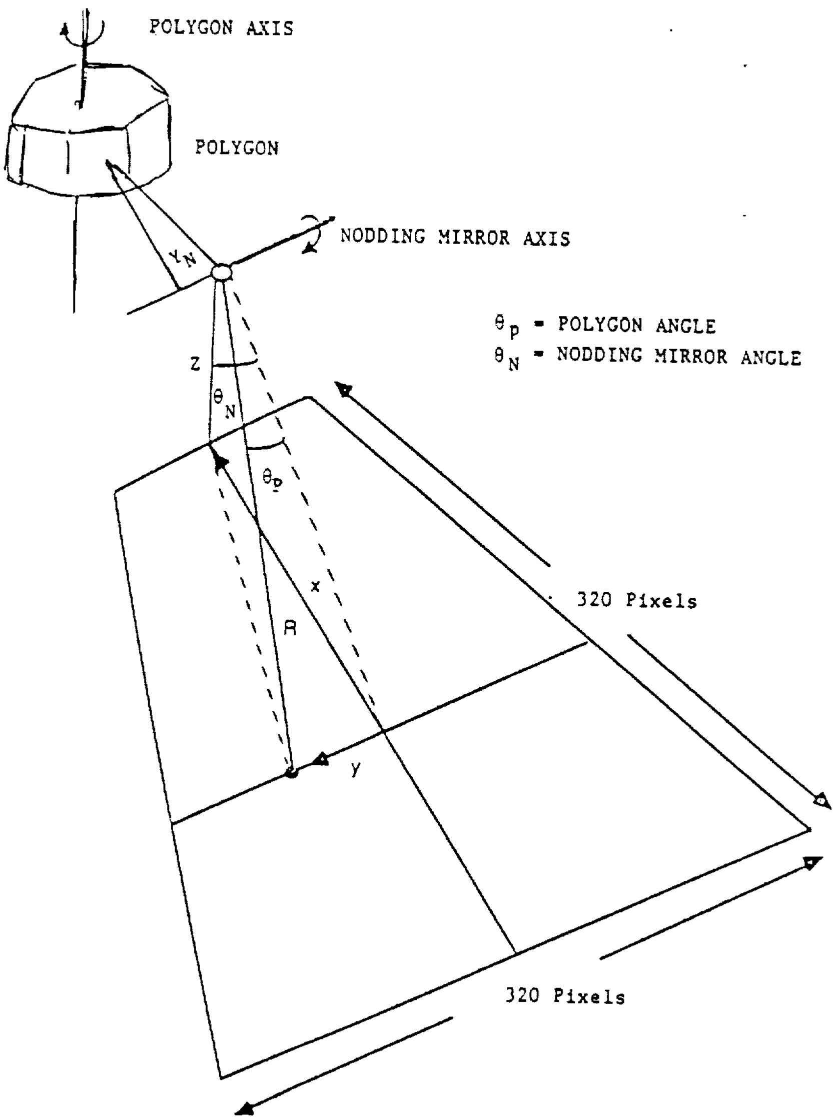


Figure 8.6: Geometric model for extracting Cartesian *x*-*y*-*z* data from a range image. (from *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.)

The coordinates of the beam on the polygon are:

$$\begin{aligned} X_p &= \frac{\frac{1.4176}{\tan(90-\theta_p')} + \frac{3.141}{\cos(\theta_p')}}{1 + \frac{0.813}{\tan(90-\theta_p')}} \\ Y_p &= 0.813X_p - 1.4176 \end{aligned}$$

The Y-coordinate of the beam on the nodding mirror is given by:

$$Y_n = (7.5 - X_p)\tan(\theta_p) + Y_p$$

The Cartesian coordinates of the target point (X_T, Y_T, Z_T) are given by:

$$\begin{aligned} X_T &= R\cos(\theta_p)\sin(\theta_n) \\ Y_T &= R\sin(\theta_p) + Y_n \\ Z_T &= R\cos(\theta_p)\cos(\theta_n) \qquad (8.3) \end{aligned}$$

The Z_T values are subtracted from a constant offset to obtain a 'height' reading.

8.1.2 Computing Architecture

Three computers were utilized in the work. The ERIM High-Speed Cytocomputer [69] drives the ERIM/USPS scanner and extracts images in the raw-sensor format. A Silicon Graphics workstation serves both as the three-dimensional display engine and the Unix host, on which most of the numeric computation takes place. A Symbolics Lisp workstation serves as the hypothesis generation engine. Most of the two-dimensional region-based image processing and display operations are also implemented on the Symbolics workstation.

During hypothesis generation and testing, the Symbolics workstation spawns a task on the Silicon Graphics workstation via a *Telnet pipe*. This *pipe* allows processes on the Symbolics machine to activate telnet sessions on other machines. From the Silicon Graphics end of the pipe, all that the operating system sees is a telnet session. The Symbolics end of the pipe is an object which takes input strings and waits for one of a set of events (return sub-strings from the

Unix station). Procedures (which include reading the return stream, executing code etc.) may be attached to the return events. The main task performed on the Silicon Graphics workstation in this mode is the fitting of three-dimensional functions to specified regions of the three-dimensional imagery.

8.1.3 Software Tools

The software tools employed in the course of the experiments are:

- *Range-angle-angle to x-y-z processing software* – This is a C program which transforms the raw sensor image in native sensor coordinates to Cartesian *x-y-z* coordinates. This program implements the transformation of equation 8.3. The raw sensor data takes the form of two byte-arrays (high and low range bytes) and the output generated are six byte-arrays $(x, y, z$ high and low byte). The resulting image scale is 0.001 inch per range count. The three pairs of images are registered such that for an image coordinate pair (i, j), the corresponding three-dimensional point is

$$\begin{aligned}&(0.001(256\mathit{HI\text{-}X}(i,j)+\mathit{LO\text{-}X}(i,j)),\\ &\ 0.001(256\mathit{HI\text{-}Y}(i,j)+\mathit{LO\text{-}Y}(i,j)),\\ &\ 0.001(256\mathit{HI\text{-}Z}(i,j)+\mathit{LO\text{-}Z}(i,j)))\end{aligned}$$

  where *HI-X* and *LO-X*, *HI-Y* and *LO-Y*, and *HI-Z* and *LO-Z* are the high and low x, y and z byte arrays respectively.

- *Besl/Jain variable order segmenter* – This is the set of C programs implemented by Paul Besl to perform the segmentation algorithms[22, 25, 26] described in section 4.2.

- *Variable order segmentation output to Lisp translation* – This is a C program that translates the output of the previous program into a run-length region-based Lisp descriptor. An example of the output is as follows:

```
(defregion
  (type Biquartic)
(state 4)
(coefficients 2694628.250000 -6355.307129 -72538.210938
              93.375771 38.125397 741.379578
              -0.417259 -0.433377 -0.073706
              -3.402546 0.000272 0.001246
              0.000571 0.000113 0.005910)
(area 428)
(top-left 62 134)
(bottom-right 101 149)
(runs
   (134 76 81) (135 73 87) (136 62 64) (136 73 86) (137 62 64)
   (137 67 83) (137 95 101) (138 63 64) (138 67 83) (138 95 101)
   (139 63 87) (139 95 101) (140 64 88) (140 92 92) (140 94 101)
   (141 64 101) (142 63 100) (143 63 100) (144 65 99) (145 65 98)
   (146 65 95) (147 69 94) (148 70 90) (149 74 75) (149 79 81)
   (149 84 86) (149 89 90)))
```

The descriptor details the order of polynomial fit performed, the coefficients obtained from the fitting, the area (in pixels) of the region in the image, the right rectangular bounding box of the region (in terms of the *top-left* and *bottom-right* coordinates), and the (*row start-column end-column*) description of the run-length regions. All the C and Lisp programs are capable of reading this data format.

- *Lisp-based hypothesis engine* – This is the bulk of the code written in the course of this work. The entire system is object-oriented using Symbolics Lisp Flavors. Each surface is an object and all the algorithms described in this thesis not implemented in the C code described in this section are implemented in Lisp on the Symbolics.

- *C program for remote fitting* – This is an interactive C program which takes as input region descriptors from the standard Unix input. It fits these regions as specified (e.g. "fit a biquadratic surface to a particular region in an *x-y-z* image set") and reports the results at the standard Unix output. This program is invoked in a telnet session on the Silicon Graphics workstation from the Symbolics Lisp workstation using a *Telnet pipe* described earlier in this chapter. The numeric fitting routines from the *Numerical Recipes Library* by Press et. al[140] were modified to operate on bivariate functions for this program. The objects which this program is capable of fitting are: polynomial surface up to and including the fourth order (biquartics), cylinders and straight lines in three dimensional space. *Singular value decomposition* is employed for the polynomial and line fits and cylinders are fitted using the *Levenberg-Marquardt method* for parameter estimation of non-linear forms.

- *C program for scene regeneration* – This program takes Lisp region descriptor files generated either by the C fitting program or the Lisp programs and produces *x-y-z* high and low byte three-dimensional images for display purposes.

- *Graphics display/plotting program* – The three dimensional perspective images in this dissertation are generated by a Silicon Graphics C program which plots the three-dimensional *x-y-z* surfaces to the Silicon Graphics screen. It assumes all adjacent data points to be connected as a surface and connects them to form surfaces. The screen images were oriented by hand and the screen buffers were saved as the byte images which appear in this chapter. The two-dimensional region-images were generated from state-labelled byte-images by a hatching program.

8.2 Experimental Sequence

This section overviews the sequence of experiments performed in the course of this work in two parts. The sequence in which the processing took place will be discussed followed by an overview of the data-sets processed.

8.2.1 Sequence of Processing

The processing took place in the sequence which conforms to the sequence of abstraction discussed in section 3.4 and diagrammed in figure 3.4. The sequence is as follows:

1. The ERIM/USPS scanner was run and the resulting range images were processed to obtain the Cartesian *x-y-z* image arrays.

2. The upper eight bits of the two-byte z image were extracted to form a byte-image. This is the input data format to the *variable-order segmentation* code described earlier. The output of this program is a set of smooth surface patches along with the coefficients of the polynomial surfaces which were fit to the patches. (The ERIM/ITA was already in byte-image format and were processed as-is). The patches were then translated to Lisp-form region descriptors.

3. The Lisp-form region descriptors were transferred to the Symbolics for the hypothesis generation and verification process to identify the geometric forms represented by the regions. The first computation performed in this process was the extraction of the Gaussian and mean curvature-based signatures from the polynomial descriptors as discussed.

4. Extraction of planar surfaces

   - If entire region has a 'FLAT' curvature signature, a plane surface was fitted to the three-dimensional data.
   - If not, the split-merge operation to extract planar surfaces was employed.

5. Extraction of cylindrical surfaces

   - If the surface satisfied the initial curvature-based analysis as cylindrical

     (a) A biquadratic surface was fitted to the three-dimensional data for cylinder parameter hypothesis

     (b) If the biquadratic fit satisfied the 'Cylinderness' test, the non-linear cylinder fit was performed on to the surface.

   - If the surface failed the initial curvature-based analysis as cylindrical or if a biquadratic fit to the surface failed the 'Cylinderness' test,

     (a) The split-merge operation to extract cylindrical surfaces one at the time (by performing biquadratic fits) was performed.

(b) The cylinder parameters were estimated from the coefficients of the biquadratic surface.

(c) The non-linear cylinder fit was performed on the three-dimensional data

8.2.2 Progression of Difficulty

A set of experiments designed along a progression of increasing difficulty was performed. These are listed here and described in greater detail in the next section.

1. *Curvature signature extraction of angle-angle-range range imagery of unknown x-y-z geometry*. Two images from a library of ERIM/ITA were processed to obtain smooth polynomial surface patches and Gaussian and mean surface curvature signatures. This demonstrates the robustness of these algorithms for images for which a true Cartesian transformation is not available. The images were those of:

   (a) a coffee cup

   (b) a space shuttle model

2. *Synthetic oriented cylinder images* – In this set of experiments, biquadratic surfaces are fitted to synthetically generated oriented cylinders to gauge the accuracy of the estimation of the axis projection of the cylinders from the polynomial coefficients.

3. *Image of blocks* – This is an ERIM/USPS scanner image of a collection of block (all the surfaces are planar).

4. *Image of a soft drink can* – This is an ERIM/USPS scanner image of a soft drink can against a flat background.

5. *Image of cylinders* – This is an ERIM/USPS scanner image of two cylindrical objects (a cylinder and a soft drink can).

6. *Image of blocks and cylinders* – This is an ERIM/USPS scanner image of a mixture of blocks and cylindrical objects.

7. *Image of a chamfered plane* – This is an ERIM/USPS scanner image of an surface made up of a plane merging smoothly into a curved chamfer. To extract the parametric description of the plane, the merge-split algorithm for planar surface recovery was applied.

8. *Image of a bent pipe elbow* – This is an ERIM/USPS scanner image of a bent fiberglass pipe comprising two straight cylindrical pipe segments merging together with a gradual 135° bend. To extract the parametric description of the two cylinders, the merge-split algorithm for cylindrical surface recovery was applied. The image is much noisier than the preceding ones because of the fiberglass material is translucent to the laser radiation and has specular characteristics.

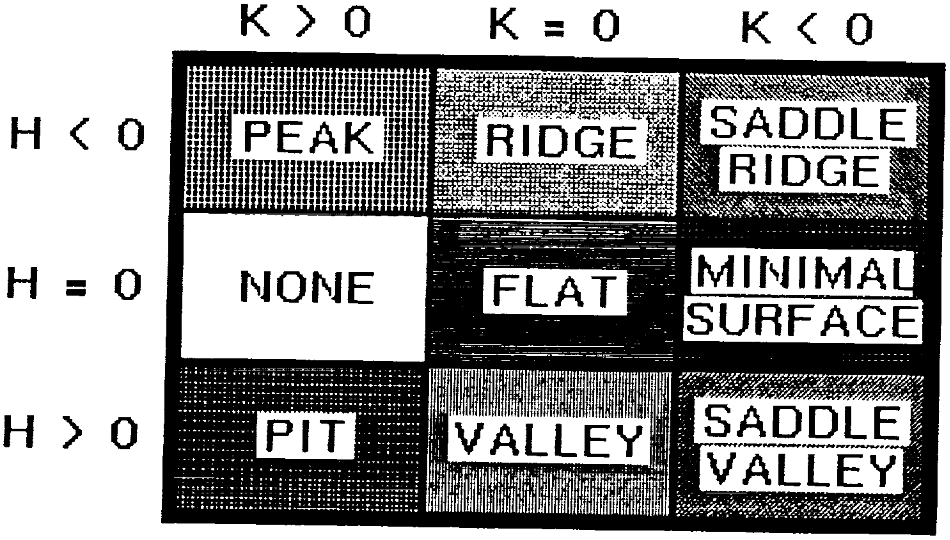
9. *Image of a PVC pipe ring* – This is an ERIM/USPS scanner image of a ring made up of four straight PVC pipe segments and four 90° PVC pipe elbows.

8.3 The Experiments

In the following experiments, when Gaussian and mean curvature sign-maps are displayed, the look-up table shown in figure 8.7 applies.

8.3.1 Curvature signature extraction of angle-angle-range range imagery of unknown x-y-z geometry.

This is a set of two experiments in which the Gaussian and mean curvature sign-maps were computed from images in the ERIM/ITA scanner's native coordinate system. Comparisons are made between the curvature sign-maps computed using the 7 × 7 kernel digital differentiation operators described in section 4.3.3 and the computation obtained from analytical computation using polynomial descriptors obtained by *variable-order segmentation*.



	$K > 0$	$K = 0$	$K < 0$
$H < 0$	PEAK	RIDGE	SADDLE RIDGE
$H = 0$	NONE	FLAT	MINIMAL SURFACE
$H > 0$	PIT	VALLEY	SADDLE VALLEY

Figure 8.7: Look-up table for Gaussian and mean curvature sign-map displays

This demonstrates the assertion that sense can be made out of the curvature sign maps which are computed on faithful smooth surface fits on images. This is true even when the images are in the native coordinate system of a scanner (which is conformally warped with respect to the scene's Cartesian dimensions).

Coffee cup

Figure 8.8 shows the 128 x 128 ERIM/ITA scanner image of the coffee cup. The display color map was randomized around to accentuate the elevation contours.

Figure 8.9 shows the curvature sign-maps computed using the 7×7 kernel operator. Although the cylindrical surface of the coffee cup was by and large labelled as a *RIDGE* region, the labelling is very noisy. Furthermore, as will be evident later, this was the best segmentation result obtained by the kernel operators among all the images on which the algorithm was run.

Figure 8.10 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation*. As discussed, the main cylindrical face of the cup was segmented into a central *RIDGE* region flanked by *FLAT* regions. This constitutes the curvature

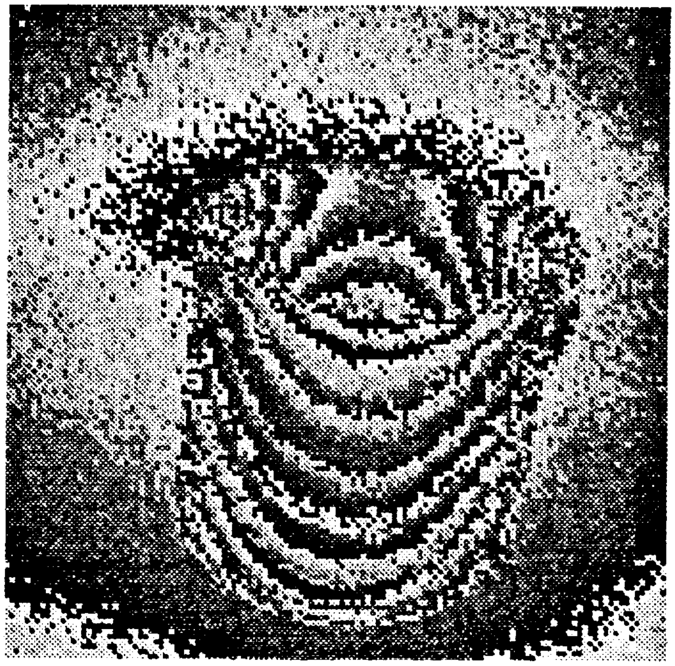


Figure 8.8: ERIM/ITA scanner image of the coffee cup

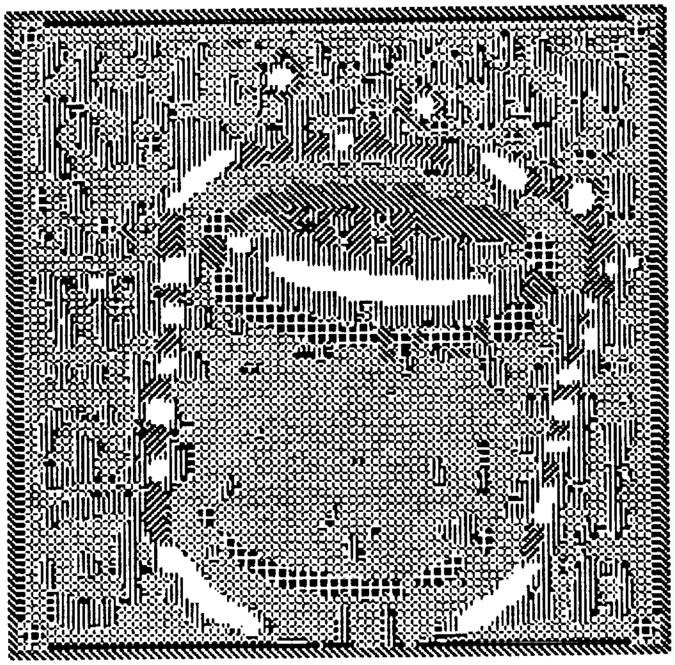


Figure 8.9: Curvature sign-map of the coffee cup obtained using kernel operators

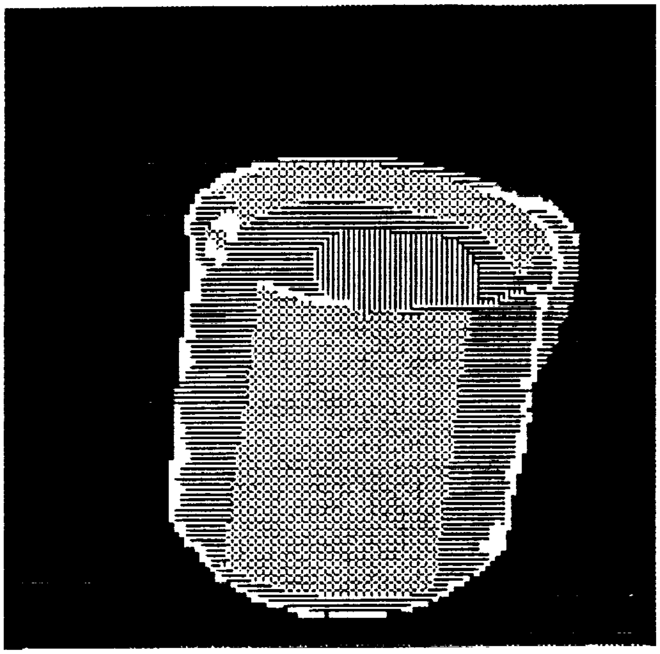


Figure 8.10: Curvature sign-map of the coffee cup computed from the polynomial descriptors extracted by *variable-order segmentation*

sign-map signature of such surfaces. The far-side of the lip of the cup was labelled as *RIDGE* and the inside of the cup was segmented into a *VALLEY* region flanked by *FLAT* regions. At the boundary between the *RIDGE* region describing the lip of the cup and the *VALLEY* region inside, was labelled as *FLAT* (as would be expected). The background plane was greyed (not hatched) to make it easier to see the labelling on the cup.

Space shuttle model

Figure 8.11 shows the 200 x 200 ERIM/ITA scanner image of a model of the space shuttle with the background plane removed. As before, a randomized display color map was used.

Figure 8.12 shows the curvature sign-maps computed using the 7 x 7 kernel operators. It is difficult to make sense of the labelling generated. No planar region, for example was found on the wings or the tail. The digital noise made the kernel-based computation of the surface derivatives unstable.

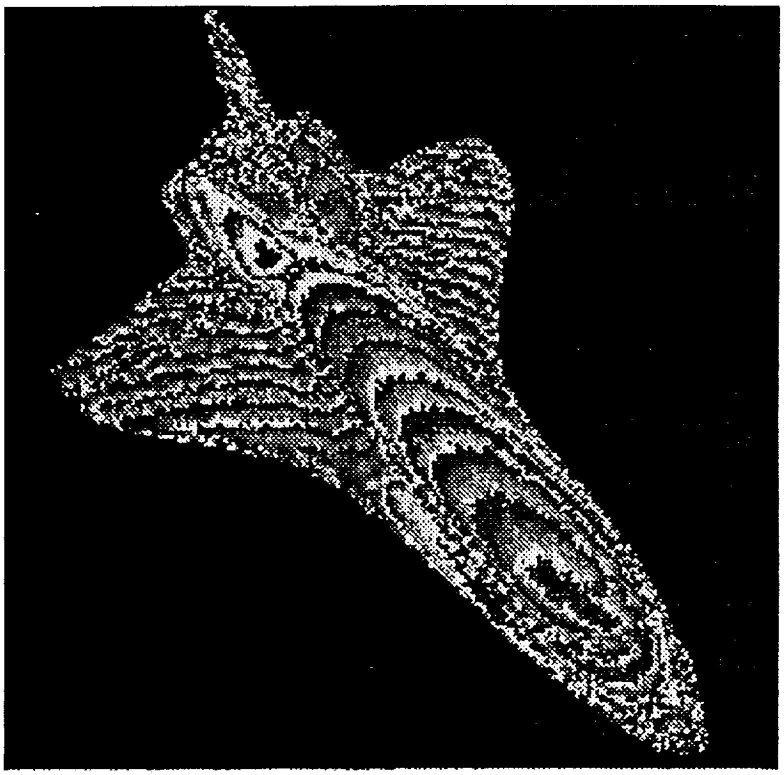


Figure 8.11: ERIM/ITA scanner image of a model of the space shuttle

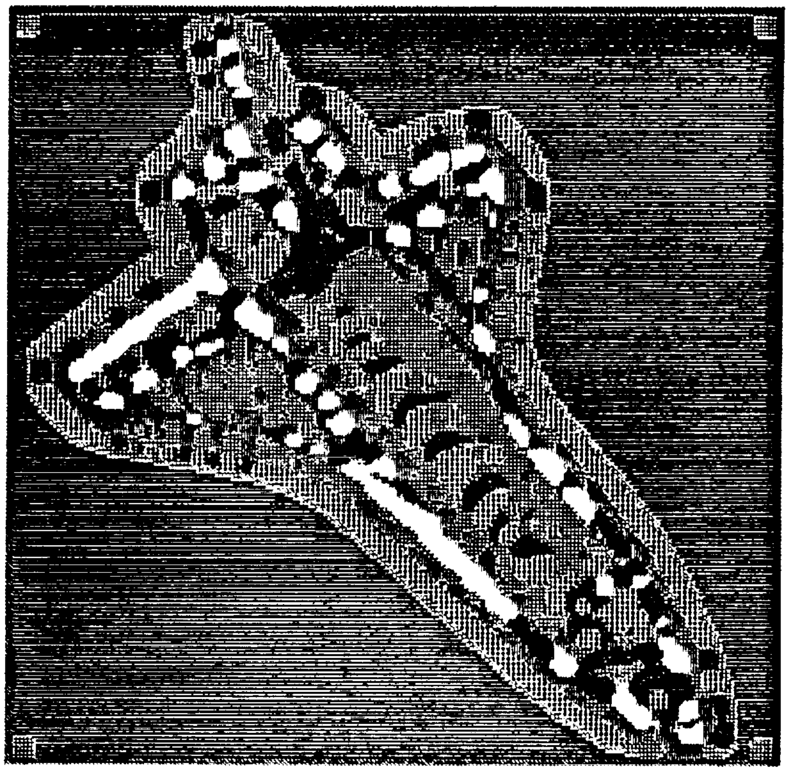


Figure 8.12: Curvature sign-map of the space shuttle model obtained using kernel operators

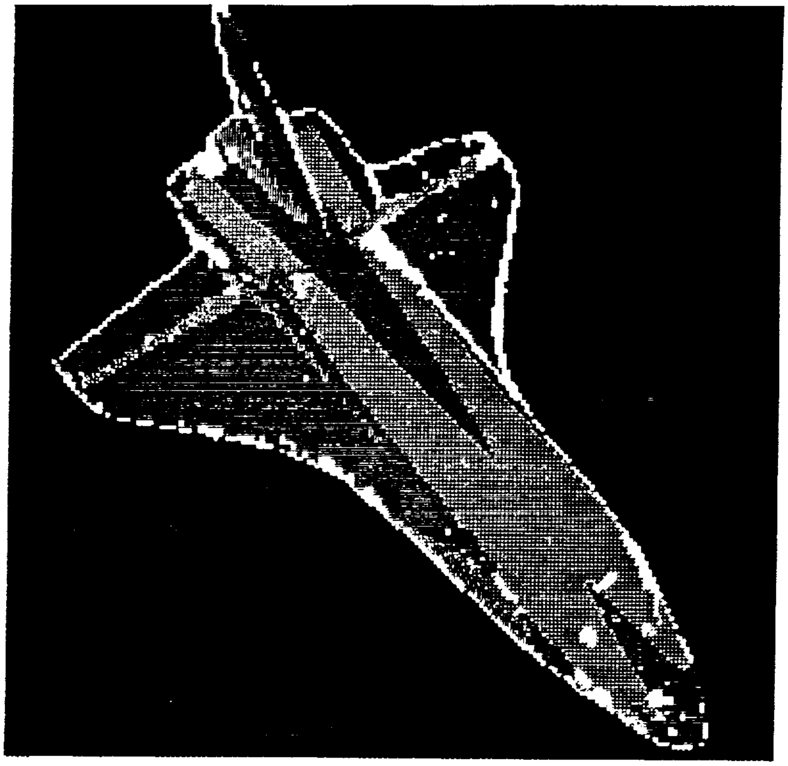


Figure 8.13: Curvature sign-map of the space shuttle model obtained from the polynomial descriptors extracted by *variable-order segmentation*

Figure 8.13 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation.* As can be seen, both wings and the tail were labelled as *FLAT*. Both engines were labelled as *RIDGE* flanked by *FLAT* regions, and the main portion of the fuselage was correctly labelled as *RIDGE*. The trough in front of the tail was identified as *FLAT*, and the boundary region between the tail and the engine was labelled as *VALLEY*.

8.3.2 Synthetic oriented cylinder images

This is a set of two experiments involving synthetically generated cylinders. The purpose of these experiments was to verify the reliability of the cylinder-axis estimates generated by biquadratic functions fitted to the data. To test if there is some comer of the parameter space for which the estimation fails, the cylinders were generated in an assortment of orientations.

In the first experiment, horizontal cylinders of radius 2.5 inches were generated at intervals of $\pi/18$ radians (10°). Biquadratic surfaces were fitted to the cylinders and the orientation of the cylinder axes were estimated from the biquadratic coefficients. In table 8.2, the angles of rotation at which the images were generated were tabulated against the estimated angles and the difference between the angles. Even though the images were created with 10% Gaussian noise content, the largest estimation error was 0.35° at an orientation of 90°. Apart from the 90° and −90° orientations, the average absolute error was 0.0061°.

In the second synthetic images experiment, tilted cylinders of radius 2.5 inches were generated at orientations (θ) of −50°, −30°, 30°, and 60°. The cylinders were tilted at $\phi = -50°, -30°, 30°, 50°$ from the horizontal plane. Gaussian noise of 10% was added to the images. As before, biquadratic surfaces were fitted to the cylinders and the orientation of the cylinder axes were estimated from the biquadratic coefficients. In table 8.3, the angle of rotation at which the images were generated are tabulated against the estimated angles, the difference between the angles and the angles of tilt. The average absolute estimation error for the entire set was 0.0103°. The average absolute estimation error for the ± 30° tilted cylinders was 0.007°, and that for the ± 50° tilted cylinders was 0.0135°. Although the errors for the tilted cylinders were higher than those for horizontal cylinders, and the average error for cylinders tilted at ± 50° was nearly double that for cylinders tilted at ± 30°, the errors in each case was still much smaller than that which would be expected with real sensing errors.

8.3.3 Image of blocks

In this experiment, a collection of four blocks (card-board boxes and wooden blocks) were imaged and processed to extract the surfaces and their orientations. Figure 8.14 is a rendered perspective image of the three-dimensional data obtained from the ERIM/USPS scanner. Notice how noisy the data is especially at the edges of the blocks. As discussed in chapter III, two kinds of errors are evident at these edges. First there are the many obvious spikes which result

θ	$\theta_{estimate}$	$\theta_{estimate} - \theta$
-90.000000	-89.668674	0.331326
-80.001886	-80.025356	-0.023470
-70.003773	-70.000170	0.003602
-59.977011	-59.981953	-0.004942
-50.007545	-50.008177	-0.000632
-39.992455	-39.997938	-0.005483
-30.022989	-30.019435	0.003554
-19.996227	-19.996967	-0.000739
-9.998114	-10.000909	-0.002795
0.000000	0.000000	0.000000
9.998114	9.975731	-0.022382
19.996227	20.000198	0.003971
30.022989	30.020099	-0.002890
39.992455	39.992506	0.000052
50.007545	50.003306	-0.004240
59.999929	60.011458	0.011528
69.998043	69.987020	-0.011024
80.001886	80.003907	0.002021
90.000000	89.649351	-0.350649

Table 8.2: Cylinder axis estimation of synthetically generated oriented horizontal cylinders

θ	$\theta_{estimate}$	$\theta_{estimate} - \theta$	ϕ
-50.007545	-50.009619	-0.002073	-30
	-49.994922	0.012623	-50
	-50.019174	-0.011629	30
	-50.003515	0.004030	50
-30.022989	-30.015716	0.007273	-30
	-30.050071	-0.027082	-50
	-30.025689	-0.002700	30
	-30.012940	0.010049	50
30.022989	30.007681	-0.015308	-30
	30.015209	-0.007780	-50
	30.019095	-0.003894	30
	30.009739	-0.013250	50
59.999929	60.009064	0.009135	-30
	59.984119	-0.015810	-50
	59.004112	0.004182	30
	60.017500	0.017571	50

Table 8.3: Cylinder axis estimation of synthetically generated oriented tilted cylinders

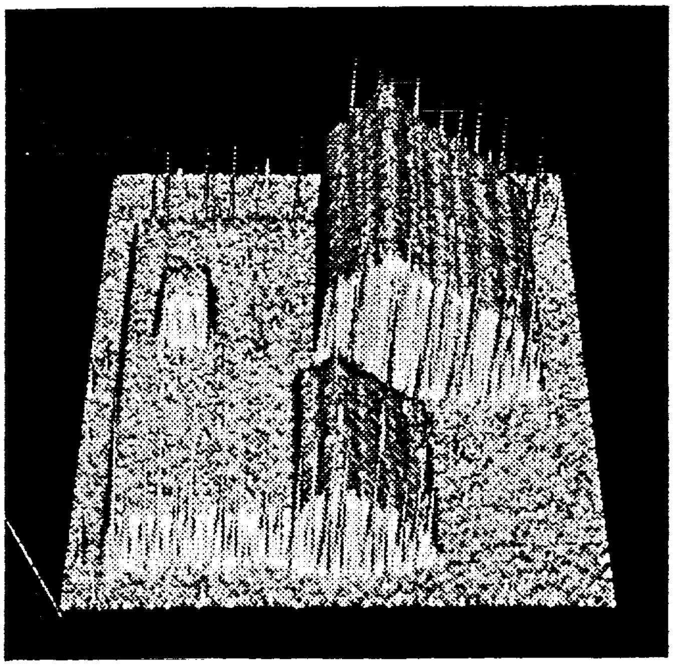


Figure 8.14: A rendered perspective view of the three-dimensional image of the blocks

from beam scatter to the sharp edges (see figure 3.1) and second, there is the averaging noise which makes the data at the edges look like corrugated slopes (see figure 3.2).

Figure 8.15 shows the curvature sign-maps computed using the 7 x 7 kernel operators. Again, the digital noise rendered the kernel-based computation of the surface derivatives ineffectual. No planar regions, for example were found on the block surfaces.

Figure 8.16 shows the smooth regions found by the *variable-order segmentation* program and figure 8.17 shows the order of the bivariate polynomials fitted to the various surfaces.

Figure 8.18 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation.* As can be seen, the majority of the block faces were labelled as *FLAT*. The little block on the left was erroneously labelled as *RIDGE* because the region was too small to obtain an extended region. The edges of one of the block faces near the top were labelled as a *FLAT* region flanked by a *RIDGE* and a *VALLEY* region. Although the background was found to be *FLAT* it was blacked-out in figure 8.18 to make it easier to see the blocks.

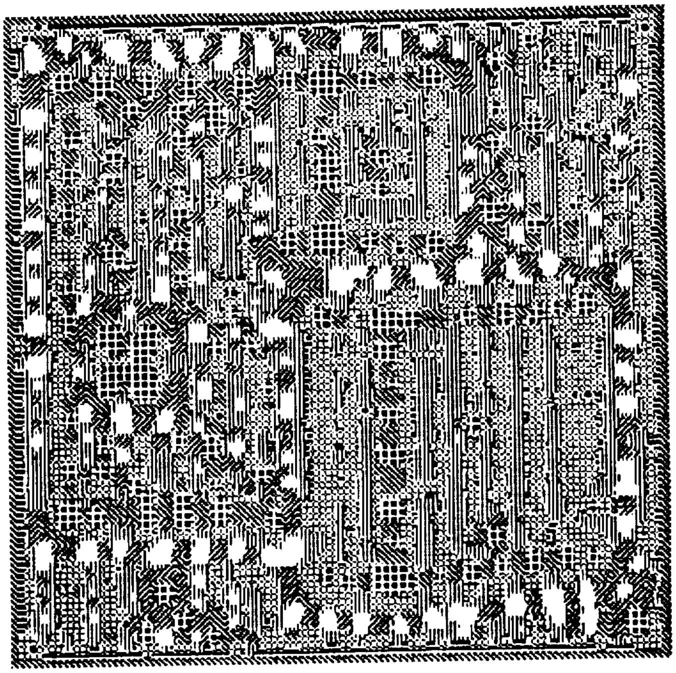


Figure 8.15: .Curvature sign-map of the image of blocks obtained using kernel operators

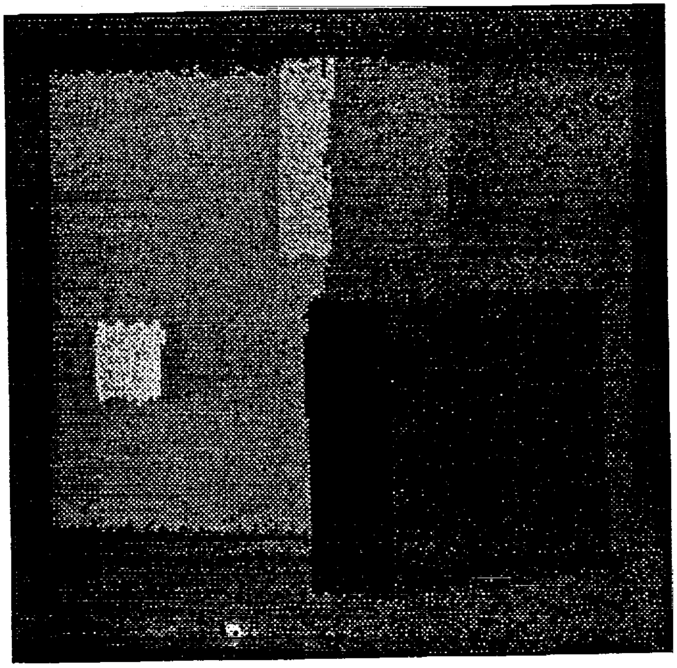


Figure 8.16: Labelled image of smooth surfaces found in the blocks image by the *variable-order segmentation* program

C-3

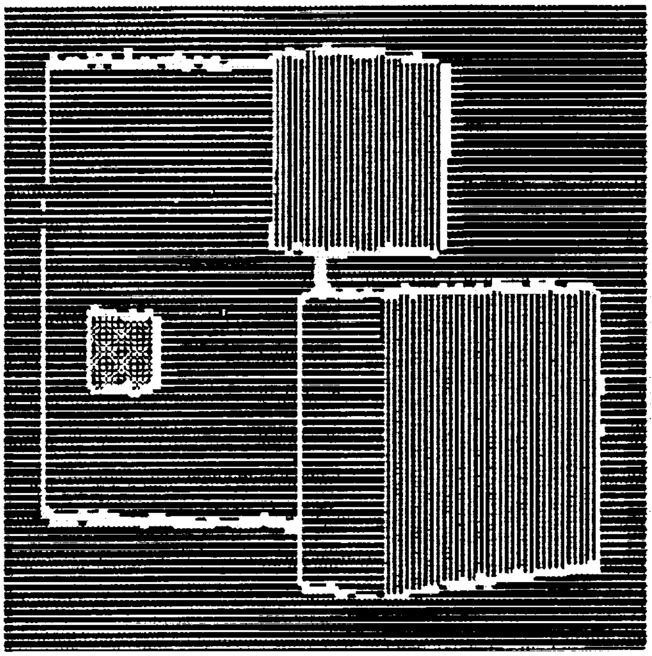


Figure 8.17: The bivariate polynomials fitted to the smooth surfaces in the blocks image by the *variable-order segmentation* program. *KEY: Horizontal lines–Planar; Vertical lines–Bicubic; Crossed '+' hatching–Biquartic*

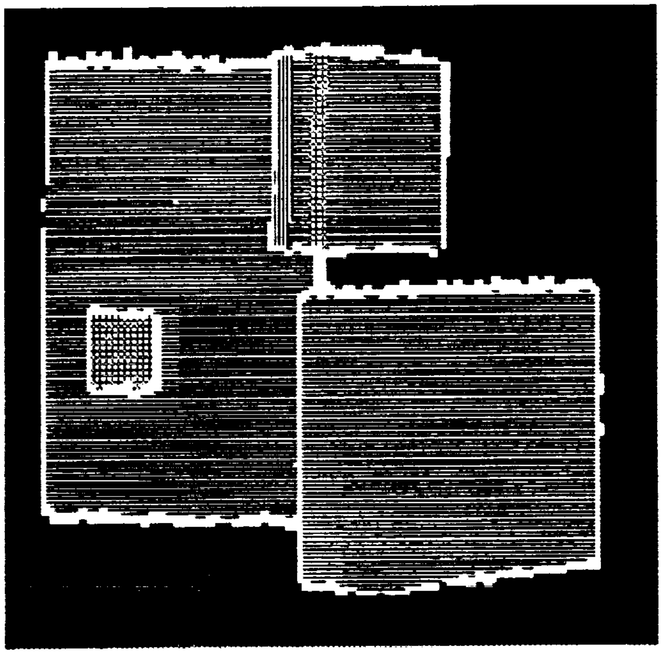


Figure 8.18: Curvature sign-map of the blocks image obtained from the polynomial descriptors extracted by *variable-order segmentation*

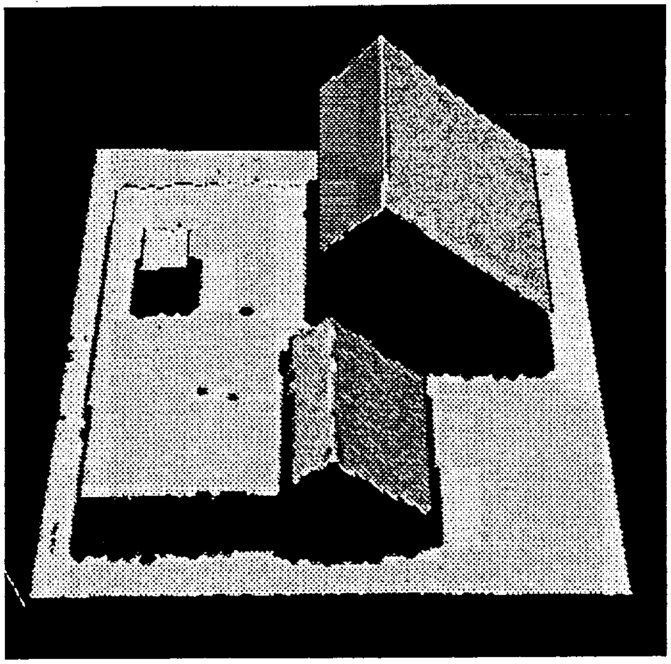


Figure 8.19: A rendered perspective view of the three-dimensional surfaces found in the blocks image

Figure 8.19 is a rendered perspective image of the three-dimensional surfaces found in the blocks image after the hypothesis-verification operation. The angle between the faces of the tilted box at the top of the image was found to be 92.5°. The areas of the surfaces in the image were 1448 and 3528 pixels. The angle between the faces of the tilted box at the bottom of the image was found to be 93.2°. The areas of the surfaces in the image were 1449 and 555 pixels. The angles between the background plane (area was 10571 pixels) and the tops of the two boxes placed squarely on the scanning table were found to be 0.307° (area of box top was 6040 pixels) and 2.46° (area of box top was 250 pixels). As was expected, the errors were lower when the surfaces had more pixels on which to perform the fitting.

8.3.4 Image of a soft drink can

In this experiment, a soft drink can was imaged and processed. Figure 8.20 is a rendered perspective image of the three-dimensional data obtained from the ERIM/USPS scanner. Notice

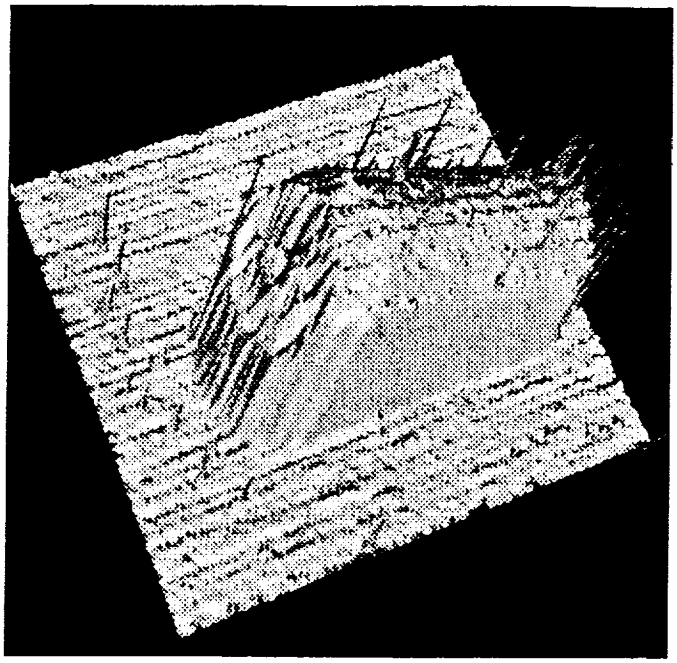


Figure 8.20: A rendered perspective view of the three-dimensional image of the soft drink can

again that the data is especially noisy at the edges of the can.

Figure 8.21 shows the curvature sign-maps computed using the 7 × 7 kernel operators. Again, the curvature labellings bear no resemblance to the surfaces one would expect to find. No *RIDGE* region, for example, was found on the cylindrical surface of the can and the background plane was not labelled as *FLAT*.

Figure 8.22 shows the smooth regions found by the *variable-order segmentation* program and figure 8.23 shows the order of the bivariate polynomials fitted to the various surfaces.

Figure 8.24 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation*. Once again, the curvature description of the cylindrical surface of the can (a *RIDGE* region flanked by *FLAT* regions) is characteristic of cylindrical surfaces. The background plane which was found to be *FLAT* was blacked-out in figure 8.24 to make it easier to see the soft drink can.

Figures 8.25 and 8.26 are rendered perspective images of the three-dimensional surfaces found in the soft drink can image by the hypothesis-verification operation. In figure 8.25, the can is

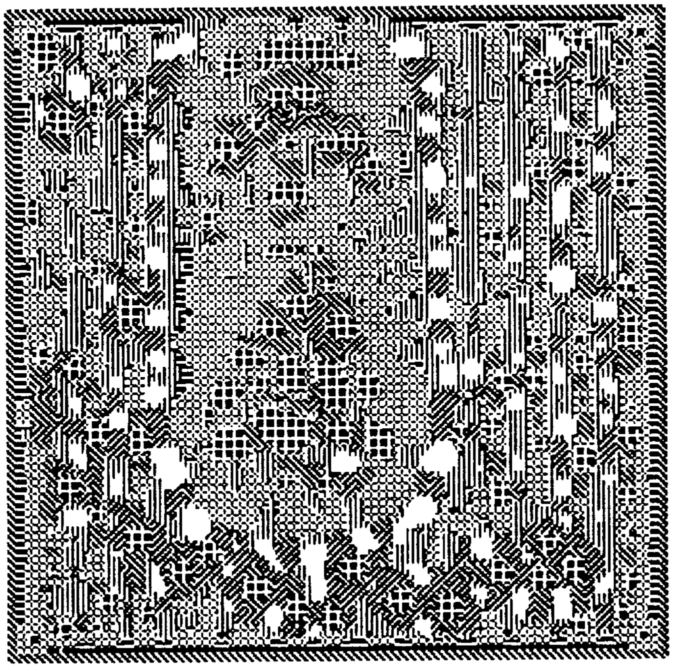


Figure 8.21: Curvature sign-map of the image of soft drink can obtained using kemel operators

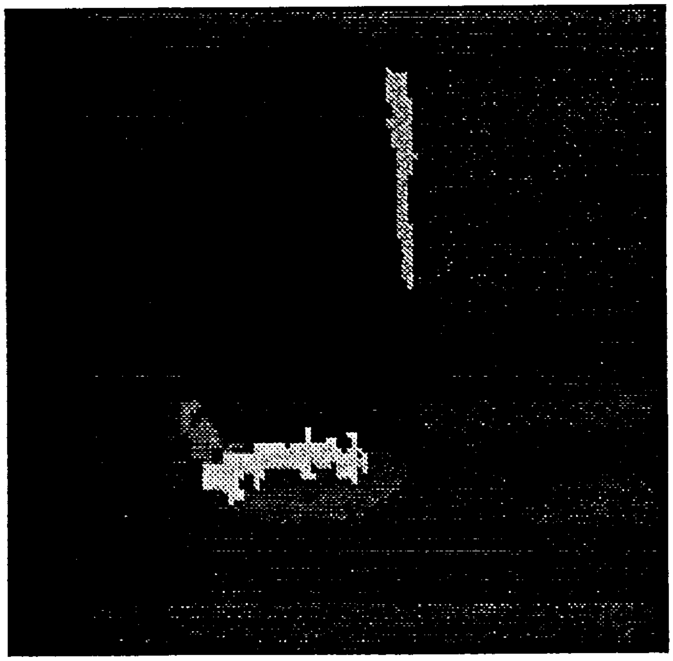


Figure 8.22: Labelled image of smooth surfaces found in the soft drink can image by the *variable-order segmentation* program

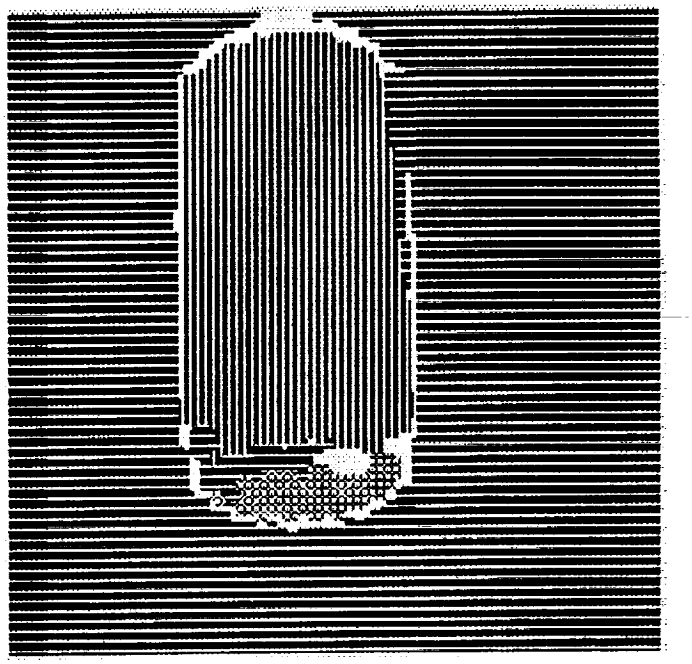


Figure 8.23: The bivariate polynomials fitted to the smooth surfaces in the soft drink image by the *variable-order segmentation* program. *KEY: Horizontal lines–Planar; Vertical lines–Bicubic; Crossed '+' hatching–Biquartic*

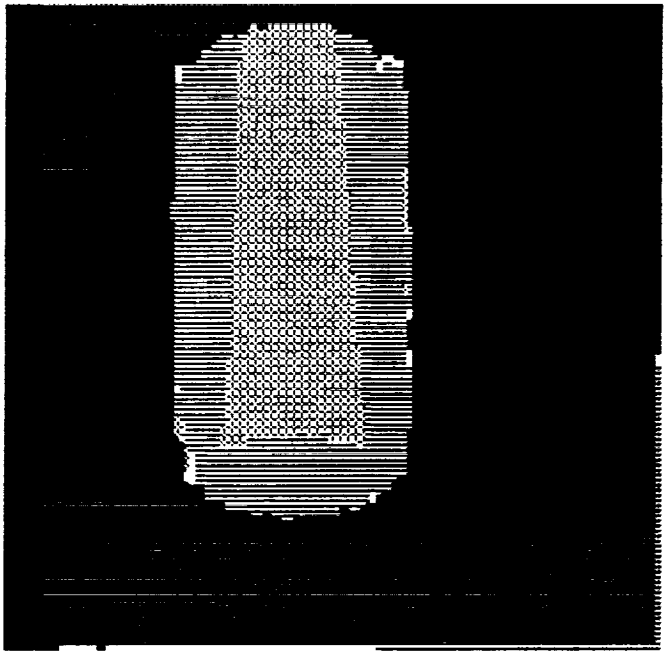


Figure 8.24: Curvature sign-map of the soft drink can image obtained from the polynomial descriptors extracted by *variable-order segmentation*

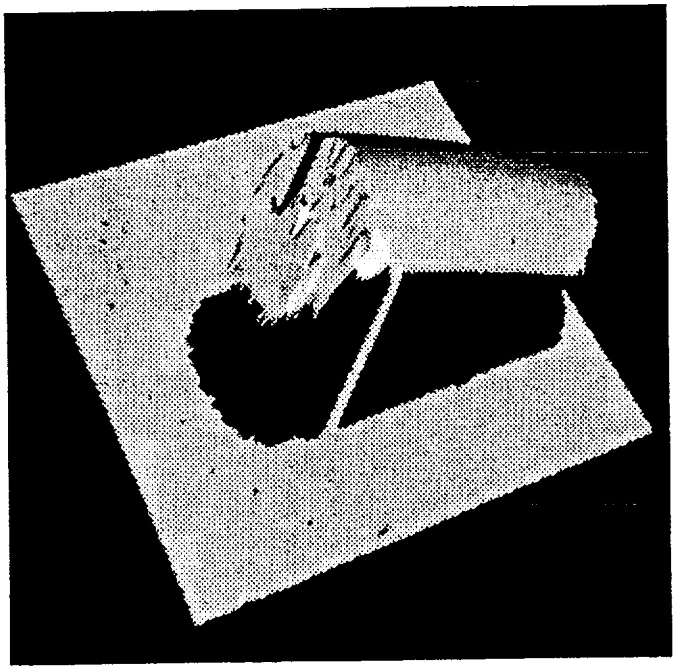


Figure 8.25: A rendered perspective view of the three-dimensional surfaces found in the soft drink can image (the cylinder was approximated by a biquadratic surface)

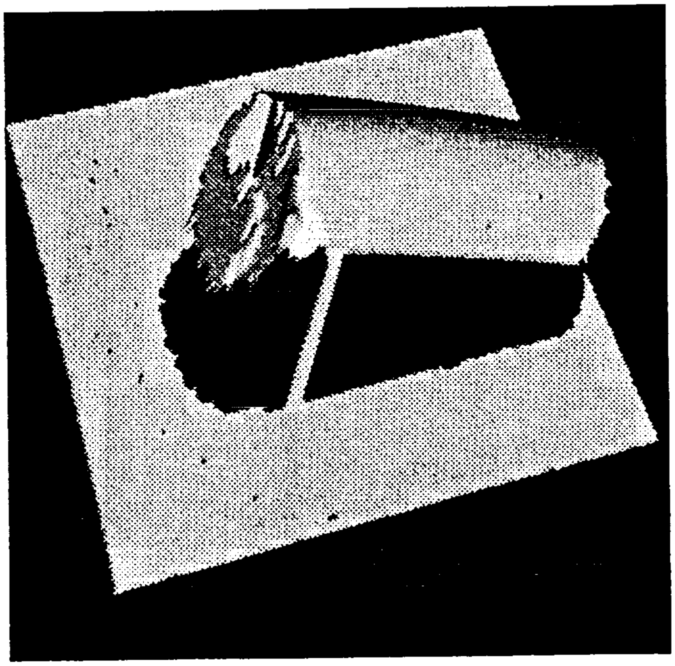


Figure 8.26: A rendered perspective view of the three-dimensional surfaces found in the soft drink can image (the can was generated as a true cylindrical surface)

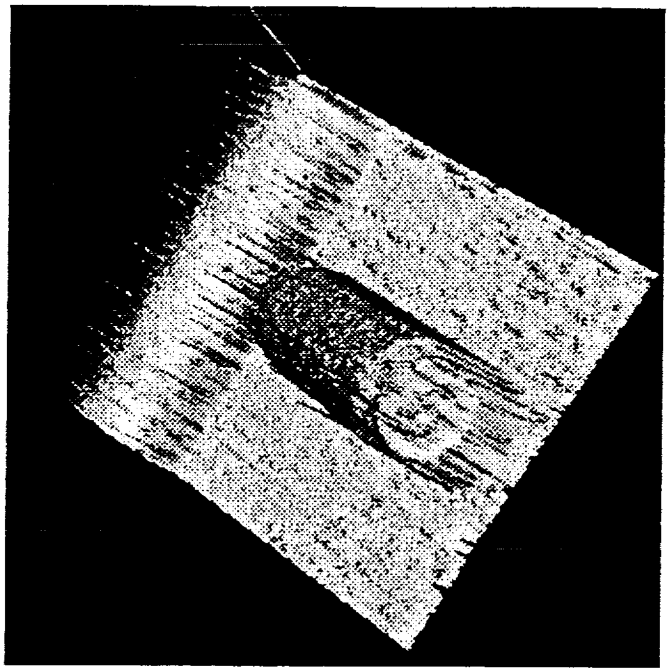


Figure 8.27: A rendered perspective view of the three-dimensional image of the cylinders

approximated by a biquadratic surface (the *companion linear function* for cylinders). Although no appreciable difference can be seen in figure 8.26, the parametric description of the true cylinder were extracted to generate it. The cylinder axis found is described by (θ = 100.65°, ϕ = 26.2°, $c_z = -2.4578$ *inches*, $c_x = 8.1643$ *inches*) (see figure 6.10). The x intercept of the axis with the y-z plane c_x was used because y intercept c_y approaches infinity as the cylinder becomes parallel to the y-axis.

8.3.5 Image of cylinders

In this experiment, a pair of cylinders (soft drink can and a lead pipe) in different orientations were imaged and processed. Figure 8.27 is a rendered perspective image of the three-dimensional data obtained from the ERIM/USPS scanner. Notice again that the data is especially noisy at the edges of discontinuity.

Figure 8.28 shows the curvature sign-maps computed using the 7 x 7 kernel operators. Again, the curvature labellings were fragmented and bear no resemblance to the surfaces one would

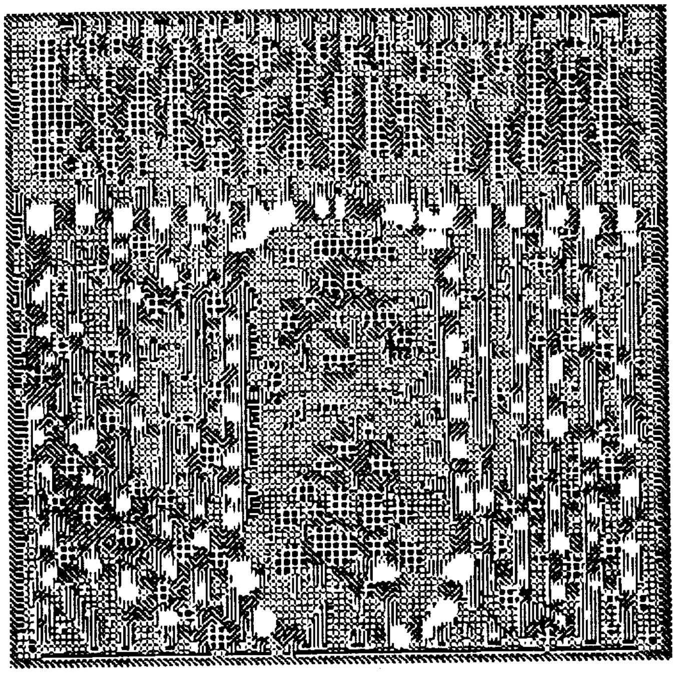


Figure 8.28: Curvature sign-map of the image of cylinders obtained using kernel operators

expect to find. No *RIDGE* regions, for example, were found on the cylindrical surfaces and the background plane was not labelled as *FLAT*.

Figure 8.29 shows the smooth regions found by the *variable-order segmentation* program and figure 8.30 shows the order of the bivariate polynomials fitted to the various surfaces.

Figure 8.31 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation.* Once again, the curvature description of the cylindrical surfaces in the image (a *RIDGE* region flanked by *FLAT* regions) are characteristic of cylindrical surfaces. The background plane which was found to be *FLAT* was blacked-out in figure 8.31 to make it easier to see the objects in the scene.

Figures 8.32 and 8.33 are rendered perspective images of the three-dimensional surfaces found in the cylinders image by the hypothesis-verification operation. In figure 8.32, the cylinders were approximated by a biquadratic surface (the *companion linear function* for cylinders). Figure 8.33, the parametric description of the true cylinders were extracted to generate the image. Again, there is no visual difference the two images. The parameters of the cylinder axis (see figure 6.10)

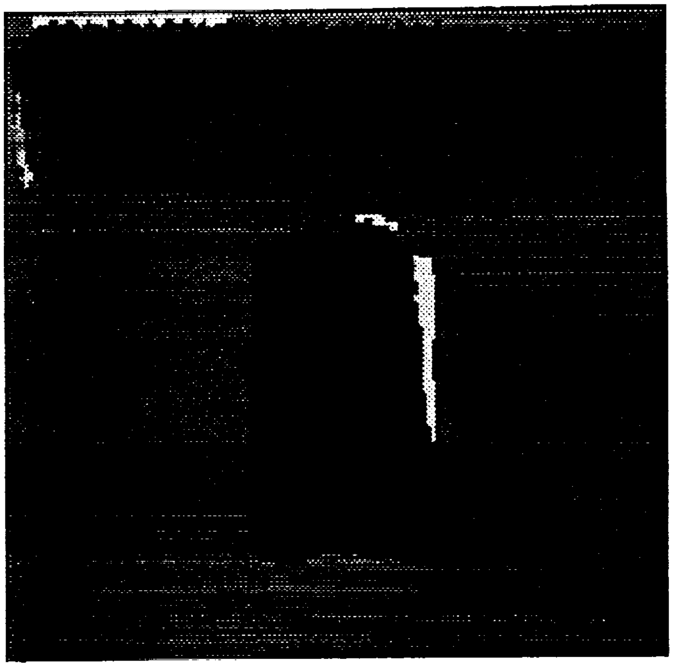


Figure 8.29: Labelled image of smooth surfaces found in the cylinders image by the *variable-order segmentation* program

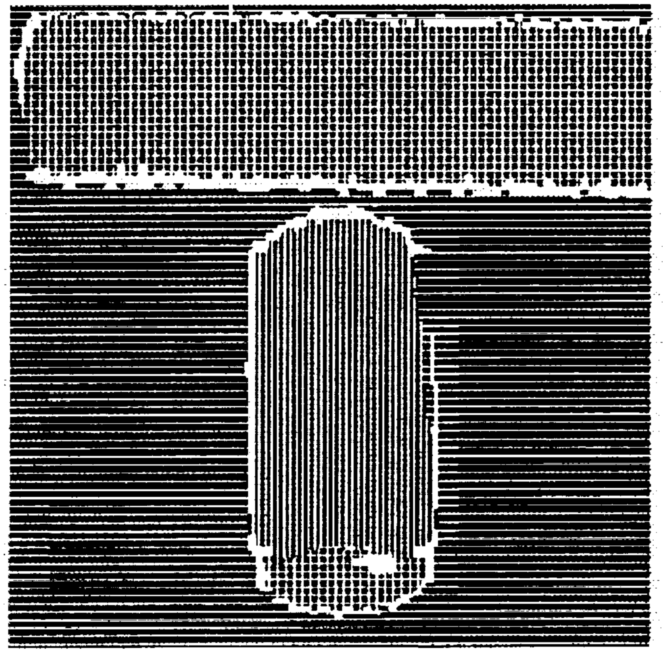


Figure 8.30: The bivariate polynomials fitted to the smooth surfaces in the cylinders image by the *variable-order segmentation* program. *KEY: Horizontal lines–Planar; Vertical lines–Bicubic; Crossed '+' hatching–Biquartic*

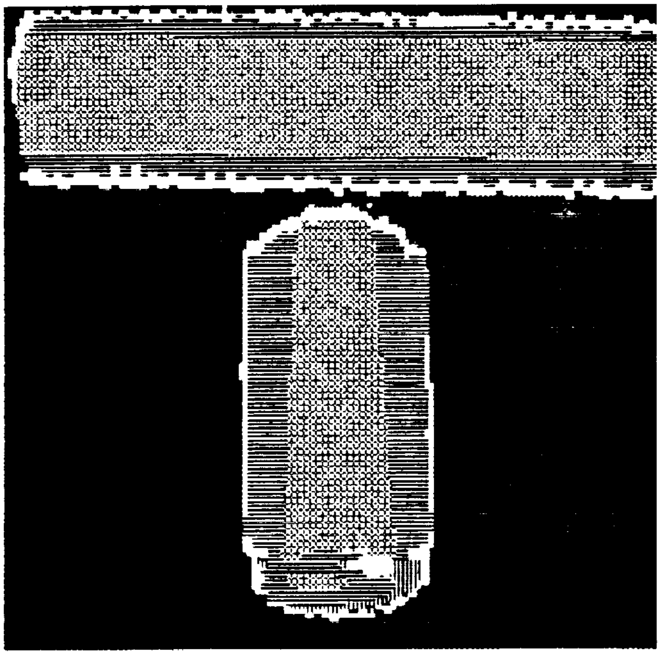


Figure 8.31: Curvature sign-map of the cylinders image obtained from the polynomial descriptors extracted by *variable-order segmentation*

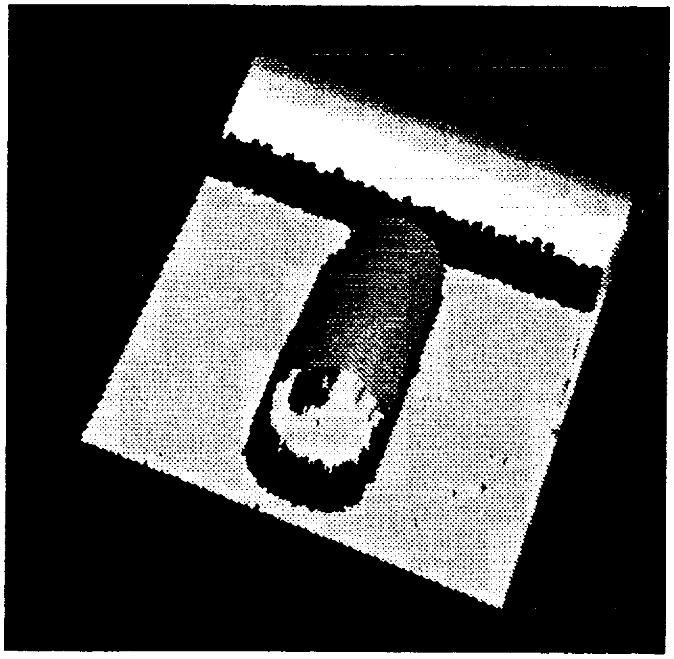


Figure 8.32: A rendered perspective view of the three-dimensional surfaces found in the cylinders image (the cylinders were approximated by biquadratic surfaces)

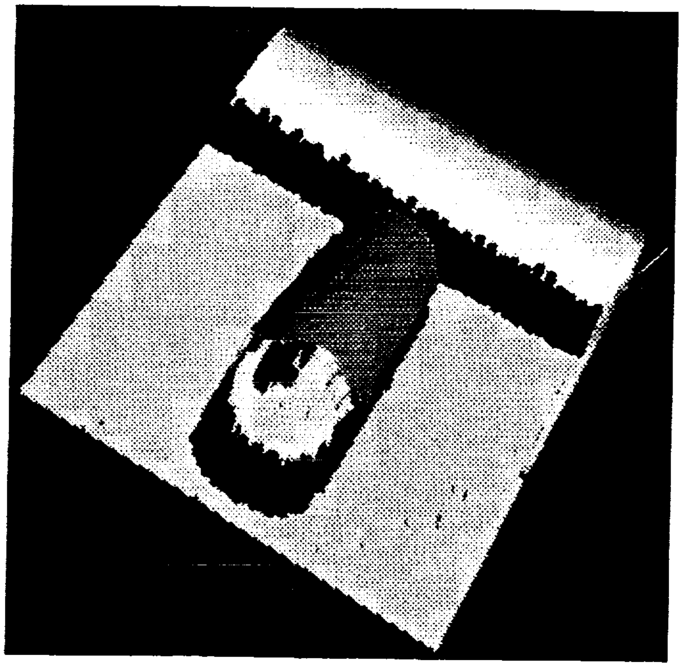


Figure 8.33: A rendered perspective view of the three-dimensional surfaces found in the cylinders image (the cylinders were generated as true cylindrical surfaces)

of the can were found to be ($\theta = 100.65°$, $\phi = 26.2°$, $c_z = -2.4578$ *inches*, $c_x = 8.1643$ *inches*) (c_x is the x intercept of the axis with the y-z plane – the y intercept c_y approaches infinity as the cylinder is parallel to the y-axis) ; and, those for the lead pipe were ($\theta = 1.48°$, $\phi = -0.467°$, $c_z = 3.6044$ *inches*, $c_y = 9.735$ *inches*).

8.3.6 Image of blocks and cylinder

In this experiment, a collection of three blocks and a cylindrical lead pipe was imaged and processed. Two of the blocks are placed one on another squarely on the scanning table. The third block was tilted so that two surfaces were visible. Figure 8.34 is a rendered perspective image of the three-dimensional data obtained from the ERIM/USPS scanner. The usual noise at the range discontinuities is again apparent.

Figure 8.35 shows the curvature sign-maps computed using the 7 x 7 kernel operators. Again, the curvature labellings were of little use for surface recognition. There was no perceptible

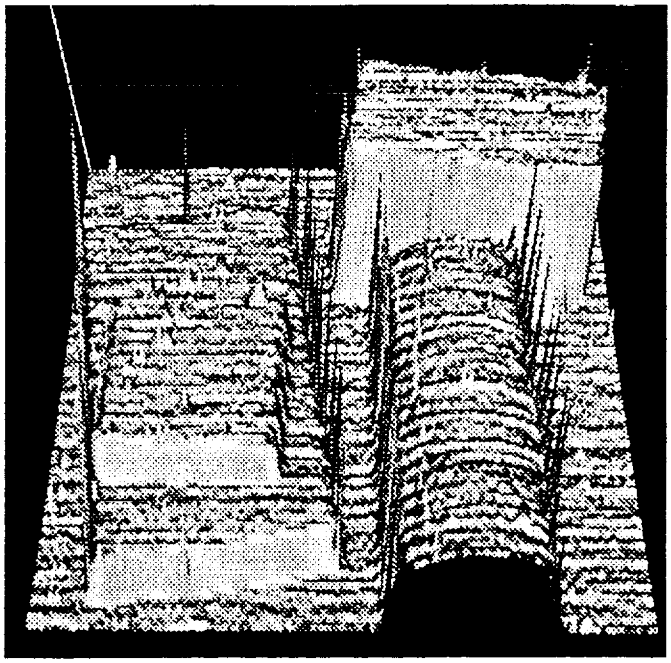


Figure 8.34: A rendered perspective view of the three-dimensional image of the blocks and cylinder

correlation between the sign-map labels and the surfaces known to be present.

Figure 8.36 shows the smooth regions found by the *variable-order segmentation* program and figure 8.37 shows the order of the bivariate polynomials fitted to the various surfaces.

Figure 8.38 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation.* Once again, the characteristic curvature signature of the cylindrical surface (a *RIDGE* region flanked by *FLAT* regions) was observed. The surfaces of the blocks and the background plane were found to be *FLAT.*

Figures 8.39 and 8.40 are rendered perspective images of the three-dimensional surfaces found in the blocks/cylinder image by the hypothesis-verification operation. In figure 8.39, the cylinder was approximated by a biquadratic surface (the *companion linear function* for cylinders). Figure 8.40, the parametric description of the true cylinder was extracted to generate it. Again, there is no visual difference the two images. The parameters of the cylinder axis (see figure 6.10) of the can were ($\theta = 1.39°$, $\phi = 0.71°$, $c_z = 3.563$ *inches*, $c_y = 13.4989$ *inches*). The angles

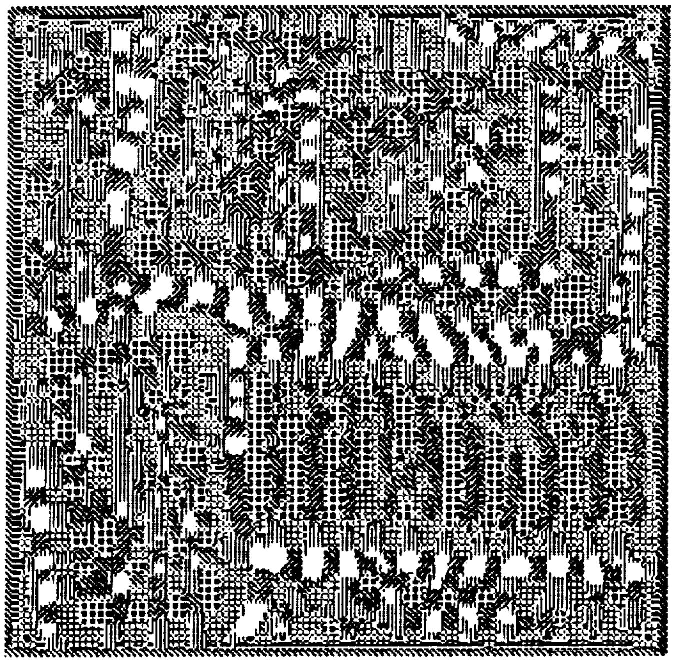


Figure 8.35: Curvature sign-map of the blocks/cylinder image obtained using kemel operators

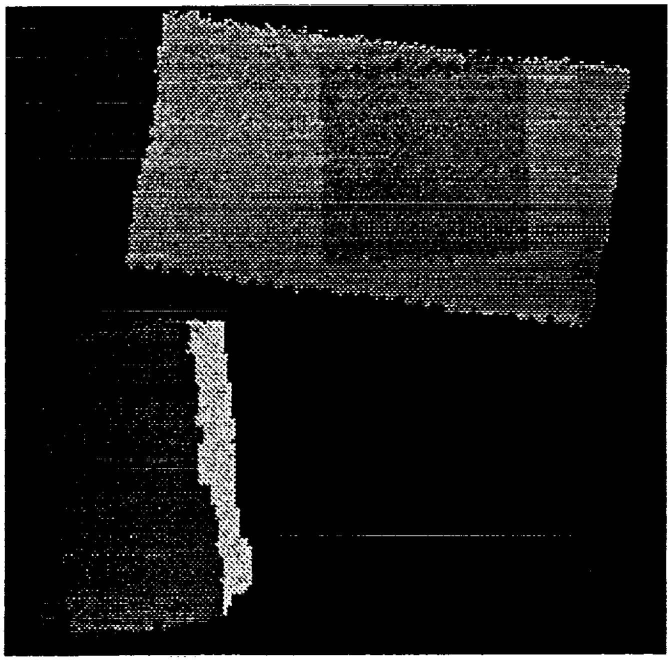


Figure 8.36: Labelled image of smooth surfaces found in the blocks/cylinder image by the *variable-order segmentation* program

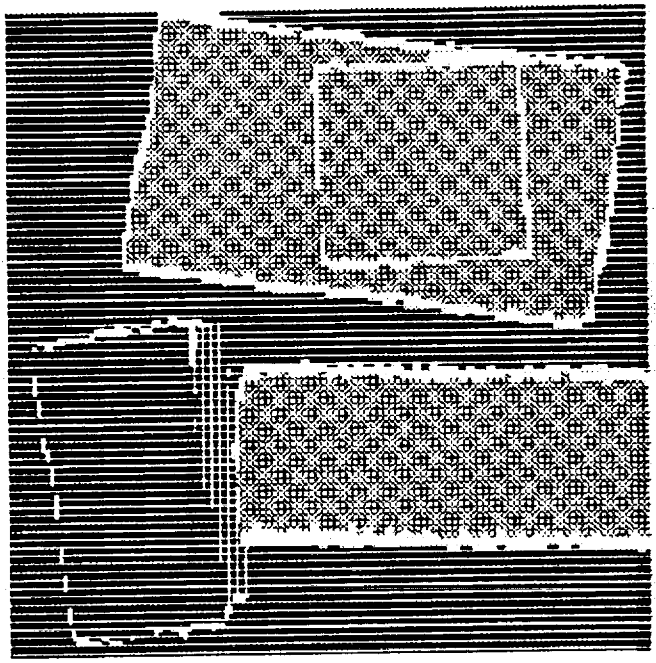


Figure 8.37: The bivariate polynomials fitted to the smooth surfaces in the blocks/cylinder image by the *variable-order segmentation* program. *KEY: Horizontal lines–Planar; Vertical lines–Bicubic; Crossed '+' hatching–Biquartic*

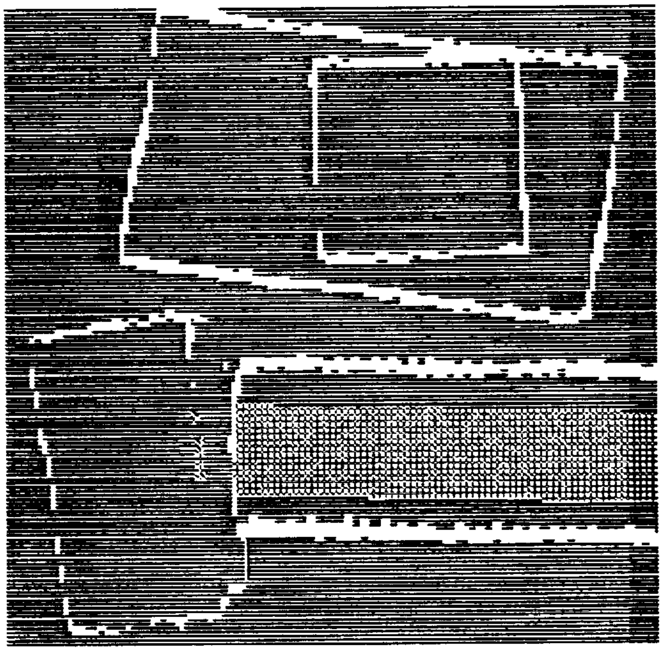


Figure 8.38: Curvature sign-map of the blocks/cylinder image obtained from the polynomial descriptors extracted by *variable-order segmentation*

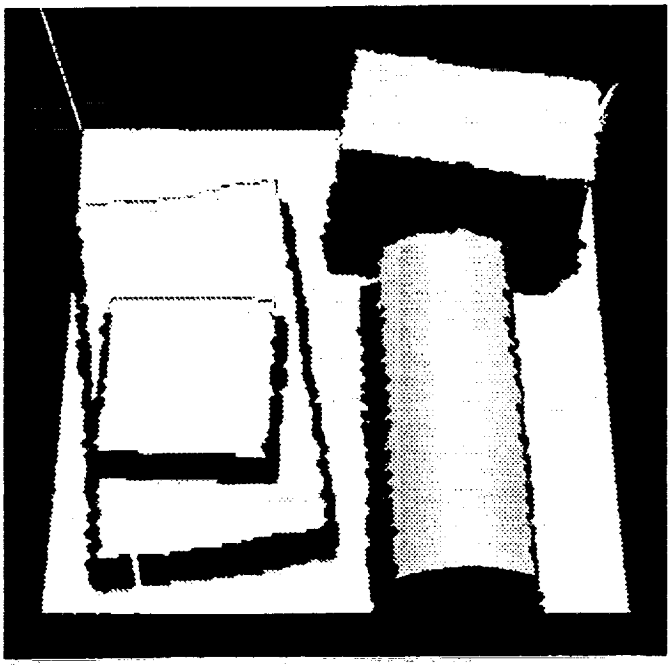


Figure 8.39: A rendered perspective view of the three-dimensional surfaces found in the blocks/cylinder image (the cylinder was approximated by a biquadratic surface)

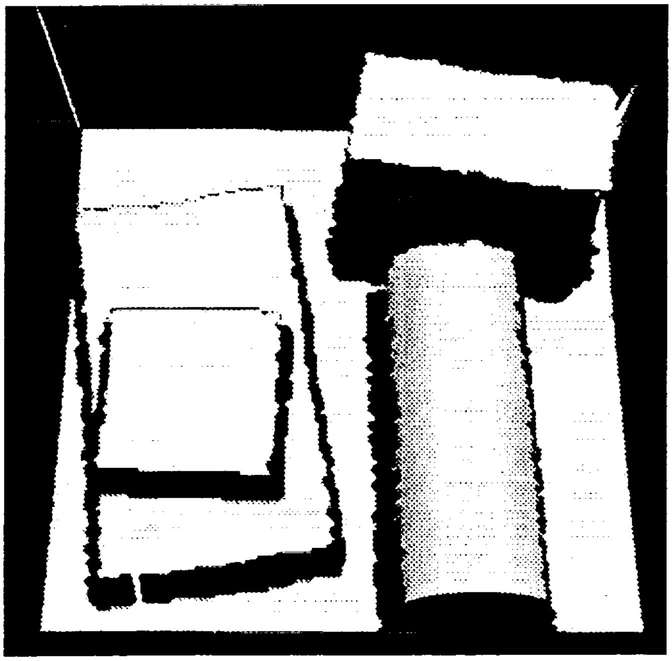


Figure 8.40: A rendered perspective view of the three-dimensional surfaces found in the image of blocks and a cylinder (the cylinder was generated as a true cylindrical surface)

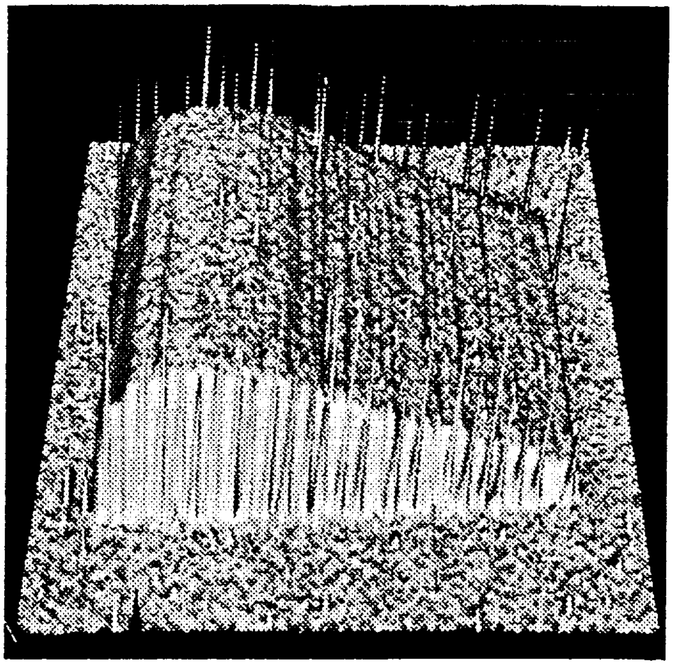


Figure 8.41:. A rendered perspective view of the three-dimensional image of the chamfered plane

between the background plane (area was 9502 pixels) and the tops of the two boxes whose bases are parallel to the scanning table were found to be 0.944° (area of box top was 2280 pixels) and 0.375° (area of box top was 4305 pixels). Again, the errors were lower when the surfaces had more pixels on which to perform the fitting. The steeper face of the tilted block was too noisy to make a good planar fit.

8.3.7 Image of a chamfered plane

In this experiment, an aluminum sheet bent to produce a plane chamfering smoothly into a curved surface was imaged and processed. This is precisely the problem discussed in section 5.8 and shown in figure 5.4. Figure 8.41 is a rendered perspective image of the three-dimensional data using the ERIM/USPS scanner.

Figure 8.42 shows the curvature sign-maps computed using the 7×7 kernel operators. Again, the curvature labellings were of little use for surface recognition. There was no perceptible difference between the labelling of the planar and curved portions of the object.

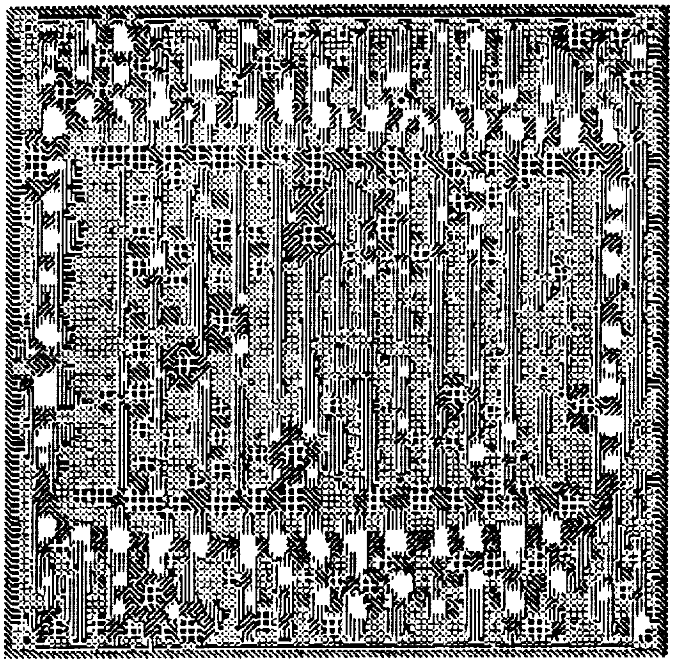


Figure 8.42: Curvature sign-map of the chamfered plane image obtained using kernel operators

Figure 8.43 shows the smooth regions found by the *variable-order segmentation* program and figure 8.44 shows the order of the bivariate polynomials fitted to the various surfaces.

Figure 8.45 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation*. The planar portion was labelled as *FLAT*. Most of the curved surface was correctly labelled as a *RIDGE* region. A small portion of the curved surface was erroneously labelled as *FLAT*.

To separate the plane from the curved chamfer (and so determine its orientation in three-dimensional space), the split-merge algorithm for plane extraction was applied. As discussed in section 5.8, the splitting was performed by applying equation 5.6 to generate the *like-normal neighbourhood* image shown in figure 8.46.

The surfaces were then merged using the merging algorithm described in section 5.5. The resulting segmentation is shown in figure 8.47 in which each surface was labelled with a unique shade. The small patch at the lower right hand corner could not be fit to the plane because the original aluminium sheet had a crease at that corner.

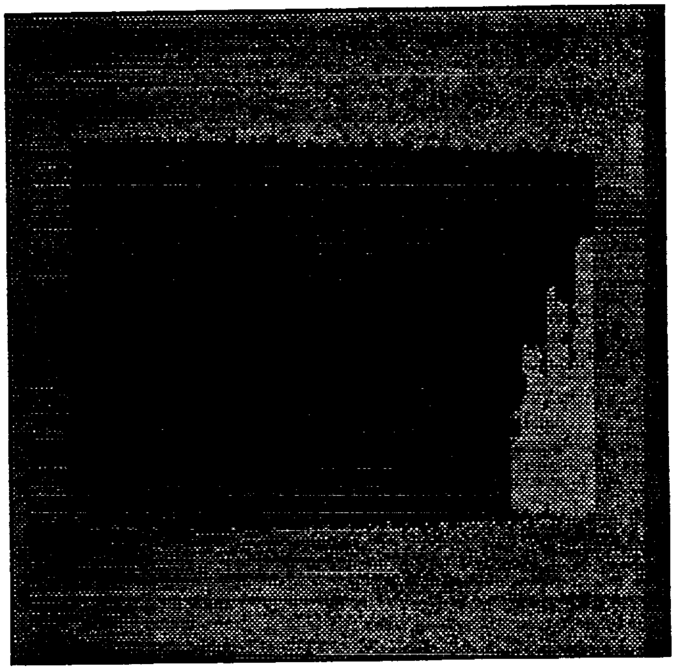


Figure 8.43: Labelled image of smooth surfaces found in the chamfered plane image by the *variable-order segmentation* program

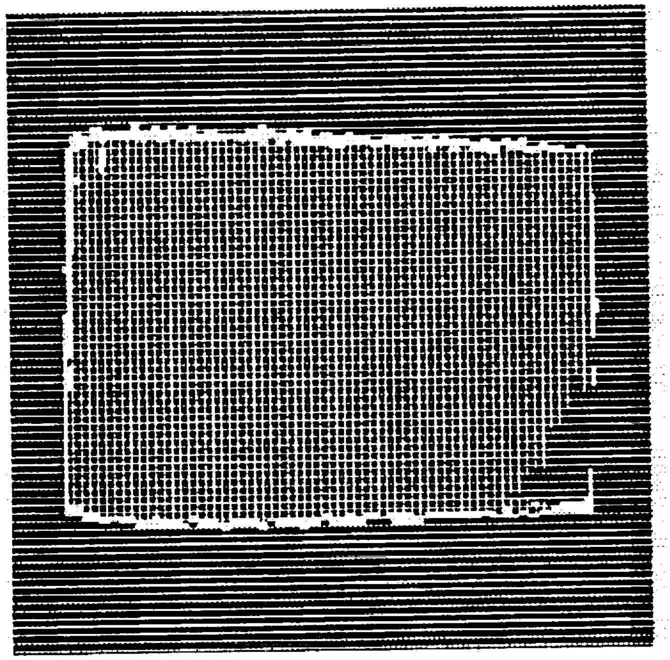


Figure 8.44: The bivariate polynomials fitted to the smooth surfaces in the chamfered plane image by the *variable-order segmentation* program. *KEY: Horizontal lines–Planar; Crossed '+' hatching–Biquartic*

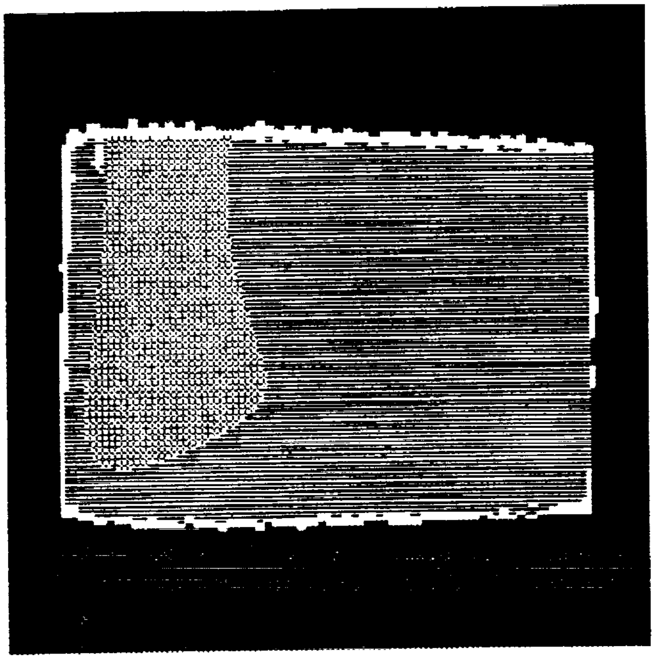


Figure 8.45: Curyature sign-map of the chamfered plane image obtained from the polynomial descriptors extracted by *variable-order segmentation*

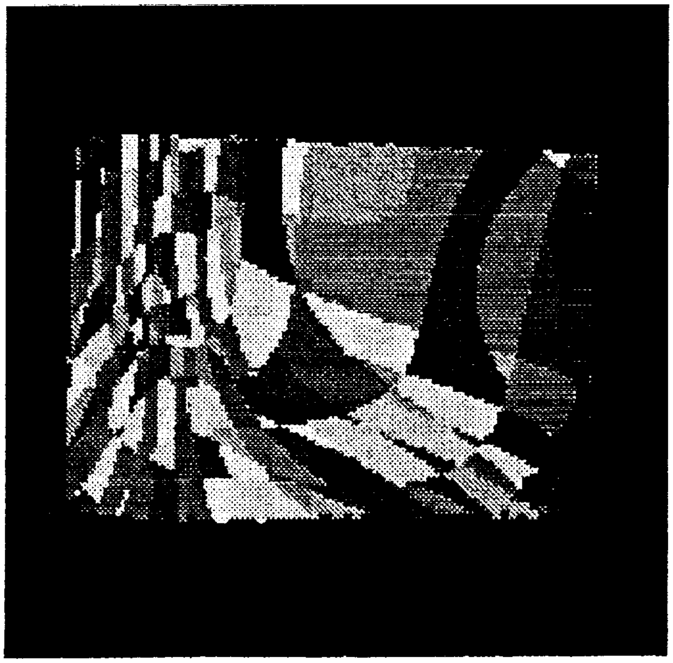


Figure 8.46: The *like-normal neighbourhood* image for the chamfered plane

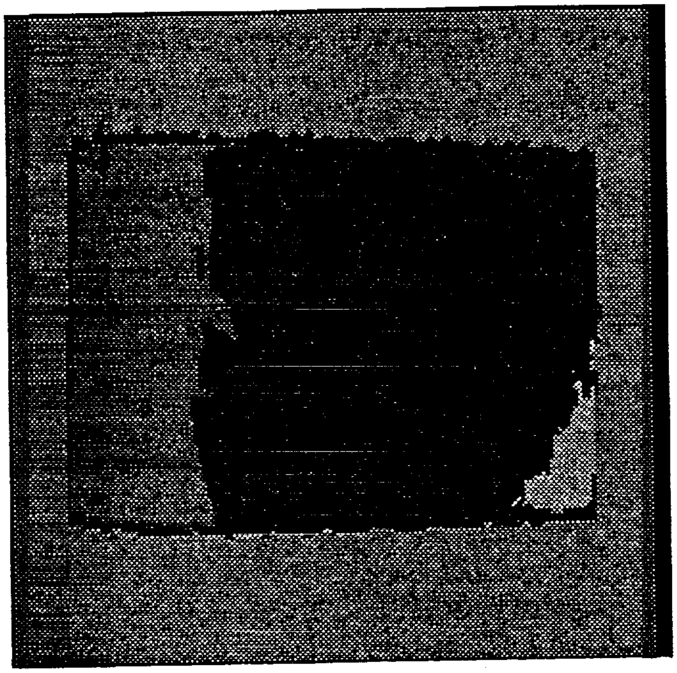


Figure 8.47: The final segmentation for the chamfered plane

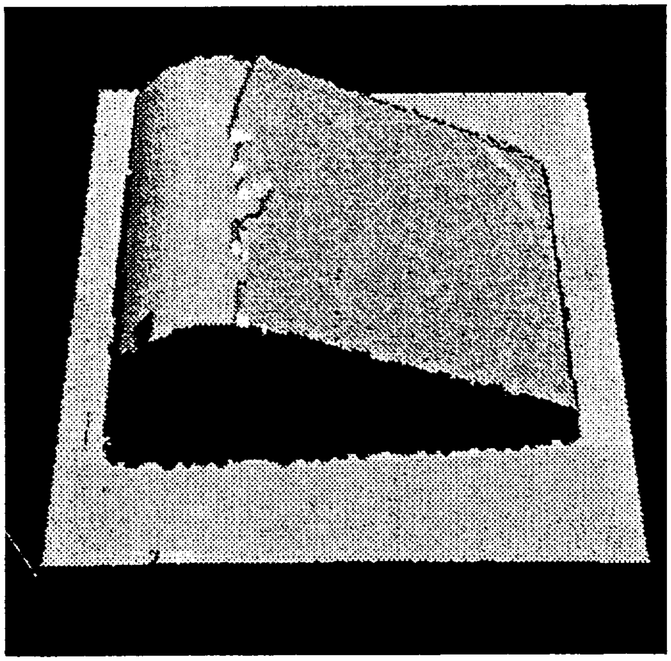


Figure 8.48: A rendered perspective view of the three-dimensional surfaces found in the chamfered plane image.

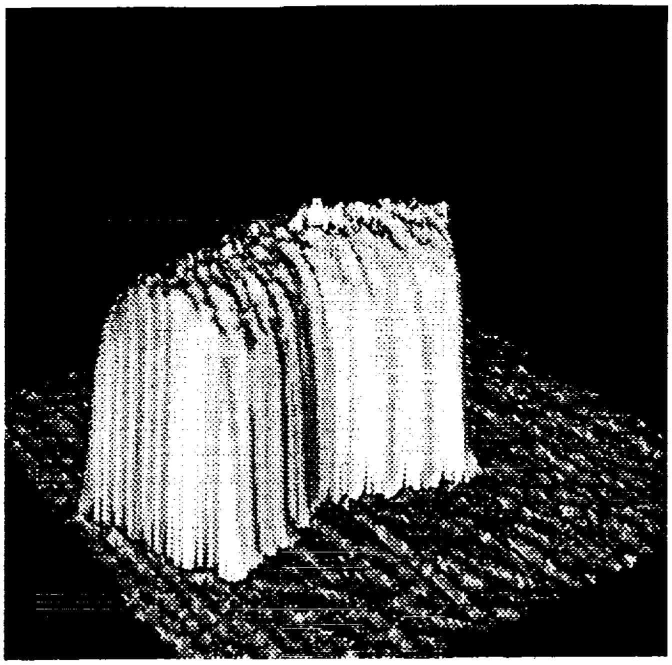


Figure 8.49: A rendered perspective view of the three-dimensional image of the pipe elbow

Figure 8.48 is a rendered perspective image of the three-dimensional surfaces found in the chamfered plane image by the hypothesis-verification operation. The plane actually grew slightly into the curved region. This could not be avoided. If the threshold for merging regions was made too sensitive, the noise in the image would have stopped the region merging before the entire plane was captured.

8.3.8 Image of a pipe elbow

In this experiment, fiberglass pipe elbow comprising two straight cylindrical pipe segments merging together with a gradual 135° bend smoothly into a curved surface was imaged and processed. To extract the parametric description of the two cylinders, the merge-split algorithm for cylindrical surface recovery described in section 5.9 was applied. Figure 8.49 is a rendered perspective image of the three-dimensional data using the ERIM/USPS scanner. The image is much noisier than the preceding ones because of the fiberglass material is translucent to the laser radiation and has specular characteristics.

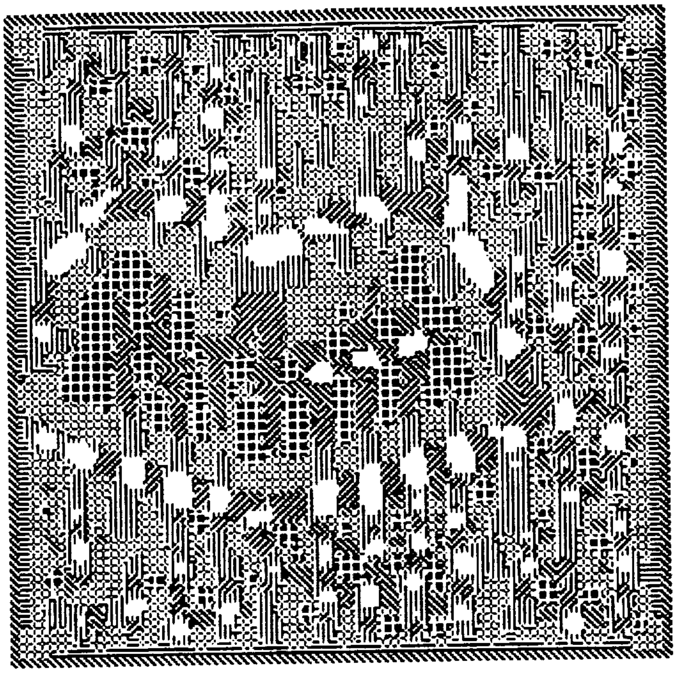


Figure 8.50: Curvature sign-map of the pipe elbow image obtained using kernel operators

Figure 8.50 shows the curvature sign-maps computed using the 7×7 kernel operators. Again, little can be made of the curvature label thus computed.

Figure 8.51 shows the smooth regions found by the *variable-order segmentation* program. All the surfaces found were approximated by biquartic functions. Figure 8.52 shows a rendered three-dimensional reconstruction of the surfaces found. Notice that the entire pipe segment was approximated by one surface, and that the surface is warped (not cylindrical).

Figure 8.53 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation.* The curvature description of the straight segments of the pipe (a *RIDGE* region flanked by *FLAT* regions) is characteristic of cylindrical surfaces. At the inside of the bend, *VALLEY* and *SADDLE VALLEY* regions were found. These describe precisely the types of surface curves actually present. The background plane which was found to be *FLAT* was blacked-out in figure 8.31 to make it easier to see the pipe elbow.

To separate the cylinders and to determine their orientations in three-dimensional space, the split-merge algorithm for cylinder extraction was employed. As discussed in section 5.9,

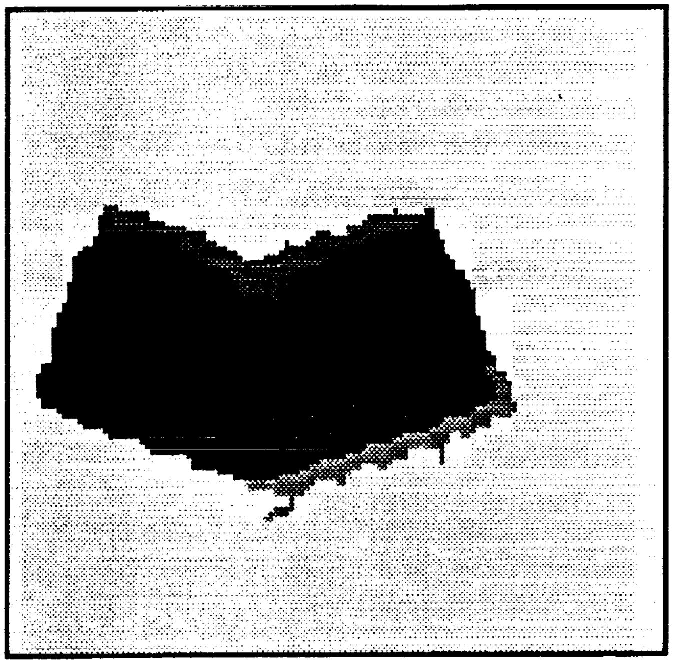


Figure 8.51: Labelled image of smooth surfaces found in the bent pipe segment image by the *variable-order segmentation* program

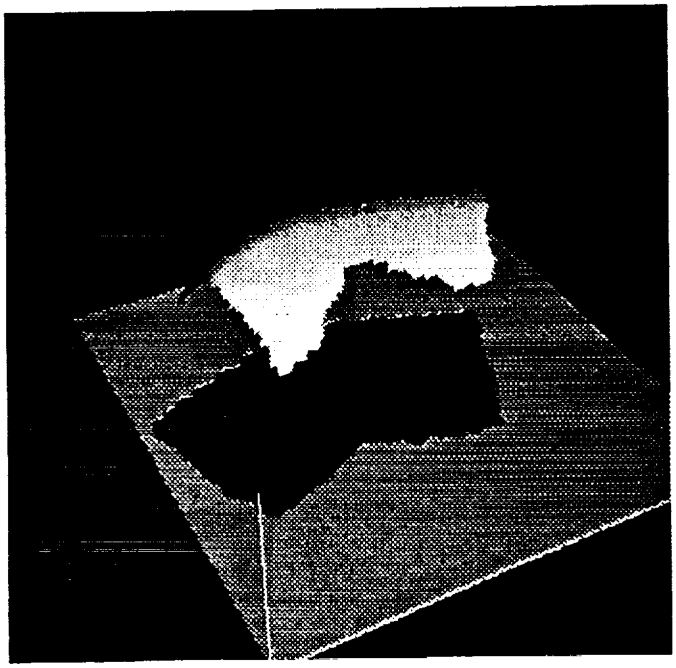


Figure 8.52: The biquartic surfaces fitted to the smooth surfaces in the pipe elbow image by the *variable-order segmentation* program.

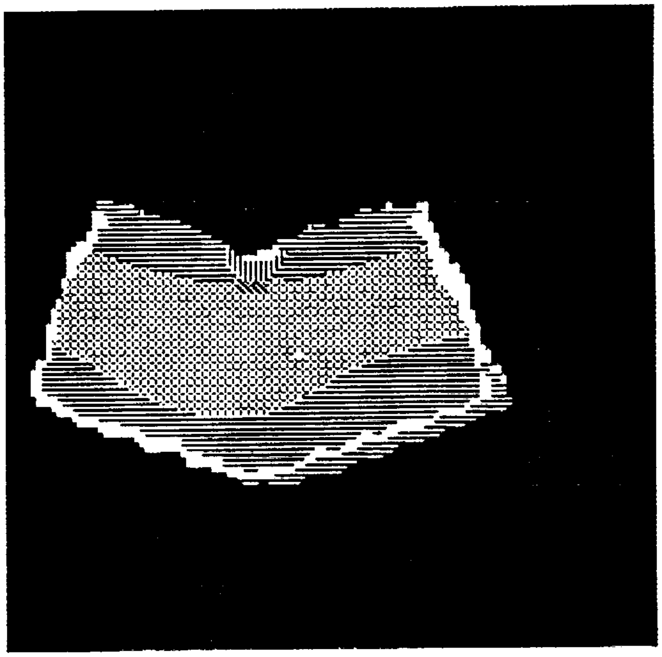


Figure 8.53: Curvature sign-map of the pipe elbow image obtained from the polynomial descriptors extracted by *variable-order segmentation*

the entire elbow was divided into *acceleration bands* (see figure 5.6). The surfaces were then merged using the merging algorithm described in section 5.5. The *acceleration bands* and the resulting segmentation are shown in figure 8.54.

Figures 8.55 and 8.56 are rendered perspective images of the three-dimensional surfaces found in the pipe elbow image by the hypothesis-verification operation. In figure 8.55, the cylinders were approximated by a biquadratic surface (the *companion linear function* for cylinders). Figure 8.33, the parametric description of the true cylinders were extracted to generate the image. The parameters of the cylinder axes (see figure 6.10) were found to be ($\theta = 157.78°$ or $-22.22°$, $\phi = -2.05°$, $c_z = -5.268$ *inches*, $c_y = 17.573$ *inches*) and ($\theta = 23.999°$, $\phi = 2.038°$, $c_z = 4.482$ *inches*, $c_y = 8.579$ *inches*). This constitutes a bend of 133.78°. This differs from the 135° of the actual pipe by 1.22°.

8.3.9 Image of PVC pipe ring

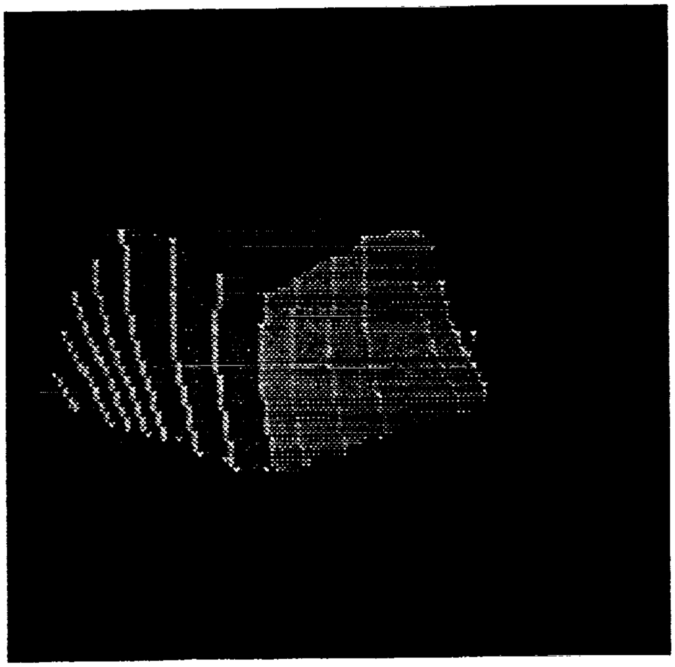


Figure 8.54: The *acceleration-band* image for the pipe elbow

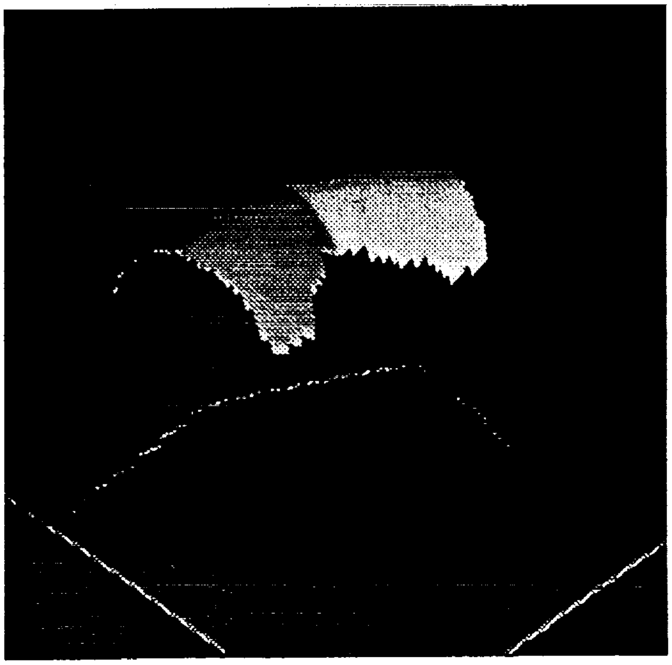


Figure 8.55: A rendered perspective view of the three-dimensional surfaces found in the pipe elbow image (the cylinders were approximated by biquadratic surfaces).

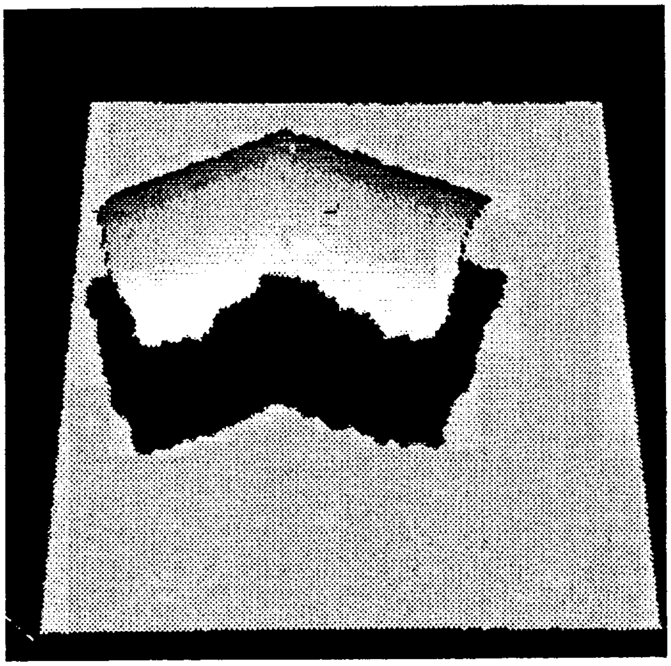


Figure 8.56: A rendered perspective view of the three-dimensional surfaces found in the pipe elbow image (the cylinders were generated as true cylindrical surfaces).

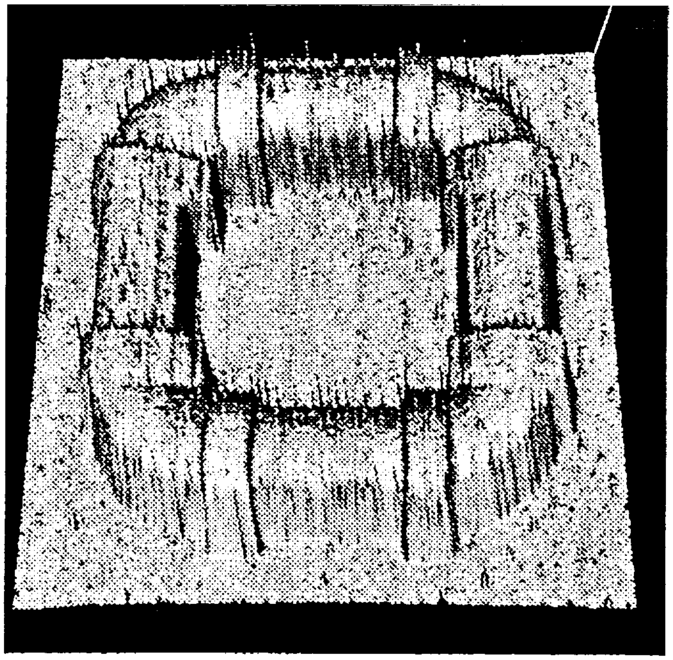


Figure 8.57: A rendered perspective view of the three-dimensional image of the PVC pipe ring

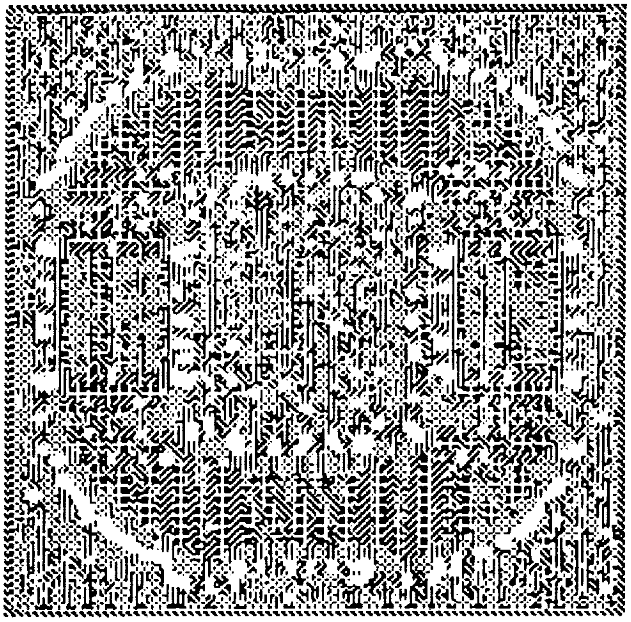


Figure 8.58: Curvature sign-map of the PVC pipe ring image obtained using kernel operators

In this experiment, a ring of PVC pipe sections (four straight $1\frac{1}{2}$ inch diameter pipe segments and four 90° elbows) was imaged and processed. Figure 8.57 is a rendered perspective image of the three-dimensional data obtained from the ERIM/USPS scanner. The usual noise at the range discontinuities is again apparent.

Figure 8.58 shows the curvature sign-maps computed using the 7×7 kernel operators. Again, there waw no perceptible correlation between the curvature sign-map labels and the surfaces known to be present.

Figure 8.59 shows the curvature sign-maps computed from the polynomial descriptors obtained by *variable-order segmentation* (The version of the *variable-order segmentation* program applied throughout the previously described experiments actually failed to segment the image satisfactorily at first – the entire ring was fitted with a 'Mexican hat'-like surface. To obtain a satisfactory segmentation, the image was augmented by a removal of edge points before *variable-order segmentation* program was run. The edge points were extracted by a morphological edge detector). Once again, the characteristic curvature signature of the cylindrical surface (a *RIDGE*

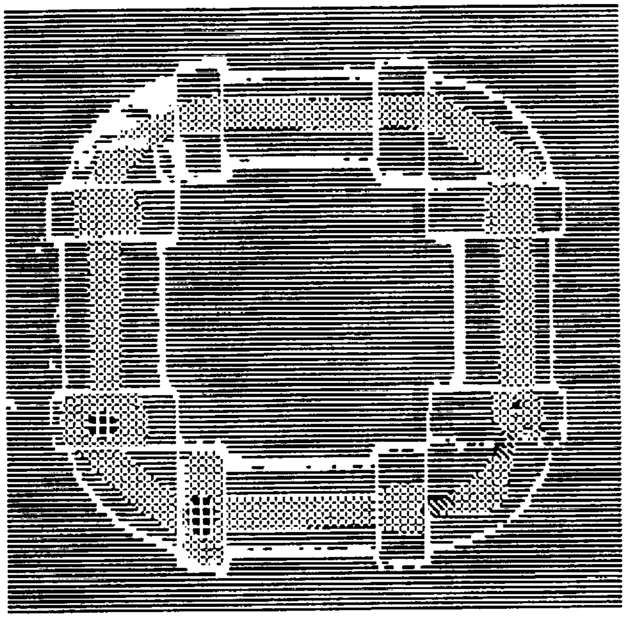


Figure 8.59: Curvature sign-map of the PVC pipe ring image obtained from the polynomial descriptors extracted by *variable-order segmentation*

region flanked by *FLAT* regions) is evident. The background plane was found to be *FLAT*.

Figures 8.60 and 8.61 are rendered perspective images of the three-dimensional surfaces found in the PVC pipe ring image by the hypothesis-verification operation. In figure 8.60, the cylinders were approximated by biquadratic surfaces (the *companion linear function* for cylinders). Figure 8.61, the parametric description of the true cylinders was extracted to generate them. Again, there is no visual difference the two images. The parameters of the cylinder axes (see figure 6.10) of the can were:

Segment	θ	ϕ	c_z (inches)	c_y or c_x (inches)
Top	0.538°	−0.991°	5.511	7.43(c_y)
Right	90.117°	−0.3818°	5.063	15.034(c_x)
Bottom	0.25°	−0.991°	5.386	15.64(c_y)
Left	90.0344°	−0.971°	5.508	7.086(c_x)

For the horizontal pipe segments, the x intercept of the cylinder axis with the y-z plane (c_x)

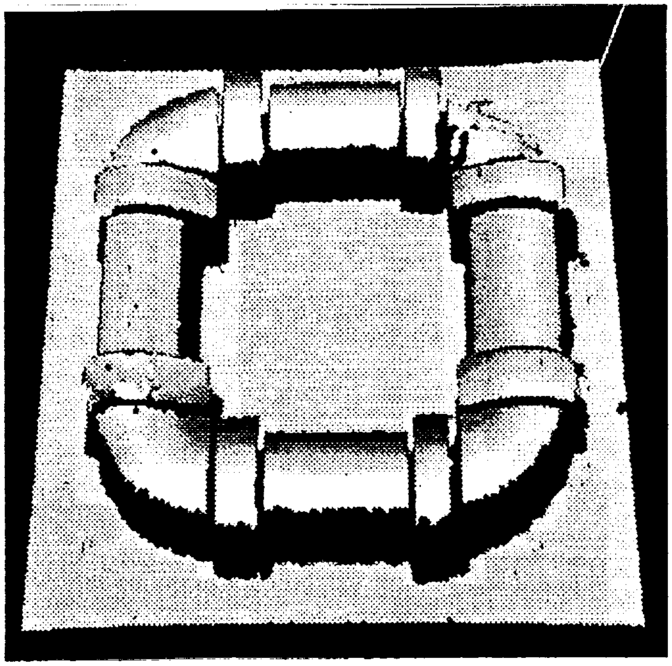


Figure 8.60: A rendered perspective view of the three-dimensional surfaces found in the PVC pipe ring image (the straight cylindrical segments were approximated by biquadratic surfaces, and the 'corners' by bicubic patches)

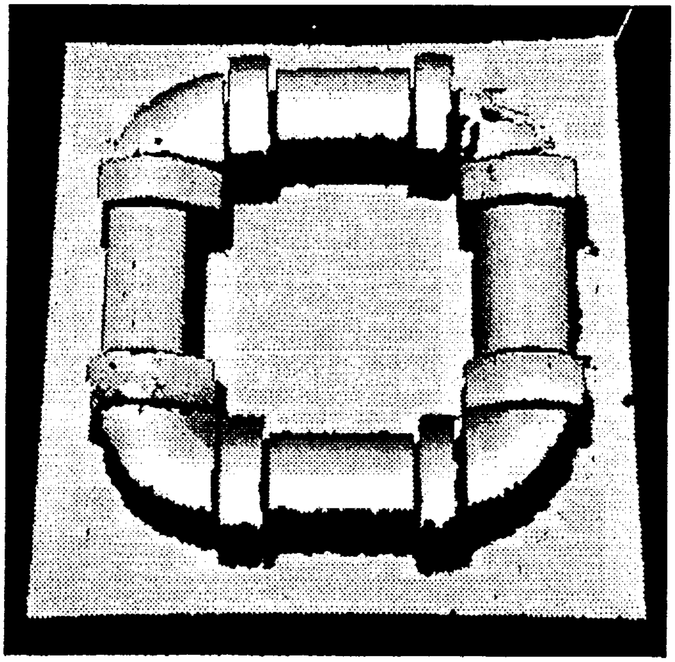


Figure 8.61: A rendered perspective view of the three-dimensional surfaces found in the image of blocks and a cylinder (the straight cylindrical segments were generated as true cylindrical surfaces)

was computed and for the vertical pipe segments, the y intercept c_y was computed.

As can be seen, the error in x-y orientation (θ) was no more than 0.5°.

CHAPTER IX

CONCLUSIONS

When work began on this thesis, it was the author's purpose to investigate issues pertaining to three-dimensional object recognition and pose determination *per se*. Simply put, the intent was to develop algorithms which would permit a robot system equipped with a laser range imaging system to recognize and locate an object and to perform an operation such as drilling a hole in it. It was thought that the surface descriptors generated by work such as Besl and Jain's *variable-order segmentation* algorithm [22, 25, 26] would provide a platform for such an endeavour. Such optimism was quickly proven wrong. Much needed to be done to shore up the foundations before the house of recognition and location may be erected. Among them were the abilities to bridge the gap between low and high level vision, to make reliable hypotheses regarding the scene, and to extract the parametric descriptions of the constructed geometric forms of which the target objects constitute. The work done on this thesis addresses these issues.

We shall conclude this work by summarizing the findings of the work of this thesis and to outline the research which remains to solve the original problem.

9.1 Summary

An *abstraction-based* paradigm for organizing perceptual tasks was developed and described in chapter I. It provides a means to span the crevasse between low and high level vision by an explicit *general to specific* refinement process. Within this paradigm, the task of specifying what is in a scene becomes one of making stronger and stronger assumptions about what is in the image. This strategy was applied to the processing of laser range imagery as outlined in the block diagram of figure 3.4.

Within the purview of *abstraction-based* refinement an initial symbolic description of the scene is necessary. The first such description was obtained in the form of smooth regions which could be approximated by bivariate polynomial surfaces. These were generated using Besl and Jain's *variable-order segmentation* program applying only the assumption that smooth contiguous regions could be modelled by bivariate polynomials.

A method for obtaining robust curvature-based surface labelling was demonstrated for the next step in the refinement process. While such differential geometry-based description has been proposed in the literature, the computation of these features directly from the imagery by kernel-type digital differentiation operators proved unstable. In the experiments presented, the Gaussian and mean curvature sign-map was computed on the polynomial description obtained in the previous abstraction process. As long as the region fits were faithful to the data, the information through the extended surface could be drawn upon to support the curvature computation. This was discussed in section 4.3 (section 4.3.4 in particular).

Once a hypothesis can be made as to the surfaces of construction present in the scene, these have to be verified and the orienting parameters have to be extracted. The majority of object recognition work found in the literature operate on polyhedra. This is because planes are the only surfaces which satisfy two criteria. They describe precisely the surface of construction and their algebraic description is linear in terms of their orienting parameters. The first criterion permits the recovery of the object centered coordinate system and the second makes their computation possible. Most constructed forms, however, are non-linear in terms of their orienting parameters. Numeric methods for the extraction of these parameters require good estimates of the parameter values. This 'chicken and egg' problem was approached using the concept of *companion linear* *forms*. These forms which are linear in their orienting parameters approximate their non-linear counterparts and are capable of estimating their orienting parameters. *Companion linear forms* are thus useful both for the detection and pose determination constructed forms.

A study was done on the application of the biquadratic form as the *companion linear form* for cylinders in sections 4.4 and 6.1. These estimates were then used in a non-linear fitting phase to determine the orienting parameters of the cylinders. The non-linear fitting was done with an implementation of the *Levenberg-Marquardt* algorithm. Results show that the parametric estimation from biquadratic fits were robust and accurate. In tests with synthetically generated cylinders (whose orientations are well known), the estimation was accurate within two hundredth of a degree for all orientations. In the experiments with the image of the fiberglass pipe elbow which was very noisy, the angle computed for the bend was off by 1.22°, and in the PVC pipe ring image, the angular errors did not exceed 0.5°.

It was shown, in chapter V, that ALL region split-merge operations are based upon hypotheses of the objects being constituted in the merging process. The criteria for region splitting and merging were defined. A hypothesis (assumption) guided split-merge algorithm was developed for determining the pose of smoothly merging planes and cylinders. For planes which merge smoothly into curved surfaces, *like-normal neighbourhoods* were advanced as a means of splitting the smooth surface containing the plane. The splitted regions were then merged to extract the plane. For cylinders which meld smoothly into other surfaces along their axes, *acceleration* bands were applied for region splitting and the biquadratic function was used to perform the region merging. Both operations were demonstrated in the experiments presented.

A fast algorithm for the extraction of the boundaries of run-length regions was developed (chapter VII). This finite-state machine algorithm takes advantage of the run list topology and boundary direction constraints implicit in the run-length encoding of the region. It extracts the external boundary of the region and the boundaries of all the holes in the region.

As mentioned in section 3.4, the laser range data (after transformation to three-dimensional Cartesian coordinates) are used in every stage of the abstraction process. As more and more specific hypotheses are imposed, the system remains data-bound. The hypotheses are always tested against the original data, and the extraction of the parametric description of the constructed geometric form is achieved by fitting the parametric models to the data.

9.2 Directions for Future Research

The measure of the success and relevance of this work is the extent to which we are closer to being able to recognize, locate and perform an operation on an industrial work piece (like drilling a specified hole). It has uncovered some areas which require more research before the task in question can be accomplished.

An architecture needs to be designed to implement the *abstraction-based* recognition and pose determination paradigm. A system which will recognize and determine the pose of objects with cylindrical and planar surfaces given laser imagery could be built directly on the work of this thesis. An alternate hierarchy of abstraction for recognition in intensity images could also be defined. The same architecture should operate on such a hierarchy.

More work needs to be done on *linear-companion forms* to cover all constructed geometric surfaces like cones, spheres etc. A promising set of functions for this task is superquadrics because of their ability to describe arbitrarily complex regular shapes. A cone, for example, could be approximated by one end of a very eccentric ellipsoid. The axis of the cone would be colinear with one principal axis of such an ellipsoid and the eccentricity of the ellipsoid would provide an estimation of the angle of taper of the cone.

In this work, the *variable-order segmentation* excluded most of the noise spikes (outliers), permitting the parameter estimation to work only on the 'good' data. The sensor's noise characteristics were not otherwise handled. In the ERIM/USPS scanner, for example, the surfaces imaged appear to possess a slight corrugation which can be seen in the images in chapter VIII.

This arises from the scanning regimen of the scanner. It was difficult to determine a least-squares threshold which would allow a region grow in a direction parallel to the corrugation and stop quickly when the surface really bends. This identifies the application of robust methods and the application of sensor noise models in the fitting process necessary for the determination of the orienting parameters of the visible surfaces.

Work also needs to be done in the fine-tuning regions at boundaries. In the example of the plane merging into a curved chamfer (see figure 8.55), the planar surface grew slightly into the curved chamfer. Surface melding functions may hide this error from the viewer, but such fixes are only cosmetic. They will not remove the errors in the estimation of the orientation of the plane introduced by the errant data. Better noise models will ameliorate this problem, but not solve it completely. One approach, perhaps, may be a 'retraction' procedure. The initial region growing could be aggressive to allow the surface to grow over the noisy data. Once the surface stops growing, overshoots must be expected. A fine tuning process may retract the surface from its boundaries using finer thresholds. This constitutes kind of a scale space approach in χ^2 threshold space.

Given a model of the work piece in terms of its constructed geometric surfaces, the algorithms developed in this thesis will extract the orienting parametric description of each surface. In order to determine the affine transformation of the entire object, methods for conflict resolution must be applied. Such methods must be able to reason with both numeric and symbolic information. For example, the significance ascribed to the orienting parameters of planes in an image should depend on the region sizes occupied by the planes and the degree to which it is oblique to the viewing direction. The estimation of the angle of orientation of a cylinder axis in the plane perpendicular to the viewing direction is more precise than the estimation of the angle of tilt of the axis away from the plane. Possible approaches for the solution of this problem are constraint-based methods, extended Hough transforms and neural networks. These techniques may then be applied in a complete object recognition and pose determination system.

Finally, the generation of recognition models (for use in a recognition system) from CAD descriptions of objects is an area where future research is needed. Such models should provide a recognition system with the types of surfaces visible and the curvature signatures one may expect. Since surface reflectivity characteristics (e.g. specularity) affect the imaging noise, methods must be investigated to incorporate these into the models.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Aloimonos, John (Yannis), "Visual shape computation," *Proceedings of the IEEE*, Vol. 76, No. 8, pp. 899–916, August 1988.

[2] Aloimonos, John, "Shape from texture," *Biological Cybernetics*, Vol. 58, pp. 345–360, 1988.

[3] Altschuler, Martin D., Posdamer, Jeffrey L., Frieder, Gideon, Altschuler, Bruce R., and Taboada, John, "The numerical stereo camera," *SPIE 3-D Machine Perception*, Vol. 283, pp. 15–24, 1981

[4] Andress, K.M., and Kak, A.C., "Evidence accumulation & flow of control in a hierarchical spatial reasoning system," *AI Magazine*, American Association for Artificial Intelligence, pp. 75–94, Summer 1988.

[5] Archibald, C., Rioux, M., "WITNESS: A system for object recognition using range images," *National Research Council Canada, Division of Electrical Engineering*, ERB-986, NRCC No. 25588, January 1986.

[6] Archibald, Colin, and Merritt, Cathy, "Pose determination of known objects from sparse range images," *Proceedings of the 2nd Conference on Intelligent Autonomous Systems, Amsterdam, The Netherlands*, December 1989.

[7] Asada, H, and Brady, M., "A curvature primal sketch," *Proc. IEEE Workshop on Computer Vision: Representation and Control*, Annapolis, Md, pp. 8–17, May 1984.

[8] Asada, Minoru, Ichikawa, Hidetoshi, and Tsuji, Saburo, "Determining surface orientation by projecting a stripe pattern," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 5, pp. 749–754, September 1988.

[9] Augusteijn, Marike F., and Dyer, Charles R. "Recognition and recovery of three-dimensional orientation of planar point patterns," *Computer Vision, Graphics, and Image Processing*, Vol. 36, pp. 76–99, 1986.

[10] Bajcsy, Ruzena, "Three-dimensional scene analysis," *Proceedings of the 5th International Joint Conference on Pattern Recognition, Miami Florida*, pp. 1064–1074, 1980

[11] Baker, H.H., and Binford, T.O., "Depth from edge and intensity based stereo," *International Joint Conference on Artificial Intelligence, Vancouver, B.C.*, Vol. 6, pp. 349–387, 1981.

[12] Ballard, Dana H., and Sabbah, Daniel, "Viewer independent shape recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5, No. 6, pp. 653–659, November, 1983.

[13] Barnard, Stephen T. and Fischler, Martin A., "Computational stereo," *ACM Computer Surveys*, Vol. 14, No. 4, pp. 553–572, December 1982.

[14] Barnard, S.T., "A stochastic approach to stereo vision," *American Association of Artificial Intelligence*, pp. 676–680, 1986.

[15] Barrow H.G. and Tenenbaum, J.M., "Interpreting line drawings as three dimensional surfaces," *Artificial Intelligence*, Vol. 17, pp. 75–117, 1981.

[16] Bastuscheck, Marc C., Schonberg, Edith, Schwartz, Jacob T., "Object recognition by 3-dimensional curve matching," *Technical Report No 203, Robotics Report No. 63*, New York University Dept. of Computer Sciences,, January 1986.

[17] Beaudet, Paul R., "Rotationally invariant image operators," *Proceedings of the* 4^{th} *International Joint Conference on Pattern Recognition*, November 7–10, 1978, Kyoto, Japan.

[18] Bergholm, Fredrik, "Edge focusing," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, Vol. PAMI-9, No. 6, pp. 726–741, November, 1987.

[19] Besl, Paul J., and Jain, Ramesh, "Three-dimensional object recognition," *Computing Surveys*, Vol. 17, No. 1, March 1985.

[20] Besl, Paul J., and Jain, Ramesh, "Intrinsic and extrinsic surface characteristics," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 119–123, June 1985

[21] Besl, Paul J., and Jain, Ramesh, "Range image understanding," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 430–449, June 1985

[22] Besl, Paul J., *Surfaces in Early Range Image Understanding*, Ph.D. Dissertation, Electr. Eng. Comp. Sci. Dept. (RSD-TR-10-86), University of Michigan, Ann Arbor, MI (March). 1986.

[23] Besl, Paul 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), pp. 33–80.

[24] Besl, Paul J., and R. Jain, "Segmentation through symbolic surface descriptions", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 77–85, June 1986.

[25] Besl, Paul J., *Surfaces in Range Image Understanding*, Springer-Verlag, New York, 1988.

[26] Besl, Paul J., and R. Jain, "Segmentation through variable-order surface fitting", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **10** No. 2, pp. 167–192, March 1988.

[27] Besl, Paul J., "Range image sensors," *Computer Science Department, General Motors Research Laboratories*, Warren, Michigan, GMR-6090, March 8, 1988.

[28] Bhanu, Bir, "Shape matching of two-dimensional occluded objects," *Proceedings of the 6th International Conference on Pattern Recognition, Munich, West Germany*, IAPR and IEEE, New York, pp. 742–744, October 19–22 1982.

[29] Bhanu, Bir, "Surface representation and shape matching of 3-D objects," *Proceedings of Pattern Recognition and Image Processing Conference, Las Vegas Nevada*, pp. 349–354, June 1982.

[30] Bhanu, Bir, "Representation and shape matching of 3-D objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, IEEE, New York, PAMI-6, No. 3, pp. 340–350, 1984.

[31] Bhanu, B., Lee, S., Ho, C.C. and Henderson, T., "Range data processing: Representation of surfaces by edges", *Proceedings of the 8th International Conference on Pattern Recognition*, pp. 236–238, October 1986

[32] Binford, T. O., "Survey of model-based image analysis systems," *International Journal of Robotics Research*, Vol. 1, No. 1, pp. 18–64, Spring, 1982.

[33] Binger, N., and Harris, S.J., "Applications of laser radar technology," *Sensors*, Vol. 4, No. 4, pp. 42–44, 1987.

[34] Boissonnat, J.D., and Faugeras, O.D., "Triangulation of 3-D objects," *Proceedings of the 7th International Joint Conference on Artificial Intelligence, Vancouver B.C. Canada*, pp. 658–660, August 24–28, 1981.

[35] Boissonnat, J.D., "Representation of objects by triangulating points in 3-D space," *Proceedings of the 7th International Conference on Pattern Recognition, Munich, W. Germany*, pp. 830–832, October 18–22, 1982.

[36] Bolle, Ruud M. and Cooper, David B., "Bayesian recognition of local 3-D shape by approximating image intensity functions with quadric polynomials," *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 4, pp. 418–429, July 1984

[37] Bolles, Robert C., and Fischler, Martin A., "A RANSAC-based approach to model fitting and its application to finding cylinders in range data," *Proc. of the 7th IJCAI*, pp. 637–643, 1984.

[38] Bolles, Robert C., and Cain, Ronald A., "Recognizing and locating partially visible objects: The local feature-focus method," Int. J. Robotics Research, Vol. 1, No. 3, pp. 57–82, 1982.

[39] Bolles, Robert C., Horaud, Patrice, and Hannah, Marsha Jo, "3DPO: A three-dimensional part orientations system," in *Robotics Research: The 1st International Symposium*, M. Brady and R. Paul, Eds. MIT Press, Cambridge, Mass., pp. 413–424, 1984.

[40] Bolles, R.C., Baker, H.H. and Marimont, D.H., "Epipolar plane image analysis: An approach to determining structure," *International Journal of Computer Vision*, Vol. 1, pp. 7–55, 1987.

[41] Boyter, Brian A., and Aggarwal, J. K., "Recognition of polyhedra from range data," *IEEE Expert*, pp. 47–59, Spring 1986.

[42] Brady, M., and Yuille, A., "An extremum principle for shape from contour," *Proceedings of the 8th International Joint Conference of Artificial Intelligence, Karlsruhe, West Germany*, IJCAI, pp. 969–972, August 1983.

[43] Brady, M., and Yuille, A., "An extremum principle for shape from contour," *IEEE Transactions on Pattern Analysis and Machine Intelligence* , Vol. PAMI-6, No. 3, pp. 288–301, 1984.

[44] Brady, Michael, "Representing shape," *IEEE International Conference on Robots* , pp. 256–265, 1984.

[45] Brice, C., and Fennema, C., "Scene analysis using regions," *Artificial Intelligence*, Vol. 1 No. 3, pp. 241–274, 1970.

[46] Brooks, Rodney A., Greiner, R., and Binford, T. O., "The ACRONYM model-based vision system," *Proc. of the 76h IJCAI*, pp. 105–113, 1979.

[47] Brooks, Rodney A., "Model-based three dimensional interpretations of two dimensional images," *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vancouver, B. C., Canada, pp. 619–624, August 1981.

[48] Brooks, Rodney A., "Symbolic Reasoning among 3-D Models and 2-D Images," *Artificial Intelligence*, Vol. 17, pp. 285–348, August 1981.

[49] Brooks, Rodney A., "Model-based three-dimensional interpretations of two-dimensional images," *Proc. of the 7th IJCAI*, pp. 619–624, 1981.

[50] Brooks, Rodney A., "Representing possible realities for vision and manipulation," *Proceedings of the Pattern Recognition and Image Processing Conference, Las Vegas, Nevada*, IEEE New York, pp. 587–592, June 14–17, 1982.

[51] Brooks, Rodney A., "Model-based three-dimensional interpretations of two-dimensional images," *IEEE Trans. Pattern Anal. and Machine Intell.*, Vol. PAMI-5 No. 2, pp. 140–149, March 1983.

[52] Brou, Philippe, "Using the Gaussian image to find the orientation of objects," *The International Journal of Robotics Research*, Vol. 3, No. 4, pp. 89–125, Winter 1984.

[53] Canny, John Francis, "Finding edges and lines in images," *Massachusetts Institute of Technology*, MIT-AI-TR-720, 1983.

[54] Canny, John Francis, "A computational approach to edge detection," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 6, 1986.

[55] Canny, John Francis, "A computational approach to edge detection," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 6, 1986.

[56] Chakravarty, Indranil, and Freeman, Herbert, "Characteristic views as a basis for three-dimensional object recognition," *Proc. of SPIE Robot Vision Conference*, Vol. 336, pp. 37–45, May 1982.

[57] Chen, Nai-Yung, Birk, John R., and Kelley, Robert B., "Estimating workpiece pose using the feature point method," *IEEE Transactions on Automatic Control*, AC-25, No. 6, December, 1980.

[58] Cheng, J. K., and Huang, T. S., "Recognition of curvilinear objects by matching relational structures," *Proc. of PRIP*, pp. 343–348, 1982.

[59] Cheng, J. K., and Huang, T. S., "Image registration by matching relational structures," *Proc. of the 6th International Conf. on Pattern Recognition*, pp. 354–356, 1982.

[60] Chin, Roland T., and Deyer, Charles R., "Model-based recognition in robot vision," *Computing Surveys*, Vol. 18, No. 1, pp. 67–108, 1986.

[61] Clark, James L., "Singularity theory and phantom edges in scale space," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, Vol. 10, No. 5, pp. 720–727, 1988.

[62] Coleman, E. North and Jain, Ramesh, "Obtaining shape of textured and specular surface using four-source photometry," *Computer Graphics and Image Processing*, Vol. 18, No. 4, pp. 309–328, 1982.

[63] Coleman Jr., E. North, and Sampson, Robert E., "Acquisition of randomly oriented workpieces through structure mating," *IEEE Proceedings on Computer Vision and Pattern Recognition, San Francisco, California*, pp.350–357, June 19–23, 1985.

[64] Dhome, M., and Kasvand, T., "Polyhedra recognition by hypothesis accumulation from range data," *National Research Council Canada, Division of Electrical Engineering*, ERB-985, NRCC No. 25655, May 1986.

[65] Douglas, Robert J., "Interpreting three-dimensional scenes: A model building approach," *Computer Graphics and Image Processing*, Vol. 17, pp. 91–113, 1981.

[66] Duda, Richard O., Nitzan, David, and Barrett, Phyllis, "Use of range and reflectance data to find planar surface regions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-1, No. 3, pp. 259–271, July 1979.

[67] Enab, Y.M., and Luh, J.Y.S, "Shape from a single view using matching method," *Pattern Recognition*, Vol. 21, No. 4, pp. 313–318, 1988.

[68] Engelbrecht, Jan R., and Wahl, Friedrich M., "Polyhedral object recognition using Hough-space features," *Pattern Recognition*, Vol. 21, No. 2, pp. 155–167, 1988.

[69] Environmental Research Institute of Michigan, *EPICAL Reference Manual – Version 1.2 (ERIM Portable Image Computer Analysis Library)*, ERIM, Ann Arbor, Michigan, 1987.

[70] Environmental Research Institute of Michigan, *Configuration description for ERIM/USPS range sensor – dynamic and static modes*, ERIM, Ann Arbor, Michigan, 1989.

[71] Ettinger, Gil J., "Hierarchical object recognition using libraries of parameterized model sub-parts," *MIT Artificial Intelligence Laboratory*, Technical Report 963, June 1987.

[72] Ettinger, Gil J., "Large Hierarchical object recognition using libraries of parameterized model sub-parts," *Proceedings of the Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 32–41, June 5–9, 1988.

[73] Fan, T.G., Medioni, G., and Nevatia, R., "Description of surfaces from range data using curvature properties," *Proc. Computer Vision and Pattern Recognition Conf.*, (Miami, Fla., June 22–26), IEEE-CS, pp. 86–91. 1986.

[74] Fan, Ting-Jun, Medioni, Gerard, and Nevatia, Ramakant, "Recognizing 3-D objects using surface descriptions," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, Vol. PAMI-11, No. 11, pp. 1140–1157, November 1989.

[75] Faugeras, G. D., Hebert, M., and Pauchor, E., "Segmentation of range data into planar and quadratic patches," *IEEE*, CH1891-1/83/0000/0008, 1983.

[76] Fekete, G., and Davis, L. S., "Property spheres: A new representation for 3-D object recognition" *Proc. IEEE Workshop on Computer Vision: Representation and Control*, pp. 192–201, May 1984.

[77] Freeman, H., and Chakravarty, I., "The use of characteristic views in the recognition of three-dimensional objects," *Pattern Recognition in Practice* E. Gelsema and L. Kanal, Eds., North-Holland Publishing Co., Amsterdam, 1980.

[78] GEOMOD *User Manual and Reference Manual*, Structural Dynamics Research Co. (SDRC), Cincinnati, Ohio, 1982.

[79] Gleason, G. J., and Agin, G. J., "A modular system for sensor-controlled manipulation and inspection," *Proc of the 9th Symp. on Industrial Robots, SME*, pp. 57–70, 1979.

[80] Goad, C., "Special-purpose automatic programming for 3D model-based vision," *Proc. of the Image Understanding Workshop*, pp. 94–104, 1983.

[81] Golub, Gene H., and Van Loan, Charles F., *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, 1983.

[82] Grimson, W.E.L., "A computer implementation of a theory of human stereo vision," MIT AI Lab Memo No. 565, January 1980.

[83] Grosky, W. I., and Lu, Yi, "Iconic indexing using generalized pattern matching," *Proceedings of the 7th International Conference on Pattern Recognition, Montreal, Canada*, pp. 1168–1170, July 30–Aug 2, 1984.

[84] Grosky, William I., and Lu, Yi, "Iconic indexing using generalized pattern matching techniques," *Computer Vision, Graphics, and Image Processing*, 35, pp. 383–403, 1986.

[85] Grosky, William I., and Jain, Ramesh, "A pyramid-based approach to segmentation applied to region matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8, No. 5, pp. 639–650, September 1986.

[86] Haralick, R.M., "The digital edge," *Proceedings of the 1981 Conference on Pattern Recognition Image Processing*, New York: IEEE Computer Society, pp. 60–73, 1981.

[87] Haralick, R.M., Watson, L.T. and Laffey, T.J., "The topographic primal sketch," *Int. J. Robotics Res.* **2**, 1 (Spring), pp. 50–72, 1983.

[88] Haralick, R. M., and Shapiro, L. G., "Image segmentation techniques," *Computer Vision, Graphics and Image Processing*, **29**, pp. 100–132, 1985.

[89] Haynes, S. M., and Jain, Ramesh, "Event detection and correspondence," *Optical Engineering*, Vol. 25, No. 3, pp. 387–393, 1986

[90] Hebert, Martial and Ponce, Jean, "A new method for segmenting 3-D scenes into primitives," *Proc. IJCPR*, pp. 836–838, 1982.

[91] Hebert, M. and Kanada, T., "The 3D-profile method for object recognition," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 458–463, June 1985.

[92] Hebert, M., "Outdoor scene analysis using range data," *Proc. IEEE Conf. Robotics and Automation*, pp. 1426–1432, April 1986.

[93] Henderson, T.C. and Bhanu, B., "Three-point seed method for the extraction of planar faces from range data," *Proc. IEEE Workshop on Industrial Applications of Machine Vision*, Research Triangle Park, North Carolina, pp. 181–186, May 1982.

[94] Henderson, T. C., "Efficient 3-D object representations for industrial vision systems," *IEEE Transactions on Pattern Recognition Analysis and Machine Intelligence*, Vol. PAMI-5 No. 6, pp. 609–617, November 1983.

[95] Holland, S. W., Rossol, L., and Ward, M. R., "CONSIGHT-1: A vision-controlled robot system for transferring parts from belt conveyors," *Computer Vision and Sensor-Based Robots*, G. G. Dodd and L. Rossol, Eds. pp. 81–97, Plenum, New York, 1979.

[96] Hopcroft, John E., and Ullman, Jeffery D., *Introduction to automata theory, languages, and computation*, Addison-Wesley Publishing Company, Reading MA, 1979.

[97] Horaud, Patrice, and Bolles, Robert C., "3DPO's strategy for matching three-dimensional objects in range data," *International Conference on Robotics, Atlanta, Georgia*, pp. 78–85, March 13-15, 1984.

[98] Horaud R., and Brady, M., "On the geometric interpretation of image contours," *Proceedings of the 1st International Conference on Computer Vision, London U.K.*, pp. 374–382, 1987.

[99] Horn, Berthold K.P. and Ikeuchi, Katsushi, "The mechanical manipulation of randomly oriented parts," *Scientific American*, pp. 100–111, July 1984.

[100] Horn, B. K. P., "Extended Gaussian images," *Proceedings IEEE*, Vol. 72, No. 12, pp. 1671-1686, 1984.

[101] Ikeuchi, Katsushi and Horn, Berthold K.P., "Numerical shape from shading and occluding boundaries," *Artificial Intelligence*, Vol. 17, pp. 141–184, August 1981.

[102] Ikeuchi, K., "Recognition of 3-D objects using the extended Gaussian image," *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vancouver, B. C., Canada, pp. 595–600, August 1981.

[103] Ikeuchi, Katsushi, Horn, Berthold K.P., and Nagata, Shigemi, "Picking up an object from a pile of objects," *M.I.T. Artificial Intelligence Laboratory*, A.I. Memo No. 726, May 1983.

[104] Ittner, D. and Jain, A., "3D surface discrimination from local curvature measures," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 119–123, June 1985.

[105] Jacobus, C., Mitchell, B., Gilles, A., and Smith, S., "Advanced Research in Recognition of Handwritten Address ZIP Codes, Phase I Report" *ERIM*, —ERIM—197200-8-T, ARVD-87-093 May, 1987.

[106] Jain, Anil K., and Hoffman, Richard, "Evidence-Based Recognition of 3-D Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-10, No. 6, pp. 783–802, November, 1988.

[107] Jarvis, R.A., "A perspective on range finding techniques for computer vision," *IEEE Trans. Pattern Anal. and Machine Intell.*, Vol. PAMI-5 No. 2, pp. 122–139 March 1983,

[108] Jarvis, R.A., "A Laser time-of-flight range scanner for robotic vision," *IEEE Trans. Pattern Anal. and Machine Intell.*, Vol. PAMI-5 No. 5, pp. 505–512, September 1983

[109] Jenkin, M., "Tracking three dimensional moving light displays," *Proc. Workshop on Motion: Representation and Control*, pp. 66–70. 1983.

[110] Jerian, C. P., and Jain, R., "Determining motion parameters for scenes with translation and rotation," *Proc. Workshop on Motion: Representation and Control*, 1983.

[111] Kanade, T., "Recovery of the three-dimensional shape of an object from a single view," *Artificial Intelligence*, Vol. 17, pp. 409–460, August 1981.

[112] Kaplan, Wilfred, *Advanced Mathematics for Engineers* Addison-Wesley Publishing Company, Reading, Massachusetts, 1981.

[113] Kelley, R. B., Birks, J. R., Martins, H. A. S., and Tella, R., "A robot system which acquires cylindrical workpieces from bins," *IEEE Trans. Syst. Man Cybern.*, Vol. 12, No. 2, pp. 204–213, 1982.

[114] "Shape from texture: An aggregation transform that maps a class of textures into surface orientation," *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 475–480, 1981.

[115] Kindle, Joseph H., *Analytical Geometry* McGraw-Hill, New York, 1950.

[116] Khan, Naseem A., and Jain, Ramesh, "Matching an imprecise object description with models in a knowledge base," *Proceedings of the 7th International Conference on Pattern Recognition, Montreal, Canada*, pp. 1131–1134, July 30-Aug 2, 1984.

[117] Knoll, Thomas F. and Jain, Ramesh "Recognizing Partially Visible Objects Using Feature Indexed Hypotheses," *IEEE J. Robotics and Automation*, Vol. RA-2, No. 1, pp. 3-13, 1986

[118] Knoll, Thomas F., and Jain, Ramesh C., "Learning to recognize objects using feature indexed hypotheses," *Proc. of the 1st International Conf. on Computer Vision (ICCV'87)*, 1987.

[119] Koenderink, J. J. and van Doorn, A.J., "Internal representation of solid shape with respect to vision," *Biological Cybernetics*, Vol. 32, pp. 211–216, 1979.

[120] Kohavi, Zvi, *Switching and finite automata theory*, McGraw-Hill Book Company, New York, 1978.

[121] Korn, Matthew R., and Dyer, Charles R., "3-D multiview object representations for model-based object recognition," *Pattern Recognition*, Vol. 20, No. 1, pp. 91–103, 1987.

[122] Kuan, Darwin T. and Drazovich, Robert J., "Model-based interpretation of range imagery," *AAAI-84*, pp. 210–215, 1984.

[123] Liou, Shih-Ping, Chiu, Arnold, H., and Jain, Ramesh C., *A parallel technique for signal-level perceptual organization*, Submitted for publication – PAMI.

[124] Laffey, T. J., Haralick, R. M., and Watson, L. T., "Topographic classification of digital image intensity surfaces," *Proceedings of the IEEE Workshop on Computer Vision: Theory Contr.* New York: IEEE Computer Society, pp. 171–177, 1982.

[125] Lipschutz, M. M., *Differential Geometry*, McGraw-Hill, New York, 1969.

[126] Little, J. J., "An iterative method for reconstruction convex polyhedra from extended Gaussian images," *Proceedings on the National Conference on Artificial Intelligence*, Washington, D. C., AAAI, pp. 247–250, August 1983.

[127] Marr, D., and Poggio, D., "Cooperative computation of stereo disparity," *Science*, Vol. 194, pp. 283–287, 1976.

[128] McClelland, J. L., Rumelhart, D. E., and Hinton, G. E., "The appeal of Parallel Distributed Processing," In *Parallel Distributed Processing, Volume 1: Foundations*, Rumelhart, D. E., McClelland, J. L., and the PDP Research Group, (Eds.), The MIT Press, Cambridge, Mass., pp. 3–44, 1986.

[129] Medioni, G. and Nevatia, R., "Description of 3-D surfaces using curvature properties," *Proc. Image Understanding Workshop*, pp. 291–299, October 1984.

[130] Milgram, D.L. and Bjorklund, C.M., "Range image processing: Planar surface extraction," *Proc. IJCPR-5*, pp. 912–919, 1980.

[131] Millman, Richard S. and Parker, George D., *Elements of Differential Geometry*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1977.

[132] O'Rourke, Joseph and Badler, Norman, "Decomposition of three-dimensional objects into spheres," *IEEE Trans. Pattern Anal. and Machine Intell.*, Vol. PAMI-1 No. 3, pp. 295–305, July 1979.

[133] O'Rourke, J., "Polyhedra of minimal as 3D models," *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vancouver, B. C., Canada, IJCAI pp. 664-666, August 1981.

[134] Parvin, B., and Medioni, G., "Segmentation of range images into planar surface by split and merge," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 415–417, June 1986.

[135] Pentland, Alex, "On the extraction of shape information from shading," *M.I.T. Media Lab Vision Sciences Group, The Media Lab, M.I.T. Cambridge, MA 02138*, , Tech. Report 102, pp. 1–11, March 15, 1988.

[136] Perkins, W. A., "A model-based vision system for industrial parts," *IEEE Trans. on Computers*, Vol. 27, No. 2, pp. 126–143, February, 1978.

[137] Potmesil, Michael, "Generating three-dimensional surface models of solid objects from multiple projections," *PhD. Dissertation, Rensselaer Polytechnic Institute*, December 1982.

[138] Potmesil, Michael, "Generating models of solid objects by matching 3-D surface segments," *Proceedings of the 8th International Joint Conference of Artificial Intelligence, Karlsruhe, West Germany*, IJCAI, pp. 1089–1093, August 1983.

[139] Prager, J. M., and Arbib, M. A., "Computing the optic flow: The MATCH algorithm and prediction," *Computer Vision, Graphics, Image Processing*, Vol. 24, pp. 271–304, 1983.

[140] Press, William, H., Flannery, Brian, P., Teukolsky, Saul A., and Vetterling, William T., *Numerical Recipes*, Cambridge University Press, Cambridge, 1986.

[141] Riggs, A.J., Tomko, L.M., Wesolowicz, K.G., and Jacobus, C.J., "Design of a laser radar range imaging sensor for postal applications," *Environmental Research Institute of Michigan*, Ann Arbor, MI, Tech. Report No. 197301-1-S1, October 1986.

[142] Rioux, M., "Laser range finder based upon synchronized scanners," *Applied Optics*, Vol. 23, No. 21, pp. 3837–3844, 1984.

[143] Rumelhart, D. E., Hinton, G. E., and McClelland, J. L., "A general framework for Parallel Distributed Processing," In *Parallel Distributed Processing, Volume 1: Foundations*, Rumelhart, D. E., McClelland, J. L., and the PDP Research Group, (Eds.), The MIT Press, Cambridge, Mass., pp. 45–76, 1986.

[144] Rumelhart, D. E., and Zipser, D., "Feature discovery by competitive learning," In *Parallel Distributed Processing, Volume 1: Foundations*, Rumelhart, D. E., McClelland, J. L., and the PDP Research Group, (Eds.), The MIT Press, Cambridge, Mass., pp. 152–193, 1986.

[145] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "A distributed model of human learning and memory," In *Parallel Distributed Processing, Volume 2: Psychological and Biological Models*, McClelland, J. L., Rumelhart, D. E., and the PDP Research Group, (Eds.), The MIT Press, Cambridge, Mass., pp. 170–215, 1986.

[146] Sampson, R.E., "3D range sensor via phase shift detection," *IEEE Computer*, Vol. 20, No. 8, pp. 23–24, 1987.

[147] Sato, K., Yamamoto, H. and Inokuchi, S., "Tuned range finder for high precision 3D data," *Proc. 8th Intl. Conf. Pattern Recognition*, pp. 1168–1171, October 1986.

[148] Sato, Y., Kitagawa, H., Fujita, H., "Shape measurement of curved objects using multiple slit ray projections," *IEEE Trans. Pattern Anal. and Machine Intell.*, Vol. PAMI-4, pp. 641–646, November 1982.

[149] Sethi, I.K. and Jayaramamurthy, S.N., "Surface classification using characteristic contours," *Proc. IJCPR*, pp. 438–440, August 1984.

[150] Sethi, Ishwar, K., and Jain, Ramesh, "Finding trajectories of feature points in a monocular image sequence," *IEEE Trans. on PAMI*, Vol. PAMI-9, No. 1, pp. 56–73, 1987.

[151] Shah, M. A., and Jain, R., "Detecting time varying corners," *Computer Vision, Graphics, Image Processing*, Vol. 28, pp. 345–355, 1984.

[152] Shneier, M., "A compact relational structure representation," *Proc. of the 6th IJCAI*, pp. 818–826, 1979.

[153] Shneier, M., "Models and strategies for matching in industrial vision," *Tech. Rep. TR-1073, Comp. Sc. Dept., Univ of Maryland*, College Park, July, 1981.

[154] Sripradisvarakul, Thawach, and Jain, Ramesh, "Generating aspect graphs for curved objects," *Proceedings of the Workshop on Interpretation of 3D Scenes, Austin Texas*, pp. 109–115, November 27–29 1989.

[155] Strang, Gilbert, *Linear algebra and its applications*, Academic Press Inc., New York, 1976.

[156] Svetkoff, D.J., "Towards a high resolution, video rate, 3D sensor for machine vision," *Proceedings of SPIE Conference on Optics, Illumination, and Image Sensing for Machine Vision, Cambridge MA – D.J. Svetkoff (ed.)*, Vol. 728, Oct. 30–31, 1986.

[157] Terzopoulos, Demetri, "Multilevel computational processes for visual surface reconstruction," *Computer Vision, Graphics, and Image Processing*, Vol. 24, pp. 52–96, 1983.

[158] Terzopoulos, Demetri, "The role of constraints and discontinuities in visible-surface reconstruction," *Proceedings of the 8th International Joint Conference on Artificial Intelligence, Karlsruhe, Germany*, IJCAI, August, 1983.

[159] Terzopoulos, Demetri, "Multiresolution computation of visible-surface representations," *PhD. Thesis, Massachusetts Institute of Technology*, January, 1984.

[160] Terzopoulos, Demetri, "Computing visible-surface representations," *M.I.T. Artificial Intelligence Laboratory*, A.I. Memo No. 800, March, 1985.

[161] Tropf, H., Geisselmann, H., and Foith, J. P., "Some applications of the fast industrial vision system S.A.M.," *Conf. Record of the Workshop on Industrial Applications of Mach. Vision*, pp. 73–79, 1982.

[162] Tsai, R. Y., and Huang, T. S., "Estimating three-dimensional motion parameters of a rigid planar patch," *Proc. PRIP*, pp. 94–97, 1981.

[163] Tsai, R. Y., and Huang, T. S., "Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces," *Proc PRIP*, pp.112–118, 1982.

[164] Tsai, R. Y., Huang, T. S., and Zhuy, W. L., "Estimating three-dimensional motion parameters of a rigid planar patch, II: Singular value decomposition," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP-30, pp. 525–534, 1984.

[165] Waxman, A. M., "An image flow paradigm," *Proc. Workshop on Computer Vision*, 1984.

[166] Wilson, W. A., and Tracey, J. I., *Analytic Geometry – Alternate Edition* D. C. Heath and Company, Boston, Massachusetts, 1937.

[167] Witkin, A.P., "Recovering surface shape and orientation from texture," *Artificial Intelligence*, Vol. 17, pp. 17–47, 1981.

[168] Wong, A. and Lu, S., "Recognition and knowledge synthesis of 3D object images," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 162–166, June 1985.

[169] Yachida, M., Asada, M., and Tsuji, S., "Automatic analysis of moving images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 12–19, 1981.

[170] Zuech, N., and Ray, R., "Vision guided robotic arm control for part acquisition," *Proc. of the Control Engineering Conf.*, 1983.

[171] Zuk, D.M., and Dell'eva M.L., "Three-dimensional vision system for the adaptive suspension vehicle," *Environmental Research Institute of Michigan*, Ann Arbor, MI, Final Report No. 170400-3-F, 1983.

[172] Zuk, D.M., and Dell'eva M.L., "Intelligent task automation," *Environmental Research Institute of Michigan*, Ann Arbor, MI, Tech. Report No. 165700-18-F, December 1984.