1989

Model-Based Three-Dimensional Object Recognition and Localization Using Properties of Surface Curvatures.

Wu Wang
Louisiana State University and Agricultural & Mechanical College

Order Number 9025352

Model-based three-dimensional object recognition and localization using properties of surface curvatures

Wang, Wu, Ph.D.

The Louisiana State University and Agricultural and Mechanical Col., 1989

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

Model-Based Three-Dimensional Object Recognition and Localization Using Properties of Surface Curvatures

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Computer Science

by

Wu Wang

B.S., Zhongshan University, 1984

December 1989

To Sherry: Without thee, my love,
I travel not, but stay...

Acknowledgements

To Professor S. Sitharama Iyengar, who supported me in every aspect throughout my graduate studies, taught me how to express myself, contributed important insights to the work, and encouraged me to attack these problems. I have found him to be a man of uncommon knowledge and erudition. His stimulating and friendly advice led me through the years.

I would like to thank my dissertation committee - Donald Kraft, Bush Jones and Doris Carver - who gave invaluable suggestions to the work. To Jianhua Chen, who lifted my spirits and helped me solve problems. I am grateful to the professors in my minor field - Padmanaban Sundar, Gisele Rieder, Walter Schnyder and Jorge Morales - who sharpened my mind and my logical thinking.

I would also like to thank the people around me for their friendship and support. I am especially grateful to Mohan Sharma, who gave me friendly advice on many matters; to Yuejian Sheng, who kindly helped me in many situations; to Sharon Pritchard, who reviewed early drafts of this dissertation; to Jingwen Ma, who befriended me.

Special gratitude is owed to my parents, my sisters, and my brothers-in-law, for their love and support, for their understanding, and for their accessibility. This work is dedicated to Sherry for her love and understanding.

This work was partially supported by the U.S. Department of the Navy.

Table of Contents

Abstract

The ability to recognize three-dimensional (3-D) objects accurately from range images is a fundamental goal of vision in robotics. This facility is important in automated manufacturing environments in industry. In contrast to the extensive work done in computer-aided design and manufacturing (CAD/CAM), the robotic process is primitive and ad hoc.

This thesis defines and investigates a fundamental problem in robot vision systems: recognizing and localizing multiple free-form 3-D objects in range images. An effective and efficient approach is developed and implemented as a system *Free-form Object Recognition and Localization (FORL).* The technique used for surface characterization is surface curvatures derived from geometric models of objects. It uniquely defines surface shapes in conjunction with a knowledge representation scheme which is used in the search for corresponding surfaces of an object. Model representation has a significant effect on model-based recognition. Without using surface properties, many important industrial vision tasks would remain beyond the competence of machine vision.

Knowledge about model surface shapes is automatically abstracted from CAD models, and the CAD models are also used directly in the vision process. The knowledge representation scheme eases the processes of acquisition, retrieval, modification and reasoning so that the recognition and localization process is effective and efficient.

Our approach is to recognize objects by hypothesizing and locating objects. The knowledge about the object surface shapes is used to infer the hypotheses and the CAD models are used to locate the objects. Therefore, localization becomes a by-product of the recognition process, which is significant since localization of an object is necessary in robotic applications.

One of the most important problems in 3-D machine vision is the recognition of objects from their partial view due to occlusion. Our approach is surface-based, thus, sensitive to neither noise nor occlusion. For the same reason, surface-based recognition also makes the multiple object recognition easier. Our approach uses appropriate strategies for recognition and localization of 3-D solids by using the information from the CAD database, which makes the integration of robot vision systems with CAD/CAM systems a promising future.

Chapter 1: Introduction

In recent years, there has been a tremendous spurt in the recognition of three-dimensional (3-D) objects in range images in research activity among the computer vision and robotics communities. More importantly, 3-D geometrical calculations are central to computer graphics, computer-aided design and computer-aided manufacturing (CAD/CAM), and other fields.

Fundamental to robot functions is the acquisition of relevant characteristics, task invariances, and relational object properties. Thus, an object recognition by a robot requires a good form of sensory perception in a 3-D structure. Yet recovering 3-D information from these visual (projected) images is a complex task and still remains the subject of fundamental research.

Visual data obtained from range sensors by a robot provides 3-D range information about objects directly. Interpretation of range data by a vision system has been one of the major problems of vision research in the past five years. This is because the vision system must describe the dynamic scene in terms of 3-D structural primitives of range images, such as edges, surfaces and volume, and finally recognize objects completely.

Therefore, one of the fundamental characteristics required by a robot vision system is the ability to derive properties, such as extracting features and recognizing objects, from the range data, which is a fascinating area of research in computer science. Towards this objective, we develop an efficient approach for the recognition

and localization of 3-D free-form objects in range images using properties of algebraic surfaces which are widely used in geometric modeling. They include Bezier surfaces and splines of various kinds.

We (Wang and Iyengar, 89) have developed a robust and efficient approach to recognizing and localizing multiple 3-D free-form objects. The approach is implemented as a robot vision system, *Free-form Object Recognition and Localization (FORL).* Our novel approach is able to recognize and localize real free-form objects, which has not been solved before. The approach uses boundary representation for object models, which is one of the most popular computer-aided design (CAD) model representation. Since CAD systems are popular and provide a user-friendly environment for design, they are natural sources for object models in robot vision.

We have developed a knowledge representation scheme for describing free-form surface shapes. A representation of knowledge is a combination of data structures and interpretive procedures which use the data structures. We have designed classes of data structures for storing information and procedures for intelligently manipulating these data structures to make inferences. The data structures and the procedures are well-designed so that the knowledge leads *FORL* to intelligent behavior, i.e., recognizing and localizing multiple 3-D free-form objects.

Localization is a by-product of the recognition process in *FORL.* This is significant since robot vision needs object location and orientation information to enable the robot to handle the objects. *FORL* is capable of recognizing and localizing

multiple free-form objects. Even when objects are partially occluded by one another, *FORL* is still able to recognize and localize objects from their partial view.

The remainder of this dissertation is organized as follows. We introduce some background information about machine vision in Chapter 2. Chapter 3 shows prior research in this area and compares it with our approach. In Chapter 4, we explain one of the most common CAD model representations - boundary representation of solids, which is the model representation for our system. The invariant properties of surfaces are shown in Chapter 5. We discuss the considerations for knowledge representation in intelligent systems and introduce our data structures in Chapter 6. The control strategy of the recognition and localization process is discussed in detail in Chapter 7. The experimentation of the approach is shown in Chapter 8. Finally, we give our conclusions in Chapter 9.

Chapter 2: Preliminary

Processing in a computer vision system is derived from three related fields:

a) image processing;

b) pattern recognition; and

c) scene analysis.

In image processing, the input and output are both images with the output image an improved version of the input image. Processing involves gray scale modifications to normalize scene brightness and contrast, sharpening to restore weakened high spatial frequencies, and smoothening to remove the noise in the image. If two images have to be compared, they may have to be registered, i.e., geometrically transformed to make them congruent, before matching is performed. Many of the gray scale image processing techniques are applicable to range images.

Pattern recognition provides a description of the input image based on a priori knowledge of expected patterns. The computer generally starts with a list of brightness values associated with the array of hundreds of thousands of points corresponding to the image. Recognizing a pattern means replacing this mass of undigested data with a much simpler, more useful description. It is often more convenient to first search for examples of patterns like edges and regions, which are referred to as features. A simplified description of the image constructed from these features can then be used as the basis of recognition.

Scene analysis is the transformation of simple features into abstract descriptions relating to objects that cannot be recognized based simply on pattern matching.

The functional performance of a vision system is judged according to the following characteristics [96]:

(1) 3-D measurement of objects;

(2) transformation of these measured data into a representation (data structure); and

(3) interpretation and recognition of the representation.

2.1 Progress of Machine Vision

Historically speaking, computer vision began in the mid 1960s with Roberts [71]. His system operated in the polyhedral or "blockworld" domain. After Roberts, vision systems continued to be built, but their performance seemed weak compared to the amount of work that went into them.

Ballard and Brown's [4] influential idea, favored by cognitive psychologists, was that "high-level" (cognitive) processes were at the heart of vision. This view has its points and happened to fit in with the economics of computing. However, symbolic reasoning is very difficult in machine vision, since the input often does not correspond to expectation.

The next major idea, which can be seen in Horn [44], was to use physics and applied mathematics to determine the information available in an image. The idea is

to compute "intrinsic images" or invariant physical scene parameters from an image. One way is the use of extended Gaussian images. A map between an object and the unit Gaussian sphere is defined. Gaussian curvature is

$$K = \frac{dS}{dO}$$

where dS is on the Gaussian sphere and dO is on the object. The extended Gaussian image is obtained by

$$G(\xi,\eta) = \frac{1}{K(u,v)}$$

where (ξ,η) is the point on the Gaussian sphere corresponding to the point (u,v) on the original surface. Under the assumption that all objects are convex, the map has the following properties: (a) the map is invertible; (b) a rotation of the object corresponds to an equal rotation of the Gaussian sphere; and (c) the map is unique.

Another important trend that started in the mid 1970s was to try to learn from biological systems [56]. Biological vision systems work very well compared to computer vision systems. This cross-fertilization between the neurosciences and computer sciences has been increasingly productive and promises to be a major force in the future. The neural network approach is a current trend. A memory-based reasoning approach is proposed by Wang and Iyengar [92]. Powerful parallel computers accommodate more brain-like models of computation [92, 93].

Recently, integration of vision systems with autonomous machines has been investigated in research institutes. At AT&T, Andersson [3] develops a robot ping-pong player which can recognize the ping-pong ball in real time and control the robot

manipulator holding a paddle to hit the ball. Turk et al. [87] are developing an autonomous land vehicle called VITS, which is a multi-leg moving vehicle. At Carnegie-Mellon University, Thorpe, et al. [84] are developing another autonomous vehicle called Navlab, which is an autonomous moving van.

The challenge of future vision systems is to recognize real-world objects in daily life and industrial objects with complex surface properties. The latter is the problem to be solved in this dissertation.

2.2 Vision System Input

The input to a vision system is an image of various kinds. Generally, an image function is a vector-valued function of a small number of arguments. An image function is usually the *digital (discrete) image function* where the arguments to and value of the function are all integers. The same image may be represented by different image functions. The kind of functions used to represent the image depends on what characteristics we want.

Most images are presented by functions of two *spatial* variables

$$f(\mathrm{x}) = \mathrm{f}(\mathrm{x},\mathrm{y}) \tag{2.1}$$

where $f(x,y)$ is the brightness of the gray level of the image at a spatial coordinate (x,y), where $0 \le x,y \le 2^L - 1$, for some positive integer L, and $0 \le f(\mathrm{x}) \le 2^B - 1$, for some positive integer B. Each element of the image is called a *pixel* (picture element). The ranges of x, y, and $f(\mathrm{x})$ are selected in this way for storage and computation on com-

puters. An example of the letter "C" with $0 \le x, y \le 2^3 - 1$ and $0 \le f(x) \le 2^4 - 1$ is shown in Figure 2.1.

0	0	0	0	0	0	0	0
0	0	0	15	15	15	3	0
0	0	7	15	15	15	2	0
0	3	15	15	0	0	0	0
0	3	15	15	0	0	0	0
0	2	15	15	15	15	5	0
0	0	5	15	15	15	2	0
0	0	0	0	0	0	0	0

Figure 2.1: *Image of the Letter "C"*

The above image is usually called the *gray scale image*. If function $f(x,y)$ in (2.1) is the distance from the object to the camera at a spatial coordinate (x,y), then the image which the function represents is called the *range image*. A range image is a dense range map which provides 3-D range information directly. Range image understanding has been a current trend of 3-D machine vision research.

Some special image functions are as follows. A multispectral image **f** is a vector-valued function with components $(f_1, f_2, \ldots, f_n)$. One example of a multispectral image is a color image in which the components measure the brightness values of each of three wavelengths, that is,

$$\mathbf{f}(\mathbf{x}) = \left\{ \mathrm{f}_{\mathrm{red}}(\mathbf{x}), \mathrm{f}_{\mathrm{blue}}(\mathbf{x}), \mathrm{f}_{\mathrm{green}}(\mathbf{x}) \right\}$$

Other examples are time-varying images $\mathbf{f}(\mathbf{x},t)$ which have an added temporal argument and special 3-D images where $\mathbf{x} = (x,y,z)$. In most circumstances, both the

domain and range of f are bounded by certain values.

In robot vision situations, an image is usually viewed as a piecewise-smooth graph surface contaminated by noise. The geometric shape of the image data is emphasized. A 3-D smooth graph surface is a twice-differentiable function of two variables:

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

A piecewise-smooth surface $g(x,y)$ can be partitioned into smooth surface primitives $f_i(x,y)$ over support region R_i:

$$g(x,y) = \sum_{i=1}^{N} f_i(x,y)\chi(x,y,R_i)$$

where $\chi(x,y,R_i)$ is the characteristic function of the region R_i defined as

$$\chi(x,y,R_i) = \begin{cases} 1 & (x,y) \in R_i \\ 0 & \text{otherwise} \end{cases}$$

The vision task is to find the R_i's and the image features associated with each R_i, and then to match the features with the known features about objects in order to recognize and localize objects.

2.3 Machine Vision Process

Current methods of image analysis involve three main levels: (1) low-level vision; (2) intermediate-level vision; and (3) high-level vision; each of which is shown in Figure 2.2.

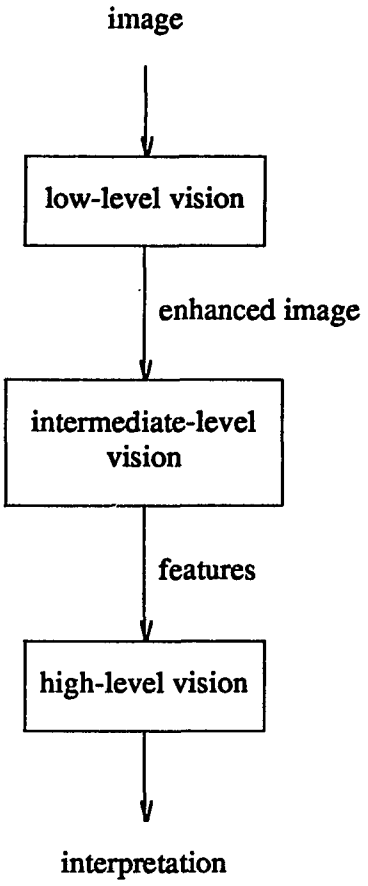

Figure 2.2: *Machine Vision System Structure*

In low-level vision processing, images are suitably pre-processed to remove random noise, enhance the image, compensate for sensor non-linearity, and so on. Also, edge pixels are identified by using various methods such as gradients or Laplacian transformations.

Intermediate-level vision processing involves segmentation and description of images. In the segmentation process, images are subdivided into their constituent parts. This stage is very important, for it is here that objects are extracted. Edge

linking (the process which follows edge detection) is essentially part of this phase. Another approach to image segmentation is splitting and merging. The description process involves extracting the features of objects in a scene.

Finally, in high-level vision, meaning is attached to the objects detected. These approaches to image analysis are best suited for the recognition of known objects, which is the primary application for them.

The vidicon camera is the conventional vision transducer for general purpose robotic tasks. Silicon detectors, called charge-coupled devices (CCD's) and charge-injected devices (CID's), were introduced in the early 1970's [99]. Vidicons depend on an electron beam scanning across an image target to create a signal electrostatically. The beam is deflected and experiences geometric distortion. Silicons generate an electronic signal proportional to incident light.

Increasingly popular state-of-the-art machine vision systems use solid state cameras because of their cost-effectiveness. Sampling intervals in the sense of a square picture of 512×512 dots are usually extracted. Such dots are called *pixels* and are usually composed of 8 bits for gray level images. That is, each pixel can be sensed in one of 256 levels of grayness. In many practical cases, images of 64×64 pixels and 16 gray levels are sufficient and provide an inexpensive transducer [1, 3, 99].

The memory which stores the image while the processor operates on the image is as important as the transducer. These devices are called *frame stores* and are usually developed as standardized chips. Some image processing tasks can be executed in

the frame store [99].

Image processing implies a transformation of an image. This step processes images and extracts image features for recognition. Much can be achieved on the image processing alone [1, 18, 19, 44, 65]. Edge detection is a typical task of image analysis. An example of edge detector is the approximation of the Laplacian operator

$$\frac{\partial^2 E}{\partial x^2} = \frac{\partial^2 E}{\partial y^2} = \frac{4}{\varepsilon^2}\left(\frac{1}{4}(E_{i-1,j} + E_{i,j-1} + E_{i+1,j} + E_{i,j-1}) - E_{i,j}\right)$$

Region growing can be viewed as the complement of edge detection. Region growning groups the image pixels into different segments, usually each of which has a distinct feature. Texture analysis is very useful to remote sensing; however, it is of little use to robot vision. Motion finding by analyzing optical flow is also very helpful for keeping track of moving rigid objects.

Finding range information from object geometry is another typical task for robot vision systems. Stereo vision and triangulation are the conventional way to find range information. Time of flight method is currently being developed; however it is cost prohibitive in applications. The structured light approach has been used in some environments where the light condition is easily controlled [85].

Recognition approaches can be categorized as *adaptive* or *algorithmic* methods. The adaptive system calculates the key differences between different objects and the similarities of objects within a particular class [1, 13, 92]. Conventional statistical pattern recognition approachs can fall into this category. This learning approach gives

a system much flexibility. The problem with this approach is that it is usually time-consuming to initialize the system.

The algorithmic approach is employed when the objects to be recognized are well-defined [3, 14, 16, 28, 48]. Traditional graph-theoretic algorithms are one way to match objects with a structure description. Symbolic logic is used to do inference. Production systems are typically used as symbolic inference systems. Scene labeling and constraint relaxation methods are used in some applications. Many AI techniques are used in recognition.

Localization is required in robot vision, but conventional computer vision does not tackle this problem. The information about the location and orientation of the object is needed for robot manipulator planning. Just recognizing an object is not enough in robotic applications. The robot system has to know the exact object location and orientation in order to control the robot actions on the object.

From Figure 2.3, we can see that the robot control and action blocks need both recognition and localization information about the object.

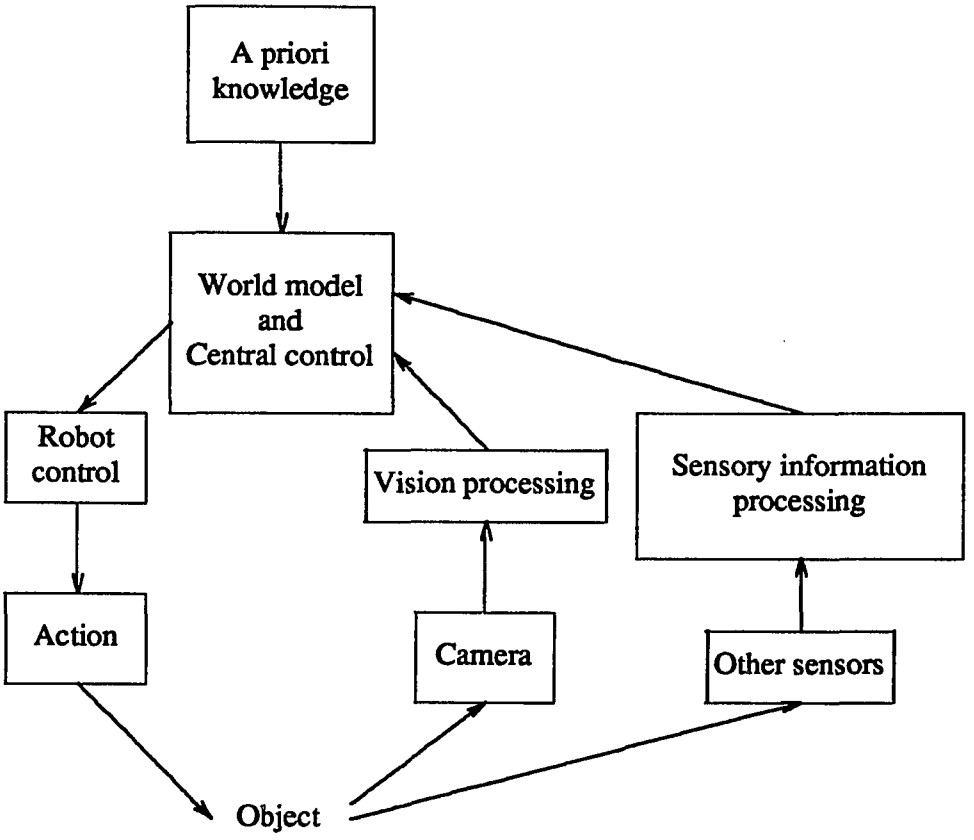

Figure 2.3: *Structure of a robot system with vision*

2.4 Characteristics of Robot Vision Systems

Some of the important performance characteristics that should be considered include the following [3, 35, 66, 73, 83].

The image resolution determines the ability of a vision system. The image resolution is the number of pixels in the image array and the image sensor's field of view. For a standard array of 256×256 pixels, the system can perceive pieces of the object as small as .0039″ if the image is $1 \times 1 in^2$. For an array of 512×512 pixels,

resolution would improve to .0020".

Resolution can be improved by using a higher magnification lens, but the field of view will then shrink. Vidicons have higher resolution than CCDs; however, vidicons are fragile and heavy.

A vision system should have a speed consistent with the speed at which parts are being presented. Typical vision systems can inspect and recognize simple parts at rates of 2-15 items per second, or higher. Slower rates will not be acceptable in applications since human workers can be more effective.

Silicon cameras can achieve higher speeds than vidicon cameras. However, silicon cameras do not have as high a resolution as vidicons. The speed of a vision system depends on the complexity of images, number of pixels, and bits per pixel.

The ability of a vision system to discriminate among variations in light intensity is determined by the number of intensity thresholds, i.e., levels of gray scale. In range images, it is the levels of distance. The trade-off is that better discrimination means increased processing time along with a higher computer memory capacity.

A trade-off can be made between speed and the ability to interpret images correctly. A higher probability of correct interpretation can be achieved by processing more image features, which increases the processing time. An acceptable accuracy rate (90%, 95%, etc.) depends on the accuracy required by the application, as in any quality control situation.

The vision system must be flexible enough to accommodate variations of multiple copies of a given part, as well as uncertainties in part placement due to individual workstation configurations. Furthermore, many robot vision tasks are distinguished by their performance in dirty and uncontrolled environments.

2.5 Scope of this Dissertation

Our approach is to recognize and localize multiple 3-D free-form objects in range images based on boundary description models, such as CAD models.

Recently, various programs and systems have been developed to derive accurate and dense range maps in times that may eventually become realistic [8, 24, 34, 46, 54]. Recent techniques for actively obtaining range images can be seen in Besl's [8] survey paper. Nitzan's [62] survey paper also presents some range imaging techniques. Range images explicitly contain depth information on the environment, which is needed in order to interpret 3-D objects in the scene.

Model representation has a significant effect on model-based recognition [9, 21]. Recent research pays more attention to model surface features. Without using surface properties, many important industrial vision tasks would remain beyond the competence of machine vision systems. Numerical features about lines and topological features about connectivity suffer from partial occlusion. Geometric features such as equations of curves and surfaces are much more stable and suitable for recovering object location.

CAD models are very stable in representing geometric features of 3-D objects and reveal model structures in detail. They are perfect for producing images from models. CAD systems provide interactive design interface which is usually user friendly. These systems help create, modify, and analyze a design.

The most general CAD database description of a surface or part is in terms of discrete points, which will be introduced in more detail in Chapter 3. The surface patch is an approximation to the characteristic polyhedron.

A coordinate frame is associated with the CAD database; this is the frame in which the part is designed and viewed on the screen of a CAD station. The location of the CAD frame in the world frame is obtained by a transformation $\mathbf{T}_l$ where

$$\mathbf{T}_l = \begin{bmatrix} n_1 & o_1 & a_1 & t_1 \\ n_2 & o_2 & a_2 & t_2 \\ n_3 & o_3 & a_3 & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

If $\mathbf{x}$ is some point on the part expressed in the CAD frame, then $\mathbf{T}_l\mathbf{x}$ is the same point in the world frame.

Another reason for a CAD model based vision is the wide availability of CAD/CAM systems in industry. Most industrial parts are designed and manufactured using CAD/CAM systems. Therefore, a CAD database is a good candidate for the model base of machine vision systems.

However, machine vision, as a reverse process, cannot easily use CAD models directly. Some research has been attempting to extract model features from CAD database for vision tasks. Bhanu and Ho [11] propose a CAD-based approach to building representations for 3-D object recognition. It is nontrivial to construct 3-D representations for vision tasks. Gunnarsson and Prinz [38] propose a method of locating an object using its CAD model. Their method assume a very limited rotation of the object.

We have developed a robust and efficient paradigm for CAD-model based robot vision from range images. The CAD representation we use is boundary representations which are the commonly used scheme in computer graphics [25, 29, 32, 59, 72]. A rigid body is represented by segmenting its boundary into a finite number of bounded subsets called surface patches. The boundary representation is unambiguous, but is not unique.

Many segmentation methods [23, 39] segment range images into surface patches in ad hoc representations. However, the surface patches derived by these segmentation procedures might be much different from those of the CAD surface patches although they may both refer to the same rigid body.

Our approach for recognition and localization is to build a knowledge base about object surface shapes from boundary representation models such as CAD models. Small segment regions in the input images are used to perform the heuristic search to form hypotheses, at which time CAD representations are used directly to match with

the range data for verifications.

Chapter 3: Review of Literature

3.1 Introduction

Any robotic system capable of executing complex sequences of dexterous tasks in manufacturing environments must have good perception ability. Typically, perception involves the integration of sensory information derived from a large number of close proximity, loosely coupled sensors into a pre-conceived knowledge and control structure.

Intelligent robotic systems can be structured and programmed in a task directed manner. Philosophically, we imply that the usefulness of an intelligent robot is justified in examples of automatic part recognition and localization in flexible manufacturing environments. All of its planning, control, and sensing processes are task-directed. A perception process which provides knowledge to a system about relationship that exists between sensed events and tasks is a task-directed perception process.

Planning and/or replanning a new task by a robot requires a vision system which can recover 3-D information from projected images of range sensors. The problem domain chosen in this dissertation is a model-based 3-D object recognition and localization using geometric modeling. Various approaches have been attempted to recognize 3-D objects in a form which is suitable to vision systems. In the following section, we review these methods and present a summary of these techniques in the form

of a table.

Before introducing various methods, we will explain some terms used throughout this dissertation. *Range images* refer to images where each pixel has associated with it the distance between the camera and the corresponding point on the object. *Objects* refer to rigid bodies or solids. Each object should have a model in the computer system for model-based vision. *CAD models* refer to models in a CAD database, which are designed by various computer-aided design methods. *Recognition* means identifying the objects in the input images according to the model database. *Localization* means determining the location and orientation of the objects.

3-D machine vision processes are composed of many components which perform particular functions in coordination with other components. First, images must be taken from a 3-D scene. Input images may be light intensity images, color images, infrared images, or range images. Range imagery is the current trend in 3-D vision research since a range image itself contains explicit 3-D information. A *range image* is a dense range map which is obtained directly by measuring point distances or is derived from multiple images. The rest of the processing sequence depends on tasks, objects, and *a priori* information, which basically consists of feature extraction, as an intermediate level process, and matching features with models, as a high level process.

Use of range data in 3-D vision research has been a current research trend. Studies were scarce until range images recently became easily obtainable. Range images

provide more reliable information on geometries of scenes than light intensity images. The information facilitates the solution of higher level problems, such as matching 3-D models to range data with multiple objects occluding one another.

3.2 Prior Approaches

Derivation of 3-D descriptions of objects from two-dimensional projected images is central to machine vision research. Towards this objective, Chien, et al. [20] present a scheme of generating volume/surface octrees from range images for representing the volume of 3-D objects. Flynn and Jain [31] describe a method for classifying a surface as planar or nonplanar through two hypothesis tests. Naik and Jain [60] propose spline-based descriptions of objects from range images by segmenting a range image and deriving spline-based descriptions for each segmented surface. Han and Volz [39] present a segmentation method of grouping range image regions and constructing a region boundary graph. Besl and Jain [7] present an algorithm of segmentation through variable-order surface fitting using an iterative region growing method. The segmentation method of Cohen and Rimey [23] segments a range image into three types of surfaces of planes, cylinders and spheres. Hoffman and Jain [43] present an approach to dividing and classifying a range image into planar, convex, and concave surfaces. Yokota and Levine [97] propose a region and edge-based hybrid method to partition images into segments with the same curvature properties. Vemuri, et al. [90] propose a curvature-based segmentation method which partitions images into segments with same curvature signs.

Early stage image processing is useful only when its output is helpful to later stages of vision processes. Therefore, which early stage image processing method is better really depends on many factors. Our approach mainly concerns the intermediate level and high level vision processes. The fundamental issues of 3-D machine vision are recognition and localization of 3-D objects.

There are some applications where the object is known and the localization of the object is the only concern. Bolle and Cooper [13] propose a method for estimating 3-D object position from range data. They introduce a Bayesian parameter estimation for data sets described by a combination of algebraic, geometric, and probabilistic models. One problem might be that the stored parameters in the reference models are often different from the parameters used in the data generation models.

Bolles and Horaud [14] present an approach to finding the configuration of objects from range data by matching preselected features. Their method starts with a distinctive feature, such as the edge at the end of a cylindrical part, and then grows a match by adding compatible features one at a time. The method is edge-based, thus, it is sensitive to occlusion and noise.

Gunnarsson and Prinz [38] propose a method for localization of industrial parts using CAD models. The method calculates the shortest distance between an object surface and a model surface and then the model iteratively approaches the object. However, the method assumes a very limited rotation of the objects from the models.

Vemuri and Aggarwal [91] propose a method for determination of the orientation of an object from a range image by searching for one corresponding point on the surfaces between the view of an object and the object model. Since real images are noisy, we suspect that the method can be useful in applications.

In more realistic situations, there are objects to be recognized and localized. The approach proposed by Oshima and Shirai [63] for recognition of objects from range images uses models built by the system itself during a learning phase. A description of each scene is built in terms of properties of regions and relations between them, which is stored as an object model. The recognition matching process is a combination of data-driven and model-driven search process, which selects a kernel region in the image and develops a hyperthesis of probable models, and then performs a matching process between the object and the model. Multiple models built for one object during the learning phase might cause problems in recognition since different view angles may generate structurally different models. Occluded surfaces can also cause the recognition to fail.

Faugeras and Hebert [28] present an approach to 3-D object recognition and localization. Their approach uses linear primitives such as points, lines and planes to approximately represent object models. The approach finds some pairings between object primitives and model primitives such that the pairings satisfy constraints on rigidity. Then, a hyperthesis is formed and verified against the model. Occlusion might be a problem to this approach. Hyperthesis formation is another major issue which is not solved satisfactorily in the paper.

Silberberg, et al. [73] present an algorithm for recognition of 3-D objects in two-dimensional intensity images. The objects to be recognized are polygons. The algorithm uses a generalized Hough transform to match image junctions to model vertices to estimate the transformation. The initial hypothesis of probable models to be matched may be the major problem in addition to the occlusion problem.

Horn [44] describes using Gaussian sphere called the extended Gaussian image (EGI) to represent objects for 3-D recognition. In the case of convex objects, the EGI representation of the object is unique. Iterative algorithms have been used to recover a convex solid from its extended Gaussian Image. One problem could be that intensive computation is required.

Grimson and Lozano-Perez [35] discuss using sparse local measurements of positions and surface normals to recognize and locate objects. Objects are modeled as polyhedra (or polygons). Their approach examines all hypotheses about pairings between sensed data and object surfaces and discards certain ones by using local constraints on distances between faces, angles between face normals, and angles of vectors between sensed points. The method depends on finding the local constraints to perform the search efficiently.

Wong et al. [96] propose an approach to 3-D object recognition based on attributed hypergraphs. The approach recognizes objects by finding the graph monomorphism between the attributed hypergraph representations for object models and the complete attributed hypergraph representation of an object. A problem might arise if

the attributed hypergraph of objects is different from attributed hypergraph of models due to noise in images. Furthermore, the method is computationally intensive.

Bhanu and Nuttal [10] present a method for recognition of 3-D objects. The method characterizes surface curvatures on curvature graphs to recognize objects. The method is effective for simple surface type objects, such as spheres, cylinders, and cubes, as their experiment showed. However, the method might fail for complex surface objects since a curvature graph is a two-dimensional plane with the principal curvatures being the coordinate axes. Thus, some information about the objects are lost after the mapping, and complex surface objects might be mapped all over the curvature graph.

The following table summarizes this review of prior research and comparison with our work.

Summary

Researcher	Image	Model	Features for Search	Object Type	Output
Bolle and Cooper, 86	Range	Parameter Vectors	Planes, Cylinder, and Spheres with boundary edges to form parameter vectors (experiment was only on objects formed by planar patches)	Single Solid	Localization
Bolles and Horaud, 87	Range	Extended CAD models, feature classification networks, planar patch models, and wire-frame models	Edges	Multiple solids of same model (industrial part)	Localization
Gunnarsson and Prinz, 87	Range	CAD models	Surfaces	Single free-form solid with limited rotation	Localization
Vemuri and Aggarwal, 88	Range	Boundary representation	One point curvature on a surface patch (noise-sensitive)	Single free-form solid with limited transformation	Localization
Oshima and Shirai, 83	Range	Description of properties (planes and quadrics) of regions and relations between them in a scene	Surfaces (planes and quadrics) with boundary edges	Multiple Solids	Recognition and localization
Faugeras and Hebert, 86	Range	Planar and quadric surface patches with boundary edges to approximate free-form solids	Surfaces (planes and quadrics) with boundary edges	Single free-form solid	Recognition and localization

Researcher	Image	Model	Features for Search	Object Type	Output
Silberberg et al., 86	Intensity	Vertex points	Edge junctions	Multiple Polyhedra	Recognition
Horn, 86	Intensity	Extended Gaussian images	Surfaces	Multiple convex free-form solids	Recognition and localization
Grimson and Lozano-Perez, 87	Range	Constraint descriptions between planar surfaces, edges, and points	Planar surfaces and edges together	Multiple polyhedra	Recognition and localization
Wong et al., 89	Range	Attributed Hypergraph	Surfaces with boundary edges	Single Solid (surfaces are isolated by edges and each surface has an explicit attribute)	Recognition
Bhanu and Nuttal, 89	Range	Curvature graphs (2D plane with principal curvatures as coordinate axes)	Surfaces	Simple Solid, such as sphere, cylinder, etc., whose curvatures should be clusters on curvature graphs	Recognition
Wang and Iyengar, 89	Range	Boundary representation, such as CAD models	Surfaces	Multiple free-form solids	Recognition and localization

Chapter 4: Three-Dimensional Object Models

Geometric modeling is the technique used to describe the shape of an object. Much of the power of current geometric modeling resides in its techniques for synthesizing and assisting us in easily describing complex shapes. Geometric modeling provides a description or model which is analytical, mathematical, and abstract. The importance of geometric modeling is rapidly increasing in many fields. It is a primary ingredient in CAD/CAM systems, computer graphics, computer art, animation, simulation, computer vision, and robotics.

An automated manufacturing system must have at least three components: a CAD/CAM system, a perception system, and a robot system. Figure 4.1 shows the structure of such an automated manufacturing system and the connections among the components.

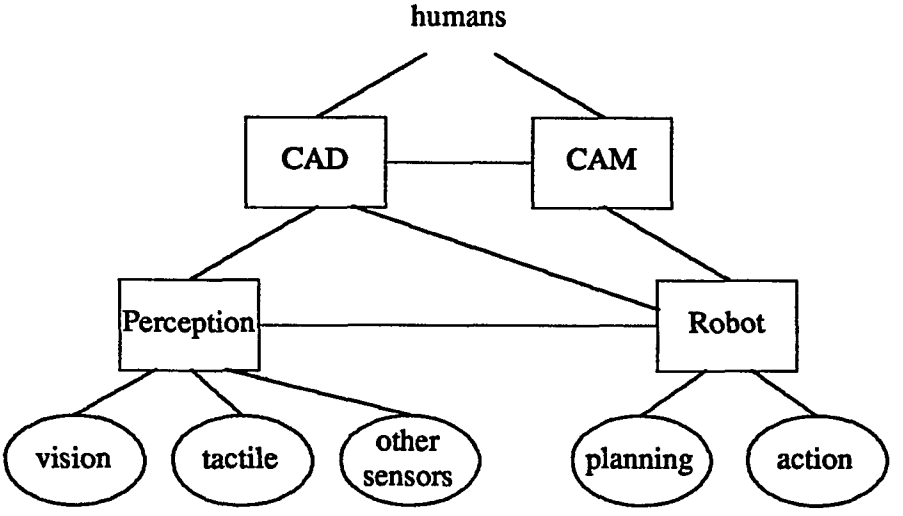

Figure 4.1: *Components of an Automated Manufacturing System*

The CAD/CAM system supports the design, analysis, simulation, and manufacturing of products and parts. The perception system integrates information from various sensors, such as visual, tactile, and ultrasonic sensors. The perception system provides the robot system with information concerning the environment and the identity, location, and orientation of the parts to be handled by the robot. The robot system plans the actions of its manipulators using the 3-D information provided by the perception system and then performs the actions. The robot system planning is task-directed, which is also controlled by the CAM system.

Many vision systems use models generated in an ad hoc manner, which have no relation to the CAD/CAM system, where the industrial objects are originally designed and manufactured. A unified system which allows vision models to be automatically generated from existing CAD databases or to use CAD models directly is desired.

4.1 Models for Machine Vision

Computer graphics is a process of generating images from object models. Computer vision can be viewed as a reverse process, which abstracts objects from images according to object models. Computer graphics has been a very successful field. Computer vision, however, is still in its infancy.

To recognize and localize rigid objects at close distance is one of the most applicable subfields of computer vision, which is also called robot vision. Industrial robots need visual ability to handle complex environments. Most industrial robots nowadays can only perform routine actions because of their lack of visual ability.

Our task is to design a robot vision system which can recognize and localize machine parts. In industrial environments, parts to be recognized are often predefined in CAD systems since CAD tools are available in most plants. Therefore, CAD models of solid objects are natural candidates for models for robot vision.

Model representation has a significant effect on model-based object recognition. Historically, many modeling methods have been proposed and used in computer vision systems. We will just introduce some approaches seen frequently.

Wireframe models which are used to describe solid object edges are commonly used in blockworld vision [71]. Wireframe models can almost be considered the earliest machine vision models. However, these models cannot handle more complex objects, since one description can correspond to several objects.

One way of building models is to take several images of one part and use hand measurements to find features. The features found are to be stored in the system [4]. Though this approach is cumbersome and crude, it is still used in many machine vision systems.

Another approach often seen is to use surface primitives to construct 3-D object models [48]. A heuristic algorithm is usually applied to place partially overlapping surface segments into one coordinate space. This approach is also troublesome and cannot obtain many useful features for recognition and localization.

Herman [40] tried to develop an automatic approach to generating object models. The approach combines solid object points from a sequence of range images corresponding to various views of the object and then applies some transformations to obtain the surface points of the complete object. This method may not be able to obtain all the necessary surface points, as compared to a CAD system which may easily describe an object by its surface points.

However, automatic model construction from multiple views of an object is a good approach for robot vision models, in addition to the CAD model approach. Especially in a situation where CAD models do not exist for some objects, the system's being able to obtain models by observing objects is absolutely a plus for the machine vision system. In other words, the system has learning ability.

The approach we introduce is suitable for using either CAD models or automatically generated models to recognize and localize objects. We will introduce the

similarity of these two kinds of models. In fact, automatic models are the same as CAD models in the sense of their data structures. Certainly the processes of building them are different. Therefore, we do not distinguish between these two kinds of models when we discuss using the models to build the knowledge base and to recognize objects. We first introduce the CAD models and then the automatic models in the last section.

4.2 Computer-Aided Design Models

CAD models contain details about solid objects. A CAD system is generally used to design new shapes for automatic manufacture. It provides an interactive design interface, which is usually user friendly, and helps create, modify, and analyze a design. CAD models are very stable in representing geometric features of 3-D objects and revealing model structures in detail. Another reason for CAD model-based vision is the wide availability of CAD/CAM systems in industry.

CAD models are perfect for producing images from models. However, machine vision, as a reverse process, does not easily use CAD models directly. We have developed an approach to use CAD models to generate hyperthesis images for verification. Our approach uses the knowledge about surface shapes of objects to perform recognization reasoning. The knowledge about the surface shapes of objects are abstracted from CAD models automatically. How to abstract knowledge about surface shapes from CAD models and store it in the system for later use will be discussed in the next two chapters.

There exist many schemes for representing solids. Among them, the most popular schemes are boundary representations (B-rep), constructive solid geometry (CSG), sweep representations, cell decompositions, spatial occupancy enumeration, primitive instances, and analytic solid modeling (ASM) [25, 32, 55].

B-rep represents a solid by its enclosing surface, which is defined by a finite number of faces or patches, which are represented in terms of bounding edges and vertices. Sweep representations are defined as the volume swept by a finite set of cross sections along some axes under some sweeping rules. Primitive instances are individual objects within a family, called generic primitives. Spatial occupancy enumeration represents a solid by a list of voxels (volume elements), which are cubes of fixed size and located in a fixed spatial grid. Cell decomposition is a generalization of spatial occupancy enumeration. Cell decomposition is a breakdown of a solid into arbitrary cells with a representation of each cell in the decomposition. CSG may be considered a generalization as a super-set of primitive instances, spatial occupancy enumeration, and cell decomposition. ASM is an extension to B-rep with the addition of mathematically described solids.

In robot vision, B-rep has many advantages. Though B-rep does not provide information about the interior of the model, it describes sculptured, free-form shapes precisely. Furthermore, B-rep does represent details about the object surfaces which are the only visible portion of the object. On the other hand, the finite nature of available primitives makes modeling of arbitrarily sculptured objects difficult with CSG.

Since only object surfaces are visible in robot vision, surface information is important. The surface evaluation of CSG models is computationally intensive, and an evaluated model requires a great deal more storage than an unevaluated one. B-rep gives surface information naturally. Thus, B-rep is a good candidate for model representation of robot vision.

Also, there exist exact conversion algorithms to B-rep from other aforementioned representations [55]. Hence, we chose B-rep CAD models to develop vision models for robot vision. Another reason for choosing B-rep is that we believe that intrinsic properties of a surface are more robust in robot vision, while edge-based recognition is subject to occlusion and noise. However, surface-based recognition is more computationally intensive. Our approach is surface-based in order to recognize free-form, sculptured solids. We dramatically reduce a great deal amount of computation using techniques introduced in the following chapters.

4.3 Boundary Representation of Solids

A boundary representation of a solid object m can be defined as a set

$$m = \left\{ f_1, f_2, \ldots, f_{p_m} \right\} \tag{4.1}$$

where f_i, $1 \leq i \leq p_m$, is a surface patch which can be defined in various ways. There are many conventional functions to describe surface patches in computer graphics, though there is no unique B-rep for a solid.

Fortunately, we are mainly concerned with the intrinsic properties of the surfaces of solids in our approach, although topological information on solids is also needed for recognition and localization. In fact, (4.1) represents the topological information implicitly since each f_i has a fixed spatial position and orientation. If more information is necessary, it can be derived from (4.1). The advantage of our approach is that the topological information about objects does not need to be abstracted from models, i.e., our approach uses these models directly. In addition to the models, our approach builds a knowledge base about the surface shapes of solids. The knowledge base helps the recognition and localization reasoning increase efficiency dramatically.

Methods of designing surface patches in CAD systems often use a set of discrete points, called control points, to help define surface patches. Since detailed discussion regarding defining functions is beyond the scope of this dissertation, we only introduce some commonly used functions, such as B-spline surfaces or a Bezier surface.

The B-spline surface is defined in terms of a characteristic polyhedron. The shape of the surface approximates the polyhedron. The surface is defined by a blending function

$$\mathbf{p}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} \mathbf{p}_{ij} N_{i,k}(u) N_{j,l}(v) \qquad u,v \in [0,1]$$

where $\mathbf{p}_{ij}$s are the vertices of the defining polyhedron. Boldfaced letters represent vectors in x, y, z, e.g. $\mathbf{p}_{ij} = (x_{ij}, y_{ij}, z_{ij})$. $N_{i,k}(u)$ and $N_{j,l}(v)$ are B-spline blending functions defined recursively by the following:

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } u_i < u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

and

$$N_{i,k}(u) = \frac{(u-u_i)\,N_{i,k-1}(u)}{u_{i+1}-u_i} + \frac{(u_{i+k}-u)\,N_{i+1,k-1}(u)}{u_{i+k}-u_{i+1}}$$

where k and l denote the parameters that control the continuity of the surface and the degree of the blending function polynomials.

A biquadratic B-spline patch can be defined by

$$\mathbf{p}_{ij}(u,v) = N(u)\,[\mathbf{p}_2]\,N(v)^T$$

where

$$N(u) = \left[\frac{(1-u)^2}{2} \quad \frac{-2u^2+2u+1}{2} \quad \frac{u^2}{2}\right], \qquad 0 \le u \le 1$$

and $N(v)$ is similarly defined as

$$N(v) = \left[\frac{(1-v)^2}{2} \quad \frac{-2v^2+2v+1}{2} \quad \frac{v^2}{2}\right], \qquad 0 \le v \le 1$$

and

$$[\mathbf{p}_2] = \begin{bmatrix} \mathbf{p}_{i,j} & \mathbf{p}_{i,j+1} & \mathbf{p}_{i,j+2} \\ \mathbf{p}_{i+1,j} & \mathbf{p}_{i+1,j+1} & \mathbf{p}_{i+1,j+2} \\ \mathbf{p}_{i+2,j} & \mathbf{p}_{i+2,j+1} & \mathbf{p}_{i+2,j+2} \end{bmatrix}$$

A biquadratic B-spline patch is described by nine control points, which can be seen in Figure 4.2.

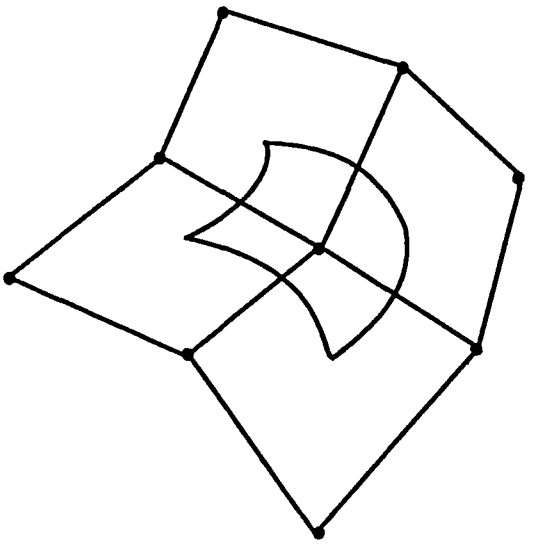

Figure 4.2: *Biquadratic B-spline surface*

A bicubic B-spline function can be

$$\mathbf{p}_{i,j}(u,v) = N(u)\,[\mathbf{p}_3]\,N(v)^T$$

where

$$N(u) = \left[\frac{(1-u)^3}{6} \quad \frac{3u^3-6u^2+4}{6} \quad \frac{-3u^3+3u^2+3u+1}{6} \quad \frac{u^3}{6}\right], \qquad 0 \le u \le 1$$

$N(v)$ is similarly defined as

$$N(v) = \left[\frac{(1-v)^3}{6} \quad \frac{3v^3-6v^2+4}{6} \quad \frac{-3v^3+3v^2+3v+1}{6} \quad \frac{v^3}{6}\right], \qquad 0 \le v \le 1$$

and

$$[\mathbf{p}_3] = \begin{bmatrix} \mathbf{p}_{i-1,j-1} & \mathbf{p}_{i-1,j} & \mathbf{p}_{i-1,j+1} & \mathbf{p}_{i-1,j+2} \\ \mathbf{p}_{i,j-1} & \mathbf{p}_{i,j} & \mathbf{p}_{i,j+1} & \mathbf{p}_{i,j+2} \\ \mathbf{p}_{i+1,j-1} & \mathbf{p}_{i+1,j} & \mathbf{p}_{i+1,j+1} & \mathbf{p}_{i+1,j+2} \\ \mathbf{p}_{i+2,j-1} & \mathbf{p}_{i+2,j} & \mathbf{p}_{i+2,j+1} & \mathbf{p}_{i+2,j+2} \end{bmatrix}$$

A bicubic B-spline patch is described by sixteen control points, as shown in Figure 4.3.

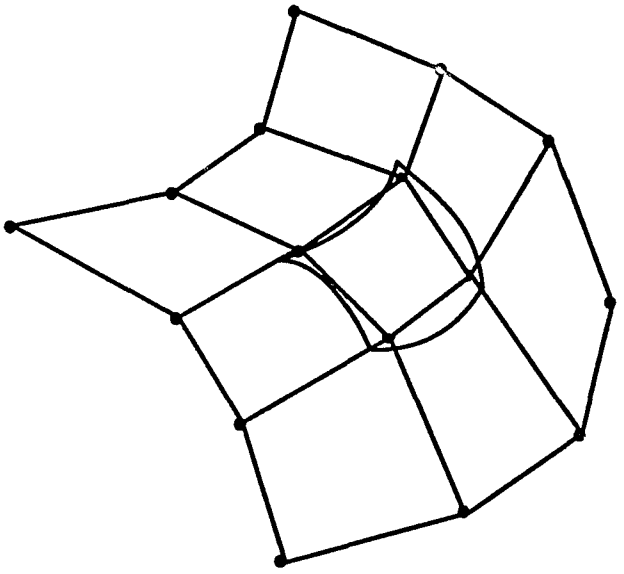

Figure 4.3: *Bicubic B-spline surface*

A Bezier surface is also defined in terms of a characteristic polyhedron. The surface is defined by a blending function

$$\mathbf{p}(u,v) = \sum_{i=0}^{m}\sum_{j=0}^{n} B_{i,m}(u)B_{j,n}(v)\mathbf{Q}_{ij} \qquad u,v \in [0,1]$$

where $\mathbf{Q}_{ij}$ are the vertices of the defining polyhedron, and

$$B_{i,m} = \binom{m}{i}(1-u)^{m-i}u^{i}$$

A bicubic Bezier surface patch is defined as

$$\mathbf{p}(u,v) = B(u)\,[\mathbf{Q}]\,B(v)^T$$

$$B(u) = \begin{bmatrix}(1-u)^3 & 3u(1-u)^2 & 3u^2(1-u) & u^3\end{bmatrix}, \quad u \in [0,1]$$

$$B(v) = \begin{bmatrix}(1-v)^3 & 3v(1-v)^2 & 3v^2(1-v) & v^3\end{bmatrix}, \quad v \in [0,1]$$

and

$$[\mathbf{Q}] = \begin{bmatrix}\mathbf{Q}i-1,j-1 & \mathbf{Q}i-1,j & \mathbf{Q}i-1,j+1 & \mathbf{Q}i-1,j+2 \\ \mathbf{Q}i,j-1 & \mathbf{Q}i,j & \mathbf{Q}i,j+1 & \mathbf{Q}i,j+2 \\ \mathbf{Q}i+1,j-1 & \mathbf{Q}i+1,j & \mathbf{Q}i+1,j+1 & \mathbf{Q}i+1,j+2 \\ \mathbf{Q}i+2,j-1 & \mathbf{Q}i+2,j & \mathbf{Q}i+2,j+1 & \mathbf{Q}i+2,j+2\end{bmatrix}$$

A bicubic Bezier surface patch is shown in Figure 4.4. Notice that the four corner points are on the surface itself, as compared with the B-spline surface patch, where the control points do not lie on the surface.

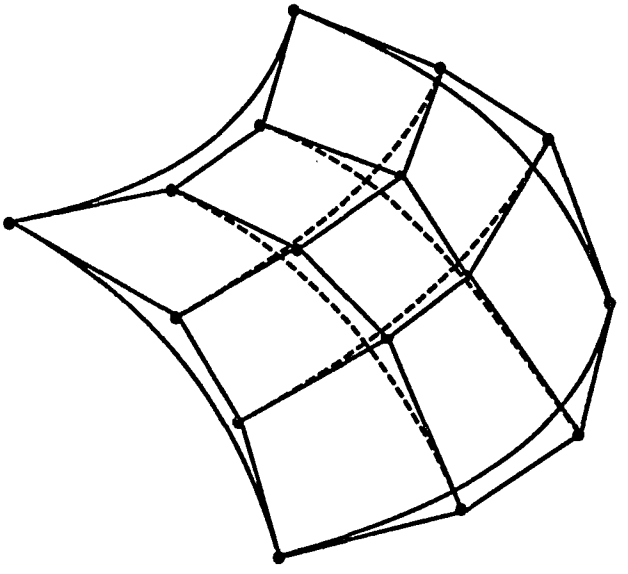

Figure 4.4: *Bicubic Bezier Surface Described by Sixteen Control Points*

4.4 Models Generated Automatically

The fundamental principle of automatically building object models is as follows. Multiple views of a solid object are first taken. Segmentation on each image is performed. Each image is then divided into a set of smooth surface segments, each of which is represented in certain mathematical function. Since the angles between different views can be predetermined, the surface segments in one view can be performed a transformation and then connected with surface segments in another view.

Segmentation has been a very active research field. Therefore, there are many different ways of representing surface segments. Most automatic model generation

research concentrates on creating models in parametric function representation as was shown in the last section. The reason is that compatibility with CAD models is emphasized. If automatic models are described in the same way as CAD models, our approach can certainly use them to build knowledge base and to recognize objects.

There are segmentation methods [7] which describe surface segment regions by an explicit bivariate function, $z = f(x,y)$. For instance, an image can be partitioned into smooth surface segments described by variable-order explicit bivariate functions. A first order function

$$z = a_0 + a_1 x + a_2 y$$

describes a planar surface. Second order functions

$$z = a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2$$

describe biquadratic surfaces. Third order functions

$$z = a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2 + a_6 x^3 + a_7 x^2 y + a_8 xy^2 + a_9 y^3$$

describe bicubic surfaces. Fourth order functions

$$\begin{aligned} z = {} & a_0 + a_1 x + a_2 y + a_3 x^2 + a_4 xy + a_5 y^2 + a_6 x^3 + a_7 x^2 y \\ & + a_8 xy^2 + a_9 y^3 + a_{10} x^4 + a_{11} x^3 y + a_{12} x^2 y^2 + a_{13} xy^3 + a_{14} y^4 \end{aligned}$$

describe biquartic surfaces. Functions more than order four are not often used in segmentation since they will be confused with the noise or edges in the images.

Explicit functions are not often used in automatic model generation research since they are not compatible with CAD models. However, these models can also be used in our approach. Our approach only requires models given in the form of (4.1). The f_i in (4.1) can be either represented in parametric functions, as CAD models, or

in explicit functions as shown above.

Therefore, our approach allows an automatic way of generating models for recognition and localization, which makes a vision system self-contained. On the other hand, if the same segmentation procedure is used to partition images in automatic model generation and to segment the input image in recognition, the recognition and localization process will be more accurate and more efficient.

Chapter 5: Invariant Properties of Surfaces

If surfaces can be recognized by their characteristics, object recognition can thus be decomposed into a surface recognition problem, which is the so-called surface-based recognition. This is in contrast to the conventional edge-based recognition that recognizes objects by using edge characteristics and their relations. In order to recognize surfaces, we must have well-defined features, or mathematical entities, which can be used to distinguish between different entities of the same type.

It is well known that curvature, torsion, and speed uniquely define the shape of 3-D curves [37, 45, 52]. In the surface case, there are two basic mathematical entities which are considered in the analysis of smooth surfaces. They are referred to as the first and second fundamental forms of a surface. We will show how these forms uniquely characterize and quantify a general, smooth surface shape.

Based on these fundamental forms, invariant surface characteristics, such as the Gaussian curvature and the mean curvature, are derived. These characteristics are *invariant* to changes in surface parameterization and to translation and rotations of object surfaces. A robust 3-D object recognition system should be view-independent. Therefore, the use of invariant surface characteristics in 3-D vision systems is significant. Furthermore, the Gaussian curvature and the mean curvature are local surface properties, which allow surface curvature to be used in occlusion situations.

5.1 Surfaces

A curved surface can be defined as a polynomial in terms of two parameters u and v. The surface S is a set of points in 3-D space. The representation

$$\mathbf{p} = \mathbf{p}(u,v) = [x(u,v), y(u,v), z(u,v)] \tag{5.1}$$

is a mapping of an open set U in the uv plane onto S. If for all (u,v) in U

$$J_x^2 + J_y^2 + J_z^2 \neq 0 \tag{5.2}$$

where J_x^2, J_y^2 and J_z^2 are Jacobians defined as

$$J_x = \frac{\partial(y,z)}{\partial(u,v)} \tag{5.3}$$

$$J_y = \frac{\partial(x,z)}{\partial(u,v)} \tag{5.4}$$

$$J_z = \frac{\partial(x,y)}{\partial(u,v)} \tag{5.5}$$

and all derivatives of $\mathbf{p}(u,v)$ of up to order m exist, and all such derivatives are continuous, then the curved surface defined by (5.1) is said to be of class C^m.

Condition (5.2) guarantees that the curved surface will not degenerate to a point or a curve, and that it does not contain any singular points. This condition requires constraints on both the curve itself and the parameters. We will assume that the condition holds, since all our geometric models are generated by piecewise-smooth surfaces, and the images are segmented into piecewise-smooth surfaces.

A parametric representation will be denoted by $\mathbf{p} = \mathbf{p}(u,v)$ and its partial derivatives by

$$\mathbf{p}_u = \frac{\partial \mathbf{p}}{\partial u}$$

$$\mathbf{p}_v = \frac{\partial \mathbf{p}}{\partial v}$$

$$\mathbf{p}_{uu} = \frac{\partial^2 \mathbf{p}}{\partial u^2}$$

$$\mathbf{p}_{vv} = \frac{\partial^2 \mathbf{p}}{\partial v^2}$$

$$\mathbf{p}_{uv} = \frac{\partial^2 \mathbf{p}}{\partial v \partial u}$$

If $\mathbf{p}$ is class $m \geq 2$, then

$$\mathbf{p}_{uv} = \mathbf{p}_{vu}$$

Though strictly speaking, a parametric representation (5.1) is a mapping, we will speak rather loosely and identify it with its image, a set of points S. Therefore, we say that P is a point on $\mathbf{p} = \mathbf{p}(u,v)$ when P is a point on the image of $\mathbf{p} = \mathbf{p}(u,v)$, or we might even say that the parametric representation $\mathbf{p} = \mathbf{p}(u,v)$ is contained in S when the image of $\mathbf{p} = \mathbf{p}(u,v)$ is a subset of S.

5.2 First Fundamental Form

A surface in 3-D space is uniquely determined by certain local invariant quantities called the first and second fundamental forms [37, 45, 52, 89]. Let $\mathbf{p} = \mathbf{p}(u,v)$ be a parametric surface patch of class ≥ 1, then the first fundamental form is

$$\begin{aligned} I &= d\mathbf{p}\cdot d\mathbf{p} \\ &= (\mathbf{p}_u du + \mathbf{p}_v dv)\cdot(\mathbf{p}_u du + \mathbf{p}_v dv) \\ &= (\mathbf{p}_u\cdot\mathbf{p}_u)du^2 + 2(\mathbf{p}_u\cdot\mathbf{p}_v)dudv + (\mathbf{p}_v\cdot\mathbf{p}_v)dv^2 \\ &= Edu^2 + 2Fdudv + Gdv^2 \end{aligned} \tag{5.6}$$

where

$$E = \mathbf{p}_u\cdot\mathbf{p}_u$$

$$F = \mathbf{p}_u\cdot\mathbf{p}_v \tag{5.7}$$

$$G = \mathbf{p}_v\cdot\mathbf{p}_v$$

E, F, and G are known as the coefficients of the first fundamental form. It is a homogeneous function of second degree in du and dv. Therefore the first fundamental form I is the quadratic form defined on (du,dv) in the uv plane by

$$I(du,dv) = Edu^2 + 2Fdudv + Gdv^2 \tag{5.8}$$

Note that $I = |d\mathbf{p}|^2 \geq 0$ and $I = |d\mathbf{p}|^2 = |\mathbf{p}_u du + \mathbf{p}_v dv|^2 = 0$ if and only if $du = 0$ and $dv = 0$. Also, $E = \mathbf{p}_u\cdot\mathbf{p}_u = |\mathbf{p}_u|^2 > 0$, $G = \mathbf{p}_v\cdot\mathbf{p}_v = |\mathbf{p}_v|^2 > 0$, and

$$\begin{aligned} EG - F^2 &= (\mathbf{p}_u\cdot\mathbf{p}_u)(\mathbf{p}_v\cdot\mathbf{p}_v) - (\mathbf{p}_u\cdot\mathbf{p}_v)(\mathbf{p}_u\cdot\mathbf{p}_v) \\ &= (\mathbf{p}_u\times\mathbf{p}_v)\cdot(\mathbf{p}_u\times\mathbf{p}_v) \\ &= |\mathbf{p}_u\times\mathbf{p}_v|^2 > 0 \end{aligned} \tag{5.9}$$

since $\mathbf{p}_u$ and $\mathbf{p}_v$ are independent, and $\mathbf{p}_u \neq 0$, $\mathbf{p}_v \neq 0$, $\mathbf{p}_u\times\mathbf{p}_v \neq 0$.

In some sense I depends only on the surface and not on the particular representation. Suppose that $\mathbf{p} = \mathbf{p}^*(s,t)$ is another coordinate patch containing a neighborhood of $\mathbf{p}(u,v)$. The transformation $s = s(u,v)$, $t = t(u,v)$ has a differential at (u,v) that maps the vector (du,dv) into the vector (ds,dt) and is given by

$$ds = s_u\,du + s_v\,dv$$

$$dt = t_u\,du + t_v\,dv$$

Then

$$\begin{aligned} I^*(ds,dt) &= |d\mathbf{p}^*|^2 \\ &= |\mathbf{p}_s^* ds + \mathbf{p}_t^* dt|^2 \\ &= |\mathbf{p}_s^*(s_u\,du + s_v\,dv) + \mathbf{p}_t^*(t_u\,du + t_v\,dv)|^2 \\ &= |(\mathbf{p}_s^* s_u + \mathbf{p}_t^* t_u)du + (\mathbf{p}_s^* s_v + \mathbf{p}_t^* t_v)dv|^2 \\ &= |\mathbf{p}_u\,du + \mathbf{p}_v\,dv|^2 \\ &= |d\mathbf{p}|^2 \\ &= I(du,dv) \end{aligned} \tag{5.10}$$

Thus, the first fundamental form I is independent of the representation in the sense that $I(du,dv)=I^*(ds,dt)$. However, the first fundamental coefficients are not invariant under a parameter transformation. They transform as follows:

$$\begin{aligned} E &= \mathbf{p}_u \cdot \mathbf{p}_u \\ &= (\mathbf{p}_s^* s_u + \mathbf{p}_t^* t_u)\cdot(\mathbf{p}_s^* s_u + \mathbf{p}_t^* t_u) \\ &= \mathbf{p}_s^* \cdot \mathbf{p}_s^* s_u^2 + 2\mathbf{p}_s^* \cdot \mathbf{p}_t^* s_u t_u + \mathbf{p}_t^* \cdot \mathbf{p}_t^* t_u^2 \\ &= E^* s_u^2 + 2F^* s_u t_u + G^* t_u^2 \end{aligned} \tag{5.11}$$

and similarly,

$$\begin{aligned} F &= E^* s_u s_v + F^*(s_u t_v + s_v t_u) + G^* t_u t_v \\ G &= E^* s_v^2 + 2F^* s_v t_v + G^* t_v^2 \end{aligned} \tag{5.12}$$

5.3 Second Fundamental Form

Suppose $\mathbf{p} = \mathbf{p}(u,v)$ is a surface patch of class ≥ 2. The unit normal to a surface

at a point $\mathbf{p}(u,v)$ is

$$\mathbf{n}(u,v) = \frac{\mathbf{p}_u \times \mathbf{p}_v}{|\mathbf{p}_u \times \mathbf{p}_v|}$$

which is a function with differential $d\mathbf{n} = \mathbf{n}_u du + \mathbf{n}_v dv$. Note that $d\mathbf{n}$ is a vector parallel to the tangent plane at $\mathbf{p}(u,v)$. This follows from $0 = d(1) = d(\mathbf{n}\cdot\mathbf{n}) = 2d\mathbf{n}\cdot\mathbf{n}$.

The second fundamental form is

$$\begin{aligned} II &= -d\mathbf{p}\cdot d\mathbf{n} \\ &= -(\mathbf{p}_u du + \mathbf{p}_v dv)(\mathbf{n}_u du + \mathbf{n}_v dv) \\ &= -\mathbf{p}_u\cdot\mathbf{n}_u du^2 - (\mathbf{p}_u\cdot\mathbf{n}_v + \mathbf{p}_v\cdot\mathbf{n}_u)dudv - \mathbf{p}_v\cdot\mathbf{n}_v dv^2 \\ &= L du^2 + 2M dudv + N dv^2 \end{aligned} \tag{5.13}$$

where

$$\begin{aligned} L &= -\mathbf{p}_u\cdot\mathbf{n}_u \\ M &= -\frac{1}{2}(\mathbf{p}_u\cdot\mathbf{n}_v + \mathbf{p}_v\cdot\mathbf{n}_u) \\ N &= -\mathbf{p}_v\cdot\mathbf{n}_v \end{aligned} \tag{5.14}$$

Since $\mathbf{p}_u$ and $\mathbf{p}_v$ are perpendicular to $\mathbf{n}$ for all (u,v), that is

$$0 = (\mathbf{p}_u\cdot\mathbf{n})_u = \mathbf{p}_{uu}\cdot\mathbf{n} + \mathbf{p}_u\cdot\mathbf{n}_u$$

$$0 = (\mathbf{p}_u\cdot\mathbf{n})_v = \mathbf{p}_{uv}\cdot\mathbf{n} + \mathbf{p}_u\cdot\mathbf{n}_v$$

$$0 = (\mathbf{p}_v\cdot\mathbf{n})_u = \mathbf{p}_{vu}\cdot\mathbf{n} + \mathbf{p}_v\cdot\mathbf{n}_u$$

$$0 = (\mathbf{p}_v\cdot\mathbf{n})_v = \mathbf{p}_{vv}\cdot\mathbf{n} + \mathbf{p}_v\cdot\mathbf{n}_v$$

Note that $\mathbf{p}$ is class ≥ 2 so that

$$\mathbf{p}_{uv} = \mathbf{p}_{vu}$$

Thus,

$$\mathbf{p}_{uu}\cdot\mathbf{n} = -\mathbf{p}_u\cdot\mathbf{n}_u$$

$$\mathbf{p}_{uv}\cdot\mathbf{n} = -\mathbf{p}_u\cdot\mathbf{n}_v = -\mathbf{p}_v\cdot\mathbf{n}_u$$

$$\mathbf{p}_{vv}\cdot\mathbf{n} = -\mathbf{p}_v\cdot\mathbf{n}_v$$

Hence, we have alternative expressions for L, M, and N.

$$\begin{aligned} L &= \mathbf{p}_{uu}\cdot\mathbf{n} \\ M &= \mathbf{p}_{uv}\cdot\mathbf{n} \\ N &= \mathbf{p}_{vv}\cdot\mathbf{n} \end{aligned} \tag{5.15}$$

Therefore, we have

$$\begin{aligned} II &= Ldu^2 + 2Mdudv + Ndv^2 \\ &= \mathbf{p}_{uu}\cdot\mathbf{n}du^2 + 2\mathbf{p}_{uv}\cdot\mathbf{n}dudv + \mathbf{p}_{vv}\cdot\mathbf{n}dv^2 \\ &= d^2\mathbf{p}\cdot\mathbf{n} \end{aligned} \tag{5.16}$$

We can view

$$d^2\mathbf{p} = \mathbf{p}_{uu}du^2 + 2\mathbf{p}_{uv}dudv + \mathbf{p}_{vv}dv^2 \tag{5.17}$$

as the second order derivative of $\mathbf{p}$ at (u,v) in the direction du, dv.

The second fundamental form II is invariant in the same sense that I is invariant under a parameter transformation which preserves the direction of $\mathbf{n}$; otherwise II changes its sign. Also, if $\mathbf{p} = \mathbf{p}^*(s,t)$ is another patch on the surface, then at a point $\frac{\partial(s,t)}{\partial(u,v)} > 0$, we have

$$
\begin{aligned}
L &= L^* s_u^2 + 2M^* s_u t_u + N^* t_u^2 \\
M &= L^* s_u s_v + M^* (s_u t_v + t_u s_v) + N^* t_u t_v \\
N &= L^* s_v^2 + 2M^* s_v t_v + N^* t_v^2
\end{aligned}
\tag{5.18}
$$

5.4 Nature of a Surface Point

Define

$$
\begin{aligned}
\delta &= \frac{1}{2} II \\
&= \frac{1}{2}(Ldu^2 + 2Mdudv + Ndv^2)
\end{aligned}
\tag{5.19}
$$

as a function called an osculating paraboloid at a point P. The nature of this paraboloid at P determines the nature of the surface in the neighborhood of P. We can distinguish four cases based on L, M, and N.

1. Elliptic: A point is called an elliptic point if $LN - M^2 > 0$. Function δ of du and dv is an elliptic paraboloid in this case. In the neighborhood of this point the surface lies on only one side of the tangent plane. Note that δ maintains the same sign for all (du, dv). The shape of the neighborhood is shown in Figure 5.1.

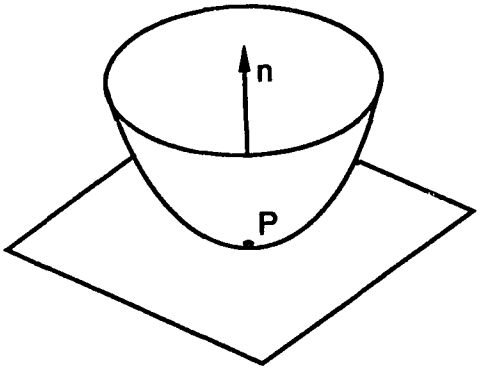

Figure 5.1: $LN - M^2 > 0$

2. Hyperbolic: A point is a hyperbolic point if $LN - M^2 < 0$. Function δ of du and dv is a hyperbolic paraboloid. There are two lines through P in the tangent plane dividing the tangent plane into four sections in which δ is positive and negative, alternatively. On the lines, $\delta = 0$. Thus, in the neighborhood of this point, the surface lies on both sides of the tangent plane. The shape is shown in Figure 5.2.

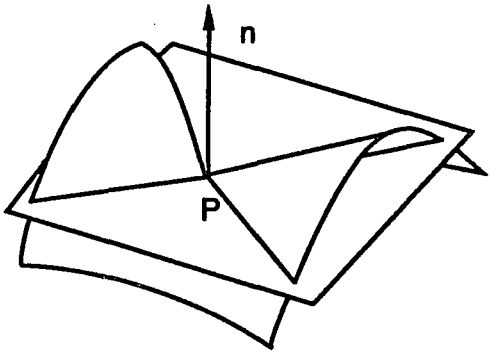

Figure 5.2: $LN - M^2 < 0$

3. Parabolic: A point is parabolic if $LN - M^2 = 0$ and $L^2 + M^2 + N^2 \neq 0$, i.e. L, M, and N are not all zero. Function δ of du and dv is a parabolic cylinder in this case. There is a line through P in the tangent plane along which $\delta = 0$, otherwise δ remains the same sign. Note that the surface itself might lie on both sides of the tangent plane. The shape of the surface is shown in Figure 5.3.

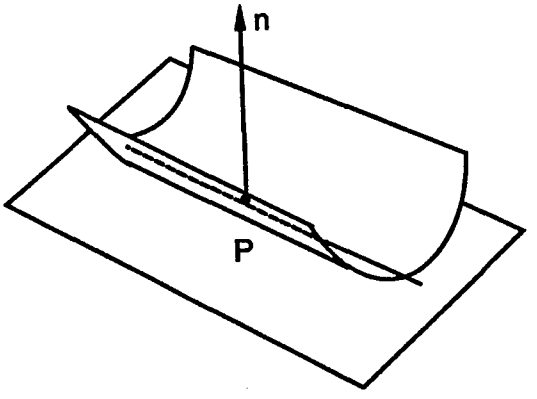

Figure 5.3: $LN - M^2 = 0$ and $L^2 + M^2 + N^2 \neq 0$

4. Planar: A point is planar if $L = M = N = 0$. In this case $\delta = 0$ for all du and dv. The degree of contact of the surface and the tangent plane is of higher order than any of the previous case.

It can be easily shown that the property of a point on a surface being elliptic, hyperbolic, parabolic, or planar is independent of the representation of the surface. If $\mathbf{p} = \mathbf{p}_*(s,t)$ is another patch on the surface, it can be verified from (5.18) that at each point

$$L^*N^* - M^{*2} = \left[\frac{\partial(u,v)}{\partial(s,t)}\right]^2 (LN - M^2) \tag{5.20}$$

Since $\frac{\partial(u,v)}{\partial(s,t)} \neq 0$, $L^*N^* - M^{*2}$ is positive, negative, or zero together with $LN - M^2$.

It also follows from (5.18) and their corresponding inverse equations that $L = M = N = 0$ if and only if $L^* = M^* = N^* = 0$.

5.5 Curvatures

Let P be a point on a surface $\mathbf{p} = \mathbf{p}(u,v)$ of class ≥ 2, and $\mathbf{p} = \mathbf{p}(u(t),v(t))$ a curve C that lies on the surface and passes through P. The normal curvature vector to C at P is the vector projection of the curvature vector $\mathbf{k}$ of C at P onto the normal $\mathbf{n}$ at P, i.e.,

$$\mathbf{k}_n = (\mathbf{k}\cdot\mathbf{n})\mathbf{n} \tag{5.21}$$

Notice that $\mathbf{k}_n$ is independent of the sense of $\mathbf{n}$ or of C. The component of $\mathbf{k}_n$ in the direction of $\mathbf{n}$ is called the *normal curvature* of C at P. That is

$$k_n = \mathbf{k}\cdot\mathbf{n} \tag{5.22}$$

Here the sign of k_n depends on the sense of $\mathbf{n}$, but it is independent of the sense of C.

Note that the unit tangent to C at P is $\mathbf{t} = \dfrac{d\mathbf{p}}{ds} = \dfrac{d\mathbf{p}/dt}{|d\mathbf{p}/dt|}$, and the curvature vector is $\mathbf{k} = \dfrac{d\mathbf{t}}{ds} = \dfrac{d\mathbf{t}/dt}{|d\mathbf{p}/dt|}$. Thus, $0 = \dfrac{d}{dt}(\mathbf{t}\cdot\mathbf{n}) = \dfrac{d\mathbf{t}}{dt}\cdot\mathbf{n} + t\cdot\dfrac{d\mathbf{n}}{dt}$ since $\mathbf{t}$ is perpendicular to $\mathbf{n}$ along the curve.

It follows that

$$k_n = \frac{L(du/dt)^2 + 2M(du/dt)(dv/dt) + N(dv/dt)^2}{E(dv/dt)^2 + 2F(du/dt)(dv/dt) + G(dv/dt)^2} \tag{5.23}$$

Note that k_n depends only on $\dfrac{du/dt}{dv/dt}$, which is the direction of the tangent line to C at

P. Otherwise, k_n is a function of the fundamental forms I and II, which depend only on P.

All curves through a point P on a surface tangent to the same line through P have the same normal curvature at P [45]. Since the normal curvature to C at P depends only on P and the direction of the tangent line to C at P, we can speak of the normal curvature in the direction $du:dv$, du and dv are not both zero, we have

$$k_n = \frac{Ldu^2 + 2Mdudv + Ndv^2}{Edu^2 + 2Fdudv + Gdv^2} \tag{5.24}$$

Notice that the above form is simply $k_n = \frac{II}{I}$. $du:dv$ are the direction numbers of the line in the tangent plane parallel to

$$\mathbf{p}_u du + \mathbf{p}_v dv$$

$du:dv$ and $du':dv'$ determine the same line if and only if they are proportional.

It is clear that k_n is invariant in the same sense as I and II. k_n does not change sign under a parametric transformation which preserves the sense of $\mathbf{n}$ and k_n changes sign under a parametric transformation which reverses the sense of $\mathbf{n}$.

Since $I \geq 0$, k_n is positive, negative, or zero together with II. That is, If P is elliptic, then $k_n \neq 0$ and remains the same sign for all $du:dv$ at P. If P is a hyperbolic point, then k_n is positive, negative, or zero, depending on $du:dv$. If P is a parabolic point, k_n remains the same sign and is zero for the direction for which $II = 0$. If P is planar, $k_n = 0$ in all directions.

The two perpendicular directions for which the values of k_n take on maximum and minimum values are called the principal directions. The maximum and minimum values of normal curvatures k_1 and k_2 are called the principal curvatures [52]. A point on the surface at which k_n is constant is called an umbilical point. If $k_n = constant \neq 0$, it is called elliptic umbilical point. If $k_n = constant = 0$, it is called parabolic umbilical point. Therefore, if all points of a connected surface S are umbilical, then S is either contained in a sphere or in a plane [89].

The principal curvatures are roots of

$$(EG - F^2)k^2 - (EN + GL - 2FM)k + (LN - M^2) = 0 \tag{5.25}$$

which can be shown by proving that k_1 is a principal curvature with principal direction $du_1:dv_1$ if and only if k_1, du_1,dv_1 satisfy

$$\begin{aligned} (L - k_1E)du_1 + (M - k_1F)dv_1 &= 0 \\ (M - k_1F)du_1 + (N - k_1G)dv_1 &= 0 \end{aligned} \tag{5.26}$$

Let us first prove that (5.25) has only real roots, i.e., the discriminant of the equation (5.25) is greater than or equal to zero. It is equal to zero if and only if $\frac{L}{E} = \frac{M}{F} = \frac{N}{G}$. From (5.25), the discriminant is

$$(EN + GL + -2FM)^2 - 4(EG - F^2)(LN - M^2)$$

which is identical to

$$4\left[\frac{EG - F^2}{E^2}\right](EM - FL)^2 + \left[EN - GL - \frac{2F}{E}(EM - FL)\right]^2$$

Thus the discriminant is not less than zero.

Since $EG - F^2 > 0$, the discriminant is zero if and only if

$$EM - FL = 0$$

and

$$EN - GL - \frac{2F}{E}(EM - FL) = 0$$

which is if and only if

$$EM - FL = 0$$

and

$$EN - GL = 0$$

i.e., if and only if

$$\frac{L}{E} = \frac{M}{F} = \frac{N}{G}$$

Now let's prove that the roots of (5.25) are principal curvatures. Suppose k_1 is a principal curvature with principal direction $du_1 : dv_1$. We know that the principal curvatures are the maximum and minimum values of the normal curvature k_n. Hence if

$$k_n = \frac{Ldu^2 + 2Mdudv + Ndv^2}{Edu^2 + 2Fdudv + Gdv^2}$$

has an extremum k_1 at (du_1, dv_1), then the partial derivatives

$$\frac{\partial k_1}{\partial du} = 0$$

$$\frac{\partial k_1}{\partial dv} = 0$$

at (du_1, dv_1), or

$$\frac{I\, II_{du} - II\, I_{du}}{I^2} = 0$$

$$\frac{I\, II_{dv} - II\, I_{dv}}{I^2} = 0$$

at (du_1, dv_1).

Multiplying by I gives

$$II_{du} - \frac{II}{I} I_{du} = 0$$

$$II_{dv} - \frac{II}{I} I_{dv} = 0$$

Since $\frac{II}{I} = k_1$ at (du_1, dv_1), we have

$$II_{du} - k_1 I_{du} = 0$$

$$II_{dv} - k_1 I_{dv} = 0$$

Since $II_{du} = 2Ldu + 2Mdv$ and $I_{du} = 2Edu + 2Fdv$, etc., we get

$$(Ldu_1 + Mdv_1) - k_1(Edu_1 + Fdv_1) = 0$$

$$(Mdu_1 + Ndv_1) - k_1(Fdu_1 + Gdv_1) = 0$$

Conversely, suppose that k_1, du_1, dv_1 satisfy (5.26) and $du^2 + dv^2 \neq 0$. Then k_1 together with the principal curvatures must satisfy

$$\det \begin{bmatrix} L - kE & M - kF \\ M - kF & N - kG \end{bmatrix} = 0$$

or by expanding we get (5.25).

Now suppose P is an umbilical point with curvature k. Since k is the same in every direction, the coefficients of (5.26) must all be zero, i.e., $k = \frac{E}{L} = \frac{M}{F} = \frac{N}{G}$. Then it follows from the above that (5.25) has a single root with multiplicity two. If P is nonumbilical, k must be one of the two distinct roots of (5.25).

5.6 Gaussian and Mean Curvature

After dividing (5.25) by $EG - F^2$, we have

$$k^2 - \frac{EN + GL - 2FM}{EG - F^2}k + \frac{LN - M^2}{EG - F^2} = 0 \tag{5.27}$$

The average of the roots of (5.27)

$$H = \frac{1}{2}(k_1 + k_2) = \frac{EN + GL - 2FM}{2(EG - F^2)} \tag{5.28}$$

is called the mean curvature at P.

The product of the roots of (5.27)

$$K = k_1 k_2 = \frac{LN - M^2}{EG - F^2} \tag{5.29}$$

is called the Gaussian curvature at P.

Since

$$EG - F^2 = (\mathbf{p}_u \cdot \mathbf{p}_u)(\mathbf{p}_v \cdot \mathbf{p}_v) - (\mathbf{p}_u \cdot \mathbf{p}_v)^2$$

by applying the formula in vector computation

$$(\mathbf{a} \times \mathbf{b}) \cdot (\mathbf{c} \times \mathbf{d}) = (\mathbf{a} \cdot \mathbf{c})(\mathbf{b} \cdot \mathbf{d}) - (\mathbf{a} \cdot \mathbf{d})(\mathbf{b} \cdot \mathbf{c})$$

we have

$$EG - F^2 = (\mathbf{p}_u \times \mathbf{p}_v) \cdot (\mathbf{p}_u \times \mathbf{p}_v)$$

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

If we introduce the triple product $[\mathbf{a}\ \mathbf{b}\ \mathbf{c}] = \mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})$ notation, we can derive the following

$$\begin{aligned} K &= \frac{LN - M^2}{EG - F^2} \\ &= \frac{(\mathbf{p}_{uu} \cdot \mathbf{n})(\mathbf{p}_{vv} \cdot \mathbf{n}) - (\mathbf{p}_{uv} \cdot \mathbf{n})^2}{|\mathbf{p}_u \times \mathbf{p}_v|^2} \\ &= \frac{1}{|\mathbf{p}_u \times \mathbf{p}_v|^2} \left[\frac{(\mathbf{p}_{uu} \cdot (\mathbf{p}_u \times \mathbf{p}_v))(\mathbf{p}_{vv} \cdot (\mathbf{p}_u \times \mathbf{p}_v)) - (\mathbf{p}_{uv} \cdot (\mathbf{p}_u \times \mathbf{p}_v))^2}{|\mathbf{p}_u \times \mathbf{p}_v|^2} \right] \\ &= \frac{[\mathbf{p}_{uu}\, \mathbf{p}_u\, \mathbf{p}_v][\mathbf{p}_{vv}\, \mathbf{p}_u\, \mathbf{p}_v] - [\mathbf{p}_{uv}\, \mathbf{p}_u\, \mathbf{p}_v]^2}{|\mathbf{p}_u \times \mathbf{p}_v|^4} \end{aligned} \tag{5.30}$$

And similarly,

$$\begin{aligned} H &= \frac{EN + GL - 2FM}{2(EG - F^2)} \\ &= \frac{(\mathbf{p}_u \cdot \mathbf{p}_u)(\mathbf{p}_{vv} \cdot (\mathbf{p}_u \times \mathbf{p}_v)) + (\mathbf{p}_v \cdot \mathbf{p}_v)(\mathbf{p}_{uu} \cdot (\mathbf{p}_u \times \mathbf{p}_v)) - 2(\mathbf{p}_u \cdot \mathbf{p}_v)(\mathbf{p}_{uv} \cdot (\mathbf{p}_u \times \mathbf{p}_v))}{2|\mathbf{p}_u \times \mathbf{p}_v|^3} \\ &= \frac{(\mathbf{p}_u \cdot \mathbf{p}_u)[\mathbf{p}_{vv}\, \mathbf{p}_u\, \mathbf{p}_v] + (\mathbf{p}_v \cdot \mathbf{p}_v)[\mathbf{p}_{uu}\, \mathbf{p}_u\, \mathbf{p}_v] - 2(\mathbf{p}_u \cdot \mathbf{p}_v)[\mathbf{p}_{uv}\, \mathbf{p}_u\, \mathbf{p}_v]}{2|\mathbf{p}_u \times \mathbf{p}_v|^3} \end{aligned} \tag{5.31}$$

If a surface is represented in an explicit form $z = z(x, y)$, we can treat this as

$$\mathbf{p} = (x, y, z(x, y))$$

with two parameters x and y. Thus, we have

$$\mathbf{p}_x = (1, 0, z_x)$$

$$\mathbf{p}_y = (0, 1, z_y)$$

$$\mathbf{p}_{xx} = (0, 0, z_{xx})$$

$$\mathbf{p}_{yy} = (0, 0, z_{yy})$$

$$\mathbf{p}_{xy} = (0, 0, z_{xy})$$

$$\mathbf{n} = \frac{1}{(1 + z_x^2 + z_y^2)^{1/2}}(-z_x \quad -z_y \quad 1)$$

Hence,

$$[\mathbf{p}_{xx}\, \mathbf{p}_x\, \mathbf{p}_y] = z_{xx}$$

$$[\mathbf{p}_{yy}\, \mathbf{p}_x\, \mathbf{p}_y] = z_{yy}$$

$$[\mathbf{p}_{xy}\, \mathbf{p}_x\, \mathbf{p}_y] = z_{xy}$$

$$|\mathbf{p}_x \times \mathbf{p}_y| = (1 + z_x^2 + z_y^2)^{\frac{1}{2}}$$

Therefore, we have

$$\begin{aligned} K &= \frac{[\mathbf{p}_{xx}\, \mathbf{p}_x\, \mathbf{p}_y][\mathbf{p}_{yy}\, \mathbf{p}_x\, \mathbf{p}_y] - [\mathbf{p}_{xy}\, \mathbf{p}_x\, \mathbf{p}_y]^2}{|\mathbf{p}_x \times \mathbf{p}_y|^4} \\ &= \frac{z_{xx} \cdot z_{yy} - z_{xy}^2}{(1 + z_x^2 + z_y^2)^2} \end{aligned} \tag{5.32}$$

We also have

$$
\begin{aligned}
H &= \frac{(\mathbf{p}_x \cdot \mathbf{p}_x)[\mathbf{p}_{yy}\,\mathbf{p}_x\,\mathbf{p}_y] + (\mathbf{p}_y \cdot \mathbf{p}_y)[\mathbf{p}_{xx}\,\mathbf{p}_x\,\mathbf{p}_y] - 2(\mathbf{p}_x \cdot \mathbf{p}_y)[\mathbf{p}_{xy}\,\mathbf{p}_x\,\mathbf{p}_y]}{2\,|\,\mathbf{p}_x \times \mathbf{p}_y\,|^3} \\
&= \frac{(1+z_x^2)z_{yy} + (1+z_y^2)z_{xx} - 2z_x z_y z_{xy}}{2(1+z_x^2+z_y^2)^{\frac{3}{2}}}
\end{aligned}
\tag{5.33}
$$

Since k_n at most changes sign with a change in orientation of the surface, the extreme values of k_n at most both change sign with a change in orientation. It follows that $K = k_1 k_2$ is independent of its representation, an invariant property of the surface. Also, the magnitude of the mean curvature $|H|$ is invariant.

Since the sign of K is the same as $LN - M^2$, we can determine the nature of the surface in the neighborhood of a point according to the value of K and H at that point.

If $K > 0$, i.e., $LN - M^2 > 0$, the point is elliptic. The surface is locally convex with respect to the tangent plane.

If $K < 0$, i.e., $LN - M^2 < 0$, the point is hyperbolic. The surface lies on both sides of the tangent plane in the neighborhood of this point.

If $K = 0$ and $H \neq 0$, i.e., $LN - M^2 = 0$ and $L^2 + N^2 + M^2 \neq 0$, the point is parabolic.

If $K = 0$ and $H = 0$, i.e., $L = M = N = 0$, the point is called a planar point.

H and K are in a sense the only invariants of the surfaces obtained algebraically from the two fundamental forms [45]. Our approach is to use H and K as heuristics

to search the model surface space. Since a brute force search of the 3-D surface space is costly, we will organize the surfaces in a way that dramatically reduces the amount of search.

Chapter 6: Representation of Knowledge about Surface Shape

We have seen that the surface curvature information is invariant on properties of surface shapes. In order to use this kind of information about the model surfaces, we have to organize the information in a way such that the computer can use it effectively and efficiently. In artificial intelligence (AI), we call this process *knowledge representation.*

An intelligent system is generally organized in a structure as shown in Figure 6.1. The *Inference engine* interacts with the environment to solve problems using information in the Knowledge base. The *Knowledge base* contains well-organized domain-specific knowledge. The knowledge base should be capable of maintaining its knowledge, such as learning new information, or modifying existing knowledge.

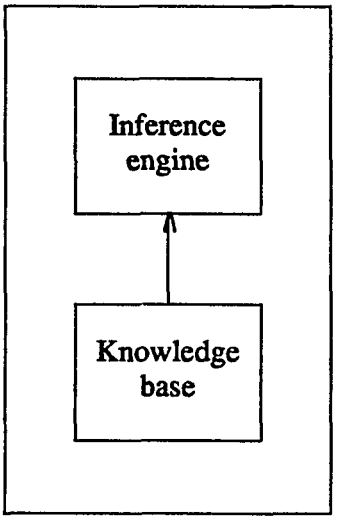

Figure 6.1: *Architecture of Intelligent Systems*

In this chapter, we discuss why we need to consider different knowledge representations for different intelligent systems, what we should consider in designing representation schemes for intelligent systems, and the representation formalism we use in our system. We refer to knowledge in intelligent systems as abstracted and organized information on specific domains, which is to be used by the intelligent systems to behave "intelligently."

6.1 Knowledge and its Representation

Knowledge is the facts and ideas acquired by study, investigation, observation, or experience. One has to know things in order to do them. We describe someone's ability to behave with intelligence in terms of his or her knowledge. This concept is also applicable to intelligent systems. We say that a computer program is intelligent if it knows some domain knowledge and solves domain problems using the knowledge.

The nature of knowledge and intelligence has been pondered by researchers in different areas, such as psychology, philosophy, and education, for thousands of years. Since AI research involves the design of computer systems which have intelligent behavior, AI researchers have often taken a pragmatic approach to the subject of knowledge, i.e., focusing on improving the behavior of the systems.

Since an intelligent process is a process of acquiring and applying knowledge in a specific domain, knowledge must be represented in a way such that it can be efficiently stored, retrieved, used, and modified. Knowledge representation research

develops techniques of representing and organizing knowledge in a domain for problem solving.

A representation of knowledge is a combination of data structures and interpretive procedures which use the data structures. If the data structures and the procedures are well-designed, the system should be led to intelligent behavior by the "knowledge." Knowledge representation involves the design of classes of data structures for storing information in computer systems as well as the development of procedures which intelligently manipulate these data structures to make inferences.

In this chapter, we first discuss some properties of knowledge representation and then introduce the data structures used to represent the surface shapes for vision process. The procedures which interpret the data structures, such as matching the image segment with the model surfaces, will be discussed in the next chapter, together with the search control strategy.

In general, data structures are not knowledge. For instance, a book is a source of knowledge, but without a reader, the book is just ink on paper. Similarly, we will talk about the curvature map data structures. We really mean that they represent the surface shapes when used by certain programs to behave in a knowledgeable way.

For a given problem, one can think of many possible representations. Furthermore, these representations will have different functional abilities which refer to how well the representation can support the system process. In general, a representation with strong ability should have clear descriptions about the problem, and at the same

time, provide convenient and efficient access for the inference engine of the intelligent system.

Given a primitive problem, there may be a lot of redundant information which contributes nothing to the problem solution process, or the information may be presented in a way too far from what the inference engine can use. Only after deleting redundant information and abstracting the useful information to a level that the inference engine can use, can the problem be solved effectively and efficiently.

Let's discuss different representations in terms of mapping, which is elegant and precise. We have the basic concepts of homomorphism and isomorphism in discrete mathematics. A homomorphism can simplify the representation, while an isomorphism can change the representation. They both are mappings maintaining the properties of computation.

Let $P = <Q, F>$ and $P' = <Q', F'>$ be two problems where Q and Q' are sets of facts for P and P' respectively, and F and F' are relations among Q and Q' respectively. If there is an onto mapping

$$h: Q \rightarrow Q'$$

such that for any ordered pair

$$<q_i, q_j> \in F$$

if and only if

$$<h(q_i), h(q_j)> \in F'$$

then P' is a homomorphic problem of P, and h is a homomorphism from P to P'.

If a homomorphism h is 1-1, then h is an isomorphism. Thus, isomorphism is a special case of homomorphism. The existence of a solution for the original problem implies the existence of a solution for its homomorphic problem. The existence of a solution of the isomorphic problem is equivalent to the existence of a solution of the original problem.

Homomorphism is an important concept which enables us to abstract useful information from problems and to delete some unimportant factors. A good homomorphism should reserve the properties of the original problem to some extent so that we do not miss the necessary information.

Homomorphism is a partial ordered relation. Thus, if h_1 is a homomorphism from P_1 to P_2, and h_2 is a homomorphism from P_2 to P_3, then $h_1 \circ h_2$ is a homomorphism from P_1 to P_3, i.e., existence of a solution for P_1 implies existence of a solution for P_3. In other words, if there are no solutions for P_3, then there are no solutions for P_1.

In addition to the abstraction process, filling in details is also an important intelligent process. In AI research, we often use homomorphism methods to simplify problems to process and then fill in the details. However, we should be careful about two points: there are some cases when P_1 is homomorphic to P by h_1, and P_2 is homomorphic to P by h_2, but P_1 and P_2 are neither isomorphic nor homomorphic;

and there are also cases when P is homomorphic to P_1 by h_1, and P is homomorphic to P_2 by h_2, but P_1 and P_2 are neither isomorphic nor homomorphic.

Using the properties of homomorphism, we can often find a homomorphism h to apply to a complex problem so that the complex problem is converted into a simpler and easier problem. After obtaining a solution for the simpler problem, we apply the inverse homomorphism h^{-1} to the solution and fill the details to obtain the solution for the original problem. The process can be shown in Figure 6.2.

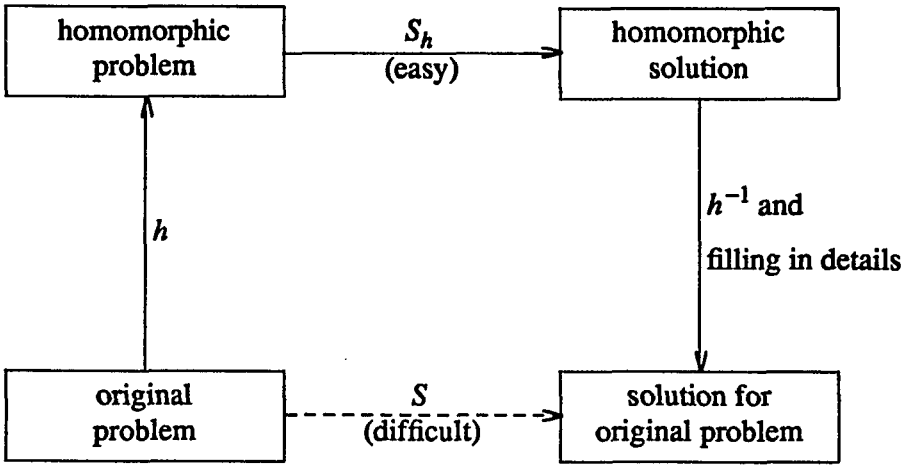

Figure 6.2: *Using Homomorphic Mapping to Solve Problems*

Isomorphism is an equivalence relation. Thus, we can represent problems in different isomorphic ways to process such that the new representations have suitable natures for computer manipulation. For instance, a digraph is represented by linked lists.

Homomorphism and isomorphism are one of the theoretical foundations for

knowledge representation research. In the following discussion, we may not express the homomorphic or isomorphic relation explicitly, however, the representations can all be formalized in terms of homomorphic or isomorphic mappings.

In the process of problem solving, finding suitable representations is very important. In some cases, intelligence is mainly seen in finding the suitable representation. Once the problem is represented suitably, the problem can be solved pretty easily. In today's AI research, the representation is always chosen by human researchers. Hopefully, in the future, computer systems can automatically select the best representation for specific problems according to the nature of the problems. Notice that since any knowledge representations are data structures and programs, they all have the same characteristics.

There is no theory of knowledge representation. We do not know why some schemes are good for certain tasks and others are not. We can only discuss some characteristics of knowledge representation schemes which have been considered important.

6.2 Considerations of Knowledge Representation

The most important consideration in designing knowledge representation schemes is the eventual use of the knowledge. The actual use of the knowledge involves three stages: (1) acquiring new knowledge; (2) retrieving information from the knowledge base relevant to the problem at hand; and (3) reasoning in search of a

solution.

To human beings, acquisition of new knowledge involves relating new materials to what we already know in a psychologically complex way. AI systems often classify the new data structure before it is added to the knowledge base. In many systems, new structures can interact with the old structures. Some other representations are concerned with acquiring knowledge in a form which is natural to human beings who are the source of the new knowledge. If the acquisition process is not designed properly, the system accumulates new facts or data structures without improving its intelligent behavior, or sometimes even degenerating its behavior.

When a system "knows" too many different things, determining what knowledge is relevant to a given problem becomes crucial. The fundamental ideas about retrieval in AI systems can be classified as two basic techniques, linking and lumping. If in a reasoning task it is known that one data structure will entail another data structure, an explicit link can be put in between the two structures. If several structures are typically going to be used together, they can be grouped into a larger structure. All these techniques will be seen in our data structures for describing surface shapes.

Inference is unavoidable when the system is required to perform a task which it has not been told explicitly how to do. The system reasons to figure out what it needs to know from what it already knows. An intelligent system must be able to deduce and verify a number of new facts beyond those it has been told explicitly.

When designing a knowledge representation scheme, we must ask ourselves what kind of reasoning is possible, easy, natural, etc., in this formalism. There are many kinds of reasoning we could think about. Formal reasoning syntactically manipulates data structures to deduce new ones following inference rules, which are prespecified. Procedure reasoning performs simulation to answer questions and solve problems. For instance, a procedure of an arithmetic model could be run to answer "what is the sum of 3 and 4." Analogy reasoning seems natural to the thought of humans, however, so far, it is difficult to accomplish in AI systems. Generalization and abstraction are also natural processes for human beings that are difficult to pin down well enough to implement in computer systems, though some systems can perform a little generalization or abstraction. This ability may be at the core of human learning; however, it has not yet become a useful technique in AI since we humans do not understand our reasoning process well enough.

When acquiring new knowledge, the system should be concerned with how the new information will be retrieved and used in reasoning. For application AI systems, efficiency and accuracy are major concerns. Thus, retrieval and use of knowledge are of more concern than acquisition of new knowledge.

Factors concerning the scope and grain size of a knowledge representation scheme can help determine whether a formalism is suitable for the solution of a particular problem. The factors are what portion of the external world should be represented in a specific system, in what detail objects and events are represented, and how much is actually needed by the reasoning mechanisms. However, it is not easy to

determine these factors. Exactly how much detail is needed depends on the performance desired. In general, uniformity of detail is desirable for a given reasoning task.

It is possible to represent almost everything that the system must know in one formalism. However, somethings are more easily represented than others. "Being represented more easily" involves the representation, the domain, and the reasoning strategies. These are part of the art of doing AI research. There is still no formal criterion for the appropriateness of a knowledge representation scheme.

In terms of mapping, as discussed in the previous section, we are designing a mapping from the objects and events of the world into some internal encoding, which is the representation scheme. Therefore, the issue is whether the mapping in a given situation is easy, natural, efficient, and the like.

The choice of the primitive attributes of the domain, which are used to build up facts in the knowledge base, strongly affects the expressiveness of the knowledge representation scheme in any representation formalism. The selection of primitive elements for the expression of knowledge in a given domain is a fundamental problem in all knowledge representation schemes.

One characteristic of knowledge representation schemes is modularity. Modularity refers to the ability to add, modify, or delete data structures more or less independently from the rest of the knowledge base. Generally, human beings understand and work with modular or decomposable systems more easily.

The problem with nonmodular systems is that the meaning of data structures in the knowledge base is dependent on the context where the knowledge is being used. Context dependence dramatically affects the modifiability of the knowledge base. If the meaning of a fact is independent of the rest of the system, modification is much easier. How much the system is capable of being understood by human beings is important in its development and performance, such as design and implementation of the system, acquisition of knowledge from humans, performance of the task, and interaction with and explanations for the final user.

Note that no system is completely modular. There is some degree of interaction between the data structures which form the knowledge base. Certainly some formalisms are more inherently modular than others.

When designing knowledge based systems, we also need to consider to what knowledge the programmer and the system have direct and manipulatory access, and what knowledge is built-in. This is the issue of what part of the system's knowledge is explicit and what is implicit in the system's program. One advantage of explicit representation schemes is the same fact can be used for multiple purposes, because the facts are in a form which allows global interpretations. In large systems, this feature is a significant advantage.

Some AI researchers emphasize declarative representation schemes for their flexibility and economy, for their completeness and the certainty of the deductions, and for the modifiability of the systems.

Other AI researchers stress procedural representation schemes for their directness of the line of inference, such as using domain-specific heuristics to avoid irrelevant or unnatural lines of reasoning, and the ease of coding and understandability of the reasoning process.

There are still many open questions, in fact serious problems, in knowledge representation research. Many issues discussed above are contradictory to each other. There are many trade-offs in practical systems. That is the reason why people talk about the art, but not about the science, of AI research.

6.3 Data Structures for our System

We have discussed the issues in designing knowledge representation schemes. We now discuss the design of data structures for our vision system. The use of the represented knowledge will be discussed in the next chapter.

In our vision task, the knowledge which the system needs is the descriptions about objects. The knowledge should be represented in a way that the system can use it efficiently to recognize objects. We already have the B-rep about objects. However the B-rep is suitable for producing image from object models but not suitable for vision tasks.

A model database is a set

$$M = \{m_1, m_2, \ldots, m_n\} \tag{6.1}$$

where each m_i, $1 \leq i \leq n$, is the B-rep for a solid object.

A B-rep is a set

$$m_i = \left\{ f_{i1}, f_{i2}, \ldots, f_{ip_i} \right\} \tag{6.2}$$

where f_{ij}, $1 \leq j \leq p_i$, is a surface patch. This set of surface patches cover the boundary of m_i.

Each model is described by a set of surface patches and the implicit spatial relationships among the surface patches. This information is sufficient for generating an image from the models, which will be used in our verification process during object recognization and localization. More information, however, is needed for the reasoning process of object recognization and localization.

The addition information about objects is the knowledge about the surface shapes of different objects. We will use the surface shape information as heuristics to direct the reasoning. We have discussed the invariant surface shape information, Gaussian curvature and the magnitude of mean curvature in the last chapter. It is natural to consider using this invariant information to recognize objects.

However, how to use it is not as simple as we might hope. The surface curvatures are continuous functions on smooth surfaces. The representation elements of the models in the database are smooth surface patches. Surface curvatures are different at different points of same surface patch. Designing data structures to represent the shapes of surface patches is not an easy task. We introduce a novel approach which

represent the surface shape clearly and is efficient and accurate for the object recognization and localization reasoning.

We are using a data structure called a curvature map to represent the surface shape. A curvature map is obtained for each surface patch. A curvature map can be viewed as a mesh on a surface patch as shown in Figure 6.3.

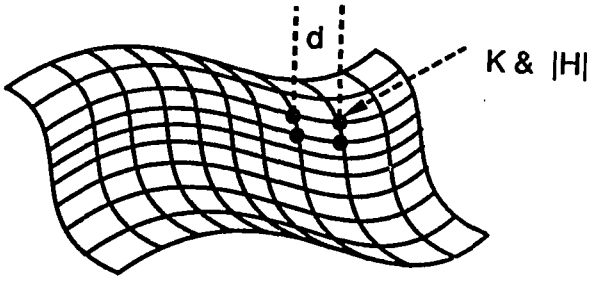

Figure 6.3: *Curvature Map as a Mesh on a Surface Patch*

We compute the Gaussian curvature K and magnitude of mean curvature $|H|$ at each node of the mesh is Figure 6.3. The density of the mesh should be higher than that of the image pixels if the surface patch is seen in an image. Thus the distance d of adjacent nodes must be less than the distance of any adjacent pixels in an image. If the distance of two adjacent pixels in an image corresponds to real world distance D. We select $d < \frac{1}{2}D$, which proved dense enough in our system. This is the issue about the grain size of knowledge representation, as discussed previously.

In the system, a curvature map is in fact a three-layer structure as shown in Figure 6.4. One layer stores the spatial coordinates of each node of the mesh. The other two layers are for the Gaussian curvature and the magnitude of mean curvature at the corresponding points.

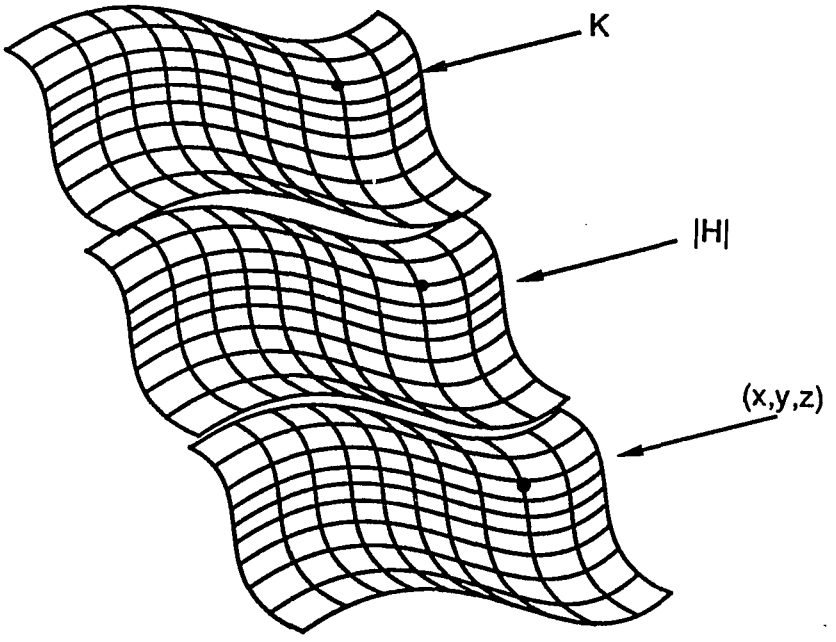

Figure 6.4: *Curvature Map as a Three-layer Structure*

To compute the Gaussian curvature and magnitude of mean curvature, we use formulas introduced in Chapter 4. For CAD models described in bivariate parametric functions, we use (5.30) and (5.31) to compute the curvatures.

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

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

Bivariate parametric surfaces are often represented as

$$\mathbf{p}(u,v) = USPTV^T$$

where

$$U = [u^n \quad u^{n-1} \cdots u \quad 1]$$

$$V = [v^m \quad v^{m-1} \cdots v \quad 1]$$

and S, T are coefficient matrices, and $\mathbf{P}$ is the matrix of control points.

Hence, we have

$$\mathbf{p}_u = U'SPTV^T$$

where

$$U' = [nu^{n-1} \quad (n-1)u^{n-2} \cdots 1 \quad 0]$$

and

$$\mathbf{p}_v = USPTV'^T$$

where

$$V' = [mv^{m-1} \quad (m-1)v^{m-2} \cdots 1 \quad 0]$$

and so on.

For instance, a bi-cubic B-spline function is

$$\mathbf{p}(u,v) = N(u)\mathbf{p}_3 N(v)^T$$

where

$$N(u) = \left[\frac{(1-u)^3}{6} \quad \frac{3u^3-6u^2+4}{6} \quad \frac{-3u^3+3u^2+3u+1}{6} \quad \frac{u^3}{6}\right], \quad 0 \le u \le 1$$

$$N(v) = \left[\frac{(1-v)^3}{6} \quad \frac{3v^3-6v^2+4}{6} \quad \frac{-3v^3+3v^2+3v+1}{6} \quad \frac{v^3}{6}\right], \quad 0 \le v \le 1$$

and

$$\mathbf{p}_3 = \begin{bmatrix} \mathbf{p}_{i-1,j-1} & \mathbf{p}_{i-1,j} & \mathbf{p}_{i-1,j+1} & \mathbf{p}_{i-1,j+2} \\ \mathbf{p}_{i,j-1} & \mathbf{p}_{i,j} & \mathbf{p}_{i,j+1} & \mathbf{p}_{i,j+2} \\ \mathbf{p}_{i+1,j-1} & \mathbf{p}_{i+1,j} & \mathbf{p}_{i+1,j+1} & \mathbf{p}_{i+1,j+2} \\ \mathbf{p}_{i+2,j-1} & \mathbf{p}_{i+2,j} & \mathbf{p}_{i+2,j+1} & \mathbf{p}_{i+2,j+2} \end{bmatrix}$$

Thus, we have

$$\mathbf{p}_u = N'(u)\mathbf{p}_3 N(v)^T$$

$$\mathbf{p}_v = N(u)\mathbf{p}_3 N'(v)^T$$

$$\mathbf{p}_{uu} = N''(u)\mathbf{p}_3 N(v)^T$$

$$\mathbf{p}_{vv} = N(u)\mathbf{p}_3 N''(v)^T$$

$$\mathbf{p}_{uv} = N'(u)\mathbf{p}_3 N'(v)^T$$

where

$$N'(u) = \left[\frac{-(u-1)^2}{2} \quad \frac{3u^2-4u}{2} \quad \frac{-3u^2+2u+1}{2} \quad \frac{u^2}{2}\right]$$

$$N'(v) = \left[\frac{-(v-1)^2}{2} \quad \frac{3v^2-4v}{2} \quad \frac{-3v^2+2v+1}{2} \quad \frac{v^2}{2}\right]$$

$$N''(u) = \begin{bmatrix} -u+1 & 3u-2 & -3u+1 & u \end{bmatrix}$$

$$N''(v) = \begin{bmatrix} -v+1 & 3v-2 & -3v+1 & v \end{bmatrix}$$

For model boundaries described in explicit function $z = f(x,y)$, we use (5.32) and (5.33) to compute Gaussian curvature and mean curvature.

$$K = \frac{z_{xx} \cdot z_{yy} - z_{xy}^2}{(1 + z_x^2 + z_y^2)^2}$$

$$H = \frac{(1 + z_x^2) z_{yy} + (1 + z_y^2) z_{xx} - 2 z_x z_y z_{xy}}{2(1 + z_x^2 + z_y^2)^{\frac{3}{2}}}$$

It is pretty straightforward to derive the derivatives of the explicit function $z = f(x,y)$. For instance, if

$$z = a_0 x^3 + a_1 x^2 y + a_2 x y^2 + a_3 y^3 + a_4 x^2 + a_5 x y + a_6 y^2 + a_7 x + a_8 y + a_9$$

Then we have

$$z_x = 3a_0 x^2 + 2a_1 xy + a_2 y^2 + 2a_4 x + a_5 y + a_7$$

$$z_y = a_1 x^2 + 2a_2 xy + 3a_3 y^2 + a_5 x + 2a_6 y + a_8$$

$$z_{xx} = 6a_0 x + 2a_1 y + 2a_4$$

$$z_{yy} = 2a_2 x + 6a_3 y + 2a_6$$

$$z_{xy} = 2a_1 x + 2a_2 y + a_5 = z_{yx}$$

The curvature map can closely represent the shape of a surface if the mesh is sufficiently dense. However, it is not easy to locate a point on the curvature map

when given curvatures for the point. Linear search is cost prohibitive. Since the system only needs the coordinates of the point when locating the given curvatures, which will be discussed in the next chapter, we reorganize the curvature map into ordered list so that the system can use binary search to locate points efficiently.

Each curvature map is reorganized into the structure shown in Figure 6.5. Thus, we can view the logical structure of curvature maps as in Figure 6.4, and the physical structure of curvature maps as in Figure 6.5. In Figure 6.5, we can still see three layers of a curvature map. The Gaussian curvature layer and the magnitude of mean curvature layer are sorted arrays. Each value in either curvature layer is associated with a point coordinates. Note that K_i and $|H_i|$ do not necessarily associated with the same point.

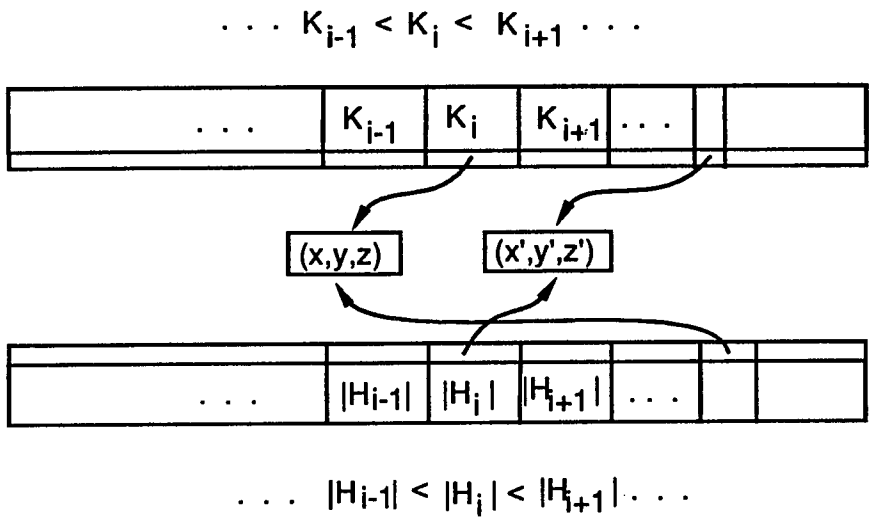

Figure 6.5: *Physical Structure of Curvature Map*

In the knowledge base, a model is a B-rep and curvature maps for surface patches of the boundary description. Notice that some curvature maps need not be stored, such planar surfaces. In fact, the curvature map of a plane is not meaningful. Only the plane normal direction is meaningful. Using the techniques discussed in the previous sections, we group some structures together if they are typically going to be used together or they have the similar characteristics.

From (6.1) and (6.2), the database can be view as a set of surface patches, each of which belongs to one of the models. That is,

$$M = \left\{ f_{ij} \mid f_{ij} \in m_i,\ 1 \le i \le n,\ 1 \le j \le p_i \right\} \tag{6.3}$$

where n is the number of models, p_i is the number of surface patches representing object model m_i. In f_{ij}, the first subscript i indicates the solid object model and the second subscript j indicates the surface patch on the solid object model.

The database M is to be divided into five disjoint subsets M_{planar}, $M_{parabolic}$, $M_{elliptic}$, $M_{hyperbolic}$ and M_{mix} such that

$$M_{planar} \cup M_{parabolic} \cup M_{elliptic} \cup M_{hyperbolic} \cup M_{mix} = M$$

and

$$M_s \cap M_t = \phi$$

$$s, t \in \{planar, parabolic, elliptic, hyperbolic, mix\}$$

The surface patches are classified into these five sets in the following way. M_{planar} contains surface patches with only planar points, i.e., $K = 0 = H$ at all points of the surface patches. $M_{parabolic}$ contains surface patches with only parabolic points, i.e., $K = 0$ and $H \neq 0$ at all points of the surface patches. $M_{elliptic}$ contains surface patches with only elliptic points, i.e., $K > 0$ at all points of the surface patches. $M_{hyperbolic}$ contains surface patches with only hyperbolic points, i.e., $K < 0$ at all points of the surface patches. M_{mix} contains surface patches which do not fall into the other four subsets.

The classification is performed easily, since we already have a curvature map for each surface patch. Apparently, it is not necessary to store the curvature maps for planar surface patches. Similarly, the Gaussian curvature layer of the curvature map for a parabolic surface patch does not need to be stored, since $K = 0$ all over the sur-

face patch.

However, we do need some other information about planes in order to perform recognition reasoning. First, each plane is associted with its surface normal. For a parametric surface function, the surface normal is

$$\mathbf{n} = \frac{\mathbf{p}_u \times \mathbf{p}_v}{|\mathbf{p}_u \times \mathbf{p}_v|}$$

For an explicit function $z = f(x,y)$, the surface normal is

$$\mathbf{n} = \frac{(-z_x, -z_y, 1)}{(1 + z_x^2 + z_y^2)^{1/2}}$$

If we have a plane represented as

$$Ax + By + Cz + D = 0$$

then the surface normal is even simpler,

$$\mathbf{n} = \frac{(A, B, C)}{(A^2 + B^2 + C^2)^{1/2}}$$

The direction of an individual plane normal will not help recognize objects. The relationship between plane normals within a model can help recognition reasoning as discussed in the next chapter.

Since

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}|\ |\mathbf{b}| \cos\theta$$

where θ is the angle between **a** and **b**, we can easily obtain the information on angles of planes within a model. Suppose

$$\left\{s_1, s_2, \ldots, s_k\right\} \subseteq m$$

where s_i, $1 \leq i \leq k$, is a plane and m is a B-rep model of surface patches. Each s_i has a surface normal $\mathbf{n}_{s_i}$. We can compute

$$\cos\theta = \mathbf{n}_{s_i} \cdot \mathbf{n}_{s_j}$$

for

$$s_i, s_j \in \left\{s_1, s_2, \ldots, s_k\right\}, \; i \neq j$$

and store this information in the database for use in recognition search. Note that if k is a large number, it is impossible to precompute and store all angle information. This extreme case happens when curved surfaces are approximated using small plane patches, which should not be the case in our system since models are described by smooth surfaces.

So far, a knowledge base of the models for solid objects is built. The knowledge base has most of the characteristics discussed before. It is very easy to update the models and recompute the curvature maps for new surface patches. Locating a point with given curvatures is simply achieved by binary search of curvature maps. We are now ready to move on to the next chapter for recognition and localization reasoning process.

Chapter 7: Control Strategy

The vision process in an autonomous robot normally involves the recognition of complex objects in scenes and the problems of estimating the position and orientation of an object. The above process is computationally intensive, and new models are needed that permit efficient ways of recognizing and localizing 3-D objects.

This portion of our work is about two topics. One is the recognition problem for complex objects using geometric modeling transformation. The other is the estimation of localization properties of the objects. Recognition and localization introduced for this problem are described in detail in the following sections.

7.1 Recognition and Localization Task

Suppose there are n objects, $O = \{o_1, o_2, \ldots, o_n\}$, in a scene, and there are k models, $M = \{m_1, m_2, \ldots, m_k\}$, in the model database. Our task is to find a matching

$$o_i = TRANSFORM(m_j) \tag{7.1}$$

where $o_i \in O$, $m_j \in M$ and *TRANSFORM* is a transformation. That is, we should find a matching between o_i and m_j for the recognition task, and also to find the transformation *TRANSFORM* for the localization task.

Since we can only find a set of surface segments $S = \{s_1, s_2, \ldots, s_t\}$ in an image, the task becomes one of finding the membership of an image surface segment

in the set of models, i.e., finding

$$s_i \in TRANSFORM(m_j) \tag{7.2}$$

where $s_i \in S$, and $m_j \in M$.

Since each model is described in terms of a set of surface patches

$$m_i = \left\{ f_{i1}, f_{i2}, \ldots, f_{ip_i} \right\}$$

the task becomes one of finding the membership of an image surface segment in a subset of the surface patches of a certain model, i.e.,

$$s_i \in TRANSFORM(m'_j)$$

where $s_i \in S$, and $m'_j \subseteq m$.

Since to search all possible combinations of image surface segments with model surface patches could be cost prohibitive, the task becomes an AI search problem. Since heuristic search is well-defined and understandable, we introduce our approach in terms of heuristic search.

7.2 Main Algorithm

Our algorithm is a heuristic search procedure using an evaluation function to direct search through the state space. A curved surface in an image contains more heuristic information by itself than a planar surface, which can be understood intuitively. We focus our proposed technique on curved surface segments differently from planar surface segments in an image. First, we use information from curved surface

segments to direct the search. If no curved segments can be used, the planar surface segments are used to continue searching.

The main algorithm and its flow diagram are as follows.

MAIN ALGORITHM

1. If the input image is empty, then exit.
2. Let s be the set of all models, each of which consists of a set of surface patches. Set *TEMP* and P to ϕ.
3*. Find a curved surface segment p' in the input image such that p' is close to one segment in P and p' is not in *TEMP*. Add p' to P. If not found, go to step 13.
4*. Obtain the subset $s' \subseteq s$ such that $p' \in P$ is matched to the surface patches of m_i, for all $m_i \in s'$.
5. If $|s'| = 0$, delete p' from P and place p' in *TEMP*. (If $|P| = 0$, erase p' from the image.) Go to step 3.
6*. If $|s'| = 1$, a hypothesis is formed. Go to step 9 for verification.
7*. If there are three surface segments in P that match with the same model in s', then find the model m in s' which matches the most segments in P. Delete m from s'. Go to step 9 for verification.
8. Let $s = s'$. Go to step 3.
9*. If verification succeeds, erase *TRANSFORM* (m) from the input image. Go to step 1.

10*. Check whether surfaces of m can match to segments in P at different positions. If another matching position is found, go to step 9.

11. If $|s'| = 0$, fail to recognize all segments in P. Erase all $p' \in P$ from the image. Go to step 1.

12. Go to step 6.

13*. If $|P| > 0$, find a planar surface segment in the image close to one segment in P. If found, go to step 4, otherwise go to step 15.

14*. Find three planar surface segments close to each other. If found, go to step 4.

15. Exit with failure.

Note: Steps with * will be discussed in detail in the following sections.

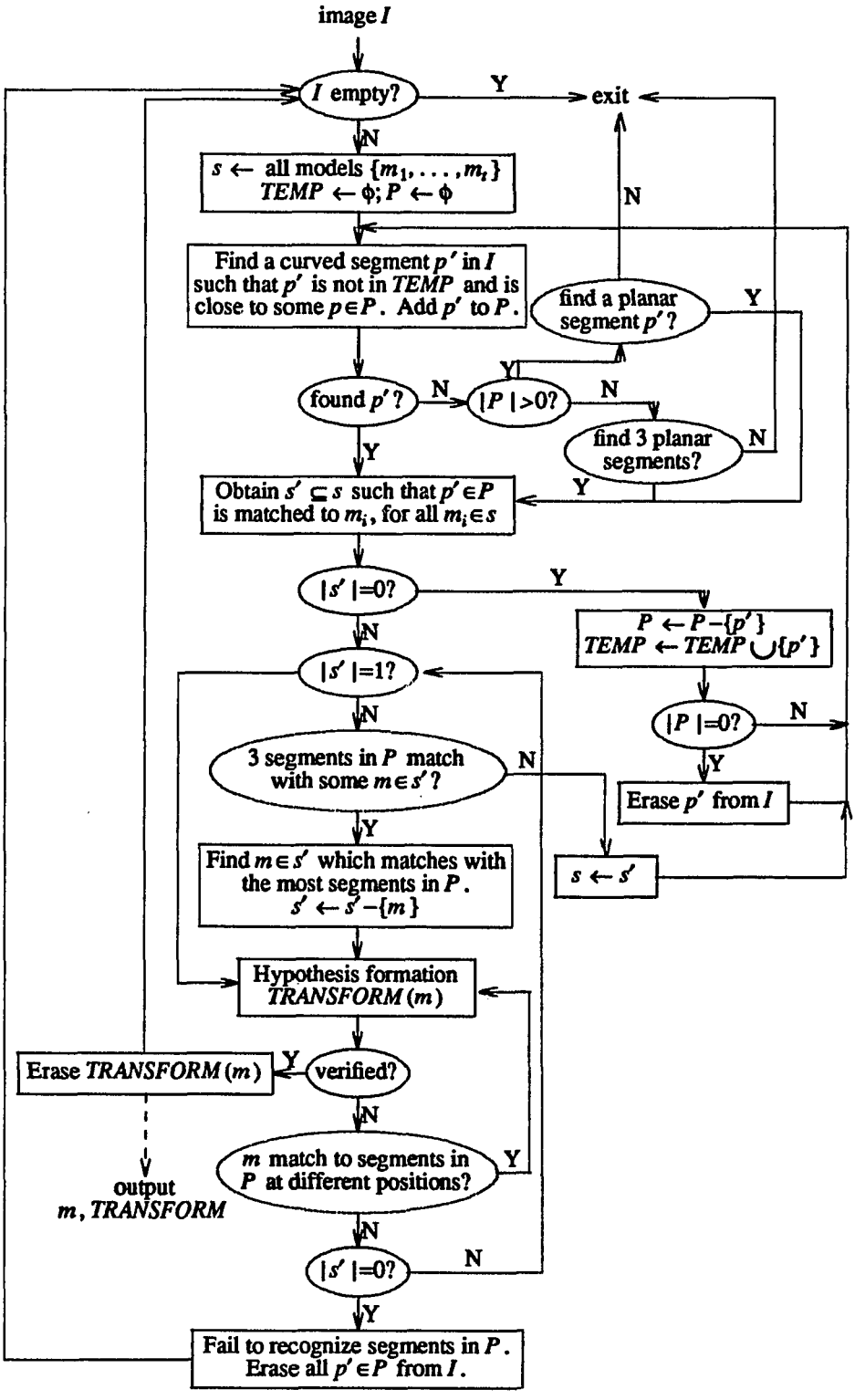

Figure 7.1: *Flow Diagram of the Main Algorithm*

The correctness of the algorithm can be established with the following theorems.

Theorem 7.1. The algorithm always terminates.

Proof. Whenever a matching (7.2) of image surface segments and model surfaces is verified at step 9, all the image surface segments $s_i \in TRANSFORM(m_j)$ are erased from the image. If a segment cannot match with any model surface patches, it is erased at step 5 from the image. If all possible matches of surface segments in P and models cannot be verified, surface segments in P are erased from the image at step 11. When the image surface segments are all erased at step 9, 11, or 5, the algorithm terminates at step 2. If each left-over surface segment in the image can match to several models but no more than two segments can match to the same model, the algorithm terminates at step 15.

Since accuracy and efficiency are the most important factors in application, we designed the algorithm to guarantee that the match is accurate and that finding the match is efficient in the cost of failing to recognize objects in some extreme cases. A heuristic search algorithm is *complete* if it guarantees finding a solution when a solution exists.

Theorem 7.2. The algorithm is not complete.

Proof. The above definition of completeness is used here. The incompleteness is caused at step 15, where the algorithm terminates when there might be some possible matches. Since at that point each surface segments in P matches to more than one

model, but no more than two segments go to the same model, it is either that the visible areas of objects in the image are too small or that there are no corresponding models for objects in the image.

Theorem 7.3. The algorithm is complete if it explores every possible match of surface segments in P and models in M before it exits at step 15.

Proof. If the algorithm checks every match before it exits at step 15, all the possible matches can be explored. If there is a solution, the algorithm guarantees finding it. However, in the extreme cases when each surface segment in P matches to more than one model but no more than two surface segments match to surface patches on the same model, the algorithm spends too much time on checking all possibilities before ending the process. This is not generally acceptable in real applications.

The original algorithm is more applicable in practice though it is incomplete. Since in robot vision applications the system must respond within a time constraint, which is called a real-time system, we cannot afford the time to explore all possibilities.

Theorem 7.4. The algorithm is admissible.

Proof. If the search algorithm returns an optimal solution, the search algorithm is *admissible.* Our algorithm is admissible in the sense that the algorithm finds the exact matching between object surface segments in the image and surface patches of the model, which is guaranteed by the verification.

There are a lot of trade-offs in application systems. Heuristics help reduce the amount of search. However, applying heuristics to find the search directions is also expensive. The more heuristics we use, the more accurately it directs the search. Therefore, the search is more efficient. However, the more heuristics we use, the higher the cost that the evaluation function has. It is very difficult to design an evaluation function which not only performs efficiently but also contains a great deal of heuristic information.

Hence, the total cost of a search system can be classified as two classes: one is the cost of the search process; another is the cost of heuristic evaluation operator. The total cost of a search system can be shown informally in Figure 7.2.

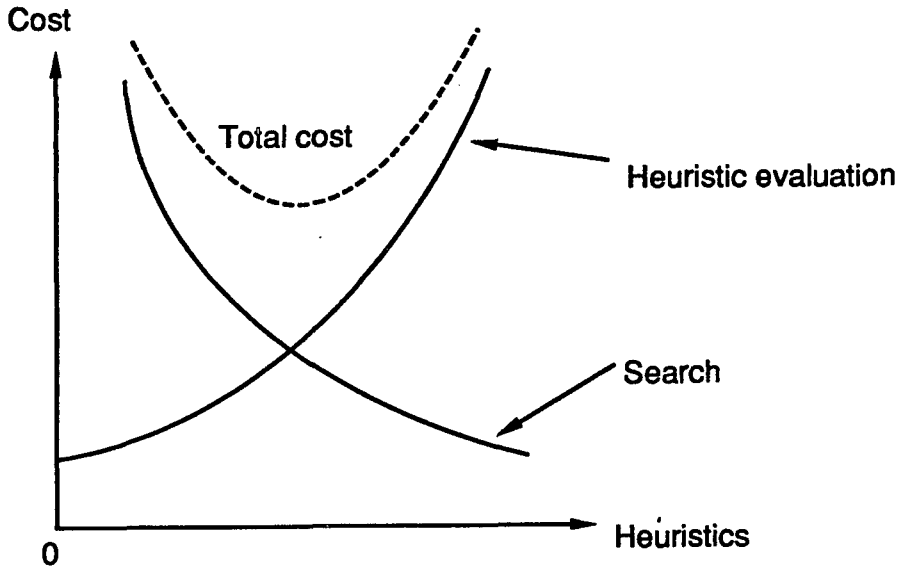

Figure 7.2: *Search System Cost*

In designing a search system, we first have to find trade-offs, considering the application requirements and the factors of the search system.

One way to design heuristic evaluation functions is to design multiple functions to deal with different situations so that each function is efficient. Our approach is to use different heuristic information and different evaluation functions to deal with curved surface segments and planar surface segments in the input image, which has been proven to be efficient.

7.3 Image Segmentation

This section discusses techniques used to obtain a surface segment in steps 3, 13 and 14.

The knowledge about model surface patches is represented as a Gaussian curvature map and a magnitude of mean curvature map for curved surfaces, and angles between surface normals for planar surface patches. Therefore, the useful features in the image are the curvatures of curved surface segments and angles between planar surface segments.

In order to find these features, the simplest method is to use an approximation function to apply to a small window of pixels to find the curvatures, and then to classify a region of neighboring pixels into surface segments such that each segment contains a smooth surface.

Suppose that we use a 5×5 window to obtain the curvatures of the central pixel of the window as shown in Figure 7.3.

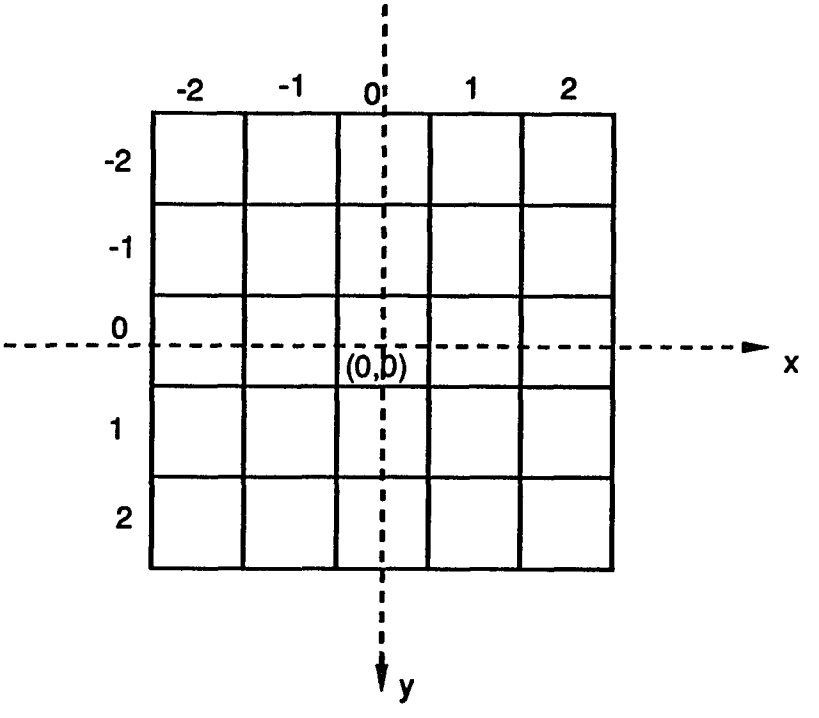

Figure 7.3: *Window for Approximation to Obtain Curvatures*

Let's use an explicit function $z = f(x,y)$ to approximate the window in order to obtain the first and second derivatives on the central pixel to compute the curvatures. Let

$$z = a_0x^3 + a_1x^2y + a_2xy^2 + a_3y^3 + a_4x^2 + a_5xy + a_6y^2 + a_7x + a_8y + a_9 \quad (7.3)$$

Let the center of the window be $(x,y) = (0, 0)$, and every pixel is one unit, as shown in Figure 7.3. We can use a least-squares approach to find the approximation efficiently. Let

$$\mathbf{x} = [a_0, a_1, \ldots, a_9]^T$$

$$\mathbf{y} = [z(-2,-2), z(-1,-2), \ldots, z(2,-2), z(-2,-1), z(-1,-1), \ldots, z(2,-1), \ldots, z(2,2)]^T$$

$$A = \begin{bmatrix} x_0^3 & x_0^2 y_0 & x_0 y_0^2 & y_0^3 & \cdots & x_0 & y_0 & 1 \\ x_1^3 & x_1^2 y_1 & x_1 y_1^2 & y_1^3 & \cdots & x_1 & y_1 & 1 \\ . & . & . & . & \cdots & . & . & . \\ . & . & . & . & \cdots & . & . & . \\ . & . & . & . & \cdots & . & . & . \\ x_{24}^3 & x_{24}^2 y_{24} & x_{24} y_{24}^2 & y_{24}^3 & \cdots & x_{24} & y_{24} & 1 \end{bmatrix}$$

$$= \begin{bmatrix} -8 & -8 & -8 & -8 & \cdots & -2 & -2 & 1 \\ -1 & -2 & -4 & -8 & \cdots & -2 & -2 & 1 \\ . & . & . & . & \cdots & . & . & . \\ . & . & . & . & \cdots & . & . & . \\ . & . & . & . & \cdots & . & . & . \\ 8 & 8 & 8 & 8 & \cdots & 2 & 2 & 1 \end{bmatrix}$$

where $(x_0, y_0) = (-2, -2)$, $(x_1, y_1) = (-1, -2)$, $(x_2, y_2) = (0,-2), \ldots,$ $(x_{24}, y_{24}) = (2,2)$.

We have an over determined system

$$A\mathbf{x} = \mathbf{y} \tag{7.4}$$

The least-squares method is to find an x such that

$$|A\mathbf{x} - \mathbf{y}|^2 = \text{minimum}$$

Multiply A^T to both sides of (7.4), thus, we have

$$A^T A\mathbf{x} = A^T \mathbf{y} \tag{7.5}$$

where

$$A^TA = \begin{bmatrix} 650 & 0 & 340 & 0 & 0 & 0 & 0 & 170 & 0 & 0 \\ 0 & 340 & 0 & 340 & 0 & 0 & 0 & 0 & 100 & 0 \\ 340 & 0 & 340 & 0 & 0 & 0 & 0 & 100 & 0 & 0 \\ 0 & 340 & 0 & 650 & 0 & 0 & 0 & 0 & 170 & 0 \\ 0 & 0 & 0 & 0 & 170 & 0 & 100 & 0 & 0 & 50 \\ 0 & 0 & 0 & 0 & 0 & 100 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 100 & 0 & 170 & 0 & 0 & 50 \\ 170 & 0 & 100 & 0 & 0 & 0 & 0 & 50 & 0 & 0 \\ 0 & 100 & 0 & 170 & 0 & 0 & 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 0 & 50 & 0 & 50 & 0 & 0 & 25 \end{bmatrix}$$

Since A^TA is nonsingular, we have

$$(A^TA)^{-1} = \begin{bmatrix} 0.013889 & 0 & 0 & 0 & 0 & 0 & 0 & -0.047222 & 0 & 0 \\ 0 & 0.007143 & 0 & 0 & 0 & 0 & 0 & 0 & -0.014286 & 0 \\ 0 & 0 & 0.007143 & 0 & 0 & 0 & 0 & -0.014286 & 0 & 0 \\ 0 & 0 & 0 & 0.013889 & 0 & 0 & 0 & 0 & -0.047222 & 0 \\ 0 & 0 & 0 & 0 & 0.014286 & 0 & 0 & 0 & 0 & -0.028571 \\ 0 & 0 & 0 & 0 & 0 & 0.01 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.014286 & 0 & 0 & -0.028571 \\ -0.047222 & 0 & -0.014286 & 0 & 0 & 0 & 0 & 0.209127 & 0 & 0 \\ 0 & -0.014286 & 0 & -0.047222 & 0 & 0 & 0 & 0 & 0.209127 & 0 \\ 0 & 0 & 0 & 0 & -0.028571 & 0 & -0.028571 & 0 & 0 & 0.154286 \end{bmatrix}$$

Thus, multiply $(A^TA)^{-1}$ to both sides of (7.3), we have

$$\mathbf{x} = (A^TA)^{-1}A^T\mathbf{y} \tag{7.6}$$

Let

$$B = (A^TA)^{-1}A^T$$

Then

$$\mathbf{x} = B\mathbf{y} \tag{7.7}$$

where B is a 10×25 matrix which is computed before the system starts. $\mathbf{y}$ is the data from the window. Therefore, the coefficients of the approximation function (7.3) are easily obtained.

Since the purpose of finding the approximation function is to compute the curvatures at the central pixel of the window, we can derive a more efficient way to perform the task.

The first derivatives of (7.3) are

$$z_x = 3a_0x^2 + 2a_1xy + a_2y^2 + 2a_4x + a_5y + a_7$$

$$z_y = a_1x^2 + 2a_2xy + 3a_3y^2 + a_5x + 2a_6y + a_8$$

and the second derivatives of (7.3) are

$$z_{xx} = 6a_0x + 2a_1y + 2a_4$$

$$z_{yy} = 2a_2x + 6a_3y + 2a_6$$

$$z_{xy} = 2a_1x + 2a_2y + a_5 = z_{yx}$$

Substituting the coordinates of the center of the window which are (0, 0). At the central pixel of the window we have

$$\begin{aligned} z_x &= a_7 \\ z_y &= a_8 \\ z_{xx} &= 2a_4 \\ z_{yy} &= 2a_6 \\ z_{xy} &= a_5 \end{aligned} \tag{7.8}$$

If we represent B in (7.7) as

$$B = \begin{bmatrix} B_0 \\ B_1 \\ B_2 \\ . \\ . \\ . \\ B_9 \end{bmatrix}$$

where B_i, $0 \le i \le 9$, is a 1×25 matrix which is the ith row of B, then we have

$$
\begin{aligned}
z_x &= a_7 = B_7\mathbf{y} \\
z_y &= a_8 = B_8\mathbf{y} \\
z_{xx} &= 2a_4 = 2B_4\mathbf{y} \\
z_{yy} &= 2a_6 = 2B_6\mathbf{y} \\
z_{xy} &= a_5 = B_5\mathbf{y}
\end{aligned}
\tag{7.9}
$$

Since **y** is a 5×5 window on the image, we can represent the above as 5×5 window operators so that we just apply the operators to the window. An operator is represented as a 5×5 matrix. When applying the operator, we just multiply the corresponding elements in the window and then add them together. The operators are give below.

$$
z_x :: \begin{bmatrix}
0.073810 & -0.104762 & 0.0 & 0.104762 & -0.073810 \\
-0.011905 & -0.147619 & 0.0 & 0.147619 & 0.011905 \\
-0.040476 & -0.161905 & 0.0 & 0.161905 & 0.040476 \\
-0.011905 & -0.147619 & 0.0 & 0.147619 & 0.011905 \\
0.073810 & -0.104762 & 0.0 & 0.104762 & -0.073810
\end{bmatrix}
$$

$$
z_y :: \begin{bmatrix}
0.073810 & -0.011905 & -0.040476 & -0.011905 & 0.073810 \\
-0.104762 & -0.147619 & -0.161905 & -0.147619 & -0.104762 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.104762 & 0.147619 & 0.161905 & 0.147619 & 0.104762 \\
-0.073810 & 0.011905 & 0.040476 & 0.011905 & -0.073810
\end{bmatrix}
$$

$$
z_{xx} :: \begin{bmatrix}
0.057143 & -0.028571 & -0.057143 & -0.028571 & 0.057143 \\
0.057143 & -0.028571 & -0.057143 & -0.028571 & 0.057143 \\
0.057143 & -0.028571 & -0.057143 & -0.028571 & 0.057143 \\
0.057143 & -0.028571 & -0.057143 & -0.028571 & 0.057143 \\
0.057143 & -0.028571 & -0.057143 & -0.028571 & 0.057143
\end{bmatrix}
$$

$$z_{yy} :: \begin{bmatrix} 0.057143 & 0.057143 & 0.057143 & 0.057143 & 0.057143 \\ -0.028571 & -0.028571 & -0.028571 & -0.028571 & -0.028571 \\ -0.057143 & -0.057143 & -0.057143 & -0.057143 & -0.057143 \\ -0.028571 & -0.028571 & -0.028571 & -0.028571 & -0.028571 \\ 0.057143 & 0.057143 & 0.057143 & 0.057143 & 0.057143 \end{bmatrix}$$

$$z_{xy} :: \begin{bmatrix} 0.04 & 0.02 & 0.0 & -0.02 & -0.04 \\ 0.02 & 0.01 & 0.0 & -0.01 & -0.02 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.02 & -0.01 & 0.0 & 0.01 & 0.02 \\ -0.04 & -0.02 & 0.0 & 0.02 & 0.04 \end{bmatrix}$$

Now we can apply (5.32) and (5.33) to obtain the curvature.

$$K = \frac{z_{xx} z_{yy} - z_{xy}^2}{(1 + z_x^2 + z_y^2)^{/2}}$$

$$H = \frac{(1 + z_x^2) z_{yy} + (1 + z_y^2) z_{xx} - 2 z_x z_y z_{xy}}{2(1 + z_x^2 + z_y^2)^{3/2}}$$

In this way, we can obtain curvatures for each pixel in a small region. A smooth surface is a surface which has continuous first and second derivatives. Therefore, we should not include edge points in the surface segment region in order to obtain a smooth surface segment. Since edge points must have high curvature, we can determine whether a point is on an edge by testing the principal curvatures at that point. From (5.27), (5.28) and (5.29), we have

$$k_1 = H - (H^2 - K)^{\frac{1}{2}}$$

$$k_2 = H + (H^2 - K)^{\frac{1}{2}}$$

If either $|k_1|$ or $|k_2|$ is greater than a threshold $T > 0$, the point is on an edge. The selection of T is by experimentation.

Finally, we obtain a local curvature map of a smooth surface segment in the image. The surface nature can be determined according to the curvatures. If the whole map has $|K| \leq K_{zero}$ and $|H| \leq H_{zero}$, the surface is a plane.

K_{zero} and H_{zero} are also thresholds for testing the zero of K and H, since K and H cannot really be zero in the input image due to noise and quantization effects. Since K is the product of principal curvatures and H is the average of principal curvatures, we let $K_{zero} \geq H_{zero}^2$. These thresholds are also selected through experimentation.

After the surface nature and the local curvature map are obtained, they are used as heuristic information in the evaluation function at step 4 in the main algorithm.

7.4 Surface Matching Evaluation

We discuss how to perform steps 4 and 10 of the main algorithm in this section.

After the curvature map of a surface segment is obtained, we can evaluate how close the curvature map matches with the curvature maps of model surface patches in set s. s has all models initially and then contains a subset of models after one or more iterations.

If the curvature map of the surface segment indicates the surface segment as a plane, then the surface segment can match to all planar surface patches of models in s. Therefore, at step 4, all models not including any planar surface patches are deleted from s to form s'.

If the surface segment is curved, we need more work to select the subset s' of s at step 4. The surface segment curvature map can be viewed as a three-layer data structure. The first layer is the surface segment from the image where every pixel is associated with its coordinates (x,y,z) in the 3-D space, i.e., the pixel coordinates in the image associates with (x,y) and the pixel value is z, as shown in Figure 7.4.

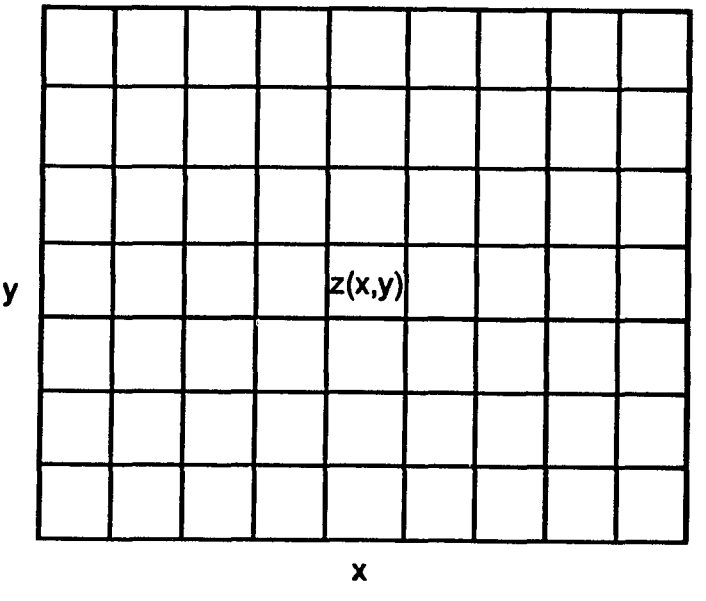

Figure 7.4: *Surface Segment Layer*

The other two layers are the Gaussian curvatures and the magnitudes of mean curvature, which are the same size as the surface segment layer. Every pixel in the segment has its Gaussian curvature and the magnitude of its mean curvature in these two layers as shown in Figure 7.5 and Figure 7.6, respectively.

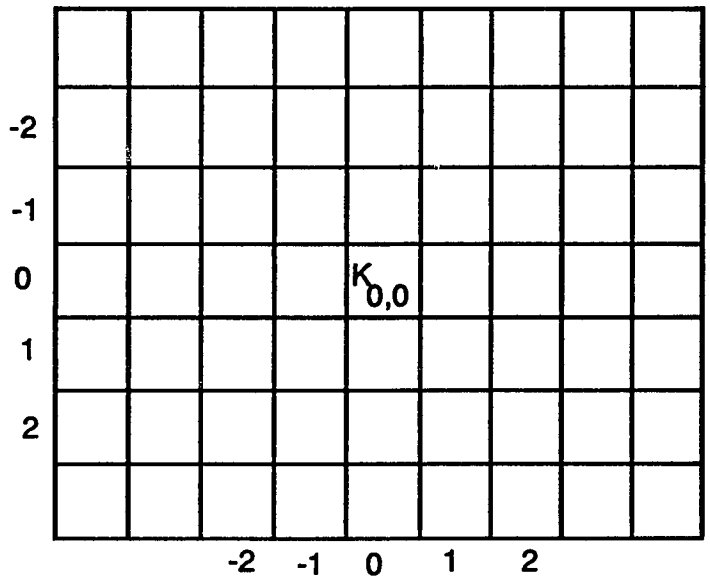

Figure 7.5: *Gaussian Curvature Layer*

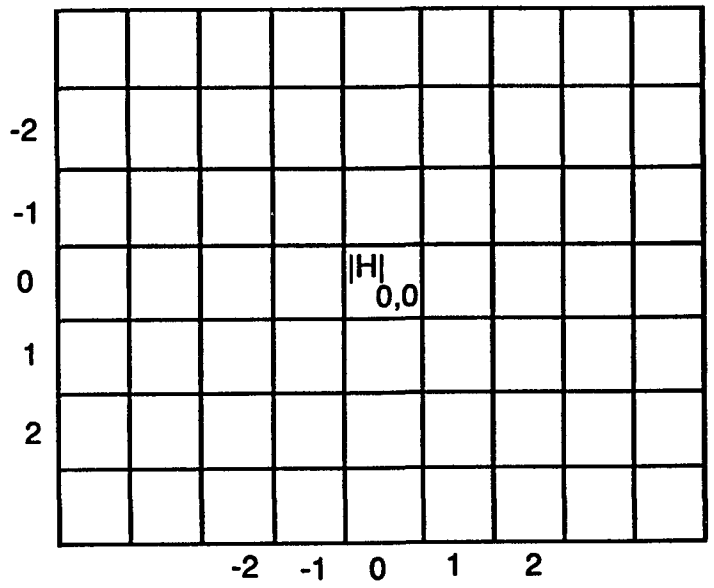

Figure 7.6: *Magnitude of Mean Curvature Layer*

Now let's take a model m from s to check whether the surface segment can match to its surface patches. As in Section 6.3 above, we can determine to which set of $M_{elliptic}$, $M_{parabolic}$, $M_{hyperbolic}$, or M_{mix} the surface segment can be classified, and then look for the possible matching surface patch in the corresponding set. Note that M_{mix} should be checked if a surface segment cannot be found in other sets.

Since the surface segment is smooth, the center shape of the surface segment is least affected by noise. We first locate the possible positions of the center pixel of the surface segment on the model m. From the initial positions, if we can develop a local fitting of the surface segment to the model surfaces, it is possible that the surface segment matches with the model. Then, the fitting position of the center of the segment

is also recorded.

In the previous chapter, we discussed the knowledge representation of surface shapes of the models. The surface shape is represented as a curvature map for a surface patch. The curvature maps are ordered lists, sorted by values of K and $|H|$.

The initial possible positions of the segment center are located in the following way. Suppose that the center of the surface segment is on the (0, 0) pixel of the segment curvature map, as shown in Figure 7.5 and Figure 7.6. Let

$$\alpha = \max\left\{ |K_{(0,0)} - K_{(i,j)}| : i,j = -1, 0, 1 \right\}$$

$$\beta = \max\left\{ |\,|H|_{(0,0)} - |H|_{(i,j)}\,| : i,j = -1, 0, 1 \right\}$$

Thus, α is the maximum difference between the central K value and its neighboring K value, and β is the maximum difference between the central $|H|$ value and its neighboring values.

A point $\mathbf{p}$ on the model with Gaussian curvature $K_{\mathbf{p}}$ and magnitude of mean curvature $|H|_{\mathbf{p}}$ is a possible matching position of the center of the surface segment if

$$|\,K_{(0,0)} - K_{\mathbf{p}}\,| < \alpha$$

and

$$|\,|H|_{(0,0)} - |H|_{\mathbf{p}}\,| < \beta$$

These points can be found efficiently by binary search from model surface curvature maps which are represented as ordered lists. Let these initial points be in set SP, let

the distance between two neighboring pixels be $\frac{1}{2}\varepsilon$, and let the noise rate be σ. Then we use the following procedure to check whether the points in SP can be selected as corresponding points to the center of the surface segment:

Procedure Fitting

1. n = total number of pixels of the surface segment. $u = 0$, which is the number of pixels unmatched.
2. If $SP = \phi$, exit with failure.
3. Take $\mathbf{p}$ from SP. $SP = SP - \{\mathbf{p}\}$.
4. For every noncentral position (i, j) of the surface segment curvature map
5. $\alpha' = \max\left\{ |K_{(i,j)} - K_{(i+i',j+j')}| : i', j' = -1, 0, 1 \right\}$
6. $\beta' = \max\left\{ |\,|H|_{(i,j)} - |H|_{(i+i',j+j')}\,| : i', j' = -1, 0, 1 \right\}$
7. For every point $\mathbf{p}'$ on the model m with
   $|K_{(i,j)} - K_{\mathbf{p}'}| < \alpha'$ and $|\,|H|_{(i,j)} - |H|_{\mathbf{p}'}| < \beta'$
8. If $|\text{distance}(seg\_p_{(0,0)}, seg\_p_{(i,j)}) - \text{distance}(\mathbf{p}', \mathbf{p})| < \varepsilon$, go to step 4.
9. End_for /* step 7 */
10. $u = u + 1$.
11. If $\frac{u}{n} > \sigma$, then go to step 2.

12. End_for /* step 4 */

13. Exit with success.

Note that the procedure **Fitting** only finds a very possible match when it exits successfully. The procedure only does checking on the distance between points, which is a necessary condition for a match but not a sufficient condition. If we want to inspect all conditions to guarantee the exact match, however, the process will be very expensive. Referring to Figure 7.2, we have a trade-off between the evaluation cost and the search cost.

7.5 Hypothesis Formation

In this section, we discuss the techniques in steps 6 and 7 of the main algorithm, which are used to form a hypothesis for verification in step 9.

In section 7.1, we defined our task as finding matchings between surface segments in the image and a transformation of model surfaces

$$s_i \in TRANSFORM(m_j)$$

where $s_i \in s$, which is a set of surface segments, and $m_j \in M$, which is a set of models. We have discussed how to find a possible membership of s_i on m_j. However, in order to verify the matching, we have to find *TRANSFORM* and then check whether s_i is really on $TRANSFORM(m_j)$. Thus, at this point, we just need to find *TRANSFORM* to obtain the hypothesis.

If $|s'| > 1$ at step 7, we select a model m from s' such that m has the most possible matching surface segments in P to form the hypothesis.

If there are three curved surface segments matched to m, and their possible positions on m have been found in previous steps as discussed in section 7.4, then we perform a distance checking between these points similar to that in procedure **Fitting**. If the distance checking fails, the hypothesis cannot be formed. Otherwise, we can obtained *TRANSFORM* from the relationship of the central points of the three surface segments and the corresponding positions on the model m since three noncollinear points uniquely determine a 3-D spatial transformation.

If there is only one curved surface segment matched with model m, which is possible at step 6, then we use the central point of the surface segment and two additional noncentral points to obtain the transformation.

Let the three points from the surface segments be (sx_0, sy_0, sz_0), (sx_1, sy_1, sz_1) and (sx_2, sy_2, sz_2). Let their corresponding points on the model m be (x_0, y_0, z_0), (x_1, y_1, z_1) and (x_2, y_2, z_2).

A transformation of points in 3-D space is represented explicitly as

$$[x' \quad y' \quad z'] = [x \quad y \quad z]R + T \tag{7.10}$$

where R is a 3×3 rotation matrix and T is a 1×3 translation matrix.

A rotation about the z-axis in 3-D is

$$R_z(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The x-axis rotation matrix is

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}$$

The y-axis rotation matrix is

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}$$

Thus, R in (7.10) is the composition of an arbitrary sequence of rotations about the x, y, and z axes. We have

$$R = \begin{bmatrix} r_{00} & r_{01} & r_{02} \\ r_{10} & r_{11} & r_{12} \\ r_{20} & r_{21} & r_{22} \end{bmatrix}$$

$$T = [t_x \quad t_y \quad t_z]$$

To find the transformation of the three points, we can first pin down the translation easily.

$$\begin{aligned} T &= [t_x \quad t_y \quad t_z] \\ &= [sx_0 - x_0 \quad sy_0 - y_0 \quad sz_0 - z_0] \end{aligned}$$

When we substitute the three points in (7.10), we have

$$[sx_0 \quad sy_0 \quad sz_0] = [x_0 \quad y_0 \quad z_0]R + T \tag{7.11}$$

$$[sx_1 \;\; sy_1 \;\; sz_1] = [x_1 \;\; y_1 \;\; z_1]R + T \tag{7.12}$$

$$[sx_2 \;\; sy_2 \;\; sz_2] = [x_2 \;\; y_2 \;\; z_2]R + T \tag{7.13}$$

From (7.11), (7.12) and (7.13), we get

$$sx_0 = x_0r_{00} + y_0r_{10} + z_0r_{20} + t_x$$

$$sx_1 = x_1r_{00} + y_1r_{10} + z_1r_{20} + t_x$$

$$sx_2 = x_2r_{00} + y_2r_{10} + z_2r_{20} + t_x$$

Similarly, we get

$$sy_0 = x_0r_{01} + y_0r_{11} + z_0r_{21} + t_y$$

$$sy_1 = x_1r_{01} + y_1r_{11} + z_1r_{21} + t_y$$

$$sy_2 = x_2r_{01} + y_2r_{11} + z_2r_{21} + t_y$$

and

$$sz_0 = x_0r_{02} + y_0r_{12} + z_0r_{22} + t_z$$

$$sz_1 = x_1r_{02} + y_1r_{12} + z_1r_{22} + t_z$$

$$sz_0 = x_2r_{02} + y_2r_{12} + z_2r_{22} + t_z$$

From the above three linear systems, we obtain R. Therefore, the hypothesis is formed, which is a matching between surfaces and their transformation.

If there are only planar segments matched with model m, we need three non-parallel planes to determine a transformation in 3-D space.

To obtain a plane equation

$$Ax + By + Cz + D = 0 \tag{7.14}$$

we need three non-collinear points on the plane. Given three noncollinear points (x_1, y_1, z_1), (x_2, y_2, z_2) and (x_3, y_3, z_3) on the plane, we have

$$\begin{aligned} Ax_1 + By_1 + Cz_1 + D &= 0 \\ Ax_2 + By_2 + Cz_2 + D &= 0 \\ Ax_3 + By_3 + Cz_3 + D &= 0 \end{aligned}$$

Since the three points are not collinear, the equations can be solved for A, B, C and D by arbitrarily assigning a value to one of them and then solving the resulting three equations in three unknowns. A better way to obtain (7.14) is to select any point (x, y, z) on the plane, then we have

$$\begin{aligned} Ax + By + Cz + D &= 0 \\ Ax_1 + By_1 + Cz_1 + D &= 0 \\ Ax_2 + By_2 + Cz_2 + D &= 0 \\ Ax_3 + By_3 + Cz_3 + D &= 0 \end{aligned}$$

If there is a nontrivial (nonzero) solution to this homogeneous system, the determinant of its coefficients

$$\det \begin{bmatrix} x & y & z & 1 \\ x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \end{bmatrix}$$

must be zero. Expanding by cofactors about the first row, we have

$$A'x + B'y + C'z + D' = 0 \tag{7.15}$$

where

$$A' = \det\begin{bmatrix} y_1 & z_1 & 1 \\ y_2 & z_2 & 1 \\ y_3 & z_3 & 1 \end{bmatrix}$$

$$B' = -\det\begin{bmatrix} x_1 & z_1 & 1 \\ x_2 & z_2 & 1 \\ x_3 & z_3 & 1 \end{bmatrix}$$

$$C' = \det\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix}$$

$$D' = -\det\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$$

Thus, (7.15) is the equation we want. The three points cannot be collinear. If all cofactors in (7.15) are zero, collinearity occurs.

The surface normal of (7.14) is

$$\mathbf{n} = \frac{(A, B, C)}{(A^2 + B^2 + C^2)^{1/2}}$$

Let the three plane segments be

$$A'_1 x + B'_1 + C'_1 z + D'_1 = 0 \tag{7.16}$$

$$A'_2 x + B'_2 + C'_2 z + D'_2 = 0 \tag{7.17}$$

$$A'_3 x + B'_3 + C'_3 z + D'_3 = 0 \tag{7.18}$$

and their surface normals be

$$\mathbf{n}'_1 = \frac{(A'_1, B'_1, C'_1)}{(A'^2_1 + B'^2_1 + C'^2_1)^{1/2}}$$

$$\mathbf{n}'_2 = \frac{(A'_2, B'_2, C'_2)}{(A'^2_2 + B'^2_2 + C'^2_2)^{1/2}}$$

$$\mathbf{n}'_3 = \frac{(A'_3, B'_3, C'_3)}{(A'^2_3 + B'^2_3 + C'^2_3)^{1/2}}$$

In order to determine a 3-D transformation uniquely, we must have $\mathbf{n}_i \neq \mathbf{n}_j$, $1 \leq i, j \leq 3, i \neq j$.

The angle between two planes is obtained from

$$\cos\theta = \mathbf{n}_i \cdot \mathbf{n}_j$$

$$= \frac{A_i A_j + B_i B_j + C_i C_j}{(A_i^2 + B_i^2 + C_i^2)^{1/2}(A_j^2 + B_j^2 + C_j^2)^{1/2}}$$

Since we have had all the angle information between planes in a model, we should inspect the angles between any two planar surface segments and compare with the angles between planes in the model. If a matching can be found, we can proceed to find the transformation from the corresponding planes to form a hypothesis.

Suppose the three corresponding planes on the model m are

$$A_1x + B_1 + C_1z + D_1 = 0 \tag{7.19}$$

$$A_2x + B_2 + C_2z + D_2 = 0 \tag{7.20}$$

$$A_3x + B_3 + C_3z + D_3 = 0 \tag{7.21}$$

Let's first determine the rotation matrix of the transformation, i.e. to rotate the model m such that after rotation (7.19) is parallel to (7.16), (7.20) is parallel to (7.17), and (7.21) is parallel to (7.18). We use the rotation matrix in (7.10). Substitute the transformation in (7.19). We have

$$\begin{aligned} &A_1(xr_{00} + yr_{10} + zr_{20} + t_x) \\ &-B_1(xr_{01} + yr_{11} + zr_{21} + t_y) \\ &+C_1(xr_{02} + yr_{12} + zr_{22} + t_z) + D_1 = 0 \end{aligned}$$

The new equation is

$$\begin{aligned} &(A_1r_{00} + B_1r_{01} + C_1r_{02})x \\ &+ (A_1r_{10} + B_1r_{11} + C_1r_{12})y \\ &+ (A_1r_{20} + B_1r_{21} + C_1r_{22})z \\ &D_1 + t_x + t_y + t_z = 0 \end{aligned} \tag{7.22}$$

Since (7.22) is parallel to (7.16), we have

$$\begin{aligned} \frac{A_1r_{00} + B_1r_{01} + C_1r_{02}}{A'_1} &= \frac{A_1r_{10} + B_1r_{11} + C_1r_{12}}{B'_1} \\ &= \frac{A_1r_{20} + B_1r_{21} + C_1r_{22}}{C'_1} \end{aligned} \tag{7.23}$$

Similarly, from (7.20) and (7.21), we have

$$\frac{A_2r_{00}+B_2r_{01}+C_2r_{02}}{A'_2} = \frac{A_2r_{10}+B_2r_{11}+C_2r_{12}}{B'_2} = \frac{A_2r_{20}+B_2r_{21}+C_2r_{22}}{C'_2} \tag{7.24}$$

and

$$\frac{A_3r_{00}+B_3r_{01}+C_3r_{02}}{A'_3} = \frac{A_3r_{10}+B_3r_{11}+C_3r_{12}}{B'_3} = \frac{A_3r_{20}+B_3r_{21}+C_3r_{22}}{C'_3} \tag{7.25}$$

Thus, we can solve nine unknowns in nine linear equations of (7.23), (7.24) and (7.25).

In order to let (7.22) and (7.16) represent the same plane, we have other conditions shown as follows:

$$\frac{D_1+t_x+t_y+t_z}{D'_1} = \frac{A_1r_{00}+B_1r_{01}+C_1r_{02}}{A'_1}$$

$$\frac{D_2+t_x+t_y+t_z}{D'_2} = \frac{A_2r_{00}+B_2r_{01}+C_2r_{02}}{A'_2}$$

$$\frac{D_3+t_x+t_y+t_z}{D'_3} = \frac{A_3r_{00}+B_3r_{01}+C_3r_{02}}{A'_3}$$

which determine the translation matrix

$$T = [t_x \quad t_y \quad t_z]$$

Therefore, we obtain the transformation from the corresponding matching surface to a hypothesis.

7.6 Verification of the Hypothesis

The task of robot recognition and localization is to find a matching between surface segments in an image and surface patches on models and the transformation from the model original position to the object position.

A hypothesis is obtained locally by analyzing some neighboring surface segments in the image. The way of obtaining hypotheses is efficient and robust to occlusion, especially when several objects are presented in a scene and each object is partially visible. However, local analysis does not guarantee a valid matching.

Given a hypothesis of

$$s_i \in TRANSFORM(m)$$

where $s_i \in s' \subseteq s = \{s_1, s_2, \ldots, s_t\}$, which are the surface segments in the image, and $m \in$ model set M, we build an image P from $TRANSFORM(m)$ and then compare P with the input image O to make decisions based on the correlation between P and O.

Suppose that each object in the scene has at least γ percent area visible, for instance 50%. We compare the image points on P with values in corresponding positions on O. If more than γ percent image points on P can match with input image O within a noise estimate δ, for instance 5%, we can believe that the hypothesis is true and then erase the matching region of P from O such that O does not contain the recognized and localized object for the next iteration.

To obtain hypothesis image P, we simply use the z-buffer technique of computer

graphics [32]. This technique is also called the depth-buffer image space algorithm. The algorithm is simple to implement. The performance of the algorithm tends to be constant since, on the average, the number of pixels covered by each surface decreases as the number of surfaces in the view volume increases [32]. The algorithm works in the following way.

The hypothesis image P is initialized to the largest representable z value. For each point (x, y) inside *TRANSFORM* (m)

1. Compute the depth $z(x,y)$ at (x,y).
2. If $z(x,y)$ is less than the image P value at (x,y), then place $z(x,y)$ into the image P at (x,y).

The algorithm records the smallest z encountered for each (x,y). The order of surfaces of the model has no effect on the resulting image. The hidden surfaces have larger z values which are replaced by visible surfaces' z values.

Chapter 8: Experimentation

We have implemented the approach as a robot vision system *Free-form Object Recognition and Localization (FORL)* and tested using synthesized range images. *FORL* is implemented in C on Sun-3/160C Workstation. A block diagram of our system is shown in Figure 8.1.

The modules in the dotted box in Figure 8.1 correspond to the main algorithm we introduced in Section 7.2. The vision process shown in the dotted box uses the information in the knowledge base about surface shapes and the CAD database to recognize and localize 3-D free-form objects. Input to the vision process is the range images. Output from the vision system is the identification of objects and their transformations with respect to the ideal positions of the CAD models.

The module *Find a segment in the image* performs the image segmentation, as we discussed in Section 7.3. The module *Find possible model surfaces to match with the segment* executes the surface matching evaluation procedure we described in Section 7.4. The *Hypothesis formation* module provides a hypothesis for verification using the technique we introduced in Section 7.5. The *Verification* module produces a synthetic image from the hypothesis and verifies it, as we showed in Section 7.6.

The CAD database is the source of the object models. The module *surface shape information abstraction* performs the process we described in Chapter 6. The module abstracts the surface shape information from the CAD models and then stores the information in the knowledge base in the representation scheme we introduced in

Section 6.3.

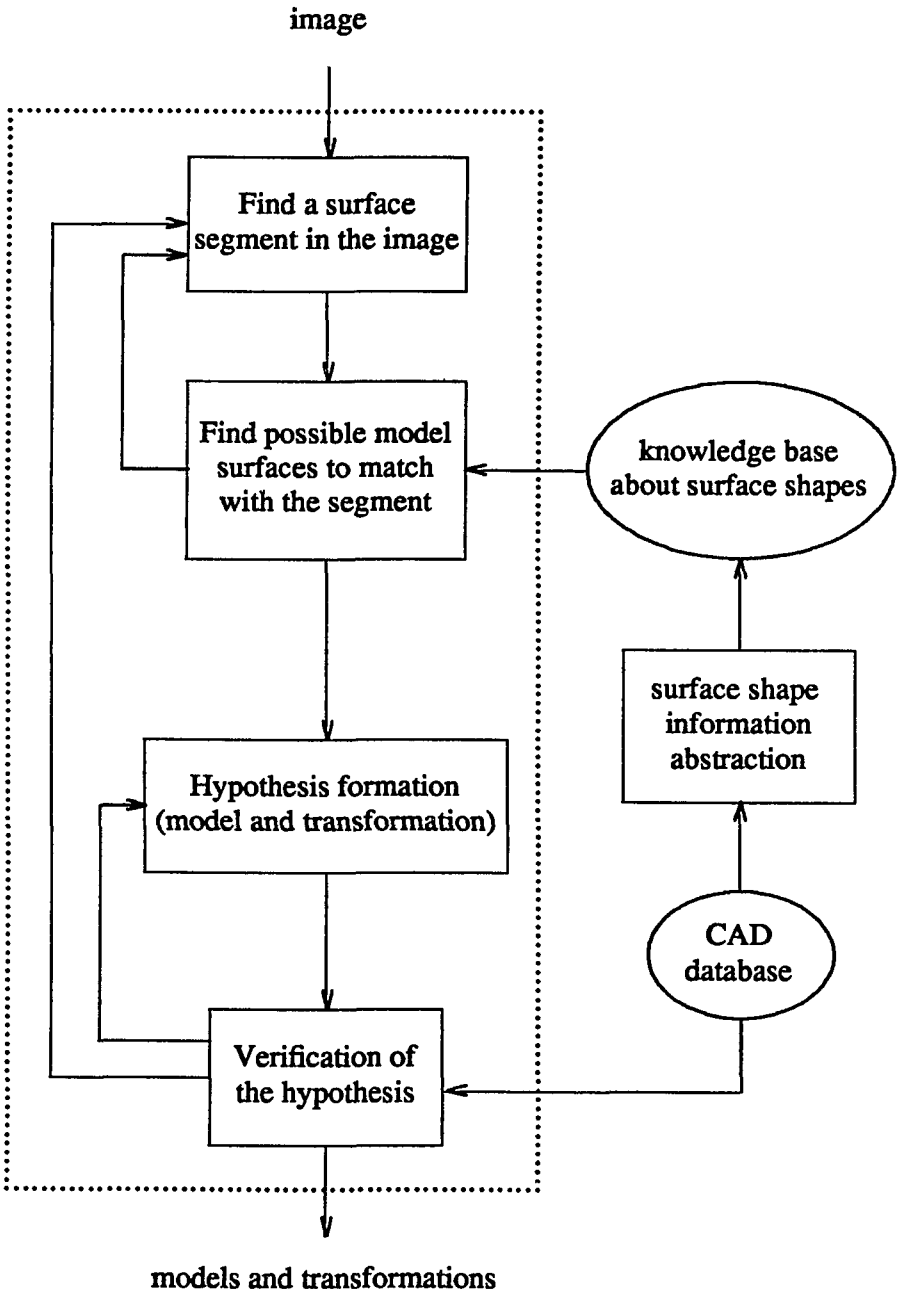

Figure 8.1: *Flow Diagram of FORL*

The following is one of the examples of recognizing and localizing multiple 3-D free-form objects in *FORL*. Figures 8.2 and 8.3 show the objects in the CAD database. We arbitrarily selected three objects and threw them in the 3-D space. Figure 8.4 is the front view of the three objects to be recognized and localized. The view at the viewpoint 25 degrees from -z axis to x axis and 25 degrees from -z axis to y axis is shown in Figure 8.5. Figure 8.6 shows the contour of the range image.

Figure 8.7 shows that one object is recognized and localized. When the first recognized object is erased, the contour of the image is shown in Figure 8.8. Figure 8.9 shows the next object is recognized and localized. Again, the contour of the image is shown in Figure 8.10 when the second object is erased. Figure 8.11 shows the recognition and localization of the last object.

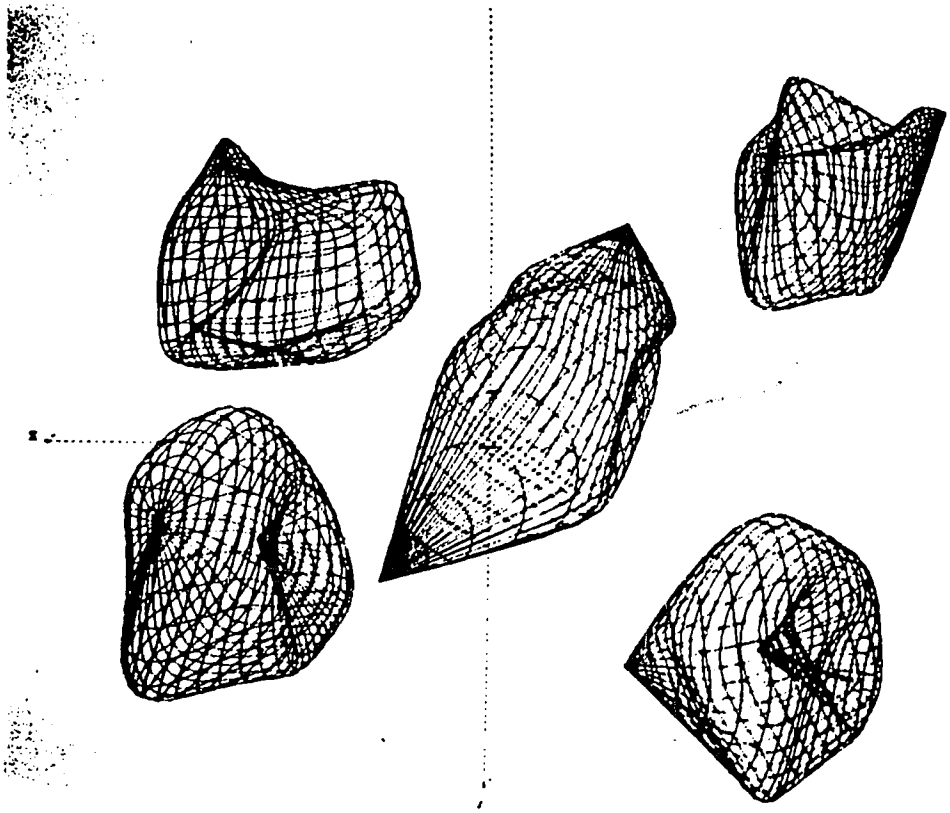

Figure 8.2: *Objects in the CAD database*

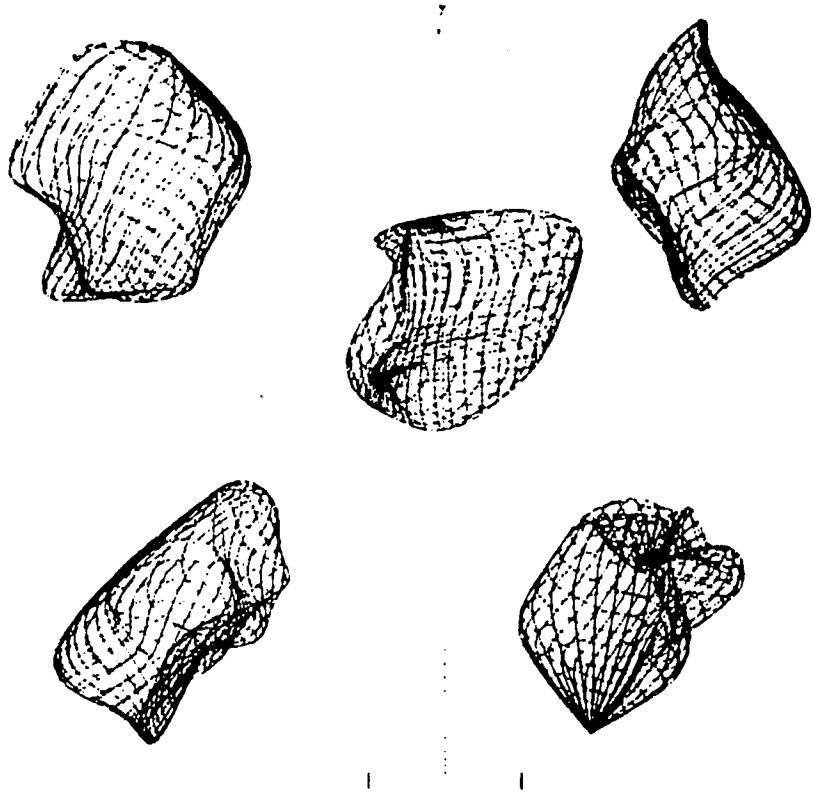

Figure 8.3: *Objects in the CAD database*

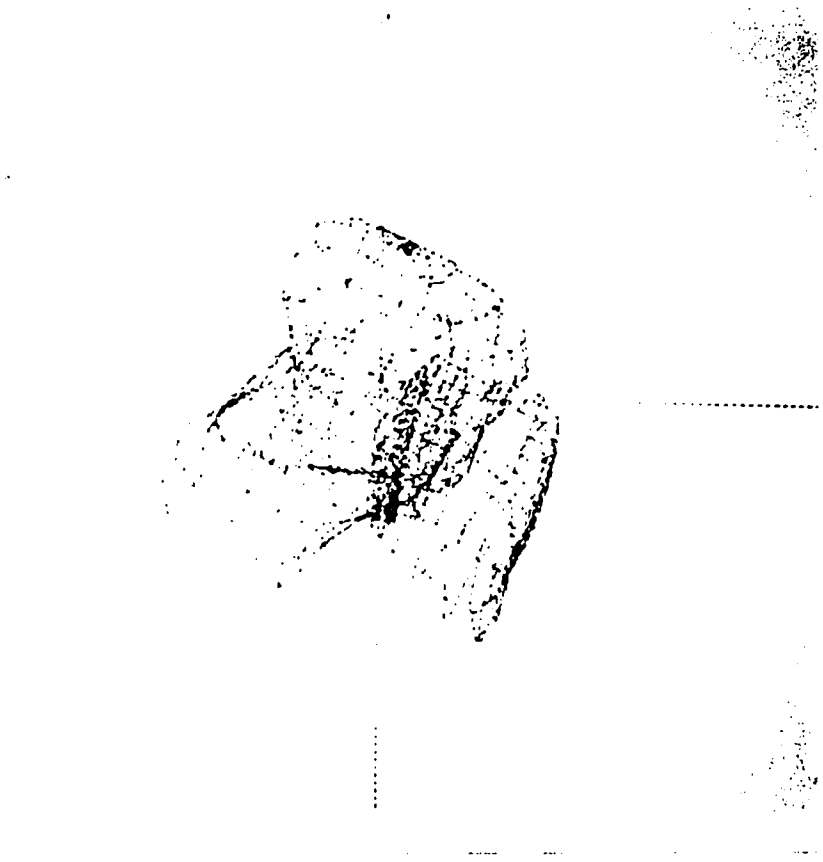

Figure 8.4: *Front View of the Objects to Be Recognized*

Figure 8.5: *The Three Objects to Be Recognized and Localized*

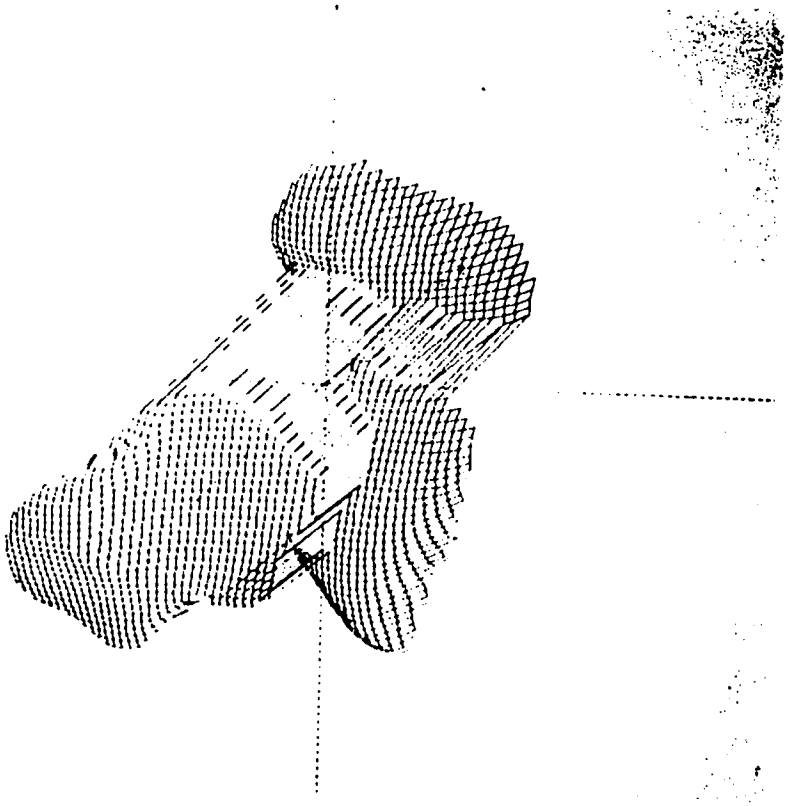

Figure 8.6: *Contour of the Range Image*

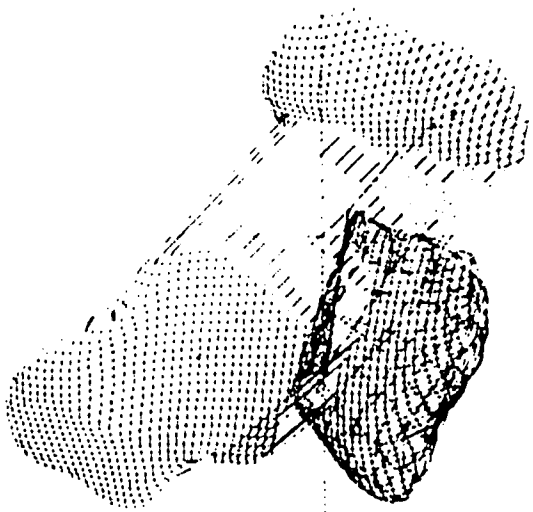

Figure 8.7: *One Object is Recognized and Localized*

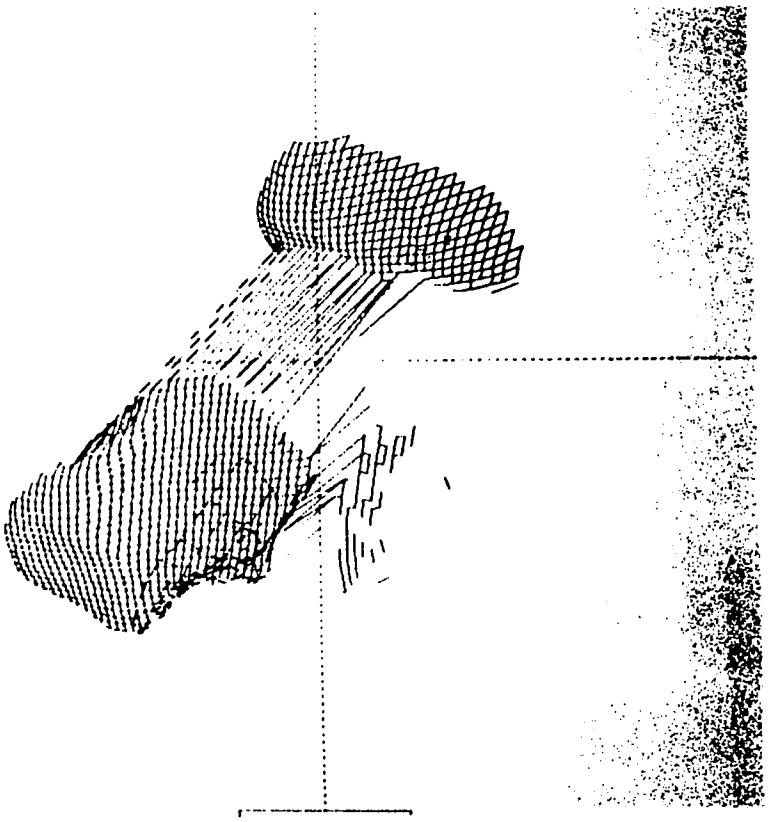

Figure 8.8: *Recognized and Localized Object is Erased*

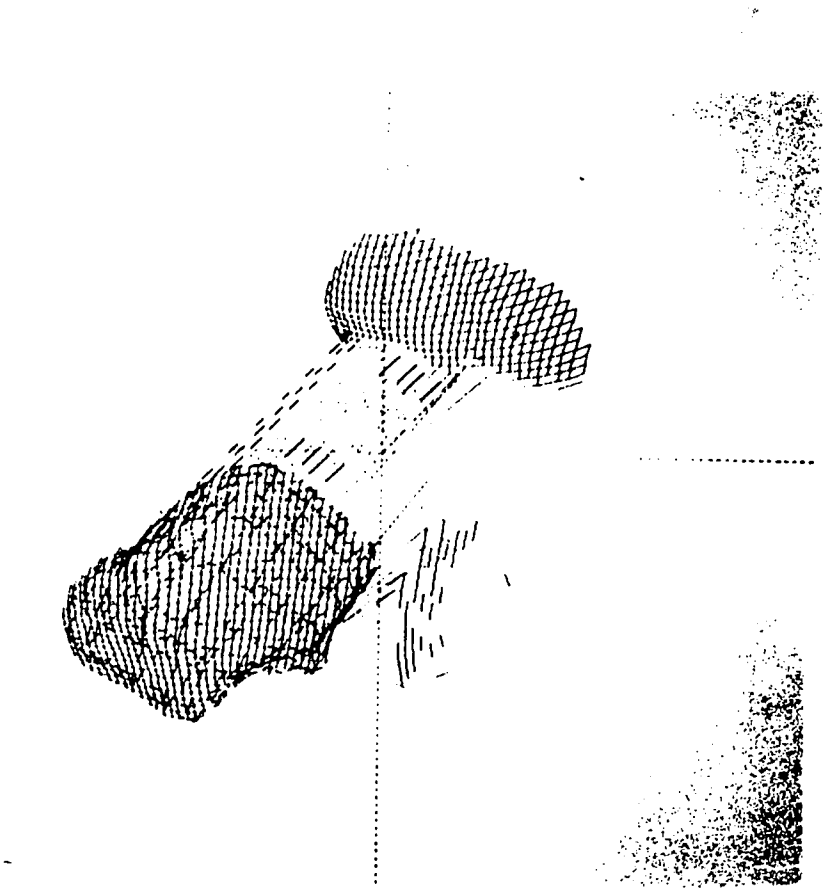

Figure 8.9: *Second Object is Recognized and Localized*

Figure 8.10: *Second Recognized and Localized Object is Erased*

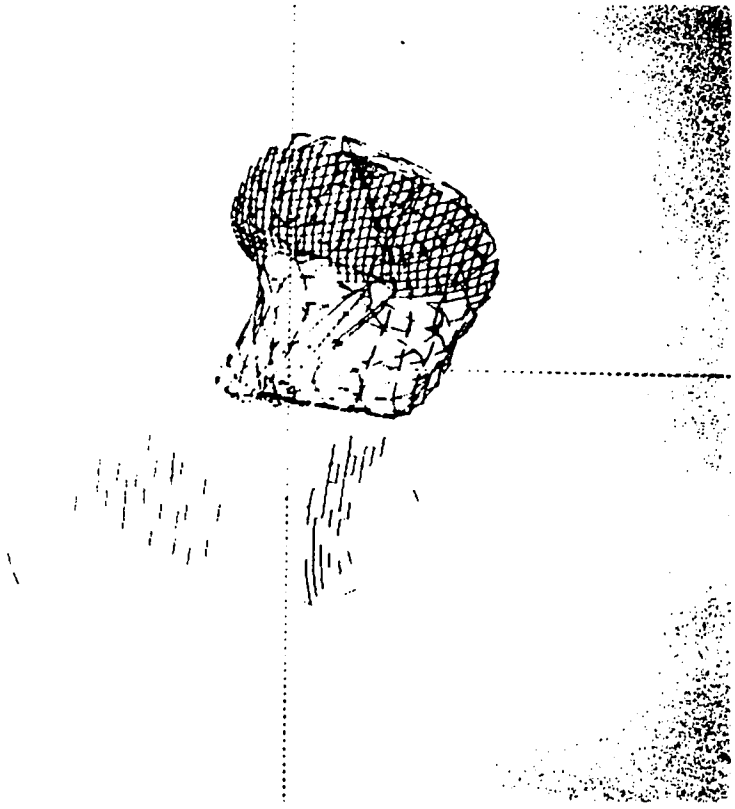

Figure 8.11: *Last Object is Recognized and Localized*

Chapter 9: Conclusions

Applications envisaged for autonomous robots which must operate in unstructured environments involve achieving operational responses in hard real-time. Fast, real-time response is especially important for robot vision. Typically, the vision problem involves the integration of recognition and localization of range data projected from 3-D objects. Yet, recovering 3-D information from range data is a challenging task. Towards this objective, we present a solution to an important problem of robot vision. The focus of our work is to develop an efficient technique to recognize and localize 3-D objects using the geometric models of objects.

We have designed an effective method of surface characterization of 3-D objects using surface curvature properties and developed an efficient approach to recognizing and localizing multiple 3-D free-form objects in range images. The approach is implemented as a robot vision system *Free-form Object Recognition and Localization (FORL)* and tested using synthesized range images. *FORL* performs satisfactorily to our expectation. Experimentation on real images is being undertaken.

The novelty of our method is in the definition of new high performance data structures for characterizing the surfaces of objects. The approach presented here is very unique and gives superior performance when compared to previous approaches. Prior work on recognition of free-form objects uses small planar or quadric patches to obtain the approximation of object models. The approximation methods certainly lose some information in many cases, and therefore cannot recognize some classes of

objects.

9.1 Contributions of this Work

The approach uses boundary representation for object models, which is one of the most popular CAD model representations. Since CAD systems are popular and provide a user-friendly environment for design, they are natural sources for solid models in robot vision. Furthermore, CAD models describe solids in detail so that recognition and localization of complex objects become possible. Previous partial models are not adequate for high requirement sensing tasks.

The most common CAD database description of a surface of a part is in terms of discrete points associated with interpolation parametric functions such as quadrics, composite Besier patches, or B-splines. Previously, these descriptions were not found to be useful in vision tasks. Different primitive concepts of different CAD systems make the automatic transfer from CAD model descriptions to descriptions for robot vision a very difficult problem. The desired information for robot vision may not be explicitly represented though CAD models provide details about objects.

Therefore, we have developed a knowledge representation scheme for describing free-form surface shapes. A representation of knowledge is a combination of data structures and interpretive procedures which use the data structures. We designed classes of data structures for storing information in computer systems and developed procedures which intelligently manipulate these data structures to make inferences.

The data structures and the procedures are well designed so that the knowledge leads the system to intelligent behavior, i.e., recognizing and localizing 3-D objects.

Our approach automatically abstracts knowledge about model surface shapes from CAD models for directing the search during vision process and uses CAD models directly in verification of the vision hypotheses. The approach makes use of the CAD models intelligently, and thus opens a door to natural sources of object models in robot vision.

The knowledge representation we developed eases the processes of knowledge acquisition, information retrieval, modification of knowledge base, and reasoning for a solution. The knowledge base used in our approach consists of a CAD database and surface shape information abstracted from CAD models automatically.

Acquiring new knowledge is straightforward to our system, which is simply adding a new model to the CAD database. Modifying the knowledge base is also made simple by modifying the CAD models. Retrieving information from the knowledge base is efficient, since the knowledge about surface shapes is organized as sorted arrays, which enables binary search to be performed. Reasoning in search of a solution using the knowledge base is effective and efficient.

Localization is a by-product of the recognition process. Since localization is necessary in robotic applications, this by-product is significant. In fact, the recognition process is to recognize objects by hypothesizing and locating objects. The approach uses the knowledge about the surface shapes to make the hypotheses and

uses the CAD models to locate the objects.

Model representation has a significant effect on model-based recognition. Without using surface properties many important industrial vision tasks will remain beyond the competence of machine vision. Numerical features about lines and topological features about connectivity suffer from partial occlusion. Geometric features such as equations of curves and surfaces are much more stable.

One of the most important problems in 3-D machine vision is the recognition of 3-D objects from their partial view. Objects may be occluded by other objects or even by themselves. However, if models of objects are given, occluded parts are supposed to be able to be inferred, which is factually a very difficult task for the previous methods that we have seen in the literature, since occlusion makes many feature relations invisible.

9.2 Advantages of our Approach

Our approach is surface-based, which is not sensitive to noise and occlusion. Surface-based recognition and localization is applicable in dirty and uncontrolled environments.

Our approach is capable of recognizing multiple objects since it is surface-based, which is not sensitive to occlusion, and forms hypotheses by local analysis of surface shapes, which does not depend on the visibility of the complete objects.

Our approach uses appropriate strategies for recognition and localization following the information from the CAD database. This makes the integration of robot vision systems with CAD/CAM systems a promising future.

9.3 Future Work

Future research should be on combining surface-based recognition systems and edge-based recognition systems since, in some situations, edges are visible and edge-based constraints are obtainable and efficient. Especially in some situations where objects have regular surfaces, such as cylinders, edge-based systems probably would be faster than surface-based systems. This should be tested. Surface-based recognition is typically useful and efficient to recognize irregularly shaped, free-form solids.

Our approach is very suitable for parallel processing to increase efficiency. Our work (Wang and Iyengar) [92,93] on parallel processing of pattern recognition tasks is an excellent candidate. Developing parallel algorithms for our approach to 3-D vision is another future research direction.

Bibliography

[1] I. Aleksander (ed.), *Artificial Vision for Robots*, Chapman & Hall, 1983.

[2] Peter K. Allen, *Robotic Object Recognition Using Vision and Touch*, Kluwer Academic Publishers, 1987.

[3] Russel L. Andersson, *A Robot Ping-Pong Player: Experiment in Real-Time Intelligent Control*, The MIT Press, 1988.

[4] D. H. Ballard and C. M. Brown, *Computer Vision*, Prentice-Hall, Inc., 1982.

[5] R. H. Bartels and J. J. Jezioranski, "Least-Squares Fitting Using Orthogonal Multinomials," *ACM Trans. Math. Software*, Vol. 11, No. 3, pp. 201-217, 1985.

[6] R. H. Bartels, J. C. Beatty and B. A. Barsky, *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*, Morgan Kaufmann Publishers, Inc., 1987.

[7] Paul J. Besl and Ramesh C. Jain, "Segmentation Through Variable-Order Surface Fitting," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 2, pp. 167-192, 1988.

[8] Paul J. Besl, "Active, Optical Range Imaging Sensors," *Machine Vision and Applications*, 1: 127-152, 1988.

[9] P. J. Besl and Ramesh C. Jain, "Three dimensional object recognition," *ACM Computing Surveys*, Vol. 17, No. 1, pp. 75-145, 1985.

[10] Bir Bhanu and L. A. Nuttall, "Recognition of 3-D Objects in Range Images Using a Butterfly Multiprocessor," *Pattern Recognition*, Vol. 22, No. 1, pp. 49-64, 1989.

[11] Bir Bhanu and Chih-Cheng Ho, "CAD-Based 3D Object Representation for Robot Vision," *Computer*, pp. 19-35, Aug. 1987.

[12] I. Biederman, "Matching Image Edges to Object Memory," *IEEE First Int. Conf. on Computer Vision*, pp. 384-392, 1987.

[13] R. M. Bolle and D. B. Cooper, "On Optimally Combining Pieces of Information with Application to Estimating 3-D Complex-Object Position from Range Data," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 5, pp. 619-638, 1986.

[14] R. C. Bolles and P. Horaud, "3DPO: A Three-Dimensional Part Orientation System," in *Three-Dimensional Machine Vision* (Takeo Kanade, ed.), 1987.

[15] I. N. Bronshtein and K. A. Semendyayev, *Handbook of Mathematics*, Verlag Harri Deutsch, 1985.

[16] R. A. Brooks, *Model-Based Computer Vision*, UMI Research Press, 1984.

[17] Christopher Brown (ed.), *Advances in Computer Vision*, v.1, Lawrence Erlbaum Associates, Inc., 1988.

[18] V. Cappellini (ed.), *Time-Varying Image Processing and Moving Object Recognition*, North-Holland, 1987.

[19] V. Cappellini and R. Marconi (eds.), *Advances in Image Processing and Pattern Recognition,* North-Holland, 1986.

[20] C. H. Chien, Y. B. Sim and J. K. Aggarwal, "Generation of Volume/Surface Octree from Range Data," *IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 254-260, 1988.

[21] Roland T. Chin and Charles R. Dyer, "Model-Based Recognition in Robot Vision," *Computing Surveys*, Vol. 18, No. 1, pp. 67-108, 1986.

[22] F. S. Cohan and D. B. Cooper, "A Decision Theoretic Approach for 3-D Vision," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 964-972, 1988.

[23] F. S. Cohen and R. D. Rimey, "A Maximum Likelihood Approach to Segmenting Range Data," *IEEE Int. Conf. on Robotics and Automation*, pp. 1696-1701, 1988.

[24] C. I. Connolly and J. R. Stenstrom, "Construction of Polyhedral Models from Multiple Range Views," *IEEE Int. Conf. on Pattern Recognition,* pp. 85-87, 1986.

[25] Bruce R. Dewey, *Computer Graphics for Engineers,* Harper & Row, Publishers, Inc., 1988.

[26] M. J. B. Duff (ed.), *Intermediate-Level Image Processing,* Academic Press, 1986.

[27] Gil J. Ettinger, "Large Hierarchical Object Recognition Using Libraries of Parameterized Model Sub-parts," *IEEE Conf. on Computer Vision and Pattern Recognition,* pp. 32-41, 1988.

[28] O. D. Faugeras and M. Hebert, "The Representation, Recognition, and Locating of 3-D Objects," *The International Journal of Robotics Research,* Vol. 5, No. 3, 1986.

[29] I. D. Faux and M. J. Pratt, *Computational Geometry for Design and Manufacture*, Ellis Horwood, Chiester, 1979.

[30] J. A. Feldman, "Connectionist Models and Parallelism in High Level Vision," *Computer Vision, Graphics, and Image Processing,* Vol. 31, pp. 178-200, 1985.

[31] P. J. Flynn and A. K. Jain, "Surface Classification: Hypothesis Testing and Parameter Estimation," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 261-267, 1988.

[32] J. D. Foley and A. Van Dam, *Fundamentals of Interactive Computer Graphics*, Addison-Wesley Pub. Co., 1984.

[33] G. H. Golub and C. F. Van Loan, *Matrix Computation*, Johns Hopkins University Press, 1983.

[34] S. J. Gordon and W. P. Seering, "Real-Time Part Position Sensing," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 10, No. 3, pp. 374-386, 1988.

[35] W. E. L. Grimson and T. Lozano-Perez, "Recognition and Localization of Overlapping Parts from Sparse Data," in *Three-Dimensional Machine Vision* (Takeo Kanade, ed.), 1987.

[36] W. I. Grosky and Ramesh Jain, "A Pyramid-Based Approach to Segmentation Applied to Region Matching," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 5, pp. 639-650, 1986.

[37] Heinrich W. Guggenheimer, *Differential Geometry*, McGraw-Hill, 1963.

[38] K. T. Gunnarsson and F. B. Prinz, "CAD Model-Based Localization of Parts in Manufacturing," *Computer*, pp. 66-74, Aug. 1987.

[39] J. H. Han and R. A. Volz, "Region Grouping from a Range Image," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 241-248, 1988.

[40] M. Herman, "Generating Detailed Scene Descriptions from Range Images," *Proc. Int. Conf. Robotics and Automation*, pp. 426-431, 1985.

[41] D. Hillis, *The Connection Machine*, The MIT Press, 1985.

[42] R. Hoffman and A. K. Jain, "Learning Rules for 3D Object Recognition," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 885-892, 1988.

[43] R. Hoffman and A. K. Jain, "Segmentation and Classification of Range Images," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 9, No. 5, pp. 608-620, 1987.

[44] Berthold Klaus Paul Horn, *Robot Vision*, The MIT Press, 1986.

[45] Chuan-Chih Hsiung, *A First Course in Differential Geometry*, John Wiley & Sons, Inc., 1981.

[46] S. Inokuchi, K. Sato and F. Matsuda, "Range Imaging System for 3-D Object Recognition," *Proc. 7th Int. Conf. Pattern Recognition*, pp. 806-808, 1984.

[47] R. A. Jarvis, "A Perspective on Range Finding Techniques for Computer Vision," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 5, No. 2, pp. 122-139, 1983.

[48] Takeo Kanade (ed.), *Three-Dimensional Machine Vision,* Kluwer Academic Publishers, 1987.

[49] V. A. Kovalevsky, *Image Pattern Recognition,* Springer-Verlag, 1980.

[50] D. T. Kuan and R. J. Drazovich, "Model-Based Interpretation of Range Imagery," *Proc. Nat. Conf. Artificial Intelligence,* pp. 210-215, 1984.

[51] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems,* Prentice-Hall, 1974.

[52] Martin M. Lipschutz, *Differential Geometry,* McGraw-Hill, 1969.

[53] W. Liu, T. Yeh, W. E. Batchelor and R. Cavin, "Bit Level Concurrency in Real-Time Geometric Feature Extractions," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 957-962, 1988.

[54] M. J. Magee and J. K. Aggarwal, "Using Multisensory Images to Derive the Structure of Three-Dimensional Objects - A Review," *Computer Vision, Graphics, and Image Processing,* Vol. 32, pp. 145-157, 1985.

[55] Nadia Magnenat-Thalmann and Daniel Thalmann, *Image Synthesis,* Springer-Verlag, 1987.

[56] David Marr, *Vision,* W. H. Freeman and Company, 1982.

[57] W. N. Martin and J. K. Aggarwal (eds.), *Motion Understanding: Robot and Human Vision,* Kluwer Academic Publishers, 1988.

[58] W. T. Miller, "Sensor-Based Control of Robotic Manipulators Using a General Learning Algorithm," *IEEE Journal of Robotics and Automation,* Vol. RA-3, No. 2, pp. 157-165, 1987.

[59] Michael E. Mortenson, *Geometric Modeling*, Wiley, 1985.

[60] S. M. Naik and R. C. Jain, "Spline-Based Surface Fitting on Range Images for CAD Applications," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 249-253, 1988.

[61] V. S. Nalwa and T. O. Binford, "On Detecting Edges," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 8, No. 6, pp. 699-714, 1986.

[62] David Nitzan, "Three-Dimensional Vision Structure for Robot Applications," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 10, No. 3, pp. 291-309, 1988.

[63] M. Oshima and Y. Shirai, "Object Recognition Using Three-Dimensional Information," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 5, No. 4, pp. 353-361, 1983.

[64] H. D. Park and O. R. Mitchell, "CAD Based Planning and Execution of Inspection," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 858-863, 1988.

[65] Theo Pavlidis, *Algorithms for Graphics and Image Processing*, Computer Science Press, Inc., 1982.

[66] Theo Pavlidis, "A Critical Survey of Image Analysis Methods," *IEEE Int. Conf. on Pattern Recognition*, pp. 502-511, 1986.

[67] Judea Pearl, *Heuristics: Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley, 1984.

[68] Alex P. Pentland (ed.), *From Pixels to Predicates: Recent Advances in Computational and Robotic Vision*, Ablex Publishing Corporation, 1986.

[69] J. Ponce and M. Brady, "Toward a Surface Primal Sketch," in *Three-Dimensional Machine Vision* (Takeo Kanade, ed.), 1987.

[70] M. J. D. Powell, *Approximation Theory and Methods*, Cambridge University Press, 1981.

[71] L. G. Roberts, "Machine Perception of Three-Dimensional Solids," in *Optical and Electro-Optical Information Processing*, (J. P. Tippet et al. eds.), The MIT Press, 1965.

[72] Joe Rooney, *Principles of Computer-Aided Design*, Pitman, 1987.

[73] A. Rosenfeld (ed.), *Techniques for 3-D Machine Perception*, North-Holland, 1986.

[74] Larry L. Schumaker, *Spline Functions: Basic Theory*, John Wiley & Sons, 1981.

[75] Jacob T. Schwartz and Chee-Keng Yap (eds.), *Advances in Robotics: Algorithmic and Geometric Aspects of Robotics*, Lawrence Erlbaum Associates, Inc., 1987.

[76] T. W. Sederberg, D. C. Anderson and R. N. Goldman, "Implicit Representation of Parametric Curves and Surfaces," *Computer Vision, Graphics, and Image Processing*, Vol. 28, pp. 72-84, 1984.

[77] Yoshiaki Shirai, *Three-Dimensional Computer Vision*, Springer-Verlag, 1987.

[78] D. R. Smith and T. Kanade, "Autonomous Scene Description with Range Imagery," *Computer Vision, Graphics, and Image Processing*, Vol. 31, pp. 322-334, 1985.

[79] John Stark, *Managing CAD/CAM: Implementation, Organization, and Integration* , McGraw-Hill Book Company, 1988.

[80] Andrew C. Staugaard, Jr., *Robotics and AI: An Introduction to Applied Machine Intelligence*, Prentice Hall, Inc., 1987.

[81] T. M. Strat and M. A. Fischler, "One-Eyed Stereo: A General Approach to Modeling 3-D Scene Geometry," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, pp. 730-741, 1986.

[82] K. Sugihara, "Use of Vertex-Type Knowledge for Range Data Analysis," in *Three-Dimensional Machine Vision* (Takeo Kanade, ed.), 1987.

[83] Tech Tran Consultants, Inc., *Machine Vision Systems: A Summary and Forecast,* Tech Tran Corporation, 1985.

[84] C. Thorpe, M. H. Hebert, T. Kanade and S. A. Shafer, "Vision and Navigation for the Carnegie-Mellon Navlab," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. No. 3, pp. 362-373, 1988.

[85] F. Tomita and T. Kanade, "A 3D Vision System: Generating and Matching Shape Descriptions in Range Images," *Proc. Int. Conf. Robotics,* IEEE Comput. Soc., pp. 186-191, 1984.

[86] L. W. Tucker, C. R. Feynman and D. M. Fritzsche, "Object Recognition Using the Connection Machine," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 871-878, 1988.

[87] M. A. Turk, D. G. Morgenthaler, K. D. Gremban and M. Marra, "VITS - A Vision System for Autonomous Land Vehicle Navigation," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 10, No. 3, pp. 342-361, 1988.

[88] Shimon Ullman and Whitman Richards (ed.), *Image Understanding 1984*, Ablex Publishing Corporation, 1984.

[89] Izu Vaisman, *A First Course in Differential Geometry*, Marcel Dekker, Inc., 1984.

[90] B. C. Vemuri, A. Mitiche and J. K. Aggarwal, "Curvature-Based Representation of Objects from Range Data," *Image and Vision Computing*, Vol. 4, No. 2, pp. 107-114, 1986.

[91] B. C. Vemuri and J. K. Aggarwal, "Localization of Objects from Range Data," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 893-898, 1988.

[92] Wu Wang, S. Sitharama Iyengar and L. M. Patnaik, "Memory-Based Reasoning Approach for Pattern Recognition of Binary Images," *Pattern Recognition*, Vol. 22, No. 5, pp. 505-518, 1989.

[93] Wu Wang, S. Sitharama Iyengar and Jianhua Chen, "Massively Parallel Approach to Pattern Recognition," to appear in *the Proceedings of the Ninth IEEE International Phoenix Conference on Computers and Communications.*

[94] Wu Wang and S. Sitharama Iyengar, "Three-Dimensional Free-form Object Recognition and Localization from Range Images," in preparation.

[95] Wu Wang and S. Sitharama Iyengar, "Geometric Modeling of Free-form Solids for Robot Vision," in preparation.

[96] Andrew K. C. Wong, Si W. Lu and M. Rioux, "Recognition and Shape Synthesis of 3-D Objects Based on Attributed Hypergraphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 3, pp. 279-290, 1989.

[97] N. Yokoya and M. D. Levine, "Range Image Segmentation Based on Differential Geometry: A Hybrid Approach," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 11, No. 6, 1989, pp. 643-649.

[98] Tzay Y. Young and King-Sun Fu (eds.), *Handbook of Pattern Recognition and Image Processing*, Academic Press, Inc., 1986.

[99] Nello Zuech and Richard K. Miller, *Machine Vision*, Prentice-Hall, Inc., 1987.

Vita

Wu Wang received his B.S. degree in computer science from Zhongshan University, Canton, China in 1984. In August, 1987, he joined the Department of Computer Science at Louisiana State University, Baton Rouge, Louisiana for his Ph.D. study. He performed research on robot planning, speech recognition, natural language processing, robot vision, and machine learning. In industry, he has worked on development of medical instrument systems and real-time network control systems for some years. His technical papers have appeared in computer science journals and conferences.

DOCTORAL EXAMINATION AND DISSERTATION REPORT

Candidate: Wu Wang

Major Field: Computer Science

Title of Dissertation:

Model-Based Three-Dimensional Object Recognition and Localization Using Properties of Surface Curvatures

Approved:

Major Professor and Chairman

Dean of the Graduate School

EXAMINING COMMITTEE:

Date of Examination:

November 10, 1989