AUTOMATED TACTILE SENSING FOR OBJECT RECOGNITION AND LOCALIZATION

by

JOHN LEWIS SCHNEITER

B.S. Mech. Eng., University of Connecticut
(1978)

M.S. Mech. Eng., Massachusetts Institute of Technology
(1982)

SUBMITTED TO THE DEPARTMENT OF
MECHANICAL ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF SCIENCE IN MECHANICAL ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June, 1986

Signature of Author.......[illegible]...................
Department of Mechanical Engineering
April, 1986

Certified by.......................[illegible]........
Professor Thomas B. Sheridan
Thesis Supervisor

Accepted by................ [illegible].............
Professor Ain A. Sonin
Chairman, Department Committee

AUTOMATED TACTILE SENSING FOR OBJECT RECOGNITION AND LOCALIZATION

by

JOHN L. SCHNEITER

Submitted to the Department of Mechanical Engineering on April 28, 1986 in partial fulfillment of the requirements for the Degree of Doctor of Science in Mechanical Engineering

ABSTRACT

Automated manipulation systems operating in unstructured environments, such as undersea or in space, will be required to determine the identity, location and orientation of the various objects to be manipulated. Vision systems alone are inadequate for the successful completion of some of these recognition tasks, especially when performed where vision is partially or totally occluded. Tactile sensing is useful in such situations, and while much progress has been made in tactile hardware development, the problem of using and planning to obtain tactile information has received insufficient attention.

This work addresses the planning problem associated with tactile exploration for object recognition and localization. Given that an object has been sensed and is one of a number of modeled objects, and given that the data obtained so far is insufficient for recognition and/or localization, the methods developed in this work enumerate the paths along which the sensor should be directed in order to obtain further highly diagnostic tactile measurements. Three families of sensor paths are found. The first is the family of paths for which recognition and localization is guaranteed to be complete after the measurement. The second includes paths for which such distinguishing measurements are not guaranteed, but for which it is guaranteed that something will be learned. The third includes paths for which nothing will be learned, and thus are to be avoided.

The methods are based on a small but powerful set of geometric ideas and are developed for two dimensional, planar-faced objects. The methods are conceptually easily generalized to handle general three dimensional objects, including objects with through-holes. A hardware demonstration was developed using thick "2-D" objects, and it is shown that the strategy greatly reduces the number of required measurements when compared to a random strategy. It is further shown that the methods degrade gracefully with increasing measurement error.

Acknowledgements

The first half of this research was supported by the M.I.T. Sea Grant College Program (National Oceanographic and Atmospheric Administration), and the last half by the Ames Research Center, National Aeronautics and Space Agency, under Grant No. NCC 2-360.

I thank my thesis supervisor, Professor Thomas B. Sheridan, for providing me with the opportunity to grow, for providing support and encouragement, and for being a good friend.

I also like to thank my committee members, Professor Tomas Lozano-Perez of the M.I.T. Artificial Intelligence Laboratory, whose own work provided the springboard for this work, and whose comments and interest have been very valuable, and Professor Steven Dubowsky, for his support, comments, and valuable suggestions.

I would like to thank all the members of the Man/Machine Systems Laboratory, past and present, who have made this experience stimulating and enjoyable. Special thanks go to Jugy, Jim, Max and Leon, for the many conversations, both professional and personal, that have helped me in many ways.

A special word of thanks goes to Erik Vaaler, whose interest, support and friendship has been of great personal value.

Without my wife Andrea, I could not have gotten this far; her love, compassion and patience is a blessing.

Table of Contents

CHAPTER 1 Introduction

Automated manipulation systems operating in unstructured environments, such as undersea or in space, will be required to determine the identity, location and orientation of the various objects to be manipulated. It has been known for some time that vision systems alone are inadequate for the successful completion of some of these recognition tasks, especially when performed where vision is partially or totally occluded [1-4]. Tactile sensing is useful in such situations, and while much progress has been made in tactile hardware development [4,51-53], the problem of using and planning to obtain tactile information has received insufficient attention [2,3,5]. A few researchers in the engineering community have made attempts to develop sensing strategies [5,6] but most of the attention focused on so-called "active touch" has originated in the psychological community [7,8].

This work focuses on the planning problem associated with tactile exploration for object recognition and localization (the determination of object position and orientation). Given that an object has been sensed and is one of a number of modeled objects, and given that the data obtained so far is insufficient for recognition and/or localization, the methods developed in this work enumerate the paths along which the sensor should be directed in order to obtain further highly diagnostic tactile measurements.

Three families of sensor paths are found. The first is the family of paths for which recognition and localization is guaranteed to be complete after the measurement. The second includes paths for which such distinguishing measurements are not guaranteed, but for which it is guaranteed that something will be learned. The third family is made up of paths for which nothing will be learned, and thus such paths are to be avoided.

The recognition of an object and the determination of its position and orientation in space is a task domain that may be categorized into two classes. The first may be described as the domain of passive information gathering, in the sense that an object is presented to some suitable tactile sensor and as much information is extracted from the sensor output as is possible. An example of this is when an object is dropped onto a tactile array and the object's "footprint" is analyzed [50]. Given that most objects of interest have a finite number of stable poses on the plane and that the footprint is often unique for each pose of each object, an assessment may be made of an object's identity, location and orientation. These procedures are open-loop in the sense that feature information is extracted from the sensor "snapshot" and no attempt is made to actively pursue the gathering of more information. Such procedures are not addressed in this work because they are primarily useful only in reasonably structured environments.

The open-loop procedures contrast with the other class of recognition and localization problems in which information is actively sought by a tactile system. An example of the latter is where some suitably instrumented manipulator scans the surface of an object of interest, obtains tactile data, and performs additional planned data gathering based upon an analysis of the previously obtained data. The salient description of this (serial) process is:

1. obtain data
2. analyze the data
3. plan where to direct the sensor to obtain more data, if necessary
4. repeat as appropriate.

A good tactile scanning strategy should provide an evolving plan or schedule of sensor moves for a system to make in order to efficiently obtain tactile data of high diagnosticity. Such a plan bases the next sensor move on what has been learned from all previous measurements, including the last. As will become evident in the remainder of this thesis, a small but powerful set of geometric ideas is central to the development of such a strategy. Before proceeding directly to the development of these ideas, however, we will first explore the issues by way of a simple example, and then delve more deeply into the nature of tactile information, the notion of features, object representation and recognition, and the issues of real-world applications and hardware requirements.

The thesis is therefore structured as follows: The remainder of this chapter presents a simple example to motivate the problem and introduce some of the issues. Chapter Two provides a review of tactile work to date and explores some of the common object representation and recognition schemes (most of which were developed primarily for vision work) with critical attention paid to their suitability in the tactile domain. It is here that a representation and recognition scheme is selected and explained. Chapter Three develops a tactile scanning strategy in a two dimensional environment, assuming perfect touch sensing measurements. In Chapter Four the effects of measurement error are assessed in terms of their impact on system performance and software implementation. The generalization of the work to include three dimensional objects is discussed in Chapter Five.

A hardware demonstration system was developed that incorporated the ideas presented in this thesis. A description of the system, including the manipulator arm and tactile sensor, and an assessment of performance issues is provided in Chapter Six.

Conclusions and recommendations for further work are found in Chapter Seven.

1.2 A Simple Example

Let us assume that there is a stationary 2-dimensional object in the environment that we can obtain contact measurements from, and let us further assume that we know it is one of two objects (see figure 1.1) we are familiar with.

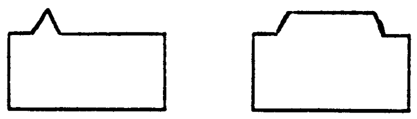


Figure 1.1. Two Simple Object Models

Our job is to determine which object model represents the real object and what the transformation between model and world coordinates is by reaching out and exploring the real object using touch. We are immediately faced with the following question: What is the nature of our measurements? If the objects are of different stiffness, we have only to grope until we contact the object and then simply press against it and monitor the force/displacement behavior to recognize the object. We would still be required to explore the object's surface in some (presumably) intelligent way to determine orientation. If the objects are stiff and made of the same material, then we are forced to rely exclusively on tactile surface exploration for both recognition and localization.

It is evident that, except possibly for the case in which the object is smaller than some sensor array (in which case the sensor might obtain a "snapshot"), tactile exploration of the object's surface will in general be necessary. With a contact-point sensor we have to obtain surface information from multiple contacts between the sensor and the object. If we have an array of sensitive elements, we can obtain local patches of surface data from which we might calculate surface properties such as the local surface normal and surface curvature. (This is nothing more than a parallel implementation of a single point contact sensor that provides the data in a more serial manner). In this way we can build up a sparse spatial tactile image of the object with a series of contacts.

Let us assume, then, that tactile data is comprised of contact points and measured or derived surface properties. We have to map the data onto the object models or, equivalently, fit the models to the data. If the mapping is unique the job is complete. In general, however, there will be multiple possible interpretations of the data. In such cases the data is insufficient to distinguish between the objects, or if it is sufficient to distinguish, we may still be unable to determine position and/or orientation. Figure 1.2 shows an example of contact data consisting of contact position (at the base of the arrows) and measured surface normal (represented by the arrows) which do not distinguish between the objects. The same data fits each object in only one way equally well.

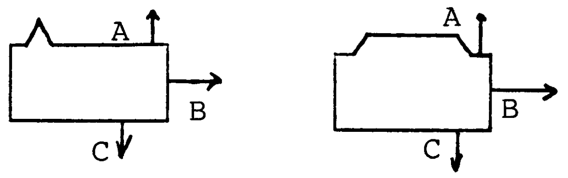


Figure 1.2. Non-Distinguishing Data

Figure 1.3 depicts the case where the data distinguishes between the objects but we are left with multiple orientations of the object.

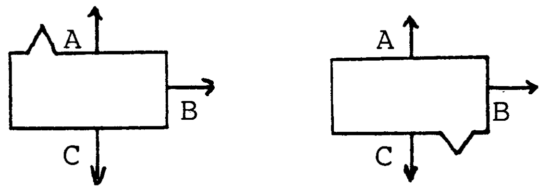


Figure 1.3. Non-Distinguishing Data

We must now determine what measurements to make next. This raises another question: Under what constraints do we operate? We have to know the relative costs of movements, measurements and time in order to respond to this question. If the cost of information processing (processing time, noise smoothing, etc.) is higher than the relative cost of moving the sensor (travel time, risky movements in an unknown environment, etc.) then it is appropriate to seek distinguishing features wherever they might be. If, conversely, information processing is relatively inexpensive and long range movements expensive, it might be more appropriate to explore a local surface. This can be very wasteful, however. For example, in Figure 1.2, local exploration of the surfaces at either B or C will yield no useful information. This lends support to the assertion that in general, a purposeful, active tactile sensing strategy should direct the sensor to the most distinguishing features available.

As an aside, we make the intuitive observation that in general, the more complex the objects, the more features there are available, hence the more likely a random strategy is to be powerful and successful. It is when the objects in a set are similar that we find we need a good strategy. Maximally different objects can possibly be distinguished on the basis of local surface analysis, whereas minimally different objects are more likely to require global or structural analysis.

1.2.1 Features

We have used the term feature without rigorously defining it. One definition (from Webster's) that is appropriate in our context is that a feature is a "specially prominent characteristic". For our purposes, the characteristic must be measurable or derivable from measurements. We therefore assume a feature to be a measurement (or a quantity derived from a measurement) that provides us with information. We notice immediately that this is dependent on the object set under consideration. For example, in Figure 1.4 the object set is comprised of objects A and B. Surface normal measurements from the triangular structure on object B, along with normal measurements from other parts of the object, inform us that the object cannot be object A.

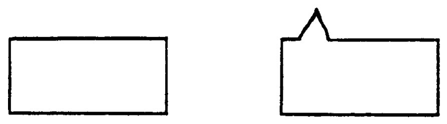


Figure 1.4. Object Set for which surface normals distinguish.

However, in figure 1.5, simply measuring surface normals is insufficient to discriminate between objects. Contact position must also be measured.

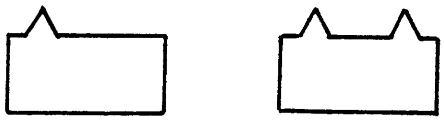


Figure 1.5. Object set for which surface normals and associated contact positions distinguish.

We see, then, that discriminating features depends upon the composition of the object set. If a scanning strategy is to be useful, it should automatically perform feature selection from among the object models and should require no more of us than correct models. We should not be required to select features a-priori (assuming that we are sophisticated and patient enough to do so) and we should be able to add objects to the object set and delete them at will.

If we are to automate active touch, we must have some way of representing objects in a computer in a way that is natural, efficient, and allows for fast processing. While these may be subjective notions, it is clear that techniques which require more computer memory than is reasonably available or routines that take days to run on standard, powerful equipment are to be avoided. The next chapter reviews many of the standard representation and recognition schemes in view of tactile sensing requirements, and describes the one chosen for this work.

CHAPTER 2 Recognition and Representation

Our ultimate objective is to produce a strategy for obtaining tactile data, but we must first discuss how we plan to represent objects and how we can recognize them using a computer. The purpose of this chapter, then, is to briefly review the previous work in tactile recognition and comment on why the various methods have proven unsatisfactory, to discuss various representation schemes that have been developed (historically, primarily for vision work), and to describe the representation and recognition methods chosen for this work. The review is rather brief because there are already a few very thorough reviews in the literature. The interested reader is referred to two reviews by Harmon [1,4], a review by Gaston and Lozano-Perez [35], and a review by Grimson and Lozano-Perez [27].

2.1 Previous Work in Recognition

There are two major alternative approaches to recognition in the tactile sensing domain: pattern recognition and description-building and matching. The basic notion in classical pattern recognition is the notion of a feature vector [9-13]. The available data is processed and a vector is created to represent the results of the processing. For example, the pressure pattern on a tactile array can be processed to determine the first, second, and higher moments of inertia of the pattern, and these moments can be used as the elements of a vector (the feature vector). The recognition process is performed by comparing the feature vector with previously computed vectors for different objects. Recognition is typically assumed complete when the feature vector matches a model vector fairly closely. The matching criterion is typically the Euclidean distance between the vectors. If the feature vector does not match any of the model vectors closely enough, then typically another feature is extracted from the data and the process is repeated.

Most of the previous work in tactile recognition using these ideas used either pressure patterns from two dimensional objects on sensor arrays [50,54] or the joint angles of the fingers that grasp the object [55,56] as the data. Some work combined the two approaches [57]. There are two major objections to these approaches. The first is that we can not expect a two-dimensional sensor array to provide enough data for recognition of complex three-dimensional shapes. Furthermore, the range of possible contact patterns that might arise in practice can be quite large and precomputation of them would be impractical. The second objection is that the range of possible graspings of an object can also be quite large, which effectively prevents precomputation of all possible finger positions and joint angles. In summary, tactile recognition based on classical pattern recognition is limited to simple objects, primarily because of the great cost in precomputing feature vectors for more complex objects. Another point, expressed in [27], is that the methods are limited because they do not exploit the rich geometric data available from complete object models.

A relatively recent branch of pattern recognition theory is called syntactic pattern recognition and is based on the observation that object shapes can in some sense be associated with an object "grammar" or rules of structure [10]. It has enjoyed some success in two dimensional vision work but has been relatively unsuccessful in 3-D recognition because the appropriate grammars are extremely difficult to devise for even fairly simple 3-D objects.

In description-based recognition methods, a partial description of an object is built up from sensor data and an attempt is made to match the partial description to an object model. Approaches have included building the description using multiple contacts of a pressure sensitive array [58] or from the displacement of the elements in a sensor comprised of long needles [59,60]. Although the description-based approach may be more general than the pattern recognition approach in its ability to handle complex 3-D shapes, it suffers from the requirement that a great deal of data must be obtained. Furthermore, there are few methods for actually matching the data to object models, and the methods are computationally expensive.

Most current researchers in the tactile field implicitly (and I feel correctly) assume that tactile sensors will be integral components of manipulation systems, and will be required to impart forces to objects as well as obtain surface information from them. This will require some moderate stiffness of the sensors, so we can not assume that we can ever obtain a great deal of dense surface data from an object with a single measurement, since that would require a very soft, easily deformable sensor that can be draped over a large part of an object. Tactile sensors useful in manipulation systems will, by their very nature, provide fairly sparse surface data. While we might obtain dense surface data from an object by an exhaustive tactile scan, it is clearly inefficient to do so, because intrinsic geometric constraints [27] associated with any object can be exploited to yield recognition and localization with only a few tactile measurements.

For this reason we assert that classical pattern recognition and description-based techniques that require dense surface data are of limited utility to the tactile problem. While the essential goal of of these methods, that of recognizing and localizing an object based on sensor measurements, is essentially what we wish to accomplish, the methods available are inappropriate in our problem context.

2.2 Previous Work in Representation

Automated recognition, or the matching of sensor data to object models, requires some representation structure that can be described mathematically or algorithmically and programmed into a computer. We need some way of representing objects that is natural and appropriate to the data we will obtain. For the case of tactile recognition, we require a surface-based representation that allows for fast, efficient processing of sparse data. Representation structures such as solid modelling [28], oct-trees [30], generalized cylinders [31,32], tables of invariant moments [22,23], fourier descriptors [29,38], and such are basically volumetric representations and generally require extensive, dense collections of data, and are therefore not particularly acceptable for tactile work.

Surface-based representation schemes have been developed for use in CAD/CAM systems [33], for object recognition using laser range data [19,27], and for some vision work [34]. These typically belong to one of two categories. In the first category, object surfaces are modeled by patches of parametric surfaces such as quadric polynomials [19-21], bicubic spline patches [33], Bezier patches and cartesian tensor product patches [33]. These patches are typically selected by the analyst and used to build up a model of an object. An objection to the use of such methods is that they are computationally very expensive. The local matching of data to model surfaces is complicated, especially if there is sensor error.

The second category of surface-based representation methods is in some sense a subset of the first, but has distinct features that make it important in its own right. This method segments an object's surface into planar facets [17-21,34], where a least squares analysis is made of the error between the modeled planar facet and the true object surface. During the modeling phase, planar model faces are "grown" until the error reaches some prescribed threshold, whereupon a new facet is started (There will be, admittedly, a large number of faces in regions of moderate to high curvature using this technique, with a concurrent increase in model complexity). An important aspect of this representation is that, during the recognition phase of contemplating where data might have come from, there is a bounded finite number of interpretations of the data, i.e., of assignments of data to faces. This still requires fairly dense sensor data, but it considerably simplifies the model matching problem that is so severe in the general parametric surface representation.

2.3 An Appropriate Representation and Recognition Scheme for Tactile Sensing

The tactile sensing work of [27,35] employs a representation and recognition structure that is quite appropriate in light of the preceding discussion, and is the one chosen for this work. It uses a surface description that segments objects into planar patches and assumes that tactile measurements are comprised of the single point of contact of the sensor with a face of an object and the surface normal of the face at that point.

Object representation is embodied in tables of face vertices, normals, and tables of constraints between distances, normals and directions between all pairs of faces for each object. The set of possible assignments of data to model faces can be structured as an Interpretation Tree [27], which is quickly and efficiently pruned by first exploiting the constraints and then performing model checks on remaining branches to determine what possible positions and orientations of which objects are consistent with the data. The method is quite fast and degrades gracefully with increasing measurement error. It is limited to planar objects and makes no use of derived properties of surfaces such as curvature, although it is easily generalized to include such information. (Such information is useful only if it can be reliably obtained. Since curvature is essentially a second derivative, it is quite sensitive to measurement error, and hence may not be useful if a numerical value is required. It may be possible, however, to reliably measure and use the sign of the curvature.)

A demonstration of the method assuming fairly complex three-dimensional objects, as well as a detailed analysis of the process, is given in [27]. A presentation of the important equations in modified form is given in Appendix 1. We motivate and discuss the salient points in what follows.

2.3.1 The Interpretation Tree

We assume that the sensed object is one of a number of modeled, possibly non-convex polyhedra. In the general case, the object may have up to six degrees of positional freedom in the global coordinate system. As previously mentioned, the sensor is capable of conveying the contact position and object surface normal in the global system. The goal of the system is to use the measurements to determine the identity, positions and orientations of objects that are consistent with the data. If there are no allowable positions and orientations of a candidate object that are consistent with the data, we can discard the object as a contender. Thus, we can solve the recognition process by doing localization, and we therefore concentrate on that problem.

Given a set of sensed points and normals and object models, we proceed as follows:

- Generate Feasible Interpretations: There are only a few credible mappings of data to faces based upon local constraints. Mappings of data to faces that violate these constraints are excluded from further attention.

- Model Test: Only a few of the remaining interpretations are actually consistent with the models in the sense that we can find a transformation from model coordinates to global (or data) coordinates. An interpretation is allowable if, when the transformation is applied to the model, the data lie on the appropriate finite faces, and not simply on the infinite surfaces defined by the face equations.

We generate feasible interpretations of the data as follows. When we obtain our first sensed point, we can assign it to any of the faces of any objects if we have access to no other information. This is graphically depicted on what is called the Interpretation Tree (IT) [27].

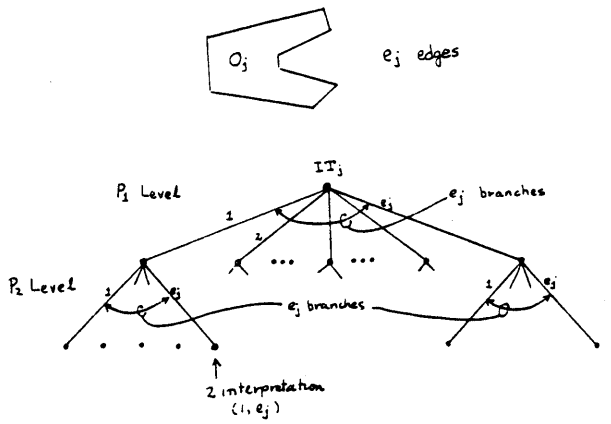


Figure 2.1 Interpretation Tree (from [27]).

At each level we have another sensed point, and if we do no analysis, we can assign that point to any of the object's faces. Each branch of the IT represents the interpretation that the sensed point at that level belongs to a particular face. There is a total of s levels in the tree, where s = number of sensed points. Since two or more points might possibly lie on the same face, each node of IT_j has e_j branches. This essentially represents the search space for feasible interpretations of the data. If we can perform some analysis to prune away entire subtrees, we can reduce the number of computationally expensive model tests to perform.

The number of possible interpretations of the sensed points is [27]

$$\sum_{i=1}^{m} (n_i)^s$$

where m = number of known objects

n_i = number of faces on object i

s = number of data points.

This can become quite large, which implies that it is not feasible to perform a model check on every conceivable interpretation. Note that the number of possible interpretations (possible combinations of assignments of data to faces) increases exponentially with the number of sensed points, whereas the set of feasible interpretations is reduced. We exploit local geometric constraints to exclude large sets of possible interpretations (subtrees) and generate the much smaller set of feasible ones.

2.3.2 Pruning the Interpretation Tree

We can make use of geometric constraints to prune the IT without having to perform model checks. Although there are many constraints that can be exploited, the following three are quite powerful [27]. The reader is referred to Appendix 1 for a detailed explanation of the constraints and of the pruning process.

1. Distance Constraint. If we are to contemplate assigning sensed point 1 to face i, and sensed point 2 to face j, then the distance between points 1 and 2 must be between the minimum and maximum distances between any points on faces i and j. See figure 2.2.

2. Angle Constraint. The angular relationship between sensed normals must be the same as that between the assigned faces in an interpretation. If we allow for angular error on the sensed normals, the range of possible angles between the sensed normals must include the angle between the normals of the assigned model faces paired with them in an interpretation.

3. Direction Constraint. The range of values of the component of the vector from sensed point 1 to sensed point 2 in the direction of the measured normal at point 1 must intersect the range of components of all possible vectors from face i to face j in the direction of the modeled normal of face i, where faces i and j are paired with sensed points 1 and 2 in the interpretation. The same must be true in the other direction, from point 2 to 1 and face j to i. Note that, in general, the ranges are different in the different directions; this test is not symmetric.

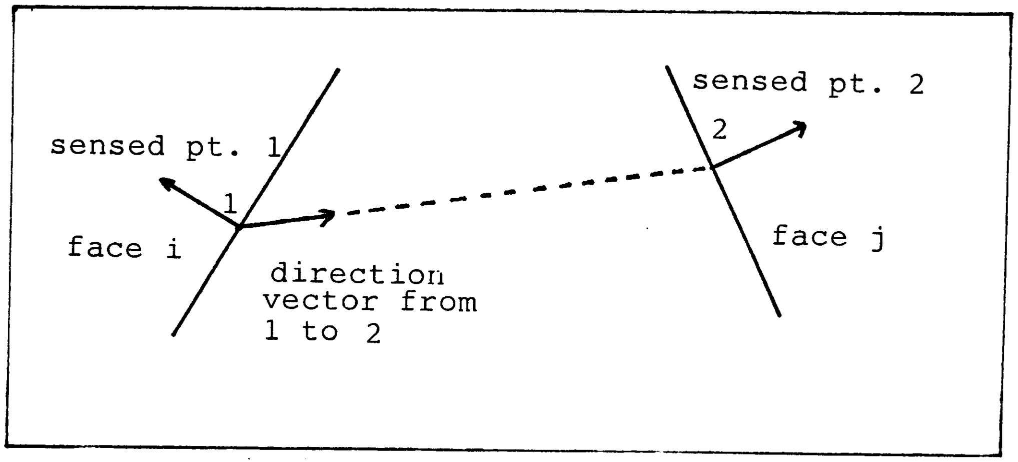


Figure 2.2 Exploiting Geometric Constraints

The application of the constraints has the effect of pruning entire subtrees of the IT, thereby vastly reducing the required number of model checks. In general, a few interpretations will survive constraint pruning and model checking. This means that there is not enough information in the data to decide on a single interpretation (presuming the object(s) is (are) not biaxially symmetric, in which case multiple interpretations are equally correct.)

We perform model checks on the surviving interpretations to insure that the data actually fits on the finite model faces, and not simply on the infinite faces described by the face equations. For each feasible interpretation we calculate the the angle θ which rotates the model, and the translation $\underline{V}_0$ which translates the model so that the model matches the data. The transformation equation applied to each vertex $\underline{V}_m$ of the model to produce $\underline{V}_g$ in the global system is

$$\underline{V}_g = R\,\underline{V}_m + \underline{V}_0$$

$$\text{or} \quad \left[\begin{array}{c} \underline{V}_g \\ \hline 1 \end{array}\right] = \left[\begin{array}{c|c} R & \underline{V}_0 \\ \hline 0 & 1 \end{array}\right] \left[\begin{array}{c} \underline{V}_m \\ \hline 1 \end{array}\right]$$

where R is 2x2 and $\underline{V}_g$ and $\underline{V}_m$ are 2x1 for our case. For our 2 D case,

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \quad \text{and} \quad \underline{V}_0 = \begin{bmatrix} x_{tr} \\ y_{tr} \end{bmatrix}$$

We find θ by computing the difference between the sensed normal direction and the normal direction of the face assigned to that sensed normal in the interpretation. If we allow for measurement error, we calculate θ by averaging the computed differences for all sensed normals. (Determining orientation in three dimensions is considerably more complex. See Grimson and Lozano-Perez [27] for development.)

We use position and normal measurements to determine the translation component of the transformation. The development of the expression relating these measurements to V_0 appears in Appendix 1. The relation is

$$[\hat{k}\cdot(R\underline{n}_i \times R\underline{n}_k)]\underline{V}_0 = (R\underline{n}_i\cdot\underline{V}_{gi}-d_i)(R\underline{n}_k \times \hat{k}) + (R\underline{n}_k\cdot\underline{V}_{gk}-d_k)(\hat{k} \times R\underline{n}_i).$$

Again, if we allow for measurement error, we compute $\underline{V}_0$ for all data pairs and average the results.

We now have to contemplate obtaining another measurement, and it is here that the notion of a strategy becomes important. We could simply choose another sensing direction at random, which might leave us with as many interpretations as we now have, or we can try to choose a path which gives us a measurement with highest information content.

We focus on 2-D objects in order to more clearly motivate and more simply develop a sensing strategy. The 2-D representation and recognition case is contained in the more general 3-D case with no significant change in the process [35]. The generalization to three dimensions of the ideas comprising the strategy are discussed in Chapter Six.

CHAPTER 3 A Sensing Strategy Assuming Perfect Measurements

This chapter addresses the planning problem associated with active touch, i.e., determining some "intelligent" schedule of sensor moves based upon a knowledge of modeled objects and the tactile information obtained thus far. Specifically, given that an object is in the environment and is one of a number of modeled objects, the objective is to determine the identity, location and orientation of the object, using tactile signals only, by selecting the best path along which to direct the sensor for the next measurement. The best path at any stage in the process is the path for which the ratio of the cost in taking the measurement with the expected gain in information is minimum. We would typically choose paths for which recognition and localization would be complete after the next measurement (a "distinguishing measurement"). If constraints such as maneuverability or time are important, however, a system might choose to take somewhat less diagnostic measurements, but under no circumstances should it take measurements of zero diagnosticity. Purposeful, directed tactile exploration should not include making moves when it is certain that nothing will be learned.

Therefore, the "strategy engine" for a system should generate three generic families of sensor moves for use by a higher level strategist, one that itself bases candidate moves on the strategy engine's output, knowledge of the physical arm configuration, torque limits, task specifications, etc.[61]. The first of these is the family of moves that guarantee recognition and localization with the next measurement, if such moves exist. The second family provides suboptimal moves in the sense that there is no guarantee of a distinguishing measurement, but at least something will have been learned. It will be shown that, assuming perfect measurements, this family of paths is not null except in the case of a single, biaxially symmetric object, where it is meaningless to talk about absolute orientation anyway. Finally, the third family contains paths that will provide absolutely no new information after the next measurement is made (and thus are to be avoided). It will be shown that this family is also not null if perfect sensing is assumed.

If one intentionally probes a specific surface in order to distinguish between objects (which implies that some minimal amount of information has already been obtained), one must have an idea of the positions and orientations for all the interpretations in order to decide upon a path. Therefore, the problem of object recognition and localization using a strategy *contains* the problem of object localization when the object is known. Hence, to fix ideas and motivate the method, we will solve the following simple problem: What are the conditions for generating the three families of paths, and what are these paths, for the case of localization of a single, known, planar 2-D object with perfect measurements of tactile contact positions and measured surface normals?

3.1 ACTIVE TOUCH: A SENSING STRATEGY

When an automated tactile recognition system begins its search, there is no more information available than perhaps that there is an object in the environment. Any strategy would simply be forced to implement some sort of blind search. Once contact has been made the situation is different, although the first measurement may not tell us much. For instance, if we are dealing with planar objects and obtain a point and normal at a face of an object of n faces, there are n possible orientations with a one-parameter family of translations for each orientation. Any other measurements m such that $|\underline{n}_1 \cdot \underline{n}_m|=1$, where $\underline{n}_i$ is the i^{th} normal measurement, lead to a similar result, although perhaps the number of orientations and the allowable ranges of translations may be reduced. The only reliable way to *constrain* the interpretations is to obtain at least two measurements with normals obeying $|\underline{n}_1 \cdot \underline{n}_2| \neq 1$, because then the translational degrees of freedom disappear.

Some methods for selecting a second measurement point might include approaching the object from a random direction, or sliding along the object, or, preferably, moving to outside the envelope of translations and approaching the object in a direction orthogonal to the first measured normal, along a ray that intersects faces for every interpretation as close to orthogonally as possible, as shown in figure 3.1.

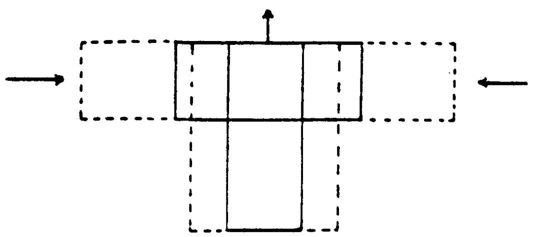


Figure 3.1 Obtaining a constraining measurement.

Once we have constrained the object we are left with some finite number of interpretations of the data, as shown in figures 3.2 and 3.3. The composite image resulting from overlaying the interpretations will be called the Multi-Interpretation Image (MII).

At this point it is appropriate to introduce definitions of the entities we shall be using for strategic path generation. Denote the enclosed area (volume in 3-D) of interpretation n by A_n. We will treat all areas (or volumes) as infinite sets on which we can perform the union and intersection set operations. See figure 3.2 for visual support of the definitions.

<u>Definition 1:</u> The area of a single interpretation i is

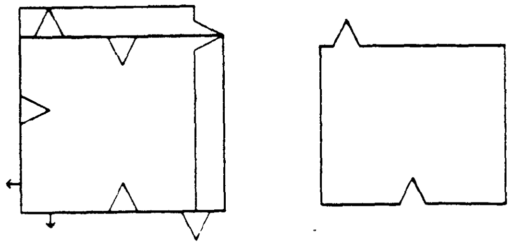


Figure 3.2 Object and Multi-Interpretation Image with respect to the data represented by the arrows.

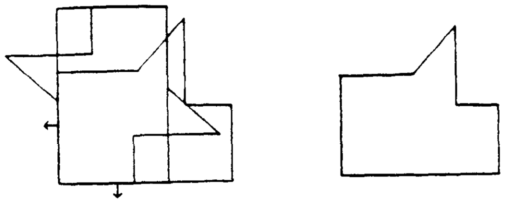


Figure 3.3 Object and Multi-Interpretation Image.

denoted by A_i and is bounded by an Interpretation Boundary.

<u>Definition 2:</u> The area A_I such that

$$A_I = A_1 \cap A_2 \cap \ldots \cap A_n$$

is the Intersection Area of the interpretations with respect to the data. The boundary containing A_I is called the Intersection Boundary. In two dimensions, it is the boundary traced out by starting at a data site and travelling along an Interpretation Boundary in a counter-clockwise direction, always choosing the left-most path at any fork.

<u>Definition 3:</u> The area A_U such that

$$A_U = A_1 \cup A_2 \cup \ldots \cup A_n$$

is the Union Area of the interpretations with respect to the data. The boundary containing A_U is called the Union Boundary. It is obtained in 2-D in precisely the same way as for the intersection boundary except that the right-most path at any fork is chosen.

<u>Definition 4:</u> Any boundary or section of boundary of the composite multi-interpretation image that is common to more than one interpretation is called an Overlapping or Blocking Boundary. For example, in figure 3.4, interpretation boundary <u>AB</u> overlaps <u>CD</u>. The blocking boundary for this situation is <u>CB</u>.

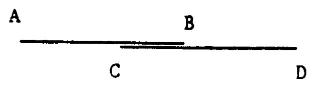


Figure 3.4 Boundary Segment CB is a Blocking Boundary.

A blocking boundary has a Degree or Strength associated with it that is determined by the number of interpretation boundaries that share it. A blocking boundary is of degree n-1 when at least n interpretation boundaries are common to it. In the example above, blocking boundary CB is of degree 1. For reasons that will become clear later, this definition allows us to view a high degree blocking boundary as a lower degree boundary. For example, a blocking boundary of degree 2 (at least 3 boundary segments overlap) can be viewed as a blocking boundary of degree 1 because at least 2 boundaries overlap. In general, then, any higher degree blocking boundary can (and will) be viewed as any lower degree blocking boundary when necessary.

With these definitions we are now in a position to introduce some observations that lead to two basic theorems in strategic path generation.

3.1.1 OPTIMAL PATHS

Optimal paths are those which are guaranteed to lead to a distinguishing measurement. In order to develop a method for finding them, an assumption and a series of observations are made regarding the nature of the measurements.

Assumption: Although we speak of the normal to a surface at a point, any device measuring a surface normal samples an area of finite size. Indeed, the surface normal is undefined at an edge or a vertex. We will therefore say that the distance and normal measurements are obtained from a data patch or site, and this patch is of finite length. (In 3-D, it is of some finite area.)

The following observations 1-3 are considered self-evident and are stated without proof (refer to figures 3.2 and 3.3). They are useful for automating the determination of intersection, union and blocking boundaries for 2-D planar objects. They are true simply because all data points are common to the interpretations.

Observation 1: Given the multi-interpretation image arrived at from the constraining data, each data site is a segment of the intersection boundary.

Observation 2: Given the multi-interpretation image, each data site is a segment of the union boundary.

<u>Observation 3</u>: Given that data sites are finite and non-zero, data sites always lie on finite blocking boundary segments of degree n-1, where n is the number of interpretations.

<u>Observation 4</u>: There will always be at least one finite, non-empty intersection area in the multi-interpretation image.

Proof: Part 1: <u>Finite</u> - The intersection area is obtained from the intersection of n finite interpretation areas. The intersection of n finite areas is less than or equal to the smallest of the areas, which will always be finite. Therefore, the intersection area is finite, and it follows that the intersection boundary is closed.

Part 2: <u>Non-Zero</u> - From Observation 1, the data site is a part of the intersection boundary. Assume the intersection area is zero. Then either the area of at least one of the interpretations is zero, or at least one intersection area does not have a common overlap with the others. But we know that the areas of the interpretations are all non-zero, and there is at least one finite data patch (from the assumption) common to all the interpretations, which implies overlap to some extent; the intersection area must be non-zero. ///

Observations 1-4 motivate and support theorems 1 and 2, which form the basis of the sensing strategy. First, let us view an example. Figure 3.5 shows the object and interpretations from figure 3.2 along with members of the family of paths that are guaranteed to produce data unique to only one interpretation. Notice that for each path, there are three distinct intersection positions and/or surface normal orientations at the intersections of the paths with each of the three interpretations. In this case, each of the paths is guaranteed to produce a distinguishing measurement.

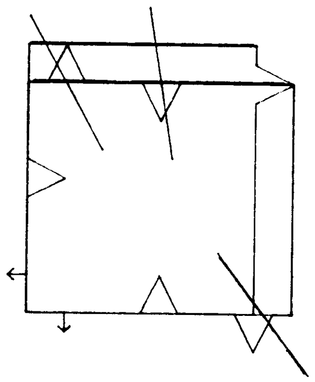


Figure 3.5 Paths for distinguishing measurements.

Theorem 1 describes when such families of paths are available and what distinguishes them from other paths.

Theorem 1: Assuming no measurement error, any path that originates outside the Union Boundary and terminates inside the Intersection Area without passing through a Blocking Boundary is certain to provide a distinguishing measurement.

Proof: There can be no boundaries outside of the union boundary by definition. Likewise, there can be no boundaries within the intersection boundary. Any ray that passes from outside the union boundary to inside the intersection boundary necessarily passes from outside to inside the area of each and every interpretation. In order to do this it must cross the boundary of each and every interpretation at least once. If, furthermore, the path does not cross a blocking boundary, then it crosses the boundary of each and every interpretation uniquely at least once. Therefore, since one of the interpretations is the "true" one, if the sensor is directed along such a path, it is guaranteed to touch the boundary corresponding to the true interpretation and report a unique measurement belonging only to that interpretation. ///

Theorem 1 implies that if a certain condition is met, namely that the blocking boundary is not closed, then at least one family of paths exists that will provide recognition and localization with the next measurement. It says nothing about when one can expect the the blocking boundary to be open. Indeed, the problem of determining whether the blocking boundary is open is a very difficult one that I suspect has no analytic solution. It appears that a detailed geometric analysis must be performed for every situation, after every measurement, in order to derive the nature of the blocking boundary.

Theorem 1 also says nothing about the nature of any paths that might be found. In the system developed, straight-line paths are sought. Straight-line paths are not guaranteed, however, and candidate paths can be curved in some situations. Finding a path in such situations is essentially a maze-running problem where the blocking boundaries act as maze walls. Automating the development of such paths is beyond the scope of this work.

Figures 3.6 and 3.7 show the results of using the theorem in a software system that performs the geometric analysis necessary for path generation. Each figure shows a known object along with the multi-interpretation image with highlighted blocking boundaries. Note path examples for which distinguishing measurements are guaranteed. Each of these cases takes on the order of 10 seconds to run on a PDP11/34.

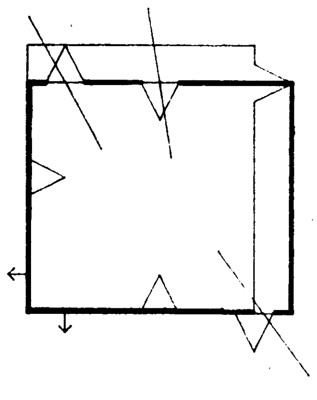


Figure 3.6 Highlighted Blocking Boundaries with distinguishing paths.

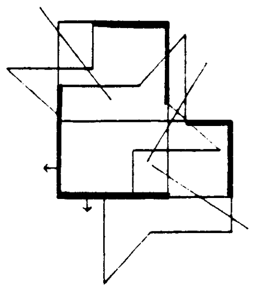


Figure 3.7 Highlighted Blocking Boundaries
with distinguishing paths.

We have assumed nothing about what a single object's shape must be, other than tacitly assuming closed boundaries and finite, non-zero area. Indeed, the theorem is equally valid in the case of multiple objects, where there might be multiple interpretations of different objects with respect to the data. Here the full power and utility of the method become apparent. For example, the difficult problem of distinguishing between two very similar objects is handled quite routinely and correctly. As an example, consider the similar objects A and B in figure 3.8 along with the multi-interpretation image associated with the constraining data shown.

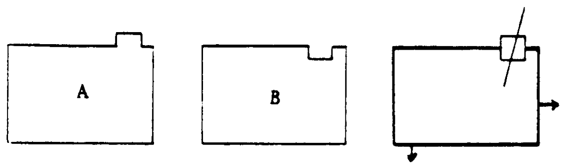


Figure 3.8 The discrimination of two similar objects.

The appropriate family of paths is easily found.

3.1.2 SUB-OPTIMAL PATHS

Let us now focus on a different problem. There may be situations in which an optimal path is unreachable or undesirable, or it is determined that the blocking boundary is closed. In any event, we may be willing to settle for a measurement that is more or less likely to provide recognition and localization in lieu of one that is certain to. The problem then becomes one of determining such paths and providing some measure of how likely they are to provide distinguishing measurements. Theorem 2 is concerned with this.

Theorem 2: Assuming that each of the interpretations in the multi-interpretation image is equally likely, any path originating outside the Union Boundary and terminating inside the Intersection Area that passes through each interpretation boundary once and through a single Blocking Boundary of degree m will, with probability (n-m-1)/n, provide a distinguishing measurement, where n is the number of interpretations.

Proof: Consider a path penetrating n boundaries, one from each of the interpretations, m+1 of which overlap. Then there are n-(m+1) distinct, distinguishable boundaries from which to obtain distinct data. Now, since we are given equal likelihood of the interpretations, the chances of obtaining the "true" data from any chosen one of the n boundaries is simply 1/n. The chances of obtaining the "true" data from any of the non-overlapping boundaries is therefore equal to the number of such boundaries divided by n, or

$$\frac{n-(m+1)}{n}$$

///

This theorem states that if a path does cross a single blocking boundary, the chances are reduced that a fully distinguishing measurement will be made. Equivalently, the chance of being left with m+1 interpretations after the measurement is (m+1)/n. This has some interesting consequences. The first is that, since any data site is on a blocking boundary of degree n-1, any path near the site that passes through the blocking boundary will generate a measurement such that the chance of being left with (n-1)+1 = n interpretations is ((n-1)+1)/n = 1. In other words, nothing will be learned. The theorem essentially advises us to avoid taking measurements near data sites. The second consequence of the theorem is that we can now trade off movements sure to produce distinguishing data with perhaps more desirable moves less certain to produce such data. Such moves might be more desirable because of geometric, mobility, or time constraints, etc.

A third consequence of the theorem is that we now have a way of dealing with the case of a low degree closed blocking boundary. It should be evident that, except for the case of a single symmetric object, there can never be a closed blocking boundary of degree n-1 everywhere, where n = number of interpretations. The blocking boundaries at the data sites will be of degree n-1 and extend for some length, but at other sites the boundaries will be of some lower degree. This means that we have only to shift our attention to blocking boundaries of higher degree until the single closed blocking boundary "opens up". In other words, the path-finding algorithm is applied to the multi-interpretation image as before, but paths are sought that do not intersect strength 2 blocking boundaries. At this point the path-finding routine will generate families of straight paths (if possible) just as before, only now distinguishing data is not guaranteed because candidate paths will pass through degree 1 blocking boundaries. We will, however, have obtained the best paths available in the sense that the chance of recognition and localization is as high as possible. As an example, consider figures 3.9 and 3.10. Here we have a situation in which the degree 1 blocking boundary is closed. By simply shifting our attention to degree 2 boundaries we can find a family of paths that, while not guaranteeing recognition and localization, gives us the highest probability of such (0.5 in this case).

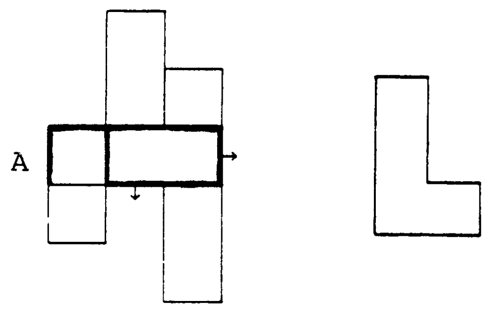


Figure 3.9 Degree 1 Blocking Boundary is closed.

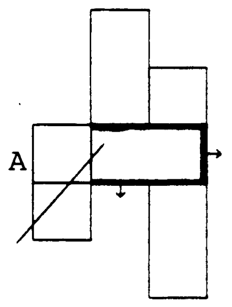


Figure 3.10 Degree 2 blocking boundary is open.

Note in figure 3.10 that the path intersects a single blocking boundary (shown in figure 3.9). If the path were to pass through Face A, which is also a blocking boundary, there would be no chance of making a distinguishing measurement. We know a-priori that there would be two interpretations remaining after taking the measurement.

An uniform, discrete distribution of the interpretations is assumed in the statement of Theorem 2. Clearly, if the physics of the problem is such that some objects or orientations are more likely than others, then some interpretations will be more likely than others. If this is the case, the more likely interpretations should be weighted more heavily in the analysis. This is discussed in the next section.

3.1.3 <u>Unequal distribution of Interpretations</u>

Theorem 2 assumes an equal, discrete distribution of the interpretations and states the probability of obtaining a distinguishing measurement if the sensor is directed along a path that passes through a single blocking boundary. There are certain situations in which the physics of the problem imply that some interpretations are more likely than others, however. For instance, consider an object resting on a plane. Some poses of the object are more stable in the presence of disturbances than others, and if the object is randomly thrown onto the plane, one would expect to observe the more stable poses more often than the less stable poses. It may also be that some faces are more likely to be sensed than others (randomly oriented object and random sense directions), which also affects the probability of occurrence of the various interpretations.

Under such circumstances it is appropriate to assign unequal probabilities to the various feasible interpretations one might obtain. This is a more general case and includes the uniform case as a subset.

We treat the problem by weighting boundaries associated with more likely interpretations more heavily than boundaries associated with less likely ones, and by slightly changing the definition of blocking boundary strength. Consider figure 3.11, which shows a fragment of a multi-interpretation image (MII) through which we contemplate directing the sensor. Each boundary segment is from a different feasible interpretation and is labeled with the probability of occurrence of that interpretation. The two leftmost boundary segments are drawn so that they may be seen separately but they are assumed to overlap and be of the same orientation. Since the interpretations are considered to be mutually exclusive events, the probability of obtaining a distinguishing measurement (P(DM)), that is, of contacting one of the two non-overlapping segments, is simply the sum of the probabilities of their occurrence, or P(DM) = 0.2.

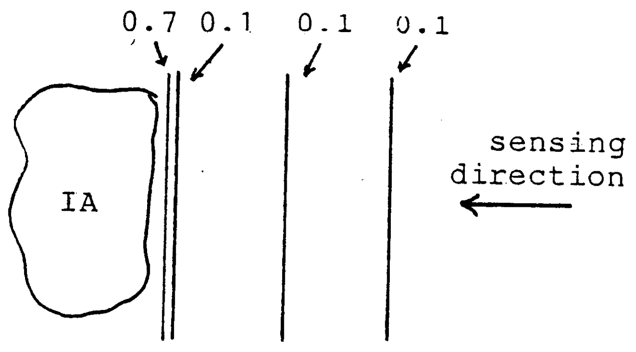


Figure 3.11 MII fragment.

In figure 3.12, the probabilities are assigned differently, and P(DM) = 0.8.

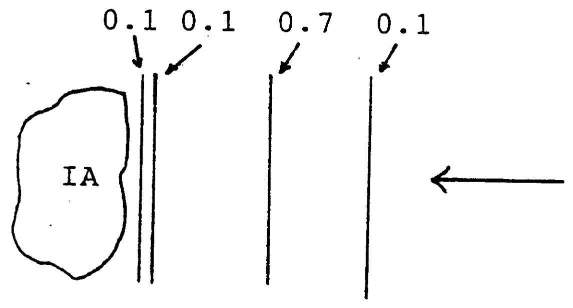


Figure 3.12. Same MII Fragment with Different Probability Distribution

Clearly, one would direct the sensor through the location depicted in figure 3.12 in preference to the one depicted by figure 3.11 because P(DM) is higher in figure 3.12. If we redefine blocking boundary strength as the sum of the probabilities of the boundaries comprising the blocking boundary, then the strategy remains the same; find the path(s) for which the strength of the intersected blocking boundary is minimum.

Figure 3.13 shows a case where, even though the blocking boundary is comprised of three boundary fragments, it is better to direct the sensor through this fragment than through the one depicted in figure 3.11.

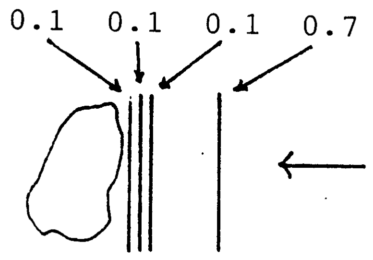


Figure 3.13 Relatively low strength blocking boundary.

3.2 Implementation

This section presents blocking boundary generation and path generation algorithms and discusses general implementation issues. All of the algorithms were developed using computational geometry (explicit manipulation of boundary vertices), and I point out some of the problems related to the "special cases" that always seem to arise when using the geometric approach. This material is here for completeness, and the reader interested in implementing the ideas discussed in this thesis are advised that it is very difficult to avoid algorithm failure due to unforeseen circumstances when using the geometric approach. Instead, the MII should be discretized using the idea of cellular decomposition [28] as discussed in the next chapter.

3.2.1 Blocking Boundary Generation

We present the case where all interpretations are assumed equally likely (The generalization to the case of unequal probability of interpretations is straightforward). The implementation of these ideas on a computer was performed by first generating the feasible interpretations of the data by constraint pruning and model checking (Chapter Two). The interpretations are described by ordered lists of vertices in global (or sensor) coordinates. These vertex lists were fed to a routine that generates blocking boundary vertices according to a blocking boundary "algebra", the explanation of which follows. In order for any two boundary segments to contribute to a blocking boundary, they must have the same normal and overlap.

One can intuitively visualize the process by imagining the interpretation boundary segments to be made of opaque glass. As the boundaries are stacked on top of each other (allowable only if they have the same orientations) and one looks from above or below at some light source, boundaries of strength 1 will be a degree darker than boundaries with no overlap, boundaries of strength 2 will be a degree darker than strength 1 boundaries, and so on. Each of the boundaries is identified in the system by endpoints and strength.

Assume the interpretation image in the lower left of figure 3.14. It is comprised of single interpretations of each of three objects. To show the method, we develop the blocking boundaries from the boundaries with surface normals pointing downwards. The boundaries from object A are combined with the boundary of object B to produce the intermediate results shown. The result is a lengthening of one of the interpretation boundaries and the creation of a strength 1 blocking boundary. The boundary from object C is then introduced for analysis. The result is one long multi-interpretation boundary, the lengthening of the previous strength 1 blocking boundary, the creation of another strength 1 blocking boundary, and the creation of a strength 2 blocking boundary.

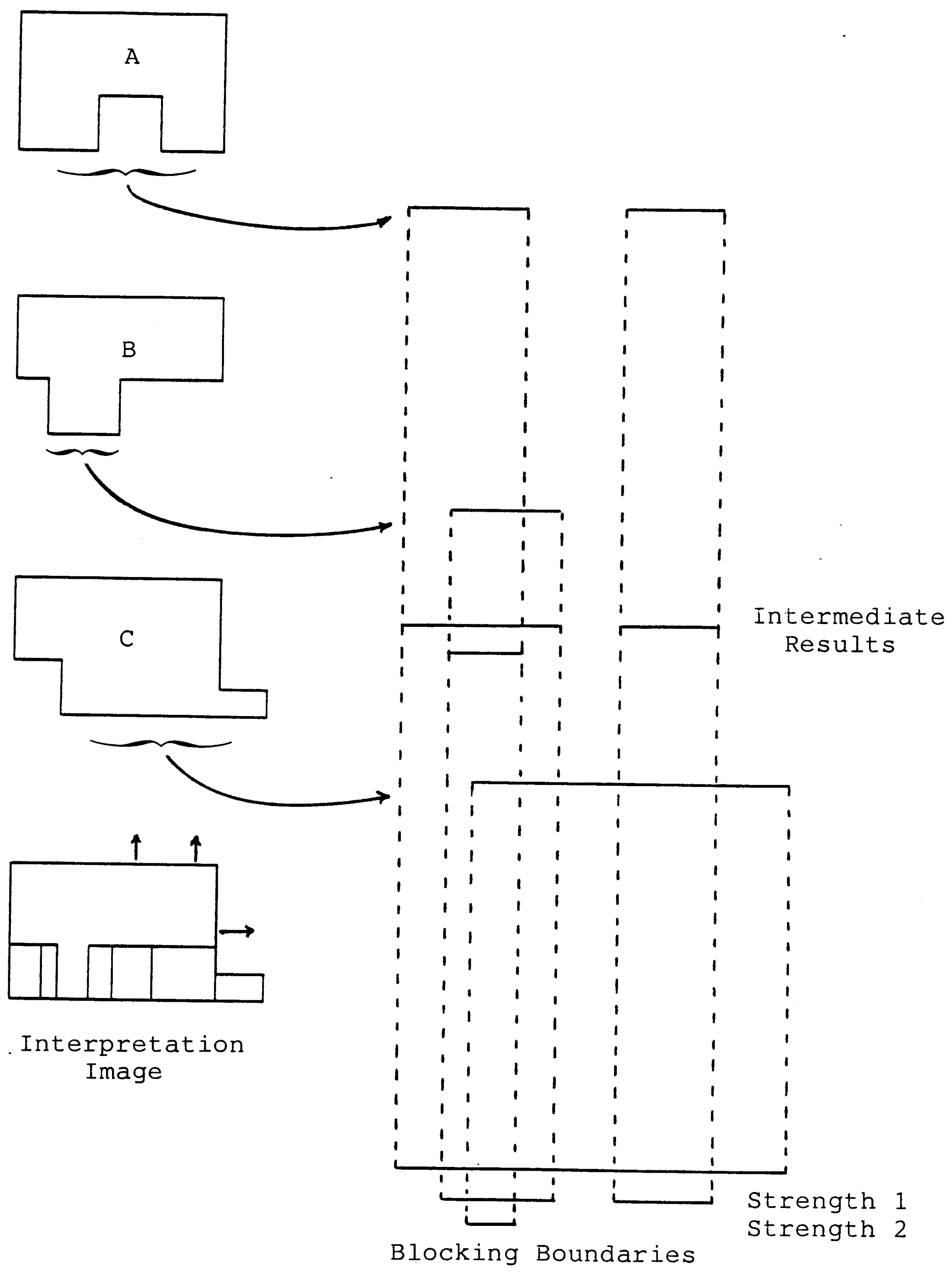


Figure 3.14 Blocking boundary generation.

3.2.2 Path Generation

Once the multi-interpretation-image (MII) has been created (with associated blocking boundaries) we can search for sensing paths. The best paths are those that originate outside the union boundary and terminate within the intersection area without intersecting the blocking boundaries of interest. The process is easily visualized in figures 3.15 and 3.16 below.

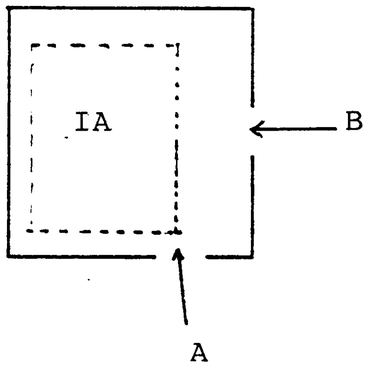


Figure 3.15 Path examples.

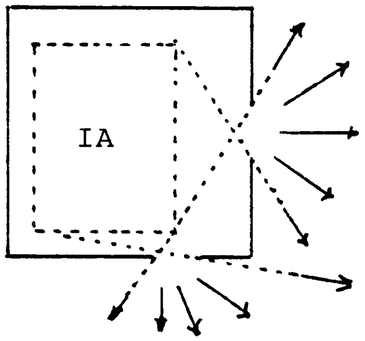


Figure 3.16 Path determination.

Paths A and B are members of the family of paths for which minimum strength blocking boundaries are not intersected. In order to compute them we can imagine a luminescent intersection area that projects light through the openings in the boundary as in figure 3.16. We can think of the openings in the boundary as "windows" through which we must direct the sensor. Since the sensor is presumably of some finite size, we should direct the sensor through the center of a window if the opening is large enough. The best entry angle is obtained by requiring the path to be as close to normal to the opening as possible and still enter the intersection areas, because the projected size of the opening as it appears along the path is greatest for such an angle. (If necessary, we could take into consideration the effective diameter of the sensor and the entry angle by using, for example, the configuration space approach of Lozano-Perez [62].) There are many problems associated with the actual implementation of this approach, however. One arises as shown in figure 3.17. In this case the sensor is directed along a path that places it on a potential discontinuity (one of the interpretation vertices).

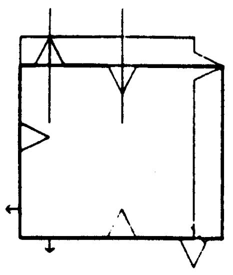


Figure 3.17 Sensor directed toward discontinuities.

Presumably situations such as these can be checked for but the check is difficult to make. (If object discontinuities are included in the representation as features and the sensor is capable of measuring them, then of course there is no problem.) A more serious problem arises when one tries to develop algorithms for determining windows. As shown in figure 3.18 an algorithm that searches for windows by sweeping a test ray through 360° at intersection area vertices and monitoring when the ray is blocked and not blocked will have problems when blocking boundary vertices are colinear with the sweep vertex. If the point D is a fraction above B (by the smallest number representable by the computer), then the vertices A and C will define a window. If D is lower than B, then A and B will define a window. This shifts the location of the window center and effectively defines two different windows.

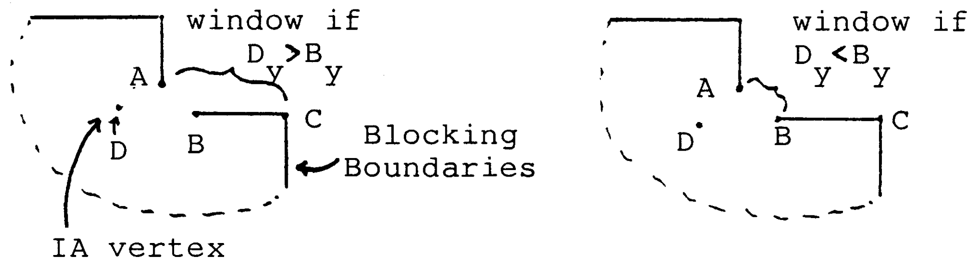


Figure 3.18 Computational geometry problems.

Another implementation problem arises when one tries to simulate the "luminescent" intersection area in real time. Loosely speaking, we are interested in determining where the blocking boundary casts shadows on some enclosing container.

These are areas from which no paths may emanate. The illuminated areas contain an infinite number of allowable paths, only one of which (for each window) meets the criterion of passing through a window center as close to perpendicular as possible. It is surprisingly difficult to implement this rather trivial geometric construct algorithmically. The implementation that was finally selected considered all pairings of effective window vertices with all intersection area (IA) vertices. Windows were tallied for each IA vertex and a list was kept of all IA vertices associated with the various windows. In this way the aperture angle for each window could be obtained and used for the final selection of entry angle. Problems arise when an intersection boundary vertex coincides with a blocking boundary edge vertex. (The algorithm must allow an IA vertex to play the role of a window vertex. The details become very tedious.) Similar consideration must also be paid when a disconnected segment of blocking boundary exists. Both of these cases appear in figure 3.19.

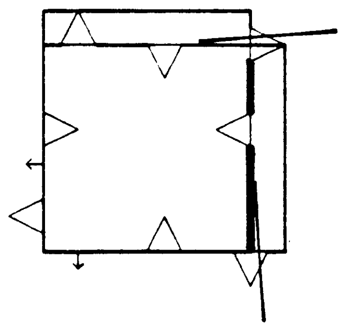


Figure 3.19 Further computational geometry problems.

There are many geometric computation problems like these that arise when the analysis is carried out analytically using vertices. The problems are exacerbated when one considers the generalization to three dimensional objects, and of course the analysis becomes extremely difficult when one allows curved boundary segments. A potential solution to this problem is presented in the next chapter where we investigate the effects of measurement error.

In summary, we have developed a method for determining strategic moves of a tactile sensor, assuming perfect measurements, that addresses some of the issues discussed in the beginning of this chapter. We can find straight paths that guarantee distinguishing measurements, if such paths exist. We have determined that some paths are clearly better than others from the viewpoint of the probability of recognition and localization, and have a reliable methodology for finding them. Finally, we can determine what measurements *not* to make, such measurements being a waste of time. The system that has been developed demonstrates that the process can be performed in real-time in the absence of measurement error.

The inclusion of error on the measurements, however, results in uncertainty about the positions and orientations of the interpretations. This necessarily affects the analysis and we focus attention on it in the next chapter.

CHAPTER 4 Strategy in a Real Environment: Measurement Errors

The previous chapter assumed no error on the positions and orientations of the interpretations after model checking. In this chapter we recognize that errors on position and normal measurements give rise to errors in the transformations from model to global coordinates, and hence to uncertainty in the positions and orientations of the interpretations. We show how these uncertainties can be included in the multi-interpretations image and how they influence the performance of our strategy. Also an alternative representation of the multi-interpretation image which eliminates the problems of computational geometry alluded to in the previous chapter will be presented.

Measurement error results in uncertainty in the computed positions and orientations of the feasible interpretations in the interpretation image. Bounds on the uncertainties due to contact measurement errors and surface normal measurement errors are described in the next section. Section 4.2 describes how to include the uncertainties for each interpretation in the interpretation image. Section 4.3 discusses the methods chosen for implementing the ideas. A method for representing the interpretation image in a way which allows for considerable simplification of computation algorithms is described.

4.1 Bounds on Transform Error

Grimson [5] investigated the effects of measurement error on the computation of the transformation matrix for the general 3-dimensional case. The results of the analysis for the specialized 2-dimensional case is presented here.

We assume (See Chapter 2) that a vector in the model coordinate system, $\underline{V}_m$, is transformed into a vector in the global system, $\underline{V}_g$, by the following transformation:

$$\underline{V}_g = R\,\underline{V}_m + \underline{V}_0$$

$$\text{or} \quad \left[\begin{array}{c} \underline{V}_g \\ \hline 1 \end{array}\right] = \left[\begin{array}{c|c} R & \underline{V}_0 \\ \hline 0 & 1 \end{array}\right] \left[\begin{array}{c} \underline{V}_m \\ \hline 1 \end{array}\right]$$

where R is 2x2 and $\underline{V}g$ and $\underline{V}m$ are 2x1 for our case. For our 2 D case,

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \quad \text{and} \quad \underline{V}_0 = \begin{bmatrix} x_{tr} \\ y_{tr} \end{bmatrix}$$

where θ is the counter-clockwise rotation angle through which the vector is rotated and x_{tr} and y_{tr} are the x and y translation components, respectively, of the vector. Two dimensional specialization of the work of Grimson shows that the upper bound on $\Delta\theta$, the angle error, is:

$$\Delta\theta \approx \phi$$

where

$$\hat{n}'_m \cdot \hat{n}_s \geqslant \varepsilon_n = \cos\theta \ ,$$

and where

$\hat{n}'_m$ is the true unit normal vector of the object face,

$\hat{n}_s$ is the measured unit surface normal

ϕ is the included angle of the surface normal error cone associated with the measurements.

This says that for the 2-D case the worst case error in the rotation component of the transformation is simply the error cone associated with the normal measurements. This is sensible because we know that θ is obtained by clustering the differences between measured and modeled surface normals for each interpretation, and that in the worst case all measured normals can simultaneously be in error by the same amount ϕ, with the result that the computed value of θ will be in error by ϕ.

The error associated with the translation component of the transform was also obtained by Grimson [5]. The form for $\underline{V}_0$ in three dimensions is obtained in a similar way to that shown in Appendix 1 and is given by

$$\begin{aligned}[\underline{n}_i \cdot (\underline{n}_j \times \underline{n}_k)]\underline{V}_0 = {} & (\underline{n}'_i \cdot \underline{P}_i - d_i)(\underline{n}'_j \times \underline{n}'_k) \\ & + (\underline{n}'_j \cdot \underline{P}_j - d_j)(\underline{n}'_k \times \underline{n}'_i) \\ & + (\underline{n}'_k \cdot \underline{P}_k - d_k)(\underline{n}'_i \times \underline{n}'_j)\end{aligned} \tag{4.1}$$

where $\underline{n}_i$ is a face normal in model coordinates, $\underline{n}'_i$ is the transformed normal in global coordinates, $\underline{P}_i$ is the position of the contact point in global coordinates, and d_i is a constant offset for face i. The error ranges are considered for each of the components

$$(\underline{n}'_i \cdot \underline{p}_i - d_i)(\underline{n}'_j \times \underline{n}'_k)$$

separately. Grimson shows that the upper bound on the magnitude of the error is given by

$$\sqrt{[s \sin\zeta - (s+\Delta)\sin(\zeta - 2\phi)]^2 + (s+\Delta)^2 \sin(2\zeta)\sin(4\phi)} \qquad (4.2)$$

where

$$s = \hat{n}'_i \cdot \underline{p}_i - d_k$$

$$\Delta \leqslant \varepsilon_d + |\underline{p}'_i|\sqrt{2}\sqrt{1-\varepsilon_n}$$

$$\cos\zeta = \hat{n}'_j \cdot \hat{n}_k$$

$\underline{p}'_i$ = contact point in model coordinates

ε_d = distance error

ε_n = surface normal error

As ϕ tends to zero the bound reduces to $|\Delta \sin\zeta|$. This expression itself tends to zero as ε_d does, so that the error in the computed translation tends to zero as the error in the measurements do, which is a result we expect. If we further restrict ourselves to the cases where faces are roughly orthogonal, then $\zeta \approx \pi/2$ ($\cos\zeta \approx 0$) and the bound becomes

$$|s - (s + \Delta)\cos(2\phi)|.$$

We note that the result is the same in 2-D as it is in 3-D by observing that the bound is computed for each component in Eqn.4.1, where we deal with two instead of three components.

4.2 Bounds on Computed Positions and Orientations

We now investigate the effects of the transform error bounds on the orientation and location of the interpretations and discuss how this impacts the analysis of path generation. The effects of uncertainty will be shown to be equivalent to a widening of the boundaries of each interpretation. These widened boundaries will be used in the multi-interpretation image.

4.2.1 Effects of Orientation Uncertainty

We consider a single interpretation without loss of generality. Figure 4.1 shows a simple example of the effect of orientation uncertainty for a single interpretation of a box.

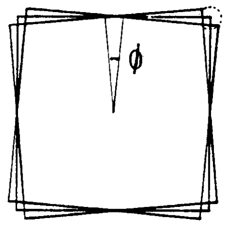


Figure 4.1 Orientation uncertainty.

The true orientation lies somewhere between $\pm\phi/2$ of the nominal.

If angular error were the only error, we could say that the true boundary lies somewhere within the union boundary and outside of the intersection boundary of the multi-interpretation image shown. Even for the single object shown, the shape of the allowable interpretation region is irregular and difficult to describe analytically. It is far easier to use the following conservative approach: Determine the vertex most distant from the model origin (the "center" of the object) and determine the distance between its endpoint positions as the interpretation is rotated from nominal - $\phi/2$ to nominal $+\phi/2$. This distance is the greatest error of position of any boundary point of the interpretation due to orientation uncertainty. Consider an error disk with this distance as the diameter and place it on the distal vertex, where it just encloses the allowable extreme vertex positions. Then run the disk along the boundary of the nominal interpretation interpretation as shown in figure 4.2. The interpretation boundary must lie within the region swept out by the error disk.

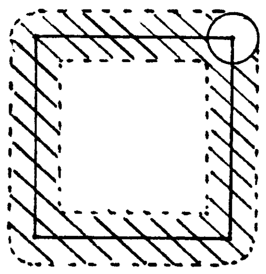


Figure 4.2 True Boundary Must Lie Within Swept Region

This is a conservative method in that it applies the worst case error to the entire interpretation boundary. For objects with large aspect ratio and sufficiently large ϕ, this can have the effect of eliminating the interior of the interpretation and preventing the existence of a non-zero intersection area, as shown in figure 4.3. In such cases, a more elaborate and less conservative analysis must be performed. Fortunately, this can be determined off-line, and only those objects with the problem need be handled differently. (Methods for doing this are beyond the scope of this thesis.)

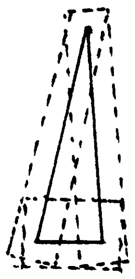


Figure 4.3 Prevention of intersection area.

A sense of the order of magnitude of the error disk may be obtained by considering a square object with 12 inch hypotenuse and 10° error on the measurements. The error disk diameter is diam $\approx 6.0 * 0.175 = 1$ inch.

4.2.2 Effects of Translational Uncertainty

The development is precisely the same as in 4.2.1. The analysis is simplified by the fact that we can choose any vertex for analysis. Conservatively, simply choose the largest member of the set of errors, Eq. 4.2, for each direction and each data pair, and use it as the diameter of an error disk that sweeps out an area in which the true interpretation must be found. Figure 4.4 shows a simple example. In figure 4.4a, the dashed outer boundary delineates the limits outside of which no boundary will be found, while the inner

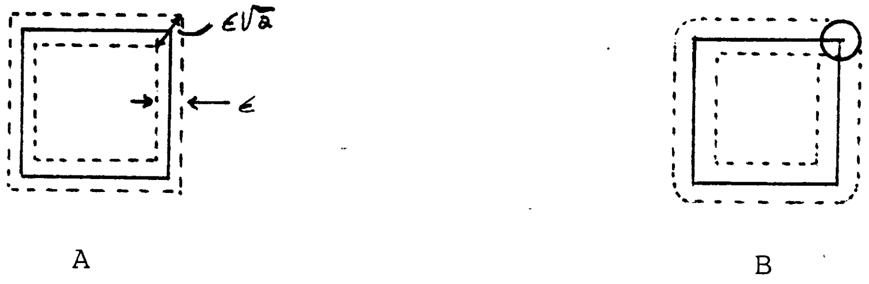


Figure 4.4 Translational error bounds.

boundary delineates the inner limits. In Figure B an error disk of diameter $\varepsilon\sqrt{2}$ conservatively sweeps out the region. For this example, if we assume errors in contact position measurement to be on the order of 0.1 inch, the error disk diam $\approx$ 0.14 inch. Note that the translational error disk does not depend on the relative size of the object, whereas the rotational error disk does.

4.2.3 Combining Uncertainties

We combine rotational and translational uncertainty by creating an error disk with diameter $\varepsilon_t = \varepsilon_a + \varepsilon_d$ where ε_a is the diameter of the error disk associated with angular uncertainty and ε_d is the diameter of the disk associated with translational uncertainty. The area swept out by this disk as it travels around an interpretation is certain to enclose the entire boundary of the interpretation as long as we have properly characterized the errors.

The use of an error disk is geometrically appealing but computationally burdensome because we can no longer simply deep track of vertices; there are curved boundaries that must be represented. We avoid this problem by "growing" each 2-D linear face. The face is widened by ε_t and each end is lengthened by $\varepsilon_t/2$. Each face is then represented by four corner vertices. This is more conservative than using an error disk because more area is swept out by "squaring out" the rounded corners; the swept area still encloses the interpretation.

Figure 4.5 demonstrates the effect of increasing total error on the multi-interpretation image. Note the shrinkage of the intersection area. Note further that smaller features become less defined as the error increases, and blocking boundaries become blocking regions and can assume complicated 2-D shapes.

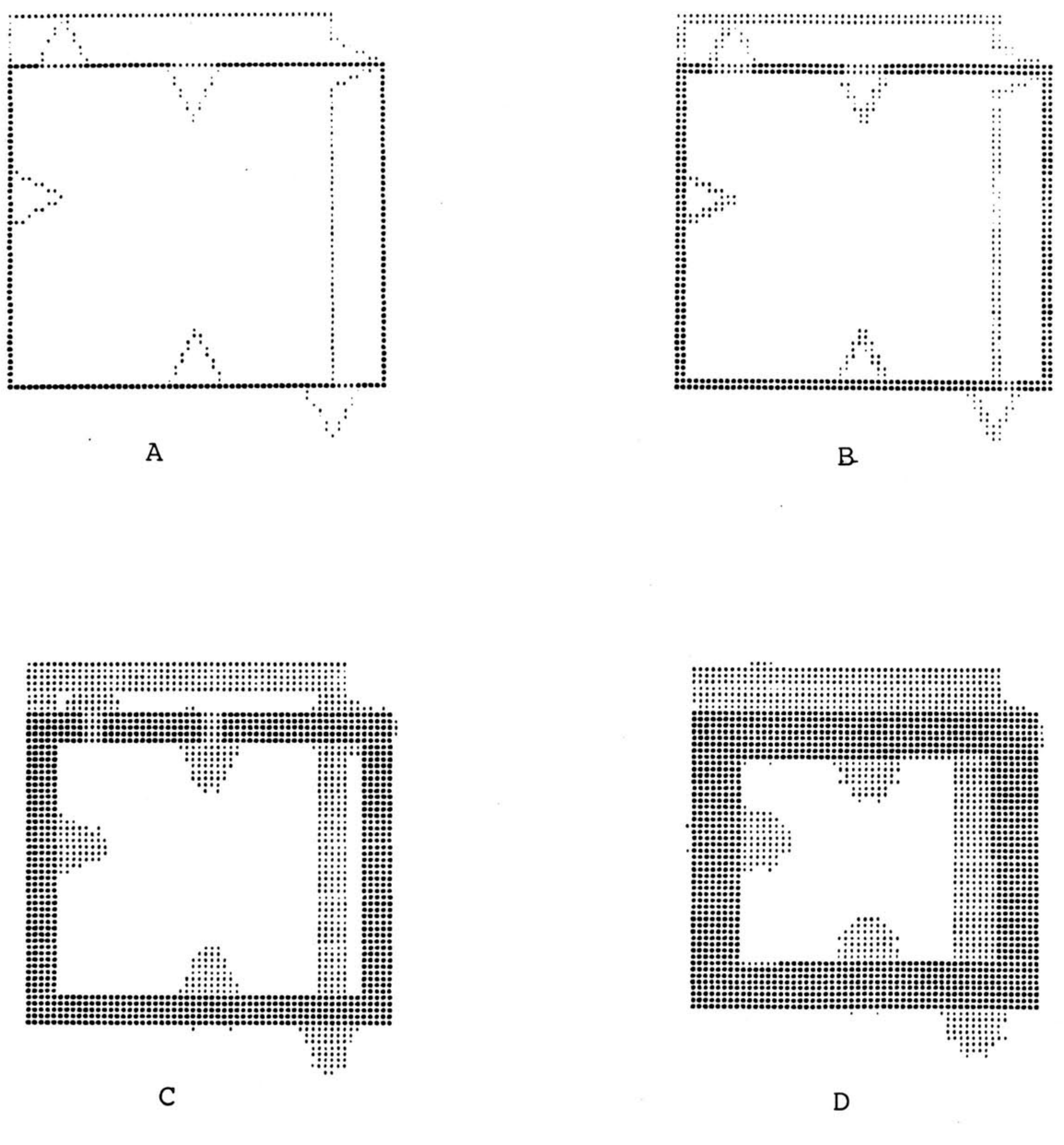


Figure 4.5

A - no error
B - 2% error
C - 10% error
D - 17% error

Percent error based on effective object diameter.

4.3 Implementation with Transform Error

The inclusion of transform error in the analysis places extra burden on computer memory requirements and algorithm complexity if we continue using geometric entities such as vertices for representation. Each face of each interpretation requires four representative vertices after error growing, and blocking boundaries can assume complex 2-dimensional shapes with no easily obtainable strong upper bound on the number of vertices required. Furthermore, checking for path obstruction is cumbersome when blocking boundary areas are represented as (necessarily) ordered lists of vertices. For this reason the problem is discretized and mapped into a grid (cellular decomposition [28]). This allows the establishment of multi-interpretation images, intersection volumes and blocking boundaries using simple addition and subtraction operations on the grid. The entire strategy is then performed by investigating the grid.

4.3.1 Discretization of the Multi-Interpretation Image

The first consideration is the spatial resolution of the grid. There is an obvious qualitative positive relationship between MII detail and the number of elements required in the grid. Unfortunately, memory requirements (and in some sense processing time) increase as the square of the size of the grid. There must therefore be some trade-off between desired spatial fidelity and reasonable computer resource requirements.

There is a somewhat subtle point with regard to this issue that is worth dwelling on for a moment. The necessary spatial resolution of the grid is a function of the object set and sensor accuracy. Since we will be mapping the MII onto a grid, there will necessarily be a loss of detail, and we must be sure that the spatial resolution of the grid is sufficient to adequately preserve important object features. We also realize that we need not preserve features so small as to be unmeasureable by our sensor. The analyst must therefore exercise some judgement and perhaps experiment with different spatial resolutions in order to strike a good balance between computer resource use and acceptable performance of the method. For the work described here a square 70x70 grid was used with very good results for the simple objects used.

4.3.2 Scaling and Mapping the MII onto the Grid

We have to scale the MII to fit into the grid with enough room left to perform the boundary growing process associated with uncertainty. The vertices of the MII in global coordinates are related to the vertices in grid coordinates by the following transformations (see Appendix 2 for equation development).

In the forward mapping,

$$\underline{X}_{gr} = \begin{bmatrix} A/\Delta & 0 & A/\Delta(\varepsilon_t - x_{glmin}) + 1 \\ 0 & A/\Delta & A/\Delta(\varepsilon_t - y_{glmin}) + 1 \\ 0 & 0 & 1 \end{bmatrix} \underline{X}_{gl}$$

where

$A = \Delta(B+2)/(\Delta+2\varepsilon_t)$

B = grid size = 70

$\Delta = \max(\Delta x_{max}, \Delta y_{max})$

ε_t = diameter of total uncertainty disk.

In the reverse mapping

$$\underline{X}_{gl} = D^{-1}C^{-1}\underline{X}_{gr} = \begin{bmatrix} \Delta/A & 0 & -\varepsilon_t - \Delta/A + x_{glmin} \\ 0 & \Delta/A & -\varepsilon_t - \Delta/A + x_{glmin} \\ 0 & 0 & 1 \end{bmatrix} \underline{X}_{gr}.$$

The effect of the forward transformation is to take the MII in global coordinates and scale and position it so it is centered in the grid and will have room enough for the boundary growing process. By allowing it to fill the grid we retain as much detail as possible.

The reverse transformation is necessary when any sensor paths found in the graph system must be converted back into the global system so that the system may direct sensor movement.

4.3.3 Boundary Growing

We present the development of boundary growing here for completeness.

The boundaries of the MII are lengthened by $\varepsilon_t/2$, where ε_t is the effective error disk diameter, and each boundary is widened by an amount ε_t. We wish to determine points $\underline{a}$, $\underline{b}$, $\underline{c}$ and $\underline{d}$ shown in figure 4.6.

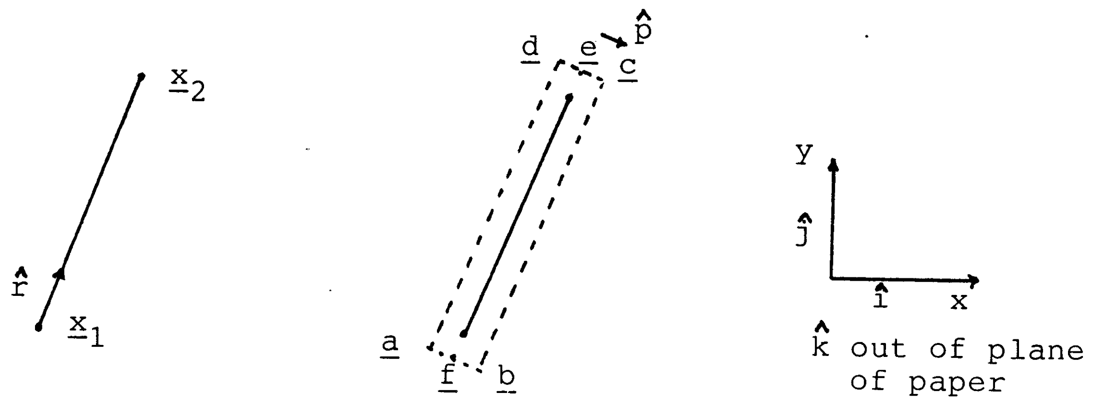


Figure 6 Face growing.

Let $\underline{r} = \underline{x}_2 - \underline{x}_1 = (\Delta x, \Delta y)$, $\hat{r} = \dfrac{\underline{r}}{|\underline{r}|}$.

Then $\underline{e} = \underline{x}_2 + \dfrac{\epsilon_t}{2}\hat{r}$

$$\underline{f} = \underline{x}_1 - \frac{\epsilon_t}{2}\hat{r}.$$

Now, $\hat{p} = \hat{r} \times \hat{k} = (\hat{r}_y, -\hat{r}_x)$

Therefore,

$$\underline{a} = \underline{f} - \frac{\epsilon_t}{2}\hat{p}$$

$$\underline{b} = \underline{f} + \frac{\epsilon_t}{2}\hat{p}$$

$$\underline{c} = \underline{e} + \frac{\epsilon_t}{2}\hat{p}$$

$$\underline{d} = \underline{e} - \frac{\epsilon_t}{2}\hat{p}.$$

These points are used as rectangular corner points on the grid so that the associated defined rectangle for each face of each interpretation may be filled in.

4.3.4 Developing the MII on the Grid

We are finally in a position to construct the multi-interpretation image on the grid. It is here that advantage is taken of simple grid operations to determine blocking boundary extent. Recall that a blocking boundary exists only when two or more boundaries simultaneously overlap and have the same normals to within angular sensor error.

We use two grids, one on which the MII is built (MGRID), and a working grid (WGRID). We start by adding all grown faces for every interpretation into MGRID, and then setting all nonzero grid elements to a 1. The result is the MII with no blocking boundary information. We next consider the first face of the first interpretation and fill it into WGRID. (Each grown face is delineated by 1's surrounded by 0's). We then add to WGRID all other faces of this interpretation and of all other interpretations whose surface normals match the first face to within sensor error. What WGRID holds after this operation is an image of all similarly oriented faces of the MII. WGRID is zero where no faces appear, 1 where a single face appears, and greater than 1 when faces overlap. The areas where WGRID (I,J) > 1 are blocking boundary regions, and we wish to increment MGRID locations by the blocking boundary strength of the corresponding WGRID locations. We do this by decrementing all nonzero elements of WGRID and sum WGRID to MGRID or, equivalently, we let

$$\mathrm{MGRID}(I,J) = \mathrm{MGRID}(I,J) + \mathrm{MAX}(0,\mathrm{WGRID}(I,J)-1) \qquad (4.3)$$

The next step in the process is to look at the next face of the first interpretation. If the surface normal is different from the first, we perform the same analysis for the rest of the faces in that interpretation and all the faces of the rest of the interpretations. In this way we step consecutively through all of the interpretations, only considering faces with surface normals different than any of the preceding normals. When we are finished MGRID contains the multi-interpretation image, complete with blocking boundaries.

Figures 4.7 and 4.8 show examples of the results of this analysis for various errors ε_t and various blocking boundary strengths.

Note the regions in figure 4.7 where the blocking boundary is of greater strength than expected (at the "corners"). These regions are the result of the summation process and represent the overlap of blocking boundaries with different orientations. We could develop methods to prevent the occurrence of such regions, but we may choose to

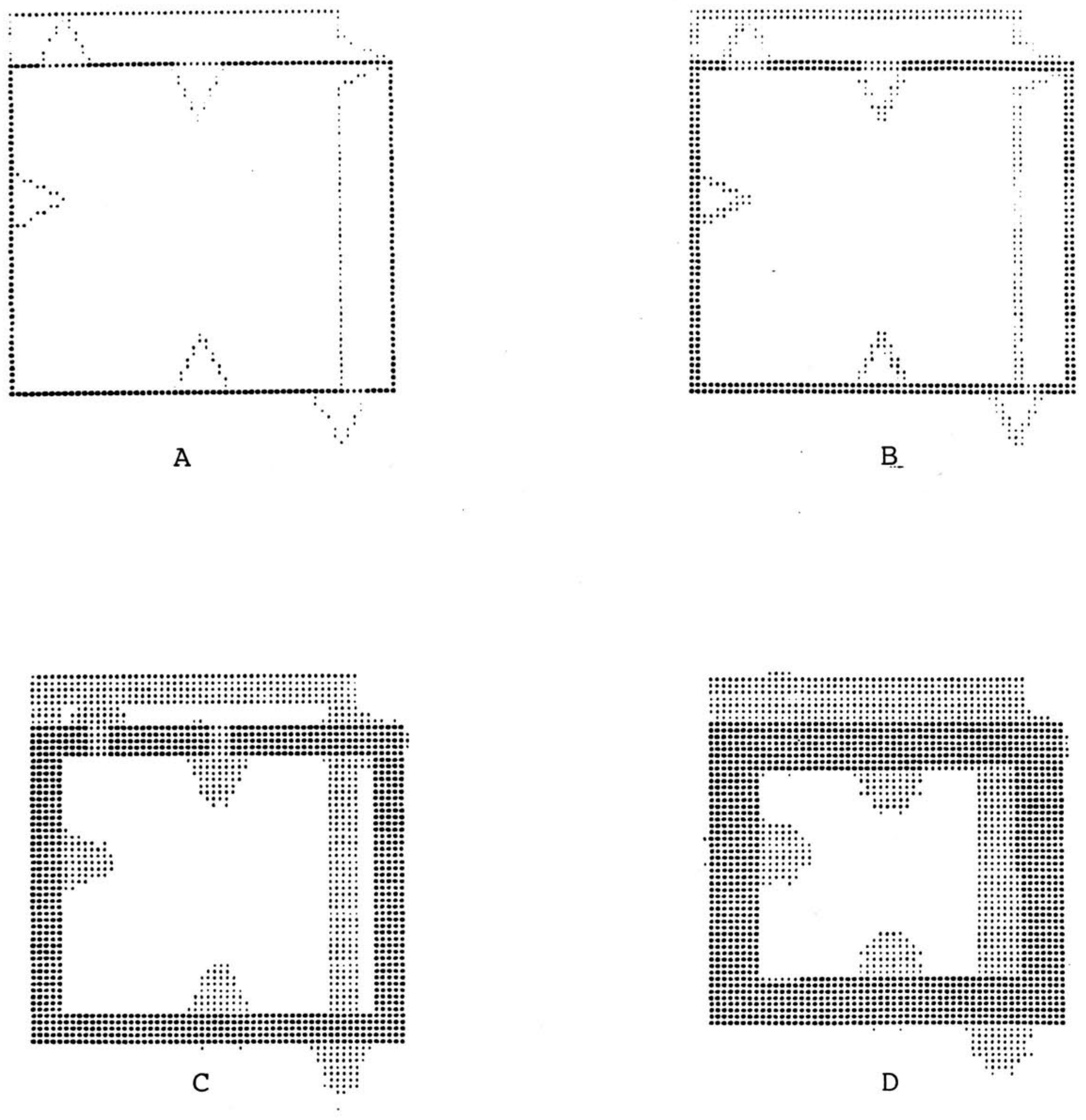


Figure 4.7 a
Degree 1 blocking boundaries.

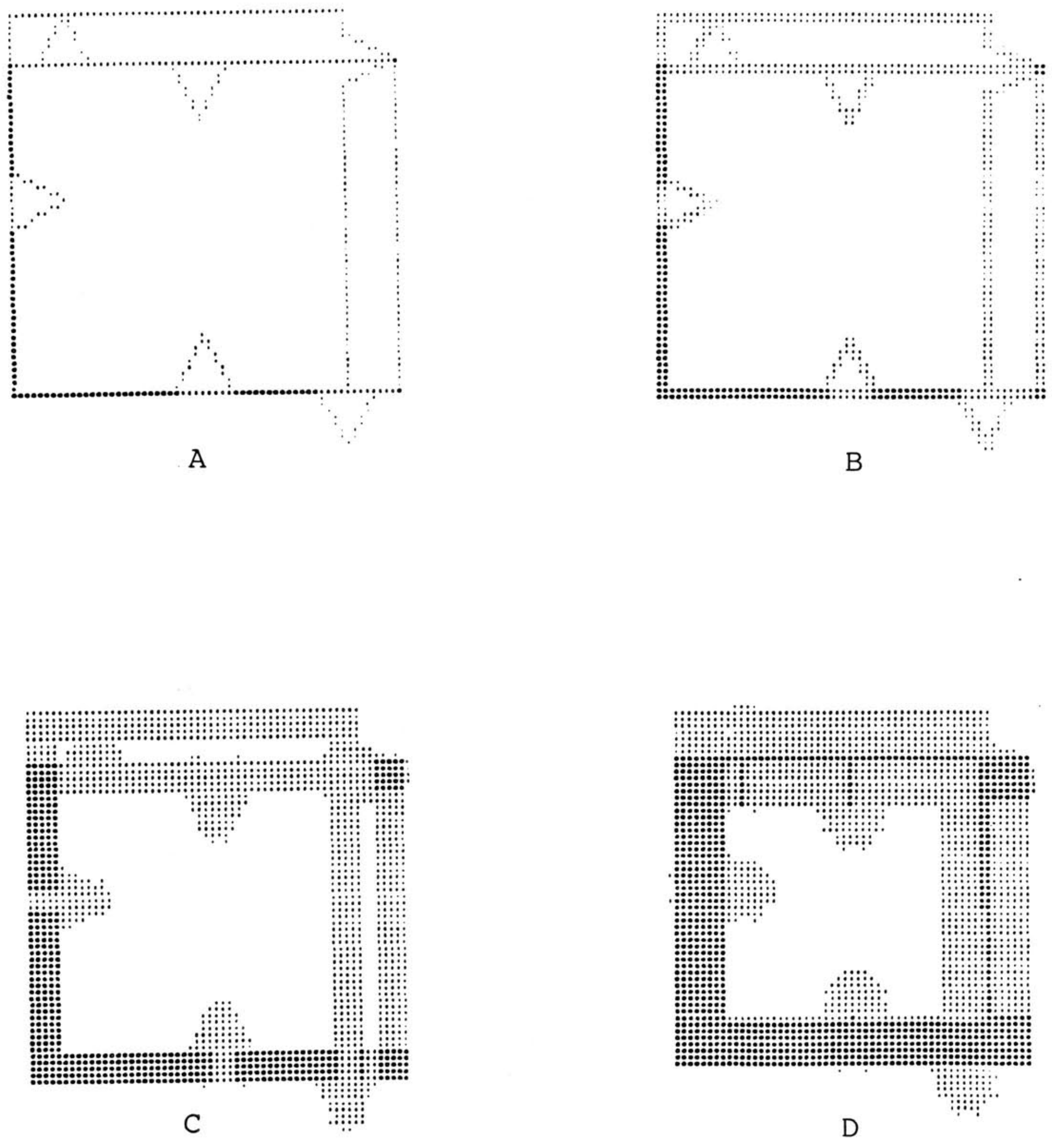


Figure 4.7 b
Degree 2 blocking boundaries.

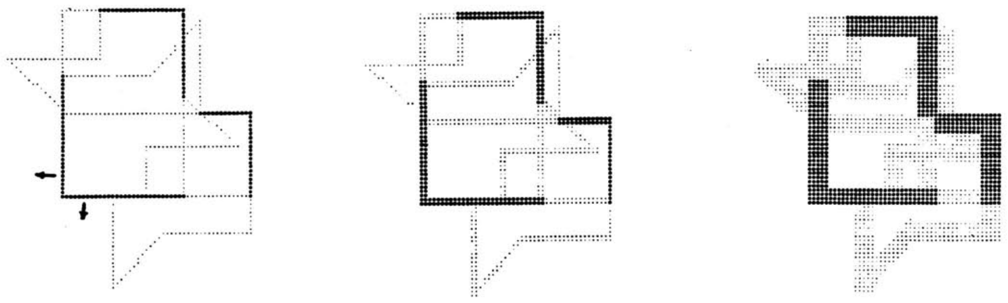


Figure 4.8a

Degree 1 blocking boundaries.

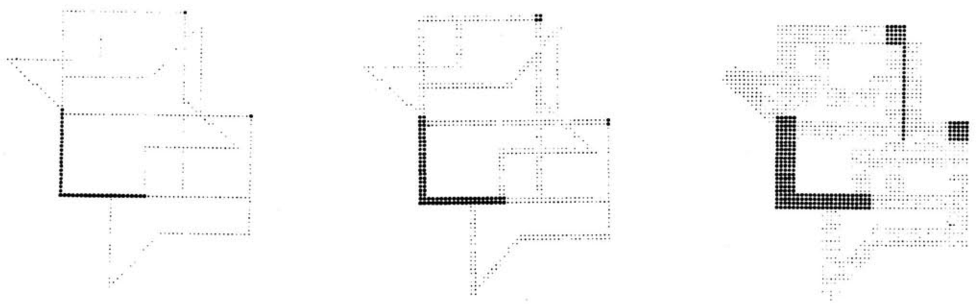


Figure 4.8b

Degree 2 blocking boundaries.

view their occurrence as serendipitous insofar as they represent "zones of confusion" about the location of vertices. If we employ a sensor that measures surface normals we will experience problems obtaining measurements at vertices because the surface normal is undefined there. It is therefore advisable to avoid such regions when taking measurements, and in this work we allow the blocking boundary regions to advise us of their existence.

If we object to the existence of artificial blocking boundary regions, we need only change equation 4.3 to read

```
IF MGRID(I,J) = 1 THEN
      MGRID(I,J) = MGRID(I,J) + MAX(0,WGRID(I,J)-1)

  ELSE IF MGRID(I,J) < WGRID(I,J)-1 THEN
      MGRID(I,J) = WGRID(I,J) - 1
```

In this way we add in the blocking boundary, replace values, or do nothing, whichever is appropriate.

There are two situations where we would object to the existence of such "artificial" blocking boundaries. The first is when we use a sensor that actually measures discontinuities (edges), where it is appropriate to use measured vertex information. The second is a bit more subtle. All of the object shapes used here have had "hard" edges in the sense that vertices represent true corners. In situations where we approximate smoothly curved surfaces by many straight segments we would observe artificial blocking boundaries at the connections of some of the segments. There is nothing to be gained by allowing the existence of artificial boundaries in this case because the true faces are smooth. In this case we would be ignoring valid measurement areas on the object.

4.3.5 Path Generation

Once we have obtained the complete MII we search for paths using the same rules as presented in Chapter 3. We try to find paths from outside the union boundary which terminate within the intersection area without passing through a blocking area. If no such paths exist, we seek paths which intersect single blocking boundaries. We note as an aside that we assume that we can distinguish between separate blocking boundary regions as a path is traversed. As it has been developed here, the grid contains no explicit information of this sort per se, and some rule can be formulated such as: If, when you travel along a path, the grid values change from blocking boundary to "clear" and back to blocking boundary, then you have intersected another blocking boundary.

A complete analysis may now be made as described in Chapter 3. We determine the intersection area(s) and determine the "windows" between the blocking boundary areas through which we establish paths. For what follows, however, we simply search for paths that terminate at the centroid of the intersection area, providing the centroid is contained within the area. If there is more than one area, we search for paths that terminate at the centroid of each. The primary reason for this restriction is that we are interested in very fast performance. We pay for this by ignoring potentially highly diagnostic paths that may not be directed at the centroid.

We therefore proceed as follows. We determine the intersection areas and their centroids (see Appendix 4 for an explanation of how they may be determined simultaneously). From each centroid we imagine a ray that extends to the grid boundary. Starting at a ray orientation of 0° (along the x-axis), we sweep the ray through 360° while continuously monitoring when the ray is obstructed by a blocking boundary region of the strength of interest. We then bisect the included angles between obstructing zones and use the results as paths (see figure 4.9).

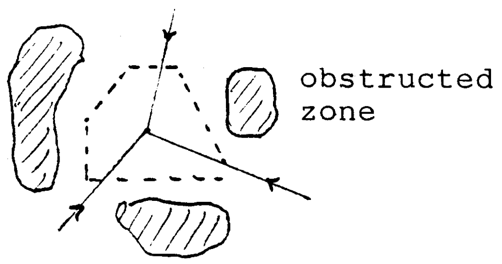


Figure 4.9. Paths to Centroid of IA through Unobstructed Zones

If no paths are found, we shift our attention to higher strength blocking boundary zones as before. We will eventually obtain a path or paths that will be the most diagnostic straight paths to the centroid along which we can direct the sensor.

Figure 4.10 shows some familiar examples with paths generated in this way. Note that, as the error increases, there are in general fewer paths found.

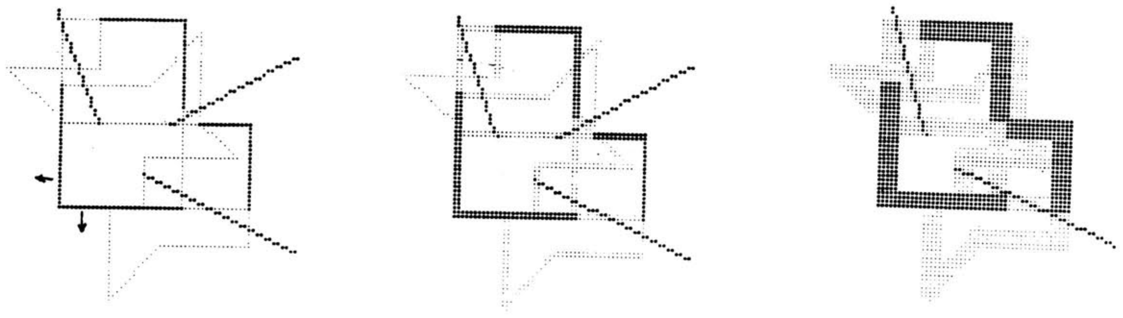


Figure 4.10 Fewer paths as error increases.

The situation is no different when we investigate multiple interpretations of multiple objects. Figure 4.11 shows two objects, the left of which is assumed to be the "true" one. Figure 4.12 shows assumed data, the resultant MII, and contending paths for the next measurement.

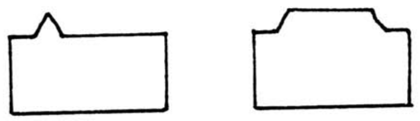


Figure 4.11 Object set.

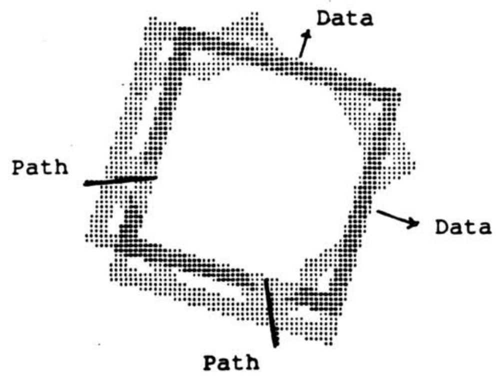


Figure 4.12 Data, MII, and available paths for the objects of figure 4.11.

4.3.6 Simulation Results

Simulations of a random sensing strategy and the strategy presented in this work were performed for comparison purposes. The simple object on the left in figure 4.11 was used for the first two simulations. Further simulations were run assuming the object set of figure 4.11; the "true" object was again the left object of the figure. Note that for all cases an error disk of 10% of the effective object diameter was used.

EXPERIMENT I. This experiment simulated the performance of a random strategy on the localization of a single object. Sensing rays were directed toward the object center from random directions, and the position of the point of intersection of the ray with the outermost object boundary, along with the surface normal at that point, were fed to the recognition and localization routines. In all cases the true object position was the same.

The results of running 469 recognition and localization cases are shown in Table 4.1. The numbers in each box represent the percentage of the total number of cases run that a particular number of probes were required for recognition and localization. For experiment I, for example, more than ten probes were required for thirty percent of the cases. Note that this percentage would decrease for more complicated objects.

Table 4.1 Simulation results

	Required Number of Measurements									
	2	3	4	5	6	7	8	9	10	>10
Exp I	12	13	10	9	6	5	4	6	4	30
Exp II	5	29	34	32	-	-	-	-	-	-
Exp III	4	19	25	52	-	-	-	-	-	-

EXPERIMENT II. This experiment simulated the performance of the strategy presented in chapters 3 and 4. Again, the single object on the left in figure 4.11 was the assumed object, and the number of probes required for localization was recorded for each case. The experiment was conducted by first choosing a sense ray toward the object center from a random direction and obtaining the sense data as in experiment I. The second sense point was obtained by choosing a sense ray orthogonal to the first. If the resulting sensed normals were parallel or anti-parallel, another ray was chosen with random direction. (This actually never happens with the object chosen. This was a test in the simulator to insure that the number of interpretations was constrained.) The data was then presented to the recognition and localization routines. The strategy routines then determined what the next sense path should be by selecting at random from among the most highly diagnostic family of paths.

The results of 115 cases are also shown in Table 4.1. In this experiment the effect of the strategy is clear. There were no cases for which more than five probes were required. Intuitively, this is correct, since the strategy essentially hunts for a measurement from the triangular structure on the object. At worst, only five measurements are necessary: two to constrain the interpretations and three probes to determine where the triangular structure is.

Note that a fourth probe is not required because the previous three will have eliminated all but the final interpretation.

EXPERIMENT III. This experiment simulated the performance of the strategy for recgonition and localization assuming the object set of figure 4.11. The "true" object was the same as in the previous experiments. This experiment was conducted as experiment II was except for the number of assumed objects.

The results of 116 cases are shown in Table 4.1. The performance of the strategy is still quite good when compared to the random strategy. The maximum number of probes was still five, but this maximum was required in more of the cases. Intuitively, we would expect that more probes would be necessary because of the extra interpretations for the extra object at each stage. We would not expect the same maximum number of probes with the addition of extra objects, however. Further simulations of more complex objects would probably show a small increase. The reason for this is that we would expect extra probes to be made to prune the extra branches of the interpretation tree due to the extra objects. The fact that the maximum number of required probes was the same in experiments I and II points out the efficiency with which the strategy effectively prunes the interpretation tree.

CHAPTER 5 Hardware Demonstration System

The two-dimensional case was implemented in hardware in order to investigate the kinds of problems that can arise and the issues that affect successful performance. The setup consisted of a PDP11/34 computer, a master/slave manipulator, an object jig and associated objects, and a 2-D tactile sensor and associated electronics. Objects were thick (3/4 inch) "2-D" plywood shapes that were mounted on the jig and could be spun about their centers and clamped into position.

Measurements were taken when the arm brought the sensor tip into lateral contact with an object's edge. All movements around the object took place along a predefined "safe circle" within which the object was guaranteed to be located. This circle served as the delineator for the manipulator workspace in the plane of the object. All contact measurements were initiated from the circle.

The first measurement was obtained by a lateral movement of the sensor to the left from the 0° point on the circle, and the object was always positioned so that contact was guaranteed. Once contact was made, the sensor was removed from contact, recalibrated at zero load, and brought back into contact until a force threshold was met. The recalibration helped to combat angular measurement errors introduced by differential drift in the two instrumentation channels.

The second measurement was made downward from the 90° point on the safe circle. (Again, contact was always guaranteed). At this point the algorithms for strategic path generation were invoked. All suitable paths were directed toward the centroid(s) of the intersection area(s). The system selected the path whose associated path ray intersected the safe circle closest to the sensor position. Figure 5.1 on the next page shows the system at various stages during the recognition process for a single modeled object.

For this case object recognition was not important (the object was known). The system was required to determine position and orientation only. Notice that, as one would expect, the system probed for the triangular feature on the object. For each measurement after the first, the monitor shows the multi-interpretation image with blocking boundaries and paths. The blocking boundaries shown are of a degree such that openings are available and paths can be found.

Figure 5.2 shows a similar sequence for the case of multiple modeled objects. Here the system had to determine which of two objects was in the environment as well as its position and orientation. Note that, as expected, the behavior changed with the inclusion of a new object model.

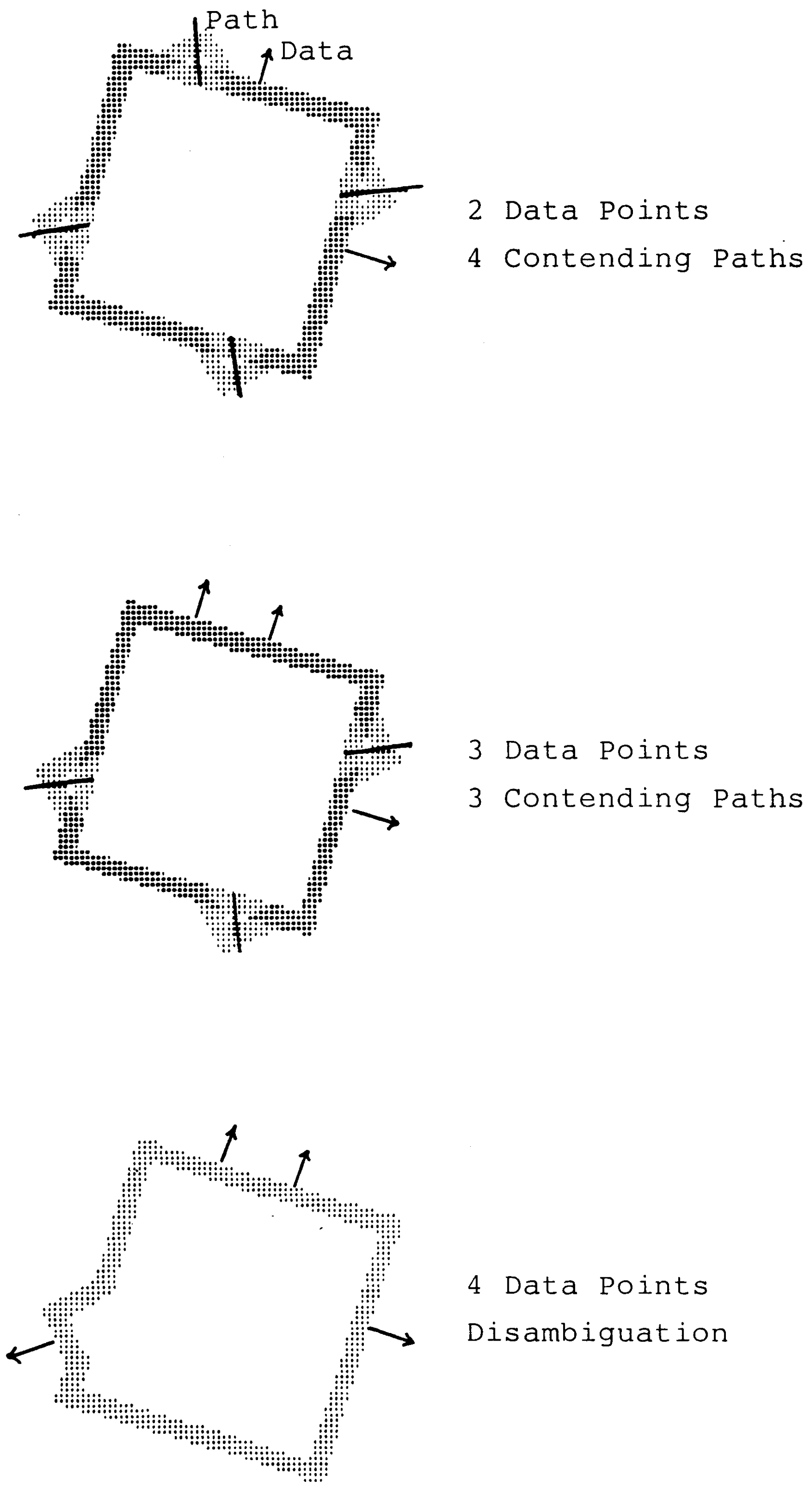


Figure 5.1
Measurement Sequence for a Single Object.

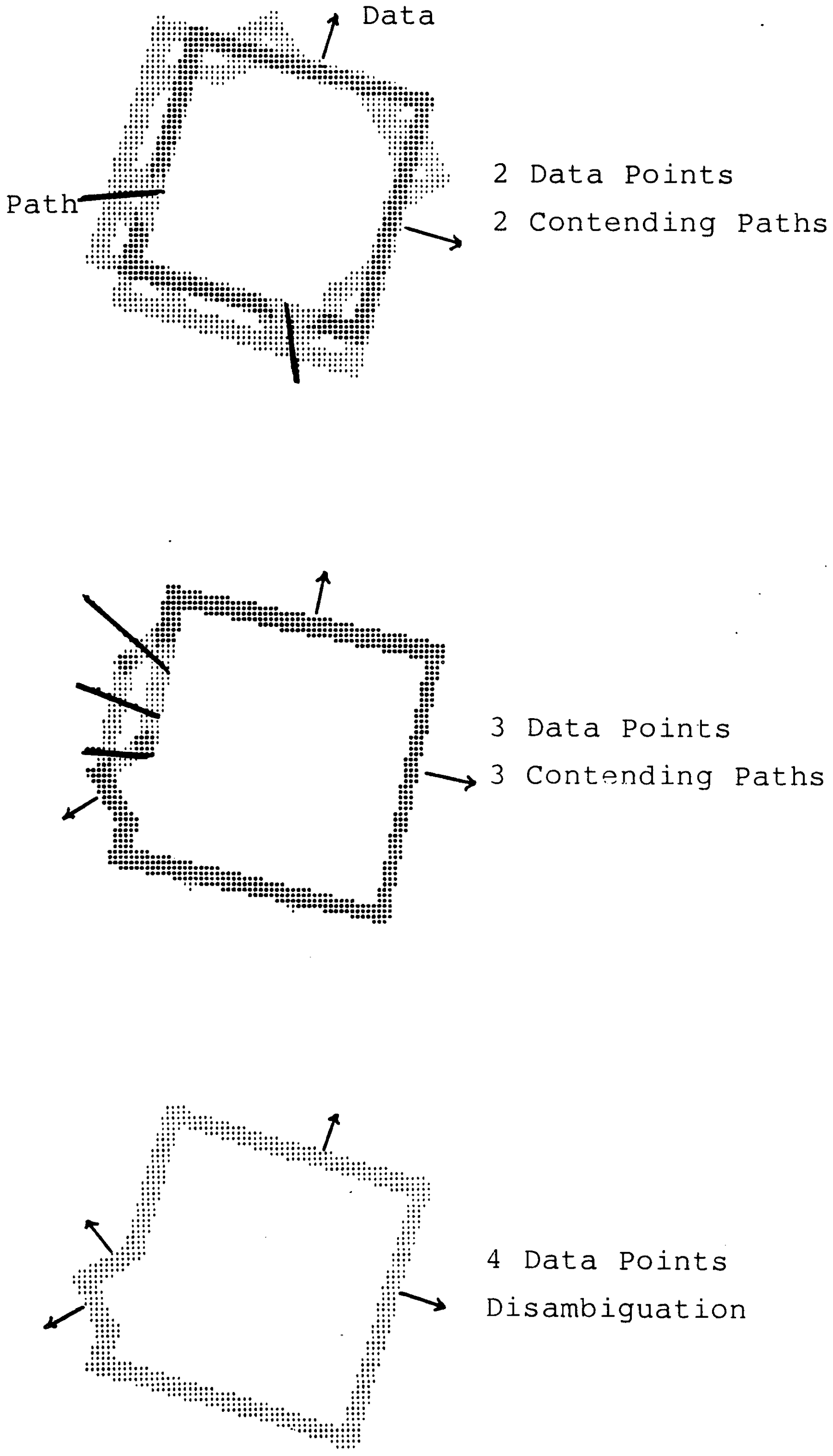


Figure 5.2

Measurement Sequence for two Modeled Objects.

5.1 E2 Master/Slave Manipulator

The E2 master/slave manipulator is a six degree-of-freedom (DOF), bilateral, force reflecting manipulator system, developed during the early 1950's primarily for the purpose of manipulating hazardous substances in nuclear hot rooms. The two arms communicate electronically and the connection can be electronically severed to allow for independent computer control of either arm.

Control of the E2 is somewhat complicated by the separation between the fourth and fifth degrees of freedom (Figure 5.3). This offset is about 1.5 inches. Such offsets are typically designed into human-controlled manipulators to help prevent gimbal lock during manipulation. However, it prevents us from finding a closed-form solution to the inverse kinematics problem, which means an iterative solution is required. (Computer-controlled manipulators are usually designed so that the last three axes of rotation intersect at a point. This can simplify considerably the inverse kinematic solution because the 6 DOF problem can be broken down into two 3 DOF problems [39].) A fairly fast approximate solution to the inverse kinematics of the E2 appears in [40], where the arm is first analysed as if the offset were not present, and a correction is made to the resulting solution.

The links of the arm are controlled by cables that are wound on capstans mounted on servo motors at the base of the

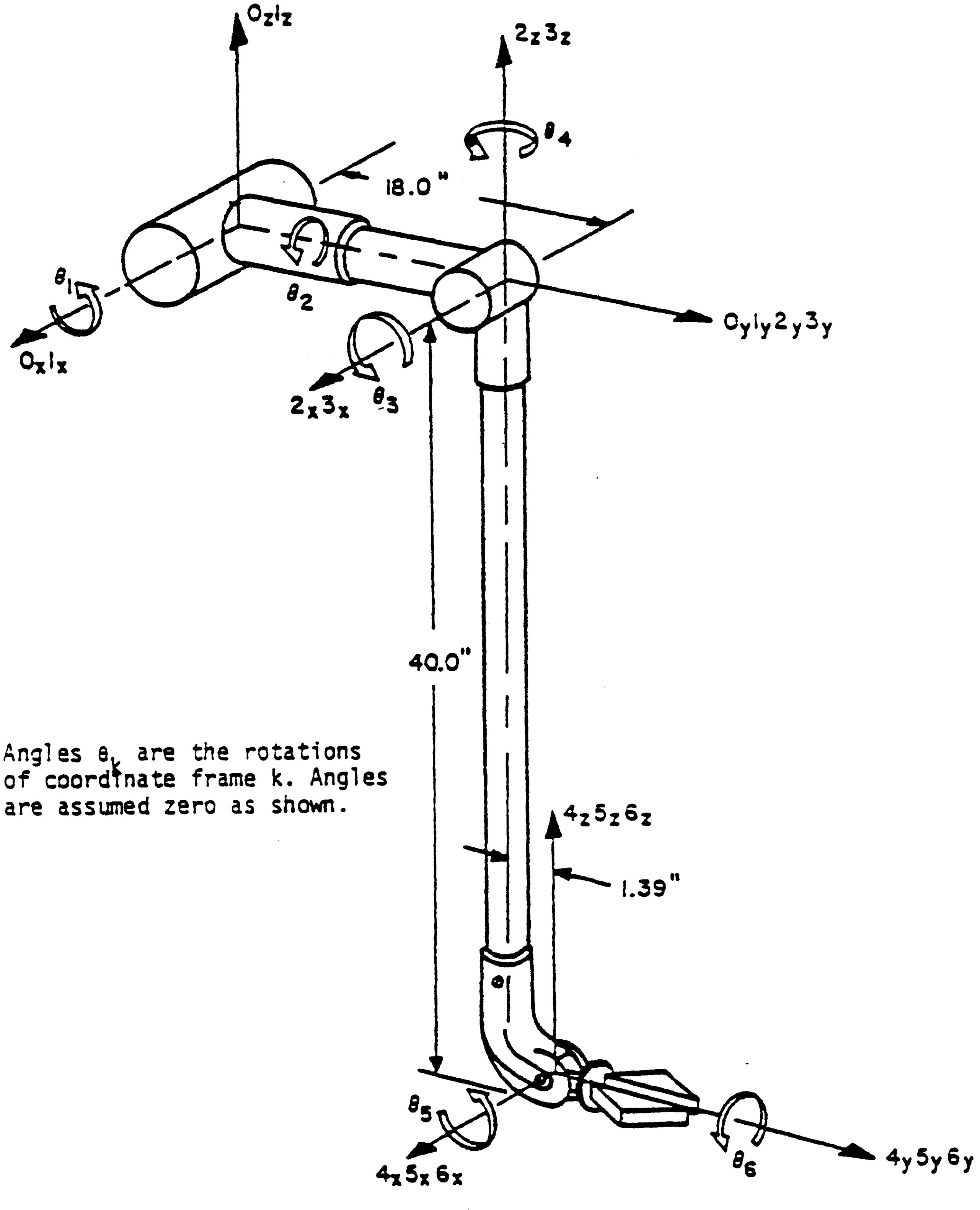


Figure 5.3Argonne E-2 Manipulator (From Yoerger)

arm. These cables act like springs when loaded, and the arm has a very high compliance as a result. When all joints are "locked" there is considerable hand deflection (about 1 inch) when reasonably small loads (less than 1 lb) are imposed. There is also considerable backlash in the wrist gearing which allows for around 10° of play in the hand. The combination of these results in considerable error in measured position (about 1 inch) when the arm is under load, and error in measured normal (about 10°) in addition to that of the sensor itself.

Further errors arise when the arm is in use for any length of time greater than about 10 minutes. Joint angles are inferred from servo motor angular positions which are measured by potentiometers mounted on the motors. The motors become quite hot during use and heat the potentiometers to the point that there are significant measurement errors. Although I did not analyse these errors in any depth, they were significant enough to cause recognition failures, even with allowable modeled position error of twenty percent. The solution to this problem was to shut the system off and allow it to cool before attempting another trial.

All of these errors taken together presented a formidable challenge to the methods. The fact that the methods were at all successful points out their power. We recognize, of course, that more complex objects with feature scales on the order of or smaller than the accumulated error are likely to cause recognition failure or indecision, no matter how many measurements are made.

There is one interesting aside to be made here. The arm compliance actually prevented recognition failures when the sensor was placed on protruding vertices. If the arm were stiff, the sensor would likely measure a surface normal that matched with no object face, and there would not likely be any correct (or even feasible) interpretation of the resulting data. Because of the compliance of the E2, the sensor flops to one side or the other of the vertex and obtains a correct measurement. Recent work by Grimson and Lozano-Perez [63] develops a method for handling erroneous data. Their methods may prove useful in situations such as those described above.

5.2 Tactile Sensor

Figure 5.4 shows a schematic of the sensor. It is grasped by the manipulator and the tip is brought into lateral contact with an object edge. Orthogonal stresses induced at the base of the "finger" are measured by strain gages, and the resulting signals are amplified and sent to the A/D converter for use by the computer. The bearing on the fingertip helps prevent lateral friction forces that would corrupt surface normal measurements. A complete description and design analysis of the sensor is given in Appendix 3.

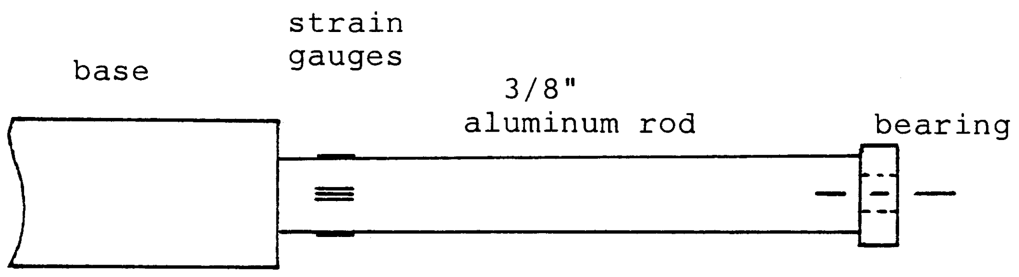


Figure 5.4 Schematic Diagram of the 2-D Tactile Sensor

Since the finger is very stiff, there are essentially no errors due to bending. The major error associated with the sensor is due to the amplifier electronics. The signal gain is 2000 for each of the two channels, and fairly rapid and large output drift can take place due to thermal transients. This problem has been isolated and is caused by a poor selection of bridge resistances (Appendix 4). High tolerance, low temperature-coefficient resistors should be used in bridge circuits.

The drift led to sizeable angular error measurements, and its effect was minimized by recalibration. After contact was first made, the sensor was pulled slightly away from the object, calibrated under no-load conditions, and again brought into object contact. Electronic interference noise played a small role, and the resulting overall error "cone" was about 5°.

The compliance of the arm also caused sensor error. The sensor was designed for lateral contact between the fingertip bearing and an object edge. The axis of the finger was supposed to be perpendicular to the object plane, but the twisting of the arm under load resulted in skewed contact that corrupted the normal measurement. This problem was particularly troublesome in cases where large vertical surface normal components were present.

5.3 Computer and Software Structure

All computations were performed on a DEC PDP11/34 system operating under RSX11-M. This system is fairly fast, even by today's standards, but available memory is severely restricted. The entire machine has less than 256K bytes of available RAM, a portion of which is occupied by the operating system. As a result, a great deal of task overlaying was employed, with a concurrent loss in speed. The computer communicated with the manipulator and sensor through a multi-channel Analogic AD5400 A/D, D/A converter. All routines were written in FORTRAN IV. (This made for some very interesting code). Figure 5.5 shows the basic structure of the software.

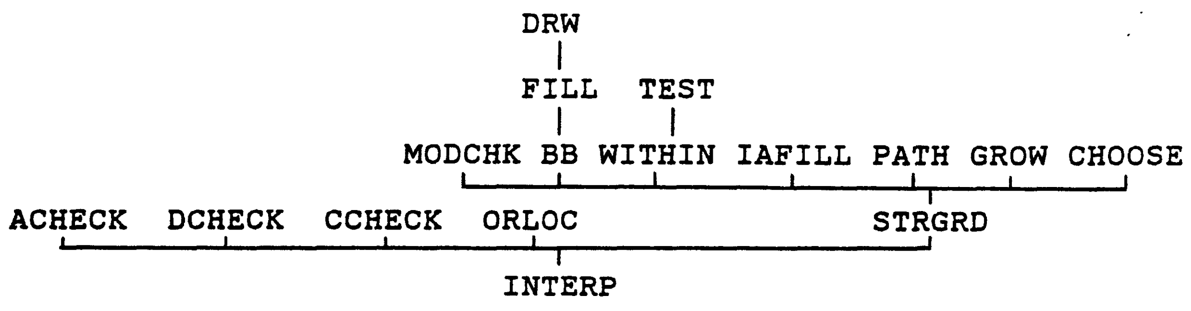


Figure 5.5 Software Structure.

A brief description of the routines follows. The reader interested in the actual code should contact the MIT Man/Machine Systems Laboratory for listings.

INTRP	- Root routine that established data bases and orchestrated subroutine calls
ACHECK	- Performed angle pruning
CCHECK	- Performed direction pruning
DCHECK	- Performed distance pruning
ORLOC	- Determined orientation and position for each feasible interpretation
MODCHK	- Performed model checking
STRGRD	- Performed grid analysis (setup, scaling, etc)
BB	- Determined blocking boundaries on the grid
GROW	- Grew object faces on grid
DRW	- Line drawing routine for raster vector draws on the grid

FILL	-	Filled grown object faces on the grid
WITHIN	-	Checked if a given point was contained by all interpretations simultaneously
TEST	-	Checked if a given point was on a finite line segment
IAFILL	-	Traced an IA boundary and filled with an identifying number. Also determined IA centroids.
PATH	-	Determined paths from the centroids of all IA's for a given blocking boundary strength
CHOOSE	-	Determined path whose intersection with the safe circle was closest to the hand position

5.4 Performance

Typical completion times for object localization only were on the order of one to two minutes. For tasks involving both object recognition and localization the times were on the order of a half minute more. Most of this time (about 70%) was spent determining blocking boundaries and searching for paths.

In most cases the object was correctly identified and/or located. Incorrect conclusions were occasionally made but were the result of heating problems in the manipulator hardware (section 5.1). Allowing the system cool and trying again led to correct performance.

The problems that arose during the implementation of this demonstration system had little to do with the strategic method itself. Indeed, correction of the backlash and compliance problems in the arm and thermal problems of the sensor should result in almost flawless performance. As it was, the method proved quite robust in spite of the hardware limitations.

CHAPTER 6 Generalization to Three Dimensions

We are interested ultimately in the practical implementation of our method in systems capable of operating in unstructured environments. The viability of our method therefore hinges upon our ability to incorporate it into systems capable of dealing with real objects. In truly unstructured environments, therefore, we must expect to handle three dimensional objects with a full six degrees of freedom of motion. We discuss in this chapter the implications of this generalization in terms of its effect on the method and its effects on the computational requirements. Full development of the 3-D case is considered to be outside the scope of this thesis and should be the focus of further work.

6.1 Strategy in 3 Dimensions

The strategy conceptually generalizes fairly easily to general three dimensional objects. The intersection area generalizes to an intersection volume, the union area to the union volume, and the blocking boundaries become three dimensional. Conceptually, the strategy remains essentially unchanged; in order to obtain a distinguishing measurement, we search for paths that originate outside the union volume and terminate within the intersection volume that do not pass through blocking boundaries. We can further rate, in order of decreasing probability of distinguishing measurements, those moves that pass through object boundaries once and through a single blocking boundary, as in the 2-D case.

Note that the method naturally handles 3-D objects with through-holes. Consider the case where only a single interpretation for each of two objects is feasible, and that the objects differ only in that one of them has a through-hole. The only paths for which a distinguishing measurement is guaranteed are those paths that pass into the hole of the appropriate interpretation. Our method would find such paths.

Although it might seem natural to assume that the method is not as powerful in three dimensions as in two (a common sentiment expressed by those newly introduced to these ideas), a moment's reflection reveals that the addition of the third dimension introduces the possibility for more discriminating features. There is therefore no reason to expect a reduction in the efficacy of the method when one deals with three dimensional objects.

6.2 Generating 3-D Volumes

The computational burden is certainly increased when one considers the general three dimensional case. In fact, it can be stated with some assurance that determining the intersection and union boundaries, even for fairly simple planar 3-D objects, can not now be performed analytically in real time with present techniques. It is almost imperative that these operations be performed volumetrically on a three dimensional grid, if we assume arbitrary object orientation. We present the basic techniques for boundary growing and determining blocking regions, intersection volumes and union volumes without developing detailed equations.

6.2.1 Boundary Growing

We discuss boundary growing associated with measurement error for the case of 3-D, planar objects only. This is performed in much the same way as for 2-D, except that instead of simply widening and lengthening the 2-D boundary, we must thicken and expand the 3-D boundary. This is shown conceptually in Figure 6.1, where the boundary is thickened by ε_t , (which is determined in the same way as for the 2-D case) after the nominal planar boundary face is widened by tracing an error disk of diameter ε_t around the face boundary.

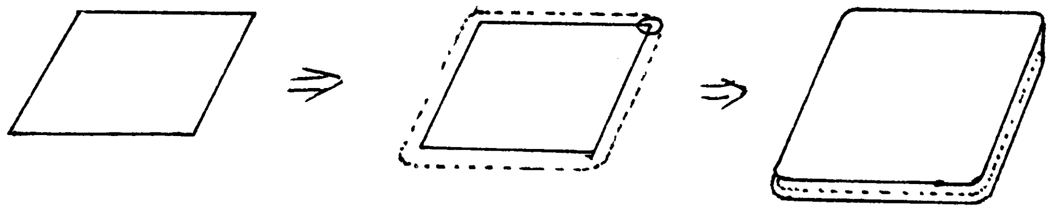


Figure 6.1 Boundary Face Growing.

We can simplify this process in a conservative manner by simply lengthening each boundary segment of the face by $\varepsilon_t/\sqrt{2}$ and insisting on geometric similarity, and then thickening by defining two identically shaped faces placed $\varepsilon_t/2$ on either side of the nominal face. We then typically have 2m vertices that define the face volume, where m = number of vertices defining the original face. (Note that vertices at concavities may disappear after boundary growth.)

6.2.2 Blocking Boundaries

Since we work with a grid, it is probably computationally fastest to develop blocking boundaries in a way similar to the 2-D case. We would build them up on a working grid and add them to the master grid. An algorithm must be developed to fill the boundaries defined by the approximately 2m vertices of 6.2.1. There is a variety of ways to approach the development of this algorithm, but I suspect that the fastest way would be the following. First, fill in all defining faces (all 2-D faces that define the boundary volume) in the grid. This requires an algorithm for drawing a plane in a grid that is similar to that required for drawing a line. When this is complete the result is, conceptually, an "empty box" in the grid that must be filled. Simply scan through the grid by, for example, fixing x and y and scanning through z, filling in the scan line between pairs of non-zero values (which are the defining face values). Of course, scanning through an entire 70x70x70 grid would be extremely time consuming, and one should define a cube on the grid which just barely encloses the boundary region, and scan through that.

6.2.3 Determining Intersection and Union Volumes

Since the 3-D multi-interpretation image has been (presumably) scaled to fit within the grid, it is not necessary to determine the union boundary explicitly. The extreme elements of the grid lie outside it and any of them may serve as a path origin. The intersection volume must be determined, however.

In the 2-D case we took advantage of a simple method for tracing out the intersection area boundary and identifying the area by "filling it in" with an identifying number (Appendix 4). There is, unfortunately no analogous method in three dimensions. We offer two approaches to this problem. The first is to repeatedly perform the 2-D method by fixing one of the coordinates, say z, and looking for 2-D intersection areas in the x-y slices of the image obtained at each z. There will have to be some method of linking adjacent slices so that emerging intersection volumes are correctly identified and labeled from one slice to the next.

An alternative method is conceptually simpler to implement. We build an image of each interpretation by itself on the working grid during boundary growing. Once we find a single zero-valued element within the interpretation, we use it to start the process of filling in the volume with a unique value in the manner explained in the previous paragraph (filling in 2-D slices). We then add a 1 to the main grid whenever the corresponding elements in the working grid with the unique value are found, and start the process over again with the next interpretation. After doing this for all interpretations, the main grid will typically have elements with values 0 through n, where n = number of interpretations. The intersection volume exists wherever the values are equal to n.

The relative desirability of the two methods depends upon what sorts of discoveries can be made to speed up the manipulations. It is not clear to me which method is faster. The first method requires 2-D slice projections of all the interpretations simultaneously, whereas the second requires that a large number of element checks be made for each interpretation. I suspect that the two methods present similar computational burdens and I would therefore opt for the latter because of its conceptual simplicity.

6.3 3-D Objects Using 2-D Techniques

If we restrict ourselves to the case of planar faced objects resting on a support plane, we can take advantage of the 2-D techniques developed in this work. Gaston and

Lozano-Perez [35] were among the first to realize this in their work on planar object recognition. We consider making all contact measurements at a constant height (or heights) above the plane (figure 6-2). For each height, each of the stable poses of each object will be associated with a 2-D planar object model. The 2-D analysis developed in this work may then be applied.

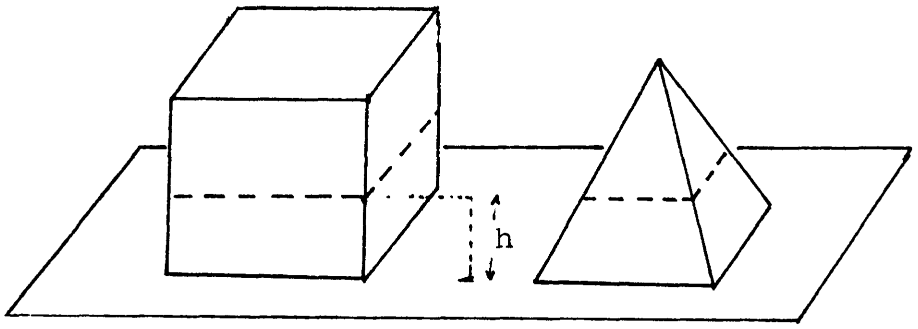


Figure 6.2 Measurements at constant height above support plane.

It is interesting to note that, if we use this method, we can and should employ an extra constraint table during the recognition phase. Associated with each modeled 2-D face is a measured component of the surface normal out of the plane for the corresponding 3-D face . A constraint table should be utilized that has the range of possible out-of-plane normal components for each face.

6.4 Computational Issues

The most obvious result of generalizing to 3-D is the increase in the number of elements in the grid. A 70x70x70 grid has 343000 elements, as compared with 4900 for the 2-D case. Although memory is relatively inexpensive and available now, accessing even a moderate fraction of the full grid takes time. This has a deleterious effect on blocking boundary region and union volume development, since for these cases full grid additions take place from the working grid to the main grid. The advent of parallel processing hardware may considerably brighten the outlook in this regard [41,42,43].

The search for paths is also negatively affected by the generalization to 3-D. Even if we limit ourselves to paths which terminate at the centroid of the intersection volume (if it is located within the volume) the search space is considerably larger. In the 2-D case we step through 360° in 1° increments. To do the analogous search in spherical coordinates requires 180x360 = 64800 checks, although we probably can reduce this considerably, since we would not make 1° steps at large azimuth angles. This may still leave us with an unacceptable number of checks with present hardware.

There might be strong heuristic rules one can apply to combat these problems, although we do not discuss them in this work. Future work should concentrate on developing such rules. Considerable attention should also be paid to investigating the theoretical foundations of the idea of a luminescent intersection volume as discussed in section 3.3.2. It may be that ray tracing is the only reasonable solution, but it is worth investigating other potential techniques.

CHAPTER 7 Summary, Conclusion and Recommendations

The objective of this thesis was to develop a strategic framework for scheduling tactile sensor moves during the process of automated tactile object recognition and localization. Such a framework is useful whenever surface data is sparse, measurements are expensive to make or to analyse, and it is known that the inspected object is one of a number of modeled objects. It is an important step towards the realization of truly autonomous machines capable of performing tasks in unstructured environments.

We have shown the strategic framework presented in this work to be successful for two-dimensional planar objects and conceptually easily generalized to three dimensional planar objects. Specifically, given that we have obtained enough data to limit the number of feasible interpretations of such data, we can find paths along which to direct the sensor that guarantee a distinguishing measurement, if such paths exist. We have determined that some paths are clearly better than others from the viewpoint of the probability of recognition and localization, and have a reliable methodology for finding them, even in the presence of measurement error. Finally, we can determine what measurements not to make, such measurements being a waste of time. Our ability to determine and rate paths gives us a rational basis with which to make measurement planning decisions.

Our method drastically reduces the number of measurements necessary for recognition and localization when compared to a random strategy. A very important result of this work is that this reduction is due to the natural and automatic selection of features that discriminate between interpretations in the multi-interpretation image. We do not have to concern ourselves with predefining them, which is convenient, not only because they change as object models are added to or deleted from the object set, but also because they depend on the state of information of the system.

The method is conceptually applicable to any general three dimensional objects. We are limited only in our ability to describe and efficiently manipulate such objects computationally. Furthermore, the nature of the measurements bears only indirectly on the method. Future implementations could very well use tactile arrays capable of providing local surface properties such as surface curvature. It reasonable to generalize the representation and recognition method of Grimson and Lozano-Perez [27] to include such information, provided sufficiently accurate measurements can be made.

The sensor paths obtained in the hardware implementation were all directed to the intersection area centroid. For the object set that we used, this was acceptable, but for general objects, many highly diagnostic paths would be missed (Consider also that the centroid of an area is not always located within that area). An algorithm should be developed to efficiently search for highly diagnostic paths throughout the intersection area. An exhaustive search at each point in the intersection area is clearly inefficient, but perhaps heuristic rules or a clever algorithm can be found that capture most of the paths that an exhaustive search would provide.

One of the most important areas for further research is the generalization to three dimensions. We discuss the associated issues and suggestions for further work in chapter six and will not repeat them here.

We have always assumed the existence of a single unoccluded object in the environment. The general environment will probably contain many objects, some of which might overlap (consider the bin-picking problem [14]). Lozano-Perez and Grimson investigate the recognition problem for such occluded objects [63], and an important problem for future work would be to determine the applicability of our strategic method in that domain and rework it as necessary.

We were careful in theorem 2 to state the assumption that we consider only those paths that intersect the boundary for each interpretation once, and pass through a single blocking boundary. There are certainly other path examples that violate this assumption. Consider figure 7.1.

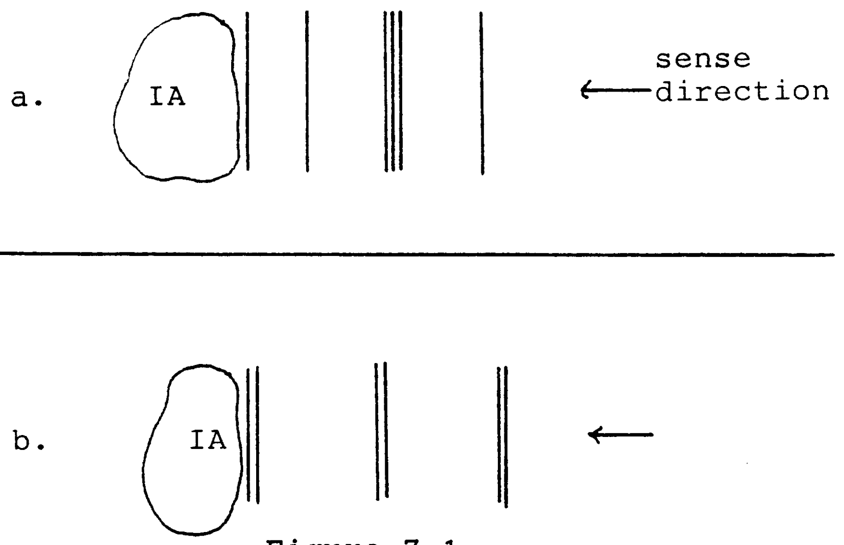


Figure 7.1

There are six interpretations and we show two possible path examples. We assume each interpretation to be equally likely to simplify matters. If we choose the path of figure 7.1a, we have a 0.5 chance of obtaining a distinguishing measurement, and the same chance of being left with three interpretations. In figure 7.1b, we are guaranteed of being left with two interpretations. It is impossible to say which path is "better" because it is not clear what "better" means. If we always maximize our chances of obtaining distinguishing measurements, then we choose the path of figure 7.1a. If, however, we are interested in some optimal performance which minimizes the expected number of measurements, then an analysis might show that path b is the appropriate choice. Work should be done to develop this analysis. It may be that results from the field of operations research would be applicable.

Another area for further research is motivated by figure 7.2.

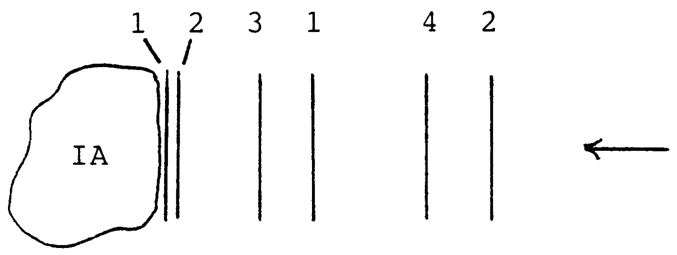


Figure 7.2

In this case there are four interpretations and the boundary fragments are identified with the interpretations to which they belong. We are guaranteed a distinguishing measurement even though the path crosses a blocking boundary. It is extremely difficult, however, to obtain an example of such a situation. To date, I have not found one. The questions to be answered include:

1. Can a situation of this type occur?

2. If it can occur, does it matter? That is, even if it occurs, are there other paths that pass through the boundaries without passing through a blocking boundary?

These are questions of global topology and are likely to be answered only after a great deal of theoretical development.

REFERENCES

[1] Harmon, L., "Automated Tactile Sensing," Intl. J. Robotics Research, Vol.1, No.2, 1982.

[2] Bejczy, A.K., "Manipulator Control using Smart Sensors," Electro/79 Convention, N.Y.,N.Y., 1979.

[3] Bejczy, A.K., "Algorithmic Formulation of Control Problems in Manipulation," Paper WEHM3-1, Proc. 1975 IEEE Intl. Conf. on Cybernetics and Society.

[4] Harmon, L. "Touch Sensing Technology: A Review," SME Technical Report MSR80-03, 1980.

[5] Grimson, W.E.L., "Sensing Strategies for Disambiguating among Multiple Objects in Known Poses," MIT Artificial Intelligence Laboratory, AI Memo 855, 1985.

[6] Hillis, W., "Active Touch Sensing," MIT Artificial Intelligence Laboratory, AI Memo 629, 1981.

[7] Gibson, J.J., "Observations on Active Touch," Psychological Review, Vol.69, No.6, 1962.

[8] Gordon, G., ed., Active Touch - The Mechanisms of Recognition of Objects by Manipulation: A Multidisciplinary Approach, Pergamon Press, 1978.

[9] Fu, K.S., ed., Applications of Pattern Recognition, CRC Press, 1982.

[10] Gonzalez, R.C., and Thomason, M., Syntactic Pattern Recogntion: An Introduction, Addison-Wesley, 1978.

[11] Fu, K.S., Sequential Methods in Pattern Recognition and Machine Learning, Academic Press, 1968.

[12] Devijver, P.A. and Kittler, J., Pattern Recognition: A Statistical Approach, Prentice-Hall Intl., 1982.

[13] Grimson, W.E.L., From Images to Surfaces: A Computational Study of the Human Early Visual System, MIT Press, Cambridge, Mass., 1981.

[14] Horn, B.K.P. and Ikeuchi, K., "Picking Parts out of a Bin," MIT Artificial Intelligence Laboratory, AI Memo 746, 1983.

[15] Cernuschi-Frias, B. and Cooper, D., "3-D Space Location and Orientation Estimation of Lambertian Spheres and Cylinders from a Single 2-D Image by Fitting Lines and Ellipses to Thresholded Data," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No.4, 1984.

[16] Faugeras, O., "New Steps Toward a Flexible 3-D Vision System for Robotics," 2nd Intl. Symp. on Robotics Research, MIT Press, 1985.

[17] Faugeras, O., Hebert, M., Ponce, J., "Object Representation, Identification, and Positioning from Range Data," First Intl. Symp. on Robotics Research, MIT Press, 1984.

[18] Ayache, N., Faugeras, O., Faverjon, B., Toscani, G., "Matching Depth Maps Obtained by Passive Stereo," 1985 IEEE Intl. Conf. on Robotics and Automation, 1985.

[19] Faugeras, O., Hebert, M., Pauchon, E., "Segmentation of Range Data into Planar and Quadratic Patches," 1983 IEEE Computer Vision and Pattern Recognition, 1983.

[20] Faugeras, O. and Pauchon, E., "Measuring the Shape of 3-D Objects," 1983 IEEE Computer Vision and Pattern Recognition, 1983.

[21] Bolle, R. and Cooper, D., "Bayesian Recognition of Local 3-D Shape by Approximating Image Intensity Functions with Quadratic Polynomials," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No.4, 1984.

[22] Hu, M., "Visual Pattern Recognition by Moment Invariants," IRE Transactions on Information Theory, Vol.IT-8, 1962.

[23] Sadjadi, F. and Hall, E., "Three-Dimensional Moment Invariants," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-2, No.2, 1980.

[24] Oshima, M. and Shirai, Y., "Object Recognition Using Three-Dimensional Information," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No.4, 1983.

[25] Ballard, D. and Sabbah, D., "Viewer Independent Shape Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No.6, 1983.

[26] Wallace, T., Mitchell, O., Fukunaga, K., "Three-Dimensional Shape Analysis Using Local Shape Descriptors," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-3, No.3, 1981.

[27] Grimson, W.E.L. and Lozano-Perez, T., "Model-Based Recognition and Localization from Sparse Range or Tactile Data," MIT Artificial Intelligence Laboratory, AI Memo 738, 1983.

[28] Requicha, A., "Representations for Rigid Solids: Theory, Methods, and Systems," Computer Surveys, Vol.12, 1980.

[29] Wilson, D., "An Exploratory Study of Complexity in Axiomatic Design," MIT Mechanical Engineering PhD thesis, 1981.

[30] Jackins, C. and Tanimoto, S., "Oct-Trees and their use in Representing 3-D Objects," Computer Graphics and Image Processing, Vol.14, 1980.

[31] Nevatia, R. and Binford, T., "Structured Descriptions of Complex Objects," Proceedings of the 3rd IJCAI, Stanford, Cal., 1973.

[32] Marr, D. and Vaina, L., "Representation and Recognition of the Movements of Shapes," MIT Artificial Intelligence Laboratory, AI Memo 597, 1980.

[33] Faux, I. and Pratt, M., Computational Geometry for Design and Manufacture, Ellis Horwood Limited, 1980.

[34] Henderson, T., "Efficient 3-D Object Representation for Industrial Vision Systems," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No.6, 1983.

[35] Gaston, P. and Lozano-Perez, T., "Tactile Recognition and Localization Using Object Models: The Case of Polyhedra on the Plane," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No.3, 1984.

[36] Hillis, D., "Active Touch Sensing," MIT Artificial Intelligence Laboratory, AI Memo 629, 1981.

[37] Albus, J., et al, "Hierarchical Control for Sensory Interactive Robots," 11th International Symposium on Industrial Robots, 1981.

[38] Persoon, E. and Fu, K.S., "Shape Discrimination Using Fourier Descriptors," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-7, 1977.

[39] Paul, R., Robot Manipulators: Mathematics, Programming, and Control, MIT Press, Cambridge, Mass., 1981.

[40] Yoerger, D., "Supervisory Control of Underwater Telemanipulators: Design and Experiment," MIT Mechanical Engineering PhD thesis, 1982.

[41] Ben-Ari, M., Principles of Concurrent Programming, Prentice/Hall International, Englewood Cliffs, N.J., 1982.

[42] Buzbe, B. and Sharp, D., "Perspectives on Supercomputing," Science, Feb.,1985.

[43] Foster, C., Content Addressable Parallel Processors, Van Nostrand Reinhold Company, New York, N.Y., 1976.

[44] Ikeuchi, K., et al, "Picking Up an Object from a Pile of Objects," MIT Artificial Intelligence Laboratory, AI Memo 726, 1983.

[45] Horn, B.K.P., and Ikeuchi, K., "Picking Parts out of a Bin," MIT Artificial Intelligence Laboratory, AI Memo 746, 1983.

[46] Considine, D. and Ross, S., Handbook of Applied Instrumentation, McGraw-Hill Book Company, New York, N.Y., 1964.

[47] Holman, J., Experimental Methods for Engineers, McGraw-Hill Book Company, N.Y., N.Y., 1971.

[48] Beckwith, T., Buck, N., Maragoni, R., Mechanical Measurements, Addison-Wesley, Reading, Mass., 1982.

[49] Popov, E., Mechanics of Materials, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1976.

[50] Togai, M., Wang, P., Rebman, J., "Design Criteria and Recognition Schemes for an Arrayed Touch Sensor," IEEE CH2008-1/84/0000/0385, 1984.

[51] Schneiter, J. and Sheridan, T., "An Optical Tactile Sensor for Manipulators," Robotics and Computer Integrated Manufacturing, Vol.1, No.1, 1984.

[52] Schneiter, J., "An Optical Tactile Sensor for Robots," MIT Mechanical Engineering Masters Thesis, 1982.

[53] Rebman, J. and Trull, M., "A Robust Tactile Sensor for Robotic Applications," Proc. 1983 Intl. Computers in Engineering Conference and Exhibit, ASME, 1983.

[54] Briot, M., "The Utilization of an 'Artificial Skin' Sensor for the Identification of Solid Objects," Ninth ISIR, Washington, D.C., 1979.

[55] Briot, M., Renaud, M., Stojilkovic, "An Approach to Spatial Pattern Recognition of Solid Objects," IEEE Transactions on Systems, Man, and Cybernetics, SMC-8, 1978.

[56] Marik, V., "Algorithms of Complex Tactile Information Processing," Seventh Intl. Joint Conf. on Artificial Intelligence, 1981.

[57] Kinoshita, G., Aida, S., Mori, M., "A Pattern Classification by Dynamic Tactile Sense Information Processing," Pattern Recognition 7, 1975.

[58] Overton, K. and Williams, T., "Tactile Sensation for Robots," Seventh Intl. Joint Conf. on Artificial Intelligence, 1981.

[59] Page, C., Pugh, A., Heginbotham, W., "Novel Techniques for Tactile Sensing in a Three-Dimensional Environment," Sixth ISIR, University of Nottingham, 1976.

[60] Takeda, S., "Study of Artificial Tactile Sensors for Shape Recognition: Algorithm for Tactile Data Input," Fourth ISIR, 1974.

[61] Albus, J., et al, "Hierarchical Control for Sensory Interactive Robots," Eleventh Intl. Symposium on Industrial Robots, 1981.

[62] Lozano-Perez, T., "Spatial Planning with Polyhedral Models," MIT Electrical Engineering PhD Thesis, 1980.

[63] Grimson, W. and Lozano-Perez, T., "Recognition and Localization of Overlapping Parts from Sparse Data," MIT Artificial Intelligence Laboratory, MIT AI Memo 841, 1985.

APPENDIX 1 Representation, Recognition and Localization Equations

In this appendix we present the equations for generating the feasible interpretations of the data that satisfy the distance, angle and direction constraints. We also develop the equations for computing the translation component of the transformation between model and global coordinates.

A1.1 Constraints

The constraint checks presented in Chapter 2 can be implemented in a very fast table lookup, where constraint tables are generated off-line, based on model vertices. The following implementation equations are modified versions of those found Grimson & Lozano-Perez [27]. The reader interested in full developmental details is referred to that report.

A1.1.1 Distance Constraint

For an object j with f_j faces, we can construct an f_j by f_j distance constraint table, the entries of which represent the minimum and maximum distances between any points for all pairings of faces. For some pair of faces (i,k) where $i=k$, we can construct the table D_j (off-line) such that the maximum distance between faces i and k is located at $D_j[\max(i,k), \min(i,k)]$ and the minimum distance

is located at $D_j[(\min(i,k), \max(i,k)]$.
If $i=k$, we simply store the largest distance of the face on the diagnonal element (i,i) of D_j, since the smallest distance is necessarily zero. Implementing the distance constraint for some measured inter-sensed-point distance d is performed by realizing that the set of all pairs of faces (i,k) on object j consistent with d is given by

$$\{(i,k) \mid D_j(\min(i,k),\max(i,k) \leqslant d \leqslant D_j(\max(i,k),\min(i,k))\}$$

plus the pair (i,i) if $d \leqslant D_j(i,i)$.

For a new sensed point at level k, where the measured distance between this point and any other previously sensed point is d_{il}, the set of possible faces that can be assigned to the new point is given by

$$\bigcap_{l=1}^{k-1}\{i \mid D_j(\min(i,i_l),\max(i,i_l)) \leqslant d_{il} \leqslant D_j(\max(i,i_l),\min(i,i_l))\}$$

unioned with the set

$$\bigcap_{l=1}^{k-1}\{i_l \mid 0 \leqslant d_{il} \leqslant D_j(i_l,i_l)\}.$$

A1.1.2 Angle Constraint

This is a constraint on the allowable range of sensed normal directions consistent with the pairing of those normals to model faces. Figure A1-1 shows an example of the situation for the 2-D case. If $\underline{u}_1$ and $\underline{u}_2$ are measured normals, but we allow for some error on the measurements, then the "true" normals are located within the error cones depicted by dashed lines.

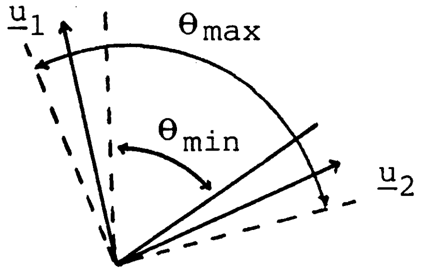


Figure A1-1 Angle constraint.

The minimum included angle is labeled θ_{min} and the maximum is labeled θ_{max}. If the nominal angle is close to π the measurement error could lead to the situation depicted in Figure A1-2.

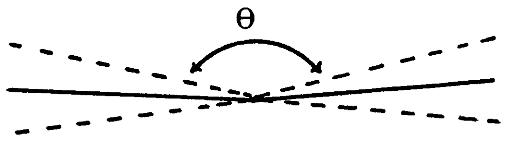


Figure A1-2 Nominal angle near π.

The minimum included angle is θ and the maximum is π. If the measured normals are such that the error cones overlap, the minimum included angle is 0.

We give a slightly different statement of angle pruning than that given in Grimson and Lozano-Perez [27]. In their development, a lower diagonal f_j by f_j table a_j is created for object j of f_j faces such that

$$a_j[\max(i,k),\min(i,k)] = \underline{V}_i \cdot \underline{V}_k$$

where i and k played roles identical to those in the distance constraint development, and $\underline{v}_i$ is the normal of model face i for object j. They then show that the set of all pairs of faces (i,k) on object j consistent with the known object normals, measured normals, and angular error is given by

$$\{(i,k) \mid \cos[\min(\pi,\gamma_{12}+\alpha_{12})] \leqslant a_j[\max(i,k),\min(i,k)] \leqslant \cos[\max(0,\gamma_{12}-\alpha_{12})]\}$$

where $\alpha_{12} = 2\phi$ and ϕ = error cone angle (if it is the same for all sensed normals), and γ_{12} is the included angle between measured normals. This implementation requires that min, max, sum and difference operations take place on measured data in return for a slight reduction in memory requirements for the table.

We can implement the check in a different way that requires a full f_j by f_j table that contains the angle ranges within it, so that we are not required to operate on the data. This trades table storage memory with computation time. With this check the set of all pairs of faces (i,k) consistent with the known object normals, measured normals and angular error is given by

$$\{(i,k) \mid A_j[\min(i,k),\max(i,k) \leqslant \underline{u}_1 \cdot \underline{u}_2 \leqslant A_j[\max(i,k),\min(i,k)]\}$$

where $\underline{u}_1$ and $\underline{u}_2$ are measured normals and

$$A_j[\min(i,k),\max(i,k)] = \cos[\min(\pi,\gamma_{12}+\alpha_{12})]$$
$$A_j[\max(i,k),\min(i,k)] = \cos[\max(0,\gamma_{12}-\alpha_{12})]$$
$$A_j(i,i) = \cos\phi \quad \text{(because the max defaults to 1).}$$

This check was used in the demonstration system because of its ameliorative effects on real-time computational requirements.

A1.1.3 Direction Constraint

We consider the situation depicted in Figure A1-3.

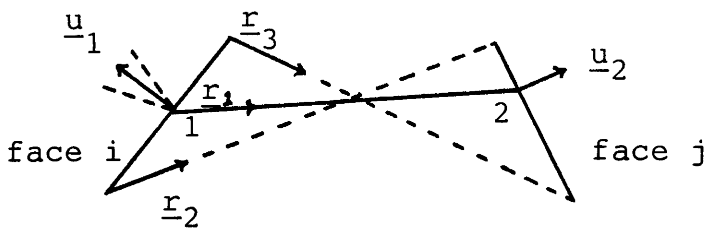
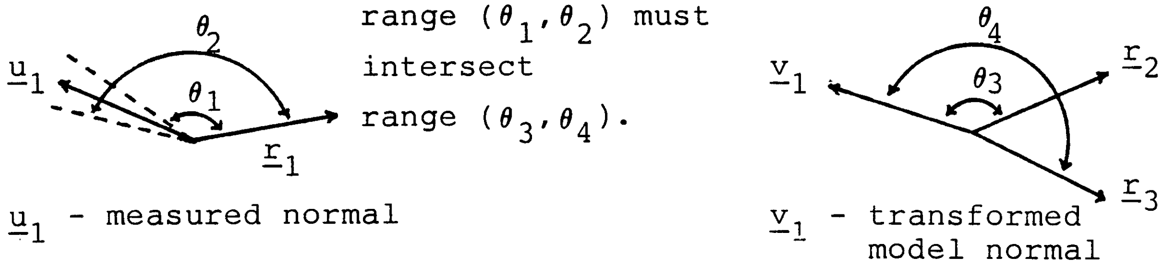


Figure A1-3 Direction constraint.

For any pairings of sensed point 1 with face *i* and point 2 with face *k* to be consistent with the direction constraint, the range of dot products of the vector from point 1 to point 2 ($\underline{r}_1$) with the vectors in the error cone range about $\underline{u}_1$ must intersect the range of dot products of the transformed model normal ($\underline{v}_1$) of face *i* with the range of possible vectors from face *i* to face *k* ($\underline{r}_1$ <--> $\underline{r}_3$). This can be seen more clearly in Figure A1-4.



θ_2
$\underline{u}_1$
θ_1
$\underline{r}_1$
range (θ_1, θ_2) must intersect range (θ_3, θ_4).
$\underline{u}_1$ - measured normal
θ_4
$\underline{v}_1$
θ_3
$\underline{r}_2$
$\underline{r}_3$
$\underline{v}_1$ - transformed model normal

Figure A1-4

This constraint may be implemented in a form similar to the distance and angle constraints. For object j with f_j faces, we create an f_j by f_j table C_j such that $C_j(i,k) = \text{range}\,[\underline{v}_1 \cdot \underline{r}_2, \underline{v}_1 \cdot \underline{r}_1]$. Then the set of all pairs of faces (i,k) on object j consistent with the measured ranges of surface normals is given by

$$\{(i,k) \mid \text{range}[\cos(\theta_{nom}-\varepsilon/2), \cos(\theta_{nom}+\varepsilon/2)] \quad C_j(i,k) \neq 0\}$$

where ε is the error cone angle.

We note that we can include the effects of angular measurement error and the range of direction vectors simultaneously in table C_j and consequently reduce the real-time computational requirements (as we did with the angle constraint). We isolate the left portion of Figure A1-3 in figure A1-5.

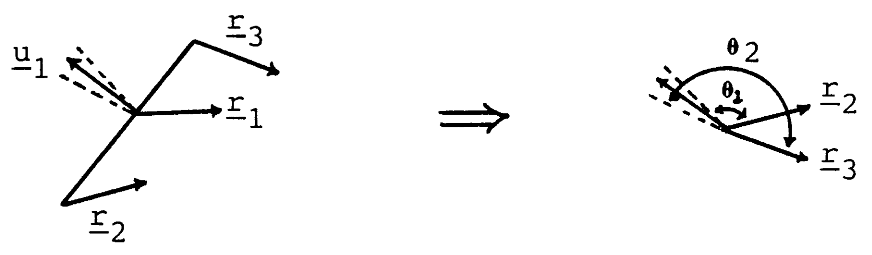


Figure A1-5

The check is simply that the measured angle between the normal and direction vectors be between θ_1 and θ_2 in figure A1-5. The previous method compares 2 smaller ranges of angles for intersection, whereas this alternative method checks a single, larger range. The first may be a bit stronger, but the second is faster.

If we describe the range of vectors between $\underline{r}_2$ and $\underline{r}_3$ in figure A1-5 as $\alpha_{ik} \pm \phi_{ik}$, then we let

$$C_j(\min(i,k),\max(i,k)) = \cos[\min(\pi, \alpha_{ik}+\phi_{ik})] \ ,$$
$$C_j(\max(i,k),\min(i,k)) = \cos[\max(0, \alpha_{ik}-\phi_{ik})]$$

and

$$C_j(i,i) = \cos(\pi/2 - \phi_{ik})$$

since the minimum defaults to zero. Then the set of all pairs (i,k) consistent with the direction constraint is given by

$$\{(i,k) \mid C_j(\min(i,k),\max(i,k)) \leqslant \underline{u}_l \cdot \underline{r}_{lm} \leqslant C_j(\max(i,k),\min(i,k))\}$$

for sensed points l and m. Since this constraint is not symmetric, the roles of l and m must be reversed and the check made again.

A1.2 <u>An Example of the Necessity of Model Checking</u>

It would seem that the constraints would weed out all but the correct interpretation, but this is sometimes not the case. It is instructive to view an example where an interpretation can only be discarded during model checking. In figure A1-6 we see two contact points and the associated measured normals. If we assign sensed point 2 to face C of the model, then careful scrutiny shows that we can assign point 1 to either face A or B if we only check constraints. The distance between point 2 and point 1 for either assignment is the same, the relationships between normals are the same, and the direction constraint is not violated in either interpretation.

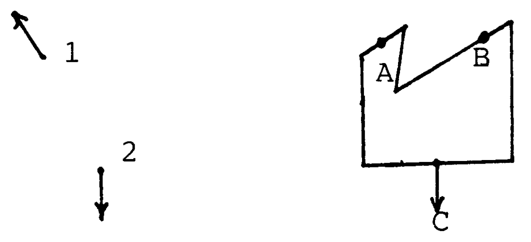


Figure A1-6 Model checking necessary.

A model check would show, however, that the assignment of point 1 to face B would not allow point 2 to remain on face C, and this interpretation would be pruned.

A1.3 Translation Component of Transform: 2-D

The angular component θ of the 2-D transform is simply found by averaging the difference between measured and modeled surface normals for each pairing in an interpretation. The translation component requires more computation and is developed here for completeness.

Any vector $\underline{V}_g$ in the global system is related to the same vector in model coordinates by the expression

$$\underline{V}_g = R\,\underline{V}_m + \underline{V}_0 \tag{1}$$

where

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

We wish to obtain $\underline{V}_0$. Note that the vector in the model system is described by

$$\underline{V}_m = R^{-1}[\underline{V}_o - \underline{V}_g]. \tag{2}$$

In the model system, any face f_i of object j is described by the set of vectors $\underline{V}$ given by

$$\{\underline{V} \mid \underline{V}\cdot\underline{n}_i = d_i\} \tag{3}$$

where $\underline{n}_i$ = unit surface normal for f_i

d_i = offset scalar for the line representing f_i.

The following expression may be obtained in the global system by using (2) in (3) to give

$$[R^{-1}(\underline{V}_{gi} - \underline{V}_0)]\cdot\underline{n}_i = d_i \qquad (4)$$

Each argument in the dot product may be transformed by R without altering the dot product relationship because R is an orthonormal rotation matrix. Therefore

$$(\underline{V}_{gi} - \underline{V}_0)\cdot R\underline{n}_i = d_i. \qquad (5)$$

Including another measurement for face f_k yields the following set of equations

$$\begin{aligned}(R\underline{n}_i)\cdot\underline{V}_0 &= (R\underline{n}_i)\cdot\underline{V}_{gi} - d_i \qquad (6)\\ (R\underline{n}_k)\cdot\underline{V}_0 &= (R\underline{n}_k)\cdot\underline{V}_{gk} - d_k\end{aligned}$$

Straightforward algebraic manipulation of (6) yields the following expression for $\underline{V}_0$:

$$\begin{aligned}[\hat{k}\cdot(R\underline{n}_i \times R\underline{n}_k)]\underline{V}_0 = (R\underline{n}_i\cdot\underline{V}_{gi}-d_i)(R\underline{n}_k \times \hat{k}) &+\\ (R\underline{n}_k\cdot\underline{V}_{gk}-d_k)(\hat{k} \times R\underline{n}_i) & \qquad (7)\end{aligned}$$

where $\hat{k}$ is the unit normal out of the paper. But $R\underline{n}_i$ and $R\underline{n}_k$ are the surface normals in global coordinates, and are the measured values if we assume perfect measurements. If the measured values of $\underline{n}_i$ and $\underline{n}_k$ are $\underline{n}'_i$ and $\underline{n}'_k$, respectively, and $\underline{V}_{gi}$ and $\underline{V}_{gk}$ are measured contact positions, then $\underline{V}_0$ is obtained from measurements by

$$[\hat{k}\cdot(\underline{n}'_i \times \underline{n}'_k)]\underline{V}_0 = (\underline{n}'_i\cdot\underline{V}_{gi}-d_i)(\underline{n}'_k \times \hat{k}) + (\underline{n}'_k\cdot\underline{V}_{gk}-d_k)(\hat{k} \times \underline{n}'_i) \qquad (8)$$

Note that, for numerical stability, $\underline{n}'_i$ and $\underline{n}'_k$ should be strongly orthogonal directions.

APPENDIX 2 Grid Equations

We wish to transform the multi-interpretation image from global to grid coordinates in a way that will allow the growing process to take place without extending beyond the grid bounds. Consider Figure A2-1.

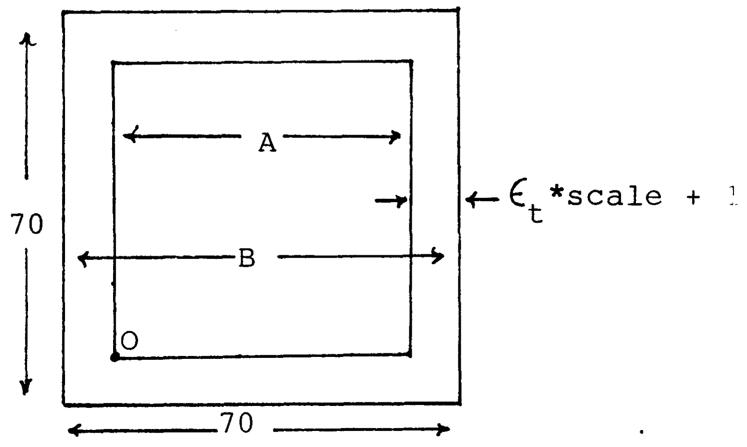


Figure A2-1 Map MII into A so that Growing Process Fills to B.

If we scale and translate the MII so that it just contacts perimeter A, then, since the worst-case growth of any face is $\varepsilon_t\sqrt{2}/2$, we conservatively select the growing region to be ε_t*scale+1, where scale is the scale factor for the mapping.

We proceed via an intermediate mapping. The first transformation translates the MII so that it contacts the x and y axes of an intermediate coordinate system with global scale, as in Figure A2-2.

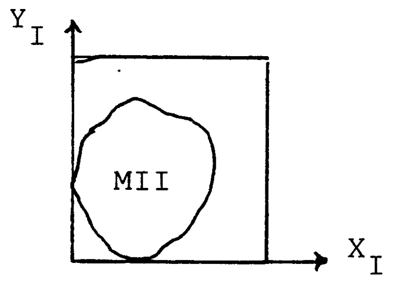


Figure A2-2. Intermediate System

We have $$\underline{X}_I = C\ \underline{X}_{gl}. \tag{A1}$$

If we denote by x_{glmin} the minimum value of the x-component of any vertex in the MII, and let y_{glmin} be the minimum y-component, then C is given by

$$C = \begin{bmatrix} 1 & 0 & -x_{glmin} \\ 0 & 1 & -y_{glmin} \\ 0 & 0 & 1 \end{bmatrix}$$

The next transformation scales and translates the intermediate MII to fill A. This is accomplished by enlarging so that the largest of the maximum x or y component differences of MII vertices fills to A. In other words, let the largest difference in x values of the vertices of the MII be denoted by ΔX_{max} and the largest difference in y values be ΔY_{max}. Then

$$\Delta = \max(\Delta X_{max}, \Delta Y_{max})$$

and scale so that $\Delta \rightarrow A$. This transformation is given by

$$\begin{bmatrix} A/\Delta & 0 & 0 \\ 0 & A/\Delta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We then translate by ε_t * scale + 1 in both X_I and Y_I to position the origin at point O in Figure A2-1. Then

$$\underline{X}_{gr} = D\ \underline{X}_I \qquad (A2)$$

where

$$D = \begin{bmatrix} 1 & 0 & \varepsilon_t{*}A/\Delta\ +1 \\ 0 & 1 & \varepsilon_t{*}A/\Delta\ +1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} A/\Delta & 0 & 0 \\ 0 & A/\Delta & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} A/\Delta & 0 & \varepsilon_t{*}A/\Delta\ +1 \\ 0 & A/\Delta & \varepsilon_t{*}A/\Delta\ +1 \\ 0 & 0 & 1 \end{bmatrix} \qquad (A3)$$

Combining (A1), (A2), (A3) gives

$$\underline{X}_{gr} = DC\underline{X}_{gl} = \begin{bmatrix} A/\Delta & 0 & A/\Delta(\varepsilon_t - x_{glmin}) + 1 \\ 0 & A/\Delta & A/\Delta(\varepsilon_t - y_{glmin}) + 1 \\ 0 & 0 & 1 \end{bmatrix} \underline{X}_{gl} \qquad (A4)$$

This is the forward mapping from global to grid coordinates. The reverse mapping from grid to global coordinates is simply $C^{-1}D^{-1}$ or

$$\underline{X}_{gl} = D^{-1}C^{-1}\underline{X}_{gr} = \begin{bmatrix} \Delta/A & 0 & -\varepsilon_t - \Delta/A + x_{glmin} \\ 0 & \Delta/A & -\varepsilon_t - \Delta/A + x_{glmin} \\ 0 & 0 & 1 \end{bmatrix} \underline{X}_{gr} \qquad (A5)$$

The value of A is determined from figure A2-1 as

$$A = B - 2(\varepsilon_t * A/\Delta + 1)$$

Simple manipulation yields

$$A = \Delta(B+2)/(\Delta+2\varepsilon_t).$$

APPENDIX 3 Sensor Design

The important requirements of the design were the following:

1. Detect contact
2. Measure contact force
3. Measure surface normal
4. Reject friction-induced corruption of surface normal measurement
5. Measurements obtained from "thick" 2-D objects

Many approaches were considered but the most natural one was selected and is described here.

The sensor is basically an instrumented cantilevered beam, on the end of which is mounted a ball bearing to help eliminate friction-induced surface normal measurement errors. The "ground" end of the sensor is held by the manipulator gripper. The sensor is shown in Figure A3-1.

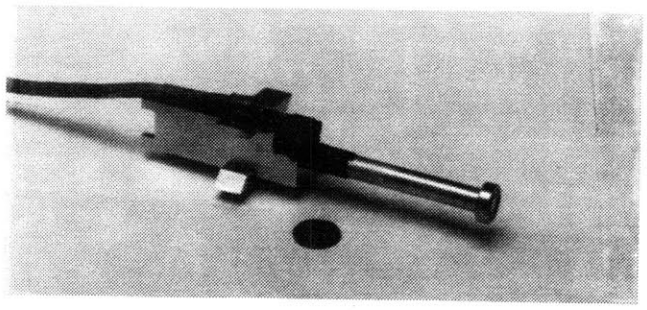


Figure A3-1 2-D Tactile Sensor.

The sensing element is a length of aluminum bar stock with two sets of strain gauges mounted at the base. A single BLH FAE-12-35-S13-ET metal strain gauge is mounted lengthwise on each quadrant of the bar. Gauges diametrically opposed on the bar form two legs of the classic Wheatstone Bridge Circuit [46,47]. This is shown schematically in Figure A3-2. There are three reasons for this arrangement: 1. Each bridge circuit is sensitive only to strains due to moments about its associated axis, with the result that the applied bending moment is obtained by a linear combination of the orthogonal measurements. 2. The strain gauges applied in this configuration are temperature compensated. 3. The bridge output is doubled, hence measurement sensitivity is increased.

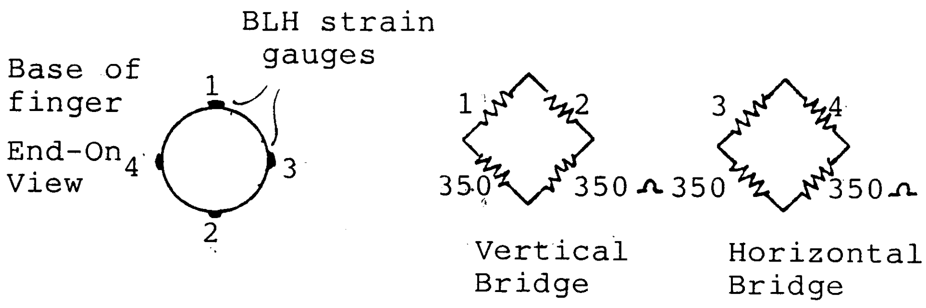


Figure A3-2 Gauge Arrangement and Bridge Circuits.

The guage style was chosen for its narrowness and relative ease of handling.

The bridge excitation and instrumentation is provided by Analog Devices 2B30 and 2B31 instrumentation modules.

These temperature compensated modules contain buffering circuitry, instrumentation amplifiers, and output signal conditioning filters. They are shown schematically in Figure A3-3. The output filters are 3-pole Butterworth, with break frequency selected at 25 Hz.

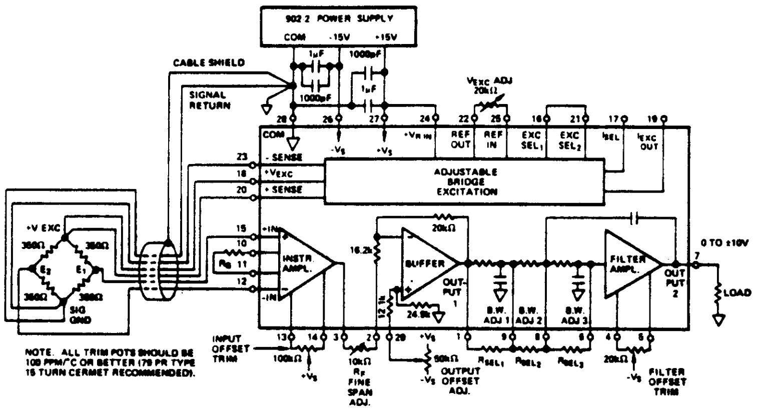


Figure A3-3 Instrumentation Modules.

The amplifiers are capable of gains to 2000, and such gains were chosen for this application.

Selection of the sensor diameter was based upon the following:

1. excitation voltage = 8 volts
2. max amplifier gain = 2000 v/v
3. strain gauge factor = 2
4. max applied lateral tip force = 4 lbf
5. sensor length = 3 inches
6. nominal maximum output voltage = 5 volts

Assume a voltage sensitive bridge with all initial resistances nominally equal, and with a single strain gauge. It can be shown that [48]

$$\frac{\Delta e_0}{e_i} = \frac{\Delta R_1/R}{4 + 2(\Delta R_1/R)}$$

where Δe_0 = change in output bridge voltage

e_i = bridge excitation voltage

ΔR_1 = change in resistance of the active gauge

R = nominal resistance of all legs of the bridge

Since $\varepsilon = \Delta R_1 / FR$, where ε is the strain of the gauge (and hence of the underlying material) and F is the gauge factor, then

$$\Delta e_0 = \frac{e_i F\epsilon}{4 + 2F\epsilon}$$

For e_i = 8 volts and $F = 2$,

$$\Delta e_0 = \frac{8(2)\epsilon}{4} = 4\epsilon \text{ volts.}$$

For ε = 1 microstrain, e_o = 4 microvolts.

For our situation the bridge factor is 2, hence e_0 = 8ε volts. A nominal maximum output of 5 volts resulting from a gain of 2000 implies e_0 = 5/2000 = 2500 μv. This would arise from a stain of $e_0/8$ or 313 μstrain. This is assumed caused by a bending moment of 4 lb x 3 inch = 12 inch-lbs.

The relationship between strain and bending moment is obtained from classical strength of materials [49] and is given by:

$$\epsilon = \frac{M\ y_{max}}{E\ I} = \frac{M(D/2)}{EI} \qquad \text{(A3-1)}$$

where	ε	= strain
	M	= applied bending moment
	E	= Young's modulus (10.E10 for aluminum)
	I	= $\pi D^4/64$ = moment of inertia for rod
	D	= Diameter of rod

We are interested in obtaining D, the nominal diameter of the sensor body. Simple algebraic manipulation of (A3-1), substituting the expression for I, gives

$$D = \sqrt[3]{\frac{32M}{E\epsilon\pi}}$$

Substituting nominal values gives D = 0.34 inches. A 0.375" diameter bar was therefore chosen. There is a slight reduction in sensitivity with the larger diameter but it is completely acceptable. The larger diameter also allows more space for the strain gauges and implies that the gauge width is a smaller fraction of the circumference. This helps reduce the effects of slight placement errors of the gauges.

We see that this design satisfies the criteria listed at the beginning of this appendix. The sensor transduces an applied tip force in directions orthogonal to the sensor axis only. An object's surface normal is transduced because a force is generated normal to the surface when the sensor is in put into contact with the object. Axial stresses are not transduced because the strain sensitive elements of each bridge cancel each other's effect. The tip bearing helps prevent off-normal components of the force vector from being generated.

APPENDIX 4 Finding The Intersection Area and Its Centroid

An inherent requirement of the strategic method is finding and identifying intersection areas on the grid. We wish to know how many there are and, for our specific implementation, where the centroids are located so that we may determine potential sensor paths in their directions. We could try to determine the areas and centroids analytically, which is difficult, or we could simply work with the multi-interpretation image on the grid after boundary growing. (There is a third method mentioned in Chapter 5).

The first part of the search is comprised of a simple raster scan of the grid array. Any zero-valued element is a potential member of the intersection area, except for the leading and trailing zeros on any scan line. A potential member must survive the check of whether it is contained by the nominal boundary of each and every interpretation simultaneously. The check is simply performed by considering a ray drawn from the point under consideration to the midpoint of every boundary face of each interpretation. If the point is within the interpretation, the dot product of the vector associated with the ray with the normal vector associated with the closest face of that interpretation intersected by that ray, must be positive or zero. If it is negative, the point is outside the interpretation. See Figure A4-1.

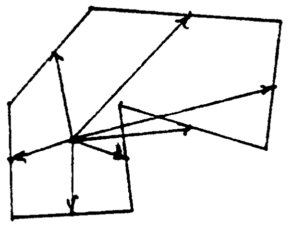
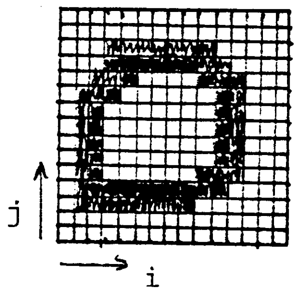


Figure A4-1 Checking membership in the IA.

We could, of course, perform this check for all zero-valued elements to determine the IA, but that would be computationally expensive and unnecessary. Instead, we proceed as follows. We choose the following because it does not require dynamic memory allocation in its implementation. We trace the boundaries of the IA's and fill their interiors with some identifying preassigned number, such as 100, 200, etc. We do this by scanning to the right from the point in the IA until we find a non-zero element. This is a boundary element. We fill the scan line to the left. We then trace the boundary counter-clockwise, filling scan lines to the left on upward movements and to the right on downward movements. See Figure A4-2 for visual support of what follows.

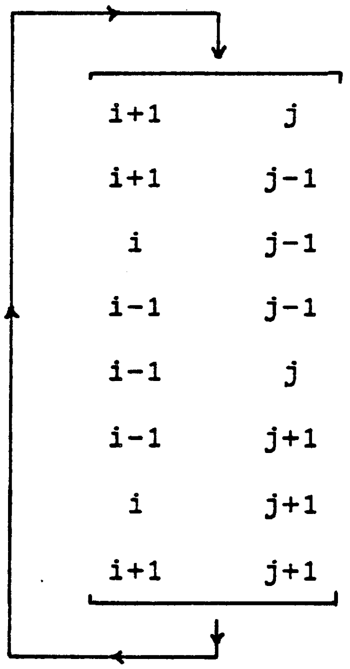


g(i,j) = element at position i,j.

Figure A4-2 IA boundary tracing.

We assume that after any move to a boundary element, the x and y indices are i and j, respectively. After our initial scan to the right and fill to the left, we find ourselves at location g(i,j). Since we know that elements immediately to the left are filled and we wish to move counter-clockwise from one boundary element to the next, we contemplate a move to g(i-1,j+1), or upwards and to the left. If the element there is non-zero and <100 we make the move. If it is zero or >100, however, we contemplate the next move clockwise, or to g(i,j+1), and so on, always contemplating the next clockwise move as long as we encounter an IA element. We will eventually find the next appropriate IA boundary element.

Note that this process is described by the following linked list.



If we denote the first element of the list as move 1, the second as move 2, etc., then we start with move 6 and step through the list as necessary. Each consecutive element of the list represents a contemplated move in a clockwise direction relative to the move represented by the previous element in the list.

We now note that we can usually predict where the ad-jacenct IA grid element will be located with respect to the grid element we've just made the move to. For instance, if we have just decided to make a 1 move directly to the right, we expect to find an IA grid element directly above the new site. In this case we would then contemplate a move not directly upward, but upward and to the right. This logic is also embodied in the linked list. We expect an IA grid element to occupy the location pointed at by the list element 6 entries away from the contemplated move entry for purely horizontal and vertical moves, and 5 entries away for diagonal moves. We therefore expect that the next move we'll have to make is 7 or 6 entries away for vertical and horizontal or diagonal moves, respectively. These expected moves are used as the next contemplated moves. If we do not find an IA grid element where we expect it, then we again contemplate moves by cycling consecutively through the list, starting from the expected move element.

If this logic is followed the intersection area will be filled and we are guaranteed to return to the boundary element with which we started, regardless of the shape of the intersection area. This method, while perhaps complicated-sounding, is very fast.

Once the intersection area is filled, we continue scanning the grid for other intersection areas as we did in the beginning. In this way all intersection areas will be identified.

In the implementation in this work, we need the location of the centroid of the intersection area. The x and y elements of the centroid are defined by

$$\bar{x} = \frac{\sum_i x_i}{n} \qquad \bar{y} = \frac{\sum_i y_i}{n} \qquad (A5-1)$$

where n = number of elements in IA

y_i = y position of the ith element

x_i = x position of the ith element

Again, we could blindly perform the calculations, but there is a much faster way of proceeding. Since we are tracing the boundary, we simply use boundary element locations for the calculations. It is fairly easy to show that the following is the appropriate equation:

$$\bar{x} = \frac{\sum_i x_i}{n_x} \qquad \bar{y} = \frac{\sum_j y_j}{n_y} \qquad \text{(A5-2)}$$

where

x_i = x position of the ith element and is only used for vertical and diagonal moves.

y_j = y position of the jth element and is only used for horizontal and diagonal moves.

n_x = number of x elements used

n_y = number of y elements used.

In this case the x_i's in the x-equation are only tallied when there has been a vertical or diagonal move along the IA boundary on the grid. Likewise, y_j's are only tabulated during horizontal or diagonal moves.

This provides a fast method for calculating the centroid based only on the boundary elements, which we determine during the IA identification process anyway. This method is very much faster than (A4-1).