CVGIP: IMAGE UNDERSTANDING
Vol. 54, No. 1, July, pp. 1–46, 1991

A Robot Vision System for Recognition of Generic Shaped Objects*

A. J. VAYDA† AND A. C. KAK

Robot Vision Lab, School of Electrical Engineering, Purdue University, West Lafayette, Indiana 47907

Received October 29, 1990; accepted January 29, 1991

Generic shape recognition is the problem of determining the pose and dimensions of objects for which only shape models are available and the object's size is unknown. One application domain for generic object recognition is the handling and sorting of postal objects. Because metrical information relating object features to one another is not available, the more common feature-based approaches are inadequate. Our system, INGEN (INference engine for GENeric object recognition), uses a data-driven approach to determine the pose and size of objects with generic shapes such as parallelepipeds and cylinders. This system successfully recognizes occluded objects in heaps. It also handles scenes which have irregularities in surfaces and edges—such irregularities are common to postal objects—as well as shadows and irregularities in the range data itself. The three most important parts of INGEN are (1) the procedures for constructing object hypotheses, computing their attributes, and evaluating how well they fit the data, (2) the geometric reasoning process which determines the size of object hypotheses by finding points of contact with other object hypotheses and also detects geometric inconsistencies in the scene interpretation, and (3) the recognition process which allows backtracking when object hypotheses are rejected due to insufficient support or geometric conflict with other object hypotheses. INGEN has been used successfully to guide a robot in removing postal objects from a pile. We show the results of these experiments.

CONTENTS

* This work was partly supported by the U.S. Postal Service Office of Advanced Technology under Contract 104230-86-H-0043.

† A. J. Vayda is now with the Environmental Research Institute of Michigan (ERIM), Ann Arbor, Michigan 48107.

1. INTRODUCTION

Over the years it has become clear to us that there cannot exist universally applicable strategies for the recognition of 3D objects from range maps—or, for that matter, from sensory data of any type. The reason for this is that the types of distinctions that a system must make between different objects depend on the uses for which object recognition is carried out. For example, the types of distinctions needed for identifying objects in a robotic assembly cell are different from the types of distinctions necessary for sorting parcels in a postal mail stream. For the former, it will usually be necessary to take into account the precise geometric attrib-

1049-9660/91 $3.00

utes of the various features of objects, since such attributes are important for making distinctions between industrial objects. For the latter, on the other hand, it will usually be sufficient to classify an object as, say, a "box" despite large variations in the dimensions of the object. If one of the dimensions becomes too small, the object might then be called a "flat," again over large variations in the other two dimensions.

The two different types of problems outlined above require fundamentally different approaches to object recognition. Where industrial objects are involved, a recognition system must make precise measurements of the various geometric attributes of the object surfaces. On the other hand, for other kinds of objects, such as postal objects, a recognition system must be able to ignore the finest level of detail, which in most cases would correspond to irrelevant details such as "crumpliness" of the surfaces, and instead concentrate on the overall object shape.

As also discussed by Bajcsy and Solina [2], these two types of problems lie at opposite ends of the spectrum of 3-D object recognition. At one end of the spectrum we have feature-based object recognition such as recognition of industrial objects where we have complete 3-D models for all objects that might appear in a scene. The goal is to match scene features with features in the object database, thus finding the identity and the pose of objects in the scene. Most efforts in object recognition have focused on this domain. At the other end of the spectrum we have generic object recognition such as classification of postal objects. In this case we have incomplete information about objects that might appear in the scene, in the sense that objects are assumed to belong to generic categories based on gross shape where each category allows for large variation in object dimensions. This domain is relatively new and the traditional feature-based approaches are not easily adapted to it.

In feature-based recognition the most important issues are the data representation and the search procedure. For example, the 3D-POLY system [5, 6], which we believe is the fastest implemented system for 3D object recognition and pose estimation, makes use of a highly optimized data structure and a hypothesize-and-verify search process to recognize industrial objects in low-order polynomial time. In this paper we discuss the INGEN (INference engine for GENeric object recognition) system which uses a hypothesize-and-refine approach to determine the pose and size of generic shaped objects from their range maps. The hypothesize-and-refine approach, in contrast to the hypothesize-and-verify approach, does not emphasize the importance of object representation, since only a few generic shapes need to be dealt with—parallelepipeds, cylinders, and irregulars. In INGEN, the task is to assign each surface segment to the appropriate object and to compute the dimensions and pose of each object.

Since INGEN is designed to do shape categorization of postal objects from their range maps, especially the parcels in a mail stream, and since such objects are often characterized by nonsmooth surfaces, one of the main challenges of INGEN is coping with the resulting oversegmentation of object surfaces. It therefore becomes imperative to design control strategies capable of efficiently pruning away large fractions of the search space whose size exhibits an exponential dependence on the total number of surface segments in the segmented range map.

INGEN begins the recognition process by creating an object hypothesis for each surface segment in segmented range data. Note that an object with multiple surfaces visible in the scene will cause multiple object hypotheses to be formed; during the combination process these hypotheses will be combined into a single object hypothesis. The same is true for cases where a single object face has been segmented into multiple parts due to noise or occlusion. The hypothesis corresponding to the largest segment is used as a seed and the lower ranked hypotheses are subsequently considered as candidates for combination with the seed hypothesis. To be considered for combination the surfaces of the two object hypotheses must meet certain criteria such as coplanarity or adjacency. If the criteria are met then the two hypotheses are combined to form a single object hypothesis and its validity is evaluated. Backtracking is initiated when the data fail to support the newly combined hypothesis. This requires that the combination be undone so that the search for other combinations can continue. After all possible combinations involving the first object hypothesis have been considered the next largest object hypothesis becomes the seed and the search begins again.

The principal mechanisms for hypothesis combination are merging and aggregation. Merging is used when two hypotheses are thought to contain segments that belong to the same surface of the same object. On the other hand, aggregation is used when two hypotheses are thought to contain segments that belong to two separate surfaces of the same object. At the very outset, the different surface segments are marked with regard to whether or not, on a pairwise basis, they can be merged or aggregated—mergability being determined on the basis of considerations such as coplanarity, cocylindricity, etc., and aggregability on the basis of perpendicularity of the associated surface normals. This mergability and aggregability information is stored in a graph data structure called the *combinability graph*. The use of combinability graphs prunes large sections of what would otherwise be an extremely large search space. In addition to merging and aggregation, a weaker criterion, adjacency, is also used for combining hypotheses on the basis of the proximity of

their constituent surfaces and certain continuity properties across their common boundaries.

INGEN uses the Dempster–Shafer theory of evidence for associating belief values with each object hypothesis. This approach allows the assignment of belief to an object hypothesis, to its negation, or to neither. Belief that cannot be assigned to the hypothesis or its negation represents the uncertainty inherent in the evaluation process. Two individual hypotheses H_1 and H_2 are combined into a single hypothesis H only if the belief associated with H is larger than the beliefs associated with H_1 and H_2 and if the belief in H is greater than the belief in the negation of H. Of course, if these conditions are not met then backtracking is initiated.

Another consideration that can trigger backtracking is the overall consistency of the scene interpretation. By consistency we mean that different object hypotheses must not occupy the same space. Consistency determination, facilitated by a geometric reasoning process that can determine the contact points between objects without requiring such points to be visible in the scene, allows us to first determine whether or not a given set of object hypotheses is valid and then permits the computation of the various dimensions of the objects corresponding to a valid interpretation. The approach is to extend each object along one of its axes, roughly in the direction away from the sensor, until it physically contacts another object in the scene. The hypothesized objects are extended one at a time; the order in which the object hypotheses are processed does not appear to matter unless it is possible to give multiple geometrically consistent interpretations to the data. When the geometric reasoning process finds an object hypothesis to be inconsistent with the rest of the scene interpretation, the system backtracks in its search by undoing the most recent hypothesis combination.

The INGEN system has been used to successfully guide robot manipulation experiments involving postal objects. Figure 1a shows a pile of postal objects. Each object in the pile has an overall generic shape, despite the irregularities of the surfaces, that belongs to one of a small number of categories. Figure 1b shows range data derived from a structured light scan of the scene and Fig. 1c shows the segmented range map. Range data are collected by a structured light scanner mounted above the robot work area. In Fig. 2, the photo in (a) shows the robot picking the first object from the pile. The photo in (b) shows the results after the first three objects have been stacked. The photo in (c) shows the final result where the objects have been moved to two stacks, one to the left of the original pile and the other to the right. Because of the depth of the original pile most of the objects are totally occluded in the initial scene. Data must be collected and interpreted several times as the robot removes the topmost objects and more objects underneath are uncovered. In this example INGEN automatically collected the data and carried out its interpretation process four times. Three objects were found on the first occasion, and two each on the second and the third. On the fourth occasion, INGEN determined that the scene was empty.

The task of postal object recognition has also been addressed by several other research groups. The work of Cowan, Mulgaonkar, DeCurtins, and de St. Vincent [7, 8, 24, 25] is the most closely related to ours, especially with regard to the geometric reasoning used for establishing the consistency of the overall scene interpretation by disallowing the interpretations in which hypothesized objects intersect. They also eliminate candidate hypotheses for scene objects on the basis of the gravitational stability of the pile of objects. Furthermore, in their work extensive use is made of range sensor geometry for the interpretation of shadows in the range data. McClain and Kenig [23] use a similar approach for low and mid level processing but do not consider geometric constraints in high level processing. Bajcsy and Solina [2, 31, 32] recognize generic shapes with an iterative procedure for fitting superquadric surfaces to range data. The single object model used is a superquadric surface which has 11 parameters (three for size, three for position, three for orientation, and two for shape). This model can be used to represent parallelepipeds, cylinders, ellipsoids, and other shapes that are deformations of these. The superquadric representation is particularly useful for modeling highly symmetric objects. The recognition process involves finding the best estimate of the parameters by solving an overdetermined optimization problem. An analytic solution to this problem is not practical so an iterative fitting procedure is used. They assume that the objects can be segmented from each other by finding concave and jump edges in the range data. Thus, they actually only address the issue of characterization of generic objects.

In what follows, in Section 2 we present the overall flow of control in INGEN. Section 3 discusses the computation of the position, orientation, and dimensions of object hypotheses. In Section 4 the hypothesis merging and aggregation criteria that are used in the construction of the combinability graph are presented. Subsequently, we discuss the object hypothesis evaluation techniques in Section 5. The geometric reasoning process which computes maximal object dimensions and detects intersections between objects is discussed in Section 6. In Appendix A we give an overview of our range data processing techniques from the acquisition of the range data through the construction of the hierarchical symbolic scene description. In Appendix B we provide an introduction to the Dempster–Shafer theory and describe

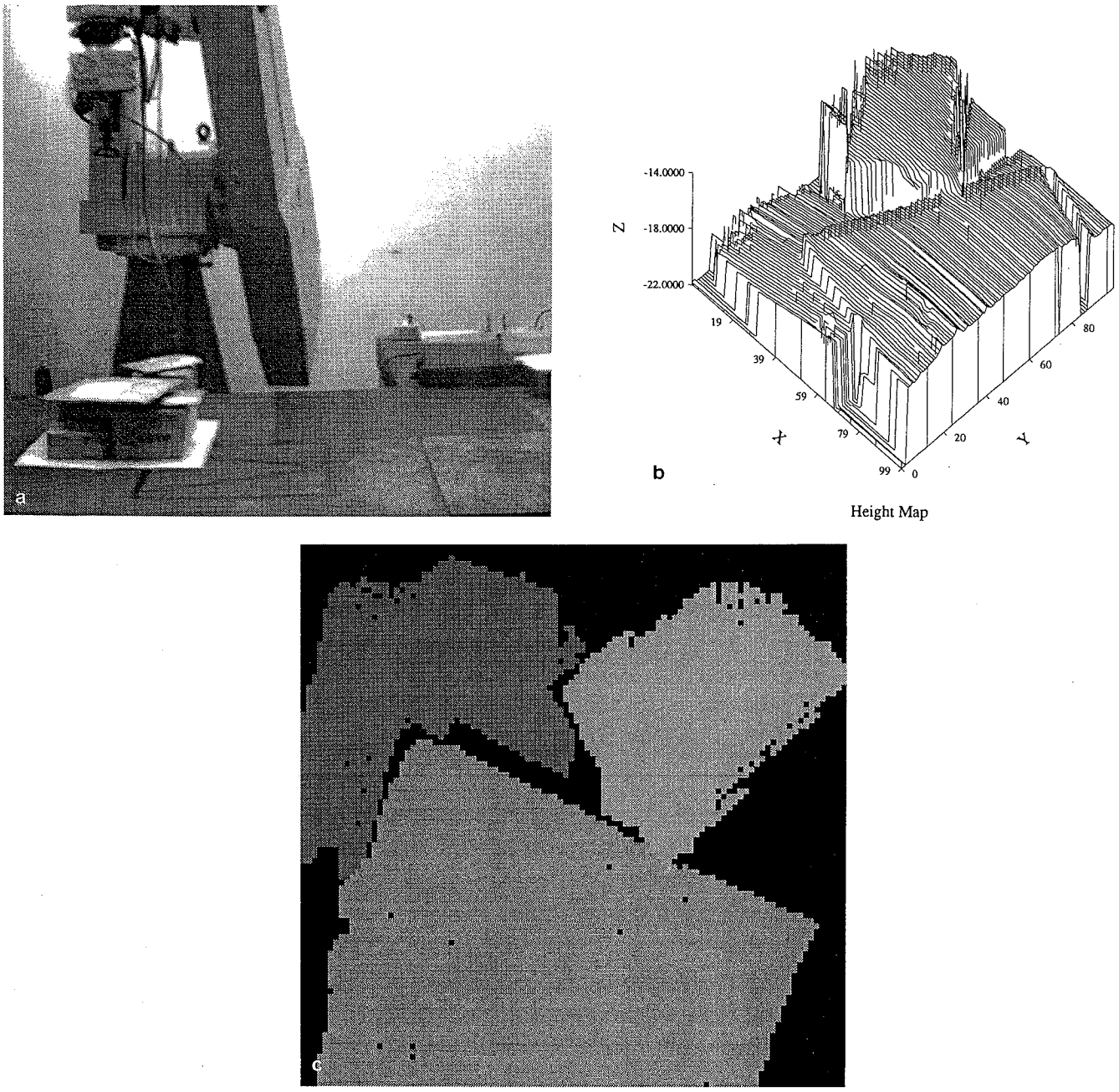


FIG. 1. (a) A pile of postal objects, (b) range data for the scene, and (c) the segmented range map for the scene.

a simplification of this theory which is applicable to our hypothesis evaluation problem.

Prior partial accounts of INGEN have appeared in [20, 16, 18, 19, 33]. INGEN has continually evolved as our own insights into the subject of generic object recognition have sharpened over time. The description in this paper represents the current reasoning architecture of INGEN and has not been discussed in any of the prior publications.

2. FLOW OF CONTROL

A diagram of the flow of control within INGEN is shown in Fig. 3. The modules in the figure have been numbered to facilitate the discussion throughout this section. We will first give an overview of the flow of control in INGEN and will then illustrate it further with the help of an example. Details regarding the operation of each of the modules in Fig. 3 will be provided in the remaining sections of the paper.

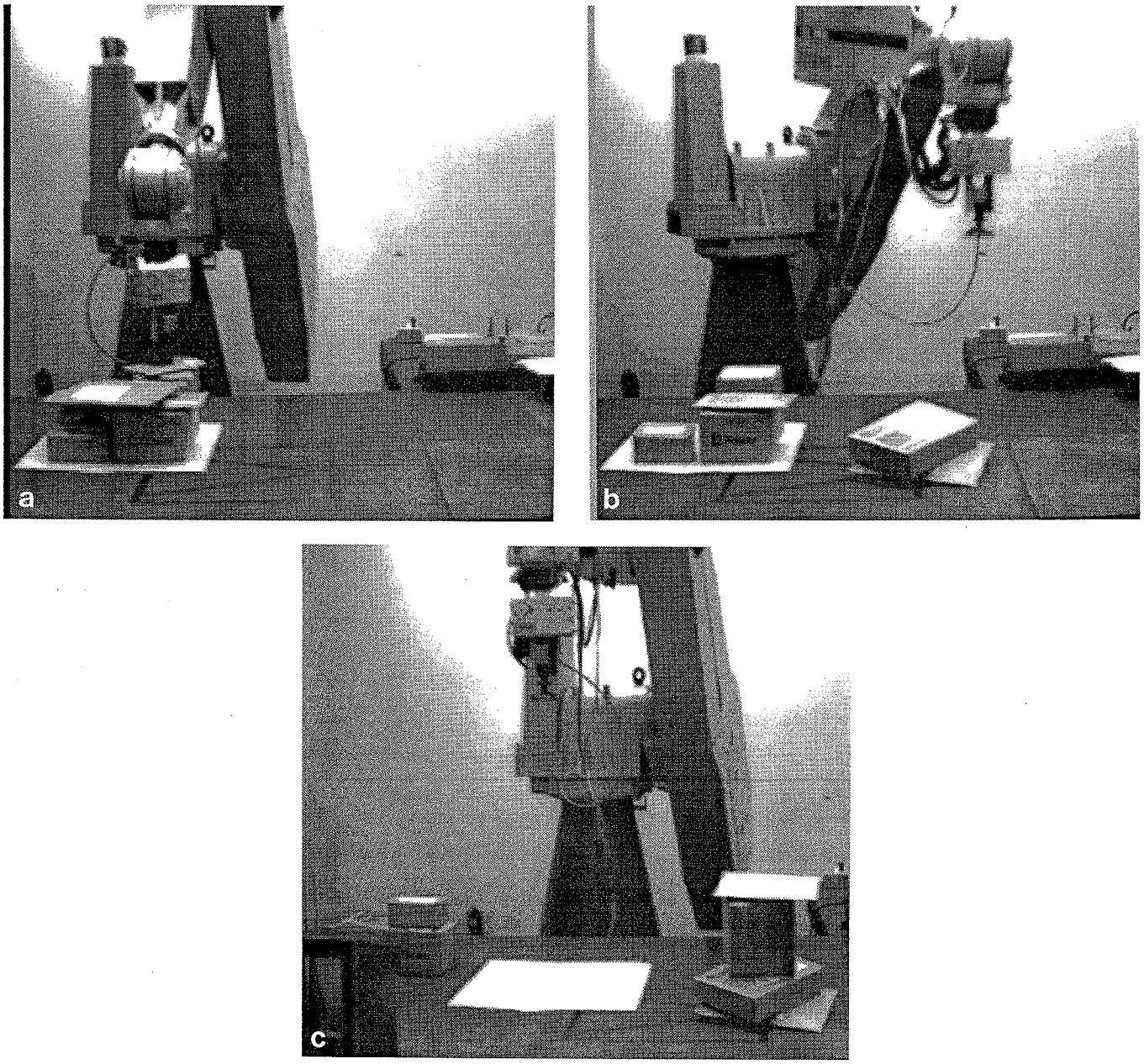


FIG. 2. Seven postal objects in a pile are picked up and stacked by a robot using INGEN.

2.1. Overview

The processing carried out by the system can be divided into four stages. The acquisition of the range data (module 1) constitutes the first stage. The output from this stage is the raw range data. The remaining three stages operate on these range data at progressively higher levels of abstraction. Low level processing (module 2) involves the computation of local properties of the range data points, such as surface normals and surface curvature, and the segmentation of the overall range map into surface segments on the basis of curvature and surface information. Based on these properties, the range data are segmented into surface segments. Mid level processing (module 3) involves the characterization of each surface segment, as well as the characterization of the edges and vertices which bound the surface segments and the relations between adjacent segments. A hierarchical symbolic scene description which contains all of this in-

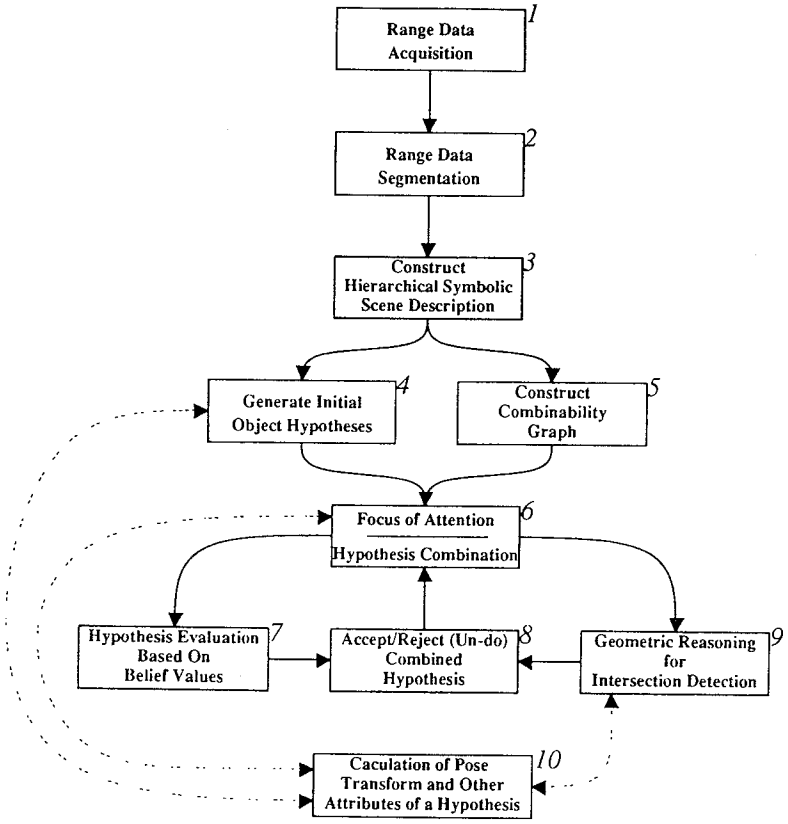


FIG. 3. Flow of control in INGEN.

formation is produced. High level processing (modules 4–10) involves the combination of surface segments into objects and the computation of the properties of these objects.

The range data acquisition stage and the low and mid level processing stages (modules 1–3) are implemented in the C programming language. As the aim of this paper is more to discuss the high level reasoning part of INGEN, we provide only an overview, in Appendix A, of the range data acquisition and the low and mid level processing modules; the low and mid level aspects are discussed in greater detail in [17]. The high level processing stage, the heart of INGEN (modules 4–10), is implemented in Quintus Prolog. Details of this implementation and the source code can be found in [34]. The robot control software is implemented in Lisp, with the communication between the computer and the robot implemented in C.

An overview of the flow of control, without concern about the actual processing in the modules, can be presented as follows:

- The actions of modules 1, 2, and 3 are carried out in series.
- The actions of modules 4 (which calls module 10) and 5 can be carried out essentially in parallel because they do not depend on each other.
- Module 6 forms the heart of the recognition process.
- Control cycles between modules 6 (which calls module 10), 7, and 8 in the Hypothesis Evaluation Loop until a scene interpretation is found in which each individual object is plausible.
- Then control shifts to module 9 in the Geometric Reasoning Loop, which evaluates the consistency of the entire scene interpretation and computes maximal dimensions for the object hypotheses.
- If an inconsistency is found then control shifts through module 8 back to modules 6, 7, and 8 where another scene interpretation is produced.
- The system alternates between these two processes until a consistent scene interpretation is found.

2.2. The Recognition Process

The recognition process begins with the acquisition of the range data (module 1 in Fig. 3). The output of the low

level processing (module 2) is a segmentation of the scene into surfaces. From these surfaces, the mid level processing (module 3) produces the hierarchical symbolic scene description. This description describes the surfaces, edges, vertices, and surface adjacency relations in the scene. The high level processing (modules 4–10) uses this scene description exclusively and does not have access to the original data.

The goal of high level processing (modules 4–10 in Fig. 3) is to partition the set of surface segments into objects in order to produce a consistent scene interpretation. This scene interpretation is constrained by two factors. First, the surfaces for each object must fit the object hypothesis sufficiently well. Second, the object hypotheses should not intersect each other in three dimensional space. A search process is used to find and test possible scene interpretations. Clearly, an exhaustive search through all possible scene interpretations is impossible for any scene of interesting complexity. So INGEN uses several techniques to reduce the size of the search space.

High level processing begins with the creation of an object hypothesis for each surface in the segmented range data (module 4 in Fig. 3). These single-surface hypotheses are then ranked according to a set of criteria for the purpose of controlling the search process. Currently the ranking is from the largest to the smallest surface based on the number of range data points because larger surfaces typically lead to more reliable object hypotheses. The first hypothesis is used as a seed and the lower ranked hypotheses are then considered, in an order based on their ranking, as candidates for combination with the seed hypothesis. If a combination is unsuccessful then the search continues with the unchanged original hypothesis as the seed. On the other hand, if a combination is successful, the search continues with the new combined hypothesis as the new seed; also, the candidate hypothesis which was combined with the seed hypothesis is removed from the scene interpretation. When no more combinations are possible with the current seed hypothesis, a new seed hypothesis is selected from the remaining hypotheses. This process is repeated until all hypotheses have been combined or considered as seeds.

Concurrently with the generation of the single-surface object hypotheses, the relations between surfaces are examined to determine which pairs of surfaces are potential candidates for combination (module 5 in Fig. 3). This makes it possible to reject totally inappropriate surface combinations based only on surface characteristics. Note that relations are tested for all pairs of surfaces, not just those that are adjacent in the range data. This results in the creation of a graph known as the *combinability graph*. Each node in the graph represents a surface segment. Each arc in the graph represents an allowable combination between the two surfaces that it connects. Combinations are allowed when surfaces meet criteria such as coplanarity, cocylindricity, perpendicularity, or adjacency. For a graph with n nodes (surfaces) there are

$$\binom{n}{2} = \frac{n(n-1)}{2}$$

possible arcs (combinations). However, the criteria for allowable combinations significantly reduces the number of arcs in the graph. The problem, then, is to determine which of the allowable combinations should be made and which should not. One could say that given a combinability graph, the task of modules 6–10 in Fig. 3 is to partition the combinability graph into subgraphs, each subgraph representing a distinct object.

The combinability criteria are derived from the two types of hypothesis combination operations: merging and aggregation. Merging is the combination of hypotheses that are separate because a single object surface was segmented into multiple parts due to noise or occlusion. Aggregation, on the other hand, is the combination of hypotheses that are separate because distinct object faces are present in the scene. The merging criteria require that the surfaces be coplanar, cocylindrical, or coirregular. The aggregation criteria require that the surfaces be perpendicular for the case of parallelepipeds and irregulars, and that the surfaces be coaxial for the case of cylinders (the latter definition is presented more precisely in Section 4). If the surfaces are adjacent in the scene then there will also be an adjacency relation which describes the boundary between them; if this relation exists then further criteria can be used when considering combinability. Note that the merging criteria depend only on the types of surfaces expected to be in the scene but the aggregation criteria depend on the types of objects expected to be in the scene.

The focus of attention module (module 6 in Fig. 3) determines the order in which the combinability graph is searched. Currently, the hypotheses are ordered based on the number of range points contained in their surfaces. This orders the search so that the strongest hypotheses are considered first.[1] Considering stronger hypotheses first should result in less backtracking and a smaller search to find the first geometrically plausible scene interpretation. Also, considering stronger hypotheses first should result in the first geometrically plausible scene

[1] Other criteria that we have considered include ordering from the topmost hypothesis in the pile to the lowest, considering merging operations before aggregation operations, and ordering based on the belief in the hypotheses. Theoretically, if the system were to look for all possible solutions (interpretations), the order in which the hypotheses are combined would not matter, since, after all, repeated backtracking would eventually discover every solution. However, in keeping with what we will have to say shortly about the combinatorics involved, a full search would be prohibitively expensive, making it necessary that hypotheses be ordered in some manner.

interpretation also being the strongest scene interpretation. For this reason, INGEN stops with the first scene interpretation that is geometrically plausible. A database containing information about the current scene interpretation is maintained by the focus of attention module and updated as the search process proceeds. Information about prior scene interpretations is saved to prevent redundant computations.

Note that the combinability graph contains information about allowable combinations of surfaces; it does not contain information about objects. This is necessary because the information about surfaces does not change throughout the recognition process. However, the objects that contain particular surfaces change as the recognition process progresses. Thus, in order to consider the combination of the seed hypothesis with a candidate hypothesis, we must first find the surfaces of the hypotheses and then must determine if there is an allowable combination between a seed hypothesis surface and a candidate hypothesis surface.

The consideration of a combination takes place in two steps (module 7 in Fig. 3). First, the combined object hypothesis must be created and its belief computed. Then, the belief in the combined hypothesis must be compared with the beliefs in the constituent hypotheses. The belief in an object hypothesis is a function of how well the data support the hypothesized object. Prior to belief computation, the pose and dimensions of the object hypothesis must be computed (module 10 in 3) and the correspondences between the data surfaces and edges and the model surfaces and edges must be determined. INGEN uses the Dempster–Shafer theory of evidence for associating belief values with object hypotheses. The belief in a hypothesis is based on two factors: how well the data surfaces fit the model surfaces with respect to orientation and surface type, and how well the data edges fit the model edges with respect to position and orientation. In Appendix B we provide an introduction to the Dempster–Shafer theory and then discuss the simplifications which are made possible by INGEN's use of dichotomous frames of discernment.

For belief computations, that is when control is in the Hypothesis Evaluation Loop in Fig. 3, each object hypothesis is evaluated individually without concern about other hypotheses in the scene. Hypothesis evaluation is carried out by accumulating the beliefs in two dichotomous propositions, O and $\neg O$, where O stands for "the data support the object hypothesis," and $\neg O$ for "the data do not support the object hypothesis." Thus a source of evidence, such as an edge or a surface segment in the data, may assign some probability mass to O, some to $\neg O$. However, as the reader will see in Section 5, not all the available probability mass need be assigned to these two dichotomous propositions; some may be withheld as a measure of ignorance if there is any chance that the source of evidence may not be relevant to either O or $\neg O$. Let $Bel(O)$ and $Bel(\neg O)$ represent, respectively, the total beliefs, as pooled from all available sources of evidence in the data, in O and $\neg O$.

On the basis of the information generated by module 5 (Fig. 3), the focus of attention module may propose that two previously established hypotheses O_1 and O_2 be combined into a single hypothesis O. Subsequently, module 7 will compute $Bel(O)$ and compare this quantity with $Bel(O_1)$ and $Bel(O_2)$. This comparison may cause O to be either accepted or rejected as a combined hypothesis.

After a complete scene interpretation has been constructed, meaning the processing in the Hypothesis Evaluation Loop has come to a halt, the geometric reasoning process begins (module 9 in Fig. 3). This process serves two purposes. First, it allows us to place constraints on the dimensions of objects based on contacts with other objects in the same heap and thus determine better estimates for the sizes of objects. Second, it allows us to detect geometrically inconsistent scene interpretations. Inconsistencies are detected by finding objects which occupy the same space (whose volumes intersect). The actual method of detecting inconsistencies involves comparison of dimensions computed by geometric reasoning with those extracted directly from the data as described in Section 6. An inconsistency is indicated when the geometric reasoning process declares that a particular dimension must be smaller than the value computed directly from the data. When the geometric reasoning process finds an object hypothesis to be geometrically inconsistent with the rest of the scene interpretation the system backtracks in the search process.

Control in the search process alternates between two phases. The hypothesis evaluation phase, executed by modules 6, 7, and 8, which form the Hypothesis Evaluation Loop, produces a scene interpretation. Then, the geometric reasoning phase, executed by modules 6, 9, and 8, which form the Geometric Reasoning Loop, tests the scene interpretation for consistency. When inconsistencies are found the hypothesis evaluation phase is restarted and a new interpretation is produced. This process repeats until a scene interpretation with no geometric inconsistencies is produced.

Backtracking is triggered under two circumstances: when a combined hypothesis is rejected during the evaluation phase (by module 7 in Fig. 3), or when a scene interpretation is rejected during the geometric reasoning phase (by module 9). In the first case it is obvious how the system should backtrack. The newly combined hypothesis is undone (by module 8) and the previous seed hypothesis becomes the seed again. How the backtracking should be carried out is not quite as obvious in the second case. If the geometric reasoning process detects an intersection between two objects then either of them could be wrong. We take the approach of backtracking on the

larger object of the conflicting pair, meaning we undo that combination (of hypotheses) which led to the formation of the larger object. Clearly, such "undoing" is impossible if the larger object is still a single-surface object hypothesis, meaning the larger object hypothesis contains only one surface. In that case, we have no choice but to use the other object for backtracking, the backtracking again being accomplished by undoing the combination that led to the other object hypothesis. There is also the issue of the order in which object hypotheses are considered for intersection detection. We start from the bottom of the pile of objects and work toward the top.[2] Thus, objects higher up in the pile will be tested after objects lower in the pile. This results in the objects higher up being tested for intersection with objects that have already been tested.

Because INGEN recognizes generic objects, the nature of the search process in INGEN differs significantly from those used in more traditional approaches. In the more traditional feature-based recognition [5, 6, 9, 11–13, 21], each object in the model library is complete with regard to geometric information and, therefore, when an object hypothesis is formed, it remains essentially fixed during subsequent verification.[3] Furthermore, during verification the test for adding a feature to a hypothesis is simply a test to verify that the feature fits the model. On the other hand, for generic object recognition it is impossible to fix permanently the pose and the dimensions of an hypothesized object at the instant of hypothesis formation due to the fact that the dimensions of the object in the scene are unknown. At any given moment in the search, one may construct the best possible pose and dimensions on the basis of, say, the surface segments included, but as new surface segments are considered for merging or aggregation, we may have to alter the pose and the dimensions associated with the hypothesized object. For this reason, for generic object recognition, it is not possible to neatly divide the overall search into hypothesis formation and verification stages; what we have now is more of a case of hypothesis formation followed by its continuous refinement as additional features are combined with the hypothesis. Therefore, whenever the compatibility of a new surface segment with a surface of the current hypothesis leads to the creation of a new hypothesis, the pose and the dimensions of the new hypothesis must be recomputed from scratch and can be quite different from those of the constituent hypotheses.

```
PROCEDURE search()
    SearchHistory ← []
    SurfaceQueue ← all data surfaces sorted by number of data points
    search1()
END PROCEDURE
```

FIG. 4. Algorithm to start the search process.

Another point of distinction between traditional feature-based recognition and generic object recognition is the difference in computational costs. In the former, after a model object has been hypothesized, the verification of the absence or presence of a new data feature in the model space has relatively low computational cost. However, in the latter, this cost can be relatively high since pose and dimension information must be recomputed every time a new entity from the data is added to a hypothesis. Therefore, in systems such as INGEN it becomes imperative that the production of new hypotheses be kept to a minimum; this we have accomplished by conducting search over what we call a combinability graph rather than over all possible objects.

2.3. The Search Algorithm[4]

Having presented a general description of the recognition process we can now present the algorithm which carries out the search process. The search algorithm can be described in pseudocode by four primary procedures: search, search1, search2, and search3. The search process begins with the procedure search( ) as shown in Fig. 4. It initializes the global variable SearchHistory to an empty list and initializes the global variable SurfaceQueue to a list of all of the data surfaces in the scene sorted by the number of data points in each. It then calls the procedure search1( ).

The procedure search1( ) recursively picks surfaces from the surface queue to determine the order in which objects are considered for combination, as shown in Fig. 5. It begins by examining the list SurfaceQueue. If it is not empty then it takes FirstSurface, the first surface from the queue, and finds Object, the object hypothesis that currently contains it. Then, search2(Object) is called to attempt combinations based on this object; as explained later, the process of hypothesis combination is actually carried out by search3 which also computes the beliefs associated with the combined hypotheses. After search2(Object) returns, search1( ) is called recursively. If SurfaceQueue is empty then we have a scene interpretation that must be checked for geometric consistency.

[2] As will be clear later, the geometric reasoning module also grows each object hypothesis to its maximum allowable dimensions along directions away from the sensor. This implies growing objects from their visible surfaces toward the work table. The directionality of this growing process dictates that for intersection detection the objects near the bottom of a pile be considered first.

[3] In some cases, the pose transform may be adjusted incrementally as additional features extracted from the data are verified as belonging to the hypothesized object.

[4] Some readers might prefer first to go through the example we have discussed in Section 2.4 before reading this section.

```
PROCEDURE search1()
    IF SurfaceQueue ≠ [] THEN
        FirstSurface ← first element of SurfaceQueue
        Object ← object containing FirstSurface
        search2(Object)
        search1()
    ELSE
        ConflictObjects ← geometric_reasoning()
        IF ConflictObjects ≠ [] THEN
            uncombine(ConflictObjects)
            SurfaceQueue ← all data surfaces sorted by number of data points
            search1()
        ELSE
            exit
        END IF
    END IF
END PROCEDURE
```

FIG. 5. Algorithm to control the search process for the entire scene.

The procedure geometric_reasoning( ), which is discussed in Section 6 and defined in Figs. 39 and 40, returns ConflictObjects, a list of objects that are in geometric conflict and must be uncombined. If this list is empty then there are no conflicts and a geometrically consistent scene interpretation has been found. If there are conflicts then these objects are uncombined, SurfaceQueue is reinitialized, and search1( ) is called recursively.

The procedure search2(Object) starts the search for combinations based on a single Object as shown in Fig. 6. First, it finds Surfaces, all of the surfaces that are adjacent to the surfaces of Object in the combinability graph. Then these surfaces are used in the call to the procedure search3(Object, Surfaces).

The procedure search3(Object, Surfaces) recursively steps through the list of Surfaces until a successful combination is made as shown in Fig. 7. When the list Surfaces is empty control reverts back to search1( ) through search2(Object). When the list Surfaces is not empty Surface, the largest surface based on the number of data points, is selected and the object OtherObject which contains it is found. Object and OtherObject are combined into NewObject. If NewObject has already been considered then it is uncombined, Surface is removed from Surfaces, and a recursive call to search3(Object, Surfaces) is made. If NewObject has not already been considered then it is added to the SearchHistory list and is evaluated. If the evaluation is successful then the surfaces of NewObject are removed from SurfaceQueue and the search continues with a call to search2(NewObject). If the evaluation is not successful then NewObject is uncombined, Surface is removed from Surfaces, and a recursive call to search3(Object, Surfaces) is made.

```
PROCEDURE search2(Object)
    Surfaces ← surfaces adjacent to surfaces of Object in combinability graph
    search3(Object, Surfaces)
END PROCEDURE
```

FIG. 6. Algorithm to control the search process for a single object.

```
PROCEDURE search3(Object, Surfaces)
    IF Surfaces = [] THEN
        SurfaceQueue ← SurfaceQueue – surfaces of Object
    ELSE
        Surface ← largest surface in Surfaces
        OtherObject ← object containing Surface
        NewObject ← combine_objects(Object, OtherObject)
        IF NewObject is in SearchHistory THEN
            uncombine(NewObject)
            Surfaces ← Surfaces – Surface
            search3(Object, Surfaces)
        ELSE
            SearchHistory ← SearchHistory + NewObject
            IF evaluate_object(NewObject) THEN
                SurfaceQueue ← SurfaceQueue – surfaces of NewObject
                search2(NewObject)
            ELSE
                uncombine(NewObject)
                Surfaces ← Surfaces – Surface
                search3(Object, Surfaces)
            END IF
        END IF
    END IF
END PROCEDURE
```

FIG. 7. Algorithm to search through a list of surfaces and make combinations.

2.4. Example of Flow of Control

The flow of control for modules 1 through 6 in Fig. 3 is linear and therefore straightforward. On the other hand, the search process embodied in modules 6 through 9 involves two interacting loops (the Hypothesis Evaluation Loop and the Geometric Reasoning Loop) and, we believe, needs to be further exemplified. This we will do with the help of an example scene consisting of the three objects shown in Fig. 8. Let us assume that for this scene the surface segments produced by processing the range data are as shown in Fig. 9, where we have also ranked the surfaces by size and numbered them from largest to

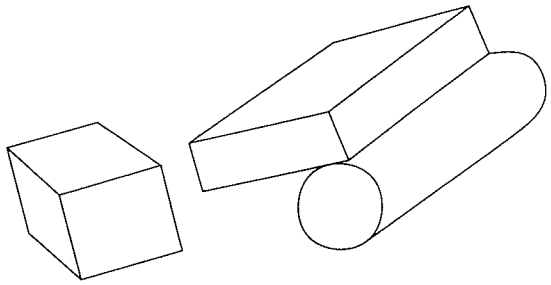


FIG. 8. Example scene.

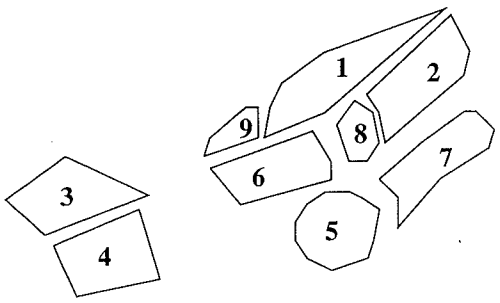


FIG. 9. Segmented range data for the example scene.

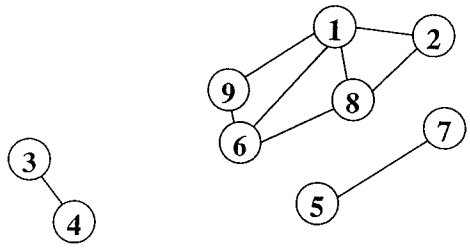


FIG. 11. Partition of the combinability graph corresponding to the correct interpretation for the example scene.

smallest, starting at 1. Even though INGEN treats any surface belonging to the background just like any other surface, for this example we disregard the background so that the explanation is simplified. Surfaces 1, 2, 6, 8, and 9 belong to a box, surfaces 5 and 7 to a cylinder, and surfaces 3 and 4 to another box. The first box is resting on top of the cylinder. Surfaces 1 and 9 correspond to the same surface of the same box and are coplanar. Surfaces 2 and 8 correspond to another surface of the same box and are also coplanar. Surface 5 of the cylinder is assumed to be coplanar with surface 6 of the box. Surfaces 1 and 3 of the two different boxes are also assumed to be coplanar.

Figure 10 shows the combinability graph for the segmented scene as would be computed by the combinability graph construction module (module 5 in Fig. 3) using the methods discussed in Section 4. To facilitate its visual assimilation, the nodes of the graph in Fig. 10 are positioned in such a manner that their locations correspond roughly to the respective surfaces in Fig. 9. The arcs (allowable combinations) in the graph are labeled to show the types of relationships between the nodes (surfaces). A represents adjacency, P represents coplanarity, X represents coaxiality in the sense that the cylindrical axis of a cylindrical surface is coaxial with the surface normal of its planar end surface, and N represents perpendicularity. Note that a graph with 9 nodes can have at most $\binom{9}{2} = 36$ arcs, but for this scene only 13 arcs have been determined to correspond to allowable combinations. Figure 11 shows the final partition of the combinability graph showing the correct scene interpretation. The goal of INGEN is to discover this partition.

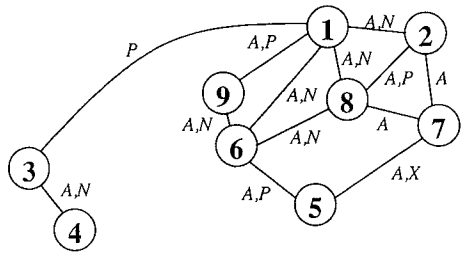


FIG. 10. Combinability graph for the example scene.

To help with the explanation and understanding of the flow of control, we make certain simplifying assumptions regarding the workings of the evaluation module (module 7 in Fig. 3) and the geometric reasoning module (module 9 in Fig. 3). We assume that the evaluation module will reject combined hypotheses which are obviously incorrect but will accept the nonobvious incorrect hypotheses which we have intentionally included in the example. To wit, we have intentionally made surfaces 5 and 6 coplanar and therefore combinable via merging. The hypothesis obtained by combining the surfaces 5 and 6 will appear reasonable to the evaluation module, since after all these two coplanar surface segments could have arisen from the same planar face of the same object. The incorrectness of such combined hypotheses will be detected by the geometric reasoning module on the basis of the overall inconsistency of all the hypotheses in the scene. Along similar lines, we assume that the geometric reasoning module will produce the obvious results concerning which hypotheses intersect and which hypotheses should be undone in the case of an intersection.

After the generation of initial object hypotheses by the object hypothesis generation module (module 4 in Fig. 3) the scene interpretation is specified by the following set of object hypotheses: {{1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, {9}}.[5] We identify each object hypothesis by the set of surfaces that it contains, and the scene interpretation by a set of object hypotheses where each surface appears in exactly one hypothesis. The nine single-surface object hypotheses are shown in Fig. 12. These hypotheses are passed on to the focus of attention module (module 6 in

[5] In actuality, as discussed in Section 3, each hypothesis is represented by a list of attribute–value pairs, one of these attributes is the list of surface segments assigned to that hypothesis. However, for this section it suffices to think of each hypothesis as the set of surfaces assigned to it. We need not be concerned with other attributes such as the pose transformation and the dimensions of the object hypothesis.

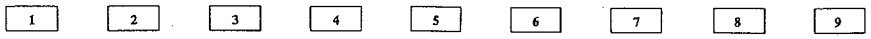


FIG. 12. Initial hypotheses.

Fig. 3) which controls the search for a scene interpretation.

The search process begins with the initialization of the surface queue to a list of surfaces sorted by size: [1, 2, 3, 4, 5, 6, 7, 8, 9]. The first surface, 1, is removed from the queue and the object containing it, {1}, becomes the first seed. A depth-first search through the combinability graph is performed as surfaces from the queue, and thus the objects that contain them, are considered for combination with the seed hypothesis. If an arc is present in the combinability graph between a surface of the seed hypothesis and a surface in the queue, then the seed hypothesis is combined with the hypothesis containing that surface. If the belief in the combined hypothesis, as computed by the evaluation module (module 7 in Fig. 3), is greater than the belief in the constituent hypotheses then it is accepted and it becomes the new seed and all of the surfaces of the new seed are removed from the surface queue. Otherwise, the hypothesis is uncombined by the accept/reject module (module 8 in Fig. 3) and the search continues with the same seed and the next surface in the surface queue. Thus, when we refer to an object hypothesis being accepted or rejected we are referring to the actions of the evaluation module. Accepted hypotheses are returned to the focus of attention module for the search to continue. Rejected hypotheses are sent to the accept/reject module to be undone and then control is returned to the focus of attention module.

The first stage of the search process is shown in Fig. 13. The search starts with the seed {1} which has been hypothesized to be a parallelepiped. {1, 2} is accepted as a combined hypothesis and becomes the new seed; in accordance with the discussion in Section 3, the object corresponding to {1, 2} will be a parallelepiped. Subsequently, the hypothesis {1, 2, 3} is not accepted because surface 3 is far from surfaces 1 and 2 and its edges are not aligned with the edges of the object hypothesis. So, as shown in Fig. 13, the search backtracks to the hypothesis {1, 2} and, then, accepts the hypothesis {1, 2, 6}. Next, the hypothesis {1, 2, 3, 6} is not accepted, again because surface 3 is far from surfaces 1, 2, and 6 and its edges are not aligned with the edges of surface 1. However, {1, 2, 5, 6} is accepted, again as a parallelepiped object. But the hypothesis {1, 2, 3, 5, 6} is not accepted for the same reasons that related to surface 3 before. {1, 2, 5, 6, 7} also is not accepted, but this time the reason is that the inclusion of surface 7 changes the object type for the hypothesis from parallelepiped to cylinder and the planar surfaces do not fit the cylindrical object model. Speaking in a more abbreviated fashion about the rest of the search: {1, 2, 5, 6, 8} is accepted. {1, 2, 3, 5, 6, 8} is not accepted. {1, 2, 5, 6, 7, 8} is not accepted. {1, 2, 5, 6, 8, 9} is accepted. At this point no further combinations are possible with the seed hypothesis {1, 2, 5, 6, 8, 9} so the surfaces assigned to this seed are removed from the surface queue. The new surface queue contains [3, 4, 7]; so {3}, hypothesized to be parallelepiped, is the new seed hypothesis. {3, 4}, also a parallelepiped, is accepted. At this point no further combinations are possible with the seed hypothesis {3, 4}. Surfaces 3 and 4 are now removed from the surface queue and the new surface queue contains [7]. So {7} is the new seed hypothesis but no combinations are possible so surface 7 is removed from the surface queue. The surface queue is now empty and therefore a complete scene interpretation has been found: {{1, 2, 5, 6, 8, 9}, {3, 4}, {7}}. This interpretation consists of three object hypotheses, the first two being parallelepipeds and the third a cylinder.

The focus of attention module sends this scene interpretation to the geometric reasoning module (module 9 in Fig. 3) where it is checked for geometric consistency; that is, the control now shifts to the Geometric Reasoning Loop. It determines that the parallelepiped hypothesis {1, 2, 5, 6, 8, 9} intersects the cylinder hypothesis {7}, so the search backtracks once again. To backtrack, module 8 in Fig. 3 undoes the latest combination that resulted in the composite hypothesis {1, 2, 5, 6, 8, 9}; in other words, {1, 2, 5, 6, 8, 9} is uncombined into {1, 2, 5, 6, 8} and {9}.

Now the flow of control is handed back to the Hypothesis Evaluation Loop. The surface queue is re-initialized:

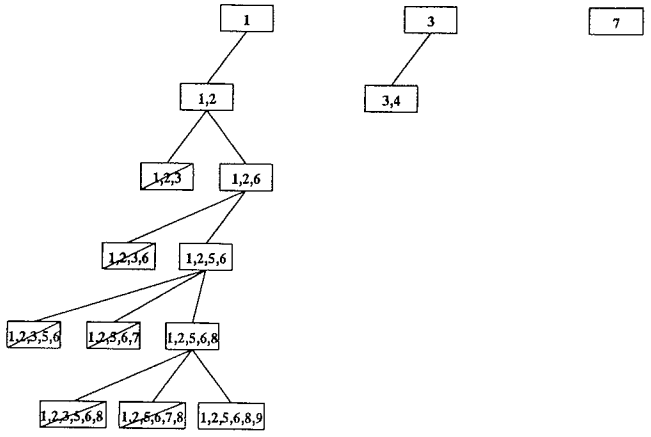


FIG. 13. First scene interpretation.

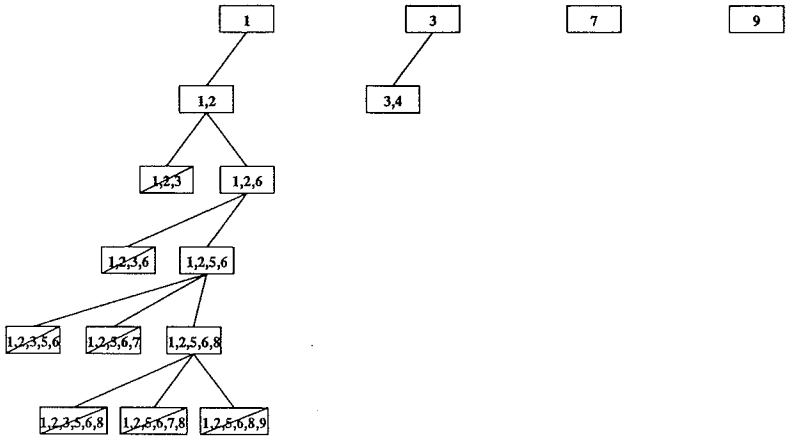


FIG. 14. Second scene interpretation.

[1, 2, 3, 4, 5, 6, 7, 8, 9], while the scene interpretation is {{1, 2, 5, 6, 8}, {3, 4}, {7}, {9}}. The hypothesis {1, 2, 5, 6, 8} becomes the new seed because it contains the first member of the surface queue. No combinations are possible so {1, 2, 5, 6, 8} is unchanged and the new surface queue is [3, 4, 7, 9]. The next seed is {3, 4}. No combinations are possible so {3, 4} is unchanged and the new surface queue is [7, 9]. The next seed is {7}. No combinations are possible so {7} is unchanged and the new surface queue is [9]. The next seed is {9}. No combinations are possible so {9} is unchanged and the new surface queue is empty. Thus, as shown in Fig. 14, a second complete scene interpretation has been found: {{1, 2, 5, 6, 8}, {3, 4}, {7}, {9}}. This interpretation consists of four object hypotheses, the first two being parallelepipeds, the third a cylinder, and the fourth a parallelepiped.

The focus of attention module sends this scene interpretation to the geometric reasoning module where it is checked for geometric consistency. In other words, the control shifts back to the Geometric Reasoning Loop. Module 9 discovers that the hypothesis {1, 2, 5, 6, 8} intersects the hypotheses {7} and {9}; this initiates backtracking which results in the undoing, by module 8, of the latest combination that resulted in the composite hypothesis {1, 2, 5, 6, 8}. The result of this uncombination is the two hypotheses {1, 2, 5, 6} and {8}.

Now the control shifts back to the Hypothesis Evaluation Loop. The surface queue is therefore re-initialized to [1, 2, 3, 4, 5, 6, 7, 8, 9]. The hypothesis {1, 2, 5, 6} becomes the new seed because it contains the first surface in the queue. The combination of the seed with {9} is accepted resulting in the composite hypothesis {1, 2, 5, 6, 9}, as shown in Fig. 15. This combined hypothesis cannot be further combined with any other hypotheses, so the new surface queue is [3, 4, 7, 8]. The next seed, {3, 4}, cannot be combined with any other hypotheses, so is left

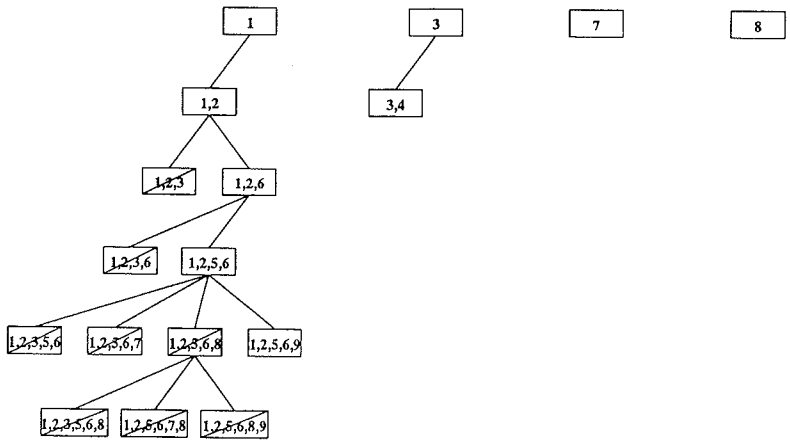


FIG. 15. Third scene interpretation.

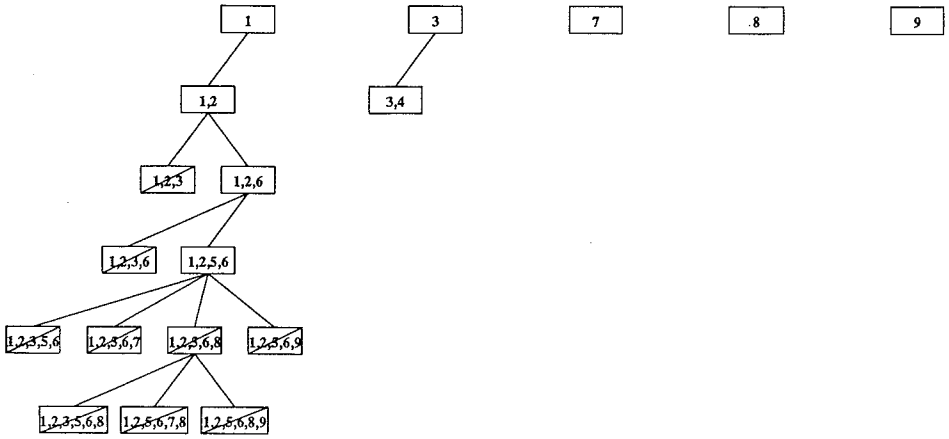


FIG. 16. Fourth scene interpretation.

unchanged, resulting in the new surface queue [7, 8]. Along similar lines, the resulting members of the surface queue are considered in turn as seeds, but neither can be further combined. Therefore, as shown in Fig. 15, a third complete scene interpretation has been found: {{1, 2, 5, 6, 9}, {3, 4}, {7}, {8}}. This interpretation consists of four object hypotheses, the first two being parallelepipeds, the third a cylinder, and the fourth a parallelepiped. The flow of control is now handed back for intersection detection to the Geometric Reasoning Loop, where it is determined that the hypothesis {1, 2, 5, 6, 9} intersects the hypotheses {7} and {8}, causing backtracking to be initiated. The backtracking undoes the latest combination that resulted in {1, 2, 5, 6, 9}, resulting in two constituent hypotheses, {1, 2, 5, 6} and {9}.

The flow of control is now handed back to the Hypothesis Evaluation Loop. Therefore, the surface queue is re-initialized to [1, 2, 3, 4, 5, 6, 7, 8, 9]. The hypothesis {1, 2, 5, 6} becomes the new seed because it contains the first surface in the queue. No combinations are possible for this seed, so the hypothesis {1, 2, 5, 6} is left unchanged. The new surface queue becomes [3, 4, 7, 8, 9] and the new interpretation is {{1, 2, 5, 6}, {3, 4}, {7}, {8}, {9}}. Subsequently, the hypotheses {3, 4}, {7}, {8}, and {9} are each considered as seeds in turn, but none can be combined with any other hypotheses. Therefore, as shown in Fig. 16, a fourth complete scene interpretation has been found: {{1, 2, 5, 6}, {3, 4}, {7}, {8}, {9}}. This interpretation consists of five object hypotheses, the first two being parallelepipeds, the third a cylinder, and the last two also parallelepipeds. The flow of control now shifts back to the Geometric Reasoning Loop, where it is determined that the hypothesis {1, 2, 5, 6} intersects the hypotheses {7}, {8}, and {9}. This initiates backtracking, resulting in the undoing of the latest combination which led to the hypothesis {1, 2, 5, 6}. The uncombining process recovers {1, 2, 6} and {5}.

The flow of control now shifts back to the Hypothesis Evaluation Loop. The surface queue is re-initialized to [1, 2, 3, 4, 5, 6, 7, 8, 9] and the hypothesis {1, 2, 6} becomes the new seed because it contains the first surface in the queue. The combination hypothesis {1, 2, 6, 7} is not accepted because the object type for this hypothesis is cylindrical owing to the presence of surface 7 and because the other surfaces, 1, 2, and 6, which are planar, do not support the cylindrical object hypothesis. Subsequently, as shown in Fig. 17, hypothesis {1, 2, 6, 8} is accepted. But, {1, 2, 3, 6, 8} is not accepted, because surface 3 is too far away. The next combination hypothesis, {1, 2, 5, 6, 8}, is not even evaluated because it was considered previously and found unacceptable. The next combination, {1, 2, 6, 7, 8}, is also not accepted for reasons that should be readily apparent by this time. Then, the hypothesis {1, 2, 6, 8, 9} is accepted, and no other combinations are possible. So the new surface queue is [3, 4, 5, 7] and the next seed is {3, 4}. No combinations

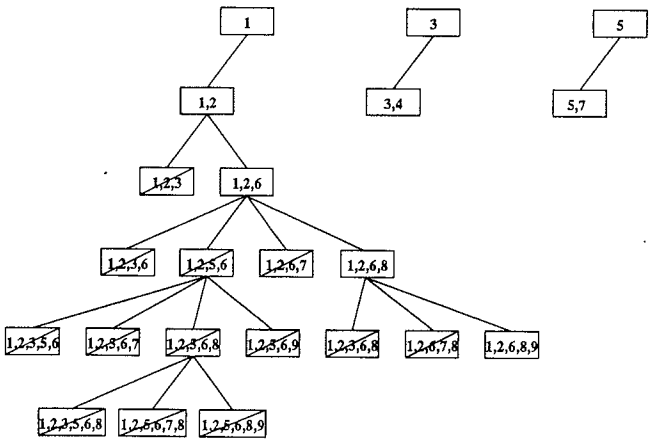


FIG. 17. Fifth (and final) scene interpretation.

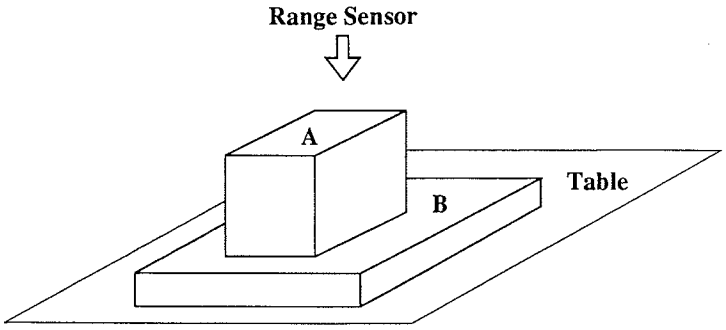


FIG. 18a. Example scene.

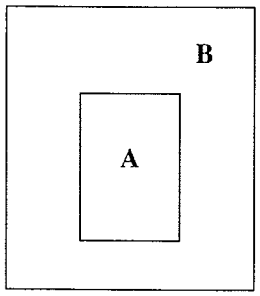


FIG. 18b. Assume that the segmentation of the range image of the scene in Fig. 18a is as shown here; in other words, the segmentation is free of artifacts.

FIG. 18c. Combinability graph for the example scene.

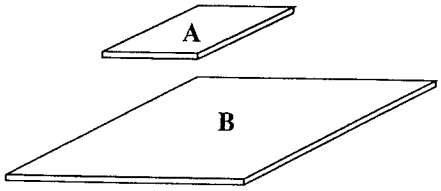


FIG. 18d. Hypothesized objects for the example scene.

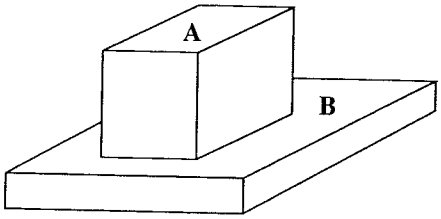


FIG. 18e. The thin-parallelepiped object hypotheses in Fig. 18d are grown in the direction away from the sensor to their maximum allowable dimensions by module 9.

are possible for the new seed, so the hypothesis {3, 4} is left unchanged, and the new surface queue is [5, 7]. The next seed is {5}, which leads to the acceptable combination hypothesis {5, 7}. Again, no other combinations are possible, leading this time to an empty surface queue. A fifth complete scene interpretation has therefore been found: {{1, 2, 6, 8, 9}, {3, 4}, {5, 7}}. This interpretation consists of three object hypotheses, the first two being parallelepipeds and the third a cylinder. The focus of attention module sends this scene interpretation to the geometric reasoning module where it is found to be free of any intersections, and, therefore, geometrically consistent. Thus, the fifth scene interpretation is the final scene interpretation.

To illustrate an aspect of the Geometric Reasoning module not made clear by the above example, assume that the scene being analyzed looks as shown in Fig. 18a and the range sensor is looking straight down. We will further assume that there are no segmentation artifacts, since the presence of these artifacts is not germane here. Therefore, we may assume that the segmentation map of this scene looks as shown in Fig. 18b. The initial hypotheses for this case would consist of two very thin parallelepipeds, shown in Fig. 18d, and the combinability graph would consist of two unconnected nodes A and B, as shown in Fig. 18c. For obvious reasons, the Hypothesis Evaluation Loop would leave these two hypotheses unchanged. Therefore, the hypotheses would be sent to the Geometric Reasoning Loop.

To augment our explanation of the Geometric Reasoning Loop in the previous example, module 9 for intersection detection does not merely do intersection detection; this module also grows the object hypotheses to their maximal allowable dimensions in directions not visible to the sensor. The object hypotheses are considered from bottom to top; so the object hypothesis B would be grown first until it contacted the table, and the object hypothesis A would be grown until it contacted either the table or object B. This growing process, discussed in Section 6, would lead to the two hypotheses shown in Fig. 18e. As these two hypotheses will be found not to intersect, they will be declared to be a valid scene interpretation.

The point being made here is that, in addition to intersection detection, the Geometric Reasoning Loop is also capable of modifying object hypotheses. As our example here has shown, the Hypothesis Evaluation Loop pro-

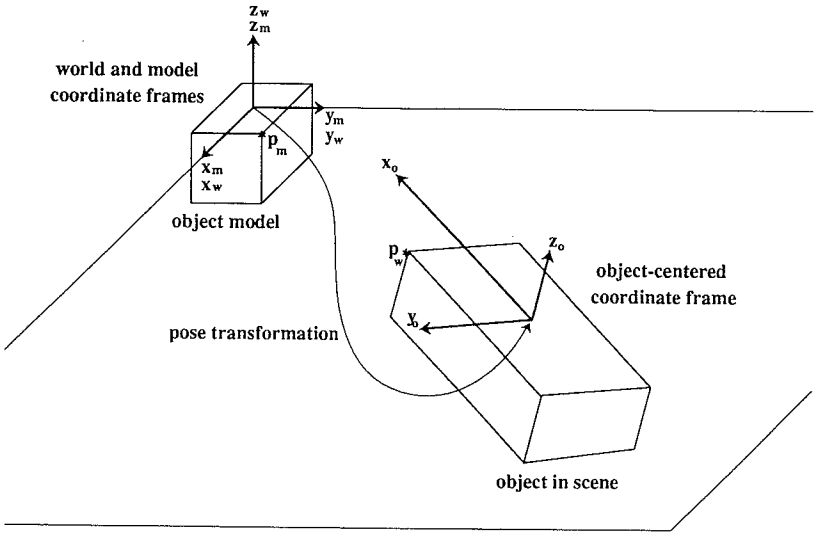


FIG. 19. Coordinate frames and the pose transformation.

duced a scene interpretation with the hypotheses as shown in Fig. 18d, and the Geometric Reasoning Loop altered the hypotheses to those shown in Fig. 18e as part of testing them for intersection detection. Note that modifications of object hypotheses by themselves are incapable of initiating backtracking.

3. COMPUTING THE POSITION, ORIENTATION, AND SCALE ASSOCIATED WITH AN OBJECT HYPOTHESIS

An object hypothesis is represented by a set of attribute–value pairs. The most important attributes are

- List of surface segments—these are the symbolic names of the surface segments participating in the hypothesis.
- Object type—rectangular parallelepiped, right elliptical cylinder, or irregular.
- Postal classification—letter, flat, box, cylinder, or irregular.
- Belief—measure of how well the data fit the model.
- Pose (position, orientation, and scale) transformation—specifies the coordinate transformation from the model coordinate frame to the world coordinate frame.

As discussed in Appendix A, the preprocessing section of INGEN (modules 1 through 3 of Fig. 3) generates a hierarchical symbolic description of the scene in which each extracted surface segment, represented by a symbolic record with fields for surface types, average location, average surface normal, etc., contains pointers to the edges that form the boundaries of the segment, with the edges themselves represented symbolically via appropriate records. The first hypothesis attribute mentioned above—list of surface segments—is simply a list of the symbolic names of the surface segments in that hypothesis.

For the second attribute for a hypothesis, object types are declared on the basis of the following criteria: An object with one or more cylindrical surface segments is classified as a cylinder; an object with one or more irregular surface segments and no cylindrical surface segments is classified as an irregular; and an object with all planar surface segments is classified as a parallelepiped.

Owing to space limitations, we will not go into issues dealing with postal classification, the determination of which is relatively straightforward and based primarily on the different dimensions associated with a hypothesis and, of course, the object type; however, further information on this subject is available in [16].

The fourth attribute associated with an object hypothesis, meaning the belief value, is discussed in detail in Section 5.

We would now like to explain how pose transformations are calculated in INGEN and to point out the differences from such calculations for the more traditional feature-based recognition systems, such as those discussed in [5, 6, 9, 11–13, 21].[6] For generic object recognition, in addition to the computation of the translation vector and the rotation matrix, it is also necessary to calculate the

[6] As soon as a new object hypothesis is created, the values of its attributes, including its pose transform, must be calculated. Since both modules 4 and 6 are capable of forming new hypotheses, the former from the segmented range map and the latter by combining hypotheses into "larger" hypotheses, the pose transform calculation is invoked by both these modules, as shown in Fig. 3.

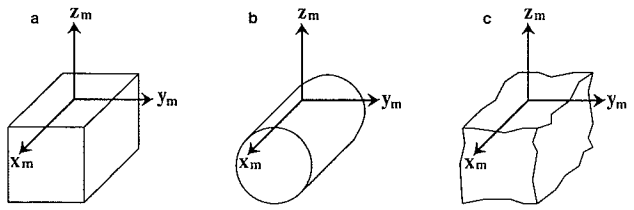


FIG. 20. Model coordinate frames.

scaling parameters since all the models are defined with unit dimensions, the purpose of the models being only to capture the shapes involved. The scaling parameters give us the actual dimensions of an object in the scene vis a vis its unit dimensions in the model space.

For computational purposes, the space in which all the models are defined may be considered to be coincident with the coordinate frame defining the world in which the scene resides. Figure 19 shows the relationship between the various coordinate frames. The axes ($\mathbf{x}_w$, $\mathbf{y}_w$, $\mathbf{z}_w$) define the world frame. When a model is defined, it initially is positioned in some standard pose at the origin of the world frame, with its various dimensions set to unity. When the same object is found in the world, as shown in Fig. 19 by the object on the right, the coordinate transformation that takes us from the unit-dimension model object to the object as it occurs in the world is the pose transformation. By "takes us" we mean that if we apply the pose transformation to, say, the point $\mathbf{p}_m$ in the model space, we get the world-frame coordinates of the corresponding point $\mathbf{p}_w$. Note that the pose transformation can be thought of as specifying an object-centered coordinate frame for an object hypothesis in the world coordinate frame. In other words, if we associate a coordinate frame with a model object—as for example depicted by the ($\mathbf{x}_m$, $\mathbf{y}_m$, $\mathbf{z}_m$) frame in Fig. 19—the pose transform then takes this frame into the corresponding object-centered coordinate frame ($\mathbf{x}_o$, $\mathbf{y}_o$, $\mathbf{z}_o$) in the world. The lengths of the axes of the ($\mathbf{x}_o$, $\mathbf{y}_o$, $\mathbf{z}_o$) frame represent the dimensional scaling of the object in the world with respect to its dimensions in the model space.

Figure 20 shows how each generic model object is defined in the model space. For a unit parallelepiped, which is the same thing as a cube, the top surface contains the origin at its center point, whereas the axes of the cube are parallel to the coordinate axes. For a cylinder, the origin is assumed to lie on the cylindrical surface and its axis parallel to the $\mathbf{x}_m$ axis. The irregular object is defined in essentially the same manner as the cube, except that the origin is now contained in a plane that is tangential to the top surface of the object. The choice of the placement of the origin, as shown in the figure, simplifies pose transform calculations and eliminates the extra computations that would otherwise be required when a suction gripper is used for manipulating the objects found in the world. As mentioned before, all the model objects have unit dimensions. For the irregular object, that means that the spacing between the planar surfaces tangential to the outside of the object is unity.

The pose transformation can be decomposed into three parts representing the rotation, translation, and scaling necessary to transform points in the model coordinate frame into the world coordinate frame. The fact that a point $\mathbf{p}_m$ in the model coordinate frame is related to the corresponding point $\mathbf{p}_w$ in the world coordinate frame by the pose transformation can be mathematically expressed by

$$\mathbf{R} \cdot \mathbf{S} \cdot \mathbf{p}_m + \mathbf{t} = \mathbf{p}_w \tag{1}$$

where $\mathbf{R}$ is a 3×3 rotation matrix, $\mathbf{S}$ is a 3×3 scaling matrix, and $\mathbf{t}$ is a translation vector. Additionally, a directional vector $\mathbf{v}_m$ in the model coordinate frame is related to the corresponding directional vector $\mathbf{v}_w$ in the world coordinate frame by the transformation

$$\mathbf{R} \cdot \mathbf{v}_m = \mathbf{v}_w. \tag{2}$$

The translation vector $\mathbf{t} = [t_x, t_y, t_z]$ specifies the position of the origin of the object-centered coordinate frame in the world with respect to the model coordinate frame. The rotation matrix $\mathbf{R}$ is given by

$$\mathbf{R} = \begin{bmatrix} x_x & y_x & z_x \\ x_y & y_y & z_y \\ x_z & y_z & z_z \end{bmatrix} \tag{3}$$

and specifies the orientation of the axes of the object-centered coordinate frame in the world with respect to the model coordinate frame. The scaling matrix $\mathbf{S}$ is given by

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix} \tag{4}$$

and specifies the scaling along the axes of the object-centered coordinate frame with respect to the model coordinate frame. The pose transformation has nine degrees of freedom. Although the rotation matrix has nine nonzero elements, these are not independent, since only three parameters—roll, pitch, and yaw, for example—are necessary to specify a rotation transformation. The scaling matrix has three nonzero elements which are independent parameters. The translation vector also contributes three parameters.

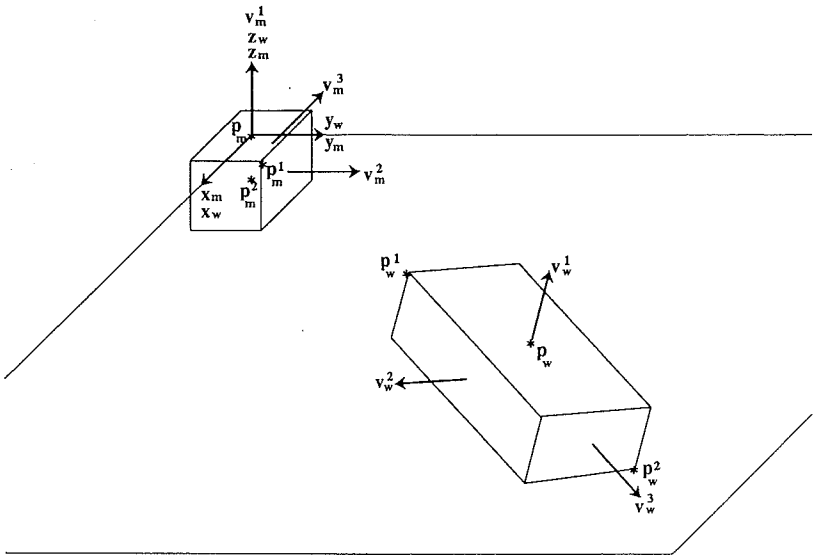


FIG. 21. Correspondences used in the computation of the pose transformation.

3.1. Computation of the Rotation Matrix

In INGEN, first the rotation matrix is computed from the correspondences of the directional vectors derived from the world. For example, Fig. 21 shows three directional vectors in the world, $\mathbf{v}_w^1$, $\mathbf{v}_w^2$, $\mathbf{v}_w^3$, and the three corresponding directional vectors in the model, $\mathbf{v}_m^1$, $\mathbf{v}_m^2$, $\mathbf{v}_m^3$. The problem of the computation of $\mathbf{R}$ may be stated as follows: Given a set of directional vectors, $\mathbf{v}_w^1$, $\mathbf{v}_w^2$, . . . , $\mathbf{v}_w^N$, associated with, say, the different surfaces of a scene object, their model correspondents, $\mathbf{v}_m^1$, $\mathbf{v}_m^2$, . . . , $\mathbf{v}_m^N$, and the set of relationships between each pair of correspondences

$$\mathbf{R} \cdot \mathbf{v}_m^i = \mathbf{v}_w^i, \tag{5}$$

we want to estimate the rotation matrix $\mathbf{R}$ which minimizes

$$\sum_{i=1}^{N} |\mathbf{R} \cdot \mathbf{v}_m^i - \mathbf{v}_w^i|^2. \tag{6}$$

On account of dependencies between the elements of $\mathbf{R}$, the minimization of this criterion is not amenable to the traditional methods found in matrix algebra, such as by the method of pseudo-inverse (for a discussion of this point, see, for example, [21]), and recourse must be made to a quaternion-based approach first advanced by Faugeras and Hebert [9] and used subsequently in [5]. In the quaternion-based approach, the object rotation is expressed by means of a four dimensional vector, called a quaternion,

$$\mathbf{Q} = \left(\cos\frac{\theta}{2}, \sin\frac{\theta}{2}\,\mathbf{a}\right), \tag{7}$$

where $\mathbf{a}$ is a unit vector along the axis of rotation and θ is the angle of rotation. It can be shown that the optimum $\mathbf{Q}$ is an eigenvector of the matrix

$$\mathbf{A} = \sum_{i=1}^{N} \mathbf{B}_i \mathbf{B}_i^T, \tag{8}$$

where

$$\mathbf{B}_i = \begin{bmatrix} 0 & -c_x^i & -c_y^i & -c_z^i \\ c_x^i & 0 & b_z^i & -b_y^i \\ c_y^i & -b_z^i & 0 & b_x^i \\ c_z^i & b_y^i & -b_x^i & 0 \end{bmatrix} \tag{9}$$

and

$$\begin{aligned} \mathbf{b}^i &= \mathbf{v}_m^i + \mathbf{v}_w^i \\ \mathbf{c}^i &= \mathbf{v}_m^i - \mathbf{v}_w^i. \end{aligned} \tag{10}$$

The eigenvector to be used must correspond to the minimum eigenvalue. A tutorial rederivation of this result of Faugeras and Hebert [9] can be found in [5].

The direction vectors used in the computation of the rotation matrix include the average surface normals for planar, irregular, and cylindrical surfaces as well as the cylindrical axis for cylindrical surfaces. The use of the

average surface normal for a cylindrical surface might come as a surprise to the reader because, if one could see the entire cylindrical surface, the average surface normal would be zero. However, in actual scenes, only a portion of a cylindrical surface will be visible; the average surface normal associated with this part serves to define the $\mathbf{z}_o$ axis in the scene, which will be a transformed version of the $\mathbf{z}_m$ axis shown in (b) in Fig. 20. (A $\mathbf{z}_o$ axis is shown in Fig. 19 for the parallelepiped case. A similar axis may be defined for the cylindrical surface in the scene; the $\mathbf{z}_o$ axis now would be perpendicular to the visible portion of the cylindrical surface.)

For the above mentioned computations to yield results for the optimum orientation matrix, it is necessary that we have at least two nonparallel directional vectors for two different entities in the scene. However, the initial hypotheses are based on individual surface segments; in other words, initially we associate a separate object hypothesis with each separate surface segment extracted from the scene. Clearly, for the very first hypotheses it will not be possible to extract two nonparallel direction vectors if these vectors are to be normal to surfaces. The same problem can arise when composite object hypotheses are formed from multiple surface segments in the scene if all these surface segments are coplanar. Two approaches are used to resolve this problem.

First, the system looks for object edges of either occluding or convex type that have directions different from the other available directional vectors. Using only these edge types ensures that the system is not fooled by shadows or occlusion effects. If the longest of these edges has a length that is a significant fraction (currently we use 50%) of the estimated length of the object, then the edge is matched with its corresponding model edge and used as a direction vector. If no appropriate edges are found then the major axes of the surfaces of the object are searched until one is found which has a direction different from the other available direction vectors. This axis is matched with a model edge along the major axis of the object and so it can be used as a direction vector.

3.2. Computation of the Scaling Parameters

The scaling parameters, represented by the matrix **S** in Eq. (1), are computed next. Suppose we know the coordinates of a pair of object points $\mathbf{p}_m^1$ and $\mathbf{p}_m^2$ in the model space and further suppose we also have found the scene correspondents of these object points, with the scene points labeled $\mathbf{p}_w^1$ and $\mathbf{p}_w^2$. Using Eq. (1), it is easy to show that these four points are related by

$$\mathbf{R} \cdot \mathbf{S} \cdot (\mathbf{p}_m^1 - \mathbf{p}_m^2) = (\mathbf{p}_w^1 - \mathbf{p}_w^2). \tag{11}$$

This equation says that the scaling matrix may be derived from

$$\mathbf{S} = \mathbf{R}^{-1} \cdot (\mathbf{p}_w^1 - \mathbf{p}_w^2) \cdot (\mathbf{p}_m^1 - \mathbf{p}_m^2)^{-1}. \tag{12}$$

Of course, if it is possible to extract more than one pair of points from the scene object and their correspondents from the model space, an optimum **S** may be computed by constructing a pseudo-inverse solution. Note that unlike the case of **R** it is possible to construct a pseudo-inverse solution here since the three elements that define **S** are all independent.

For this approach to yield results for **S**, it is necessary that the pair of points selected not lie in a plane that is perpendicular to any of the axes in the respective object centered coordinate frame. For example, the vector $\mathbf{p}_m^1 - \mathbf{p}_m^2$ should not be perpendicular to any of the axes of the $(\mathbf{x}_m, \mathbf{y}_m, \mathbf{z}_m)$ space. Similarly, the vector $\mathbf{p}_w^1 - \mathbf{p}_w^2$ should not be perpendicular to any of the axes of the $(\mathbf{x}_o, \mathbf{y}_o, \mathbf{z}_o)$ space.

At this juncture, the reader is probably wondering how we might identify points like $\mathbf{p}_w^1$ and $\mathbf{p}_w^2$ in a scene, especially in the presence of occlusions. In INGEN, these two points in the scene correspond to the "max" point and the "min" point of the minimum bounding box that contains the object and whose sides are parallel to the axes of model coordinate frame rotated through **R**. The "max" point is that vertex whose three coordinate values are the maximum of all the vertices available and the "min" point that vertex whose coordinates are the minimum. Note that a rotation of the model frame through **R** gives us a coordinate frame that is parallel to the still-unknown object-centered coordinate frame [the $(\mathbf{x}_o, \mathbf{y}_o, \mathbf{z}_o)$ frame shown in Fig. 19] and whose origin is at the world origin. For example, in Fig. 22 the rotated model coordinate frame is designated by the $(\mathbf{Rx}_m, \mathbf{Ry}_m, \mathbf{Rz}_m)$ axes. This approach is guaranteed to yield two points that meet the necessary conditions for the calculation of the scaling parameters. For example, for the scene object shown in Fig. 21, the points selected would be $\mathbf{p}_w^1$ and $\mathbf{p}_w^2$ as shown there.

The calculation of the "max" and the "min" points is carried out by computing the distances of the data vertices from the three perpendicular planes defining the model coordinate frame rotated through **R**. By retaining the maximum and minimum values of such distances, we obtain the minimum bounding box and the points $\mathbf{p}_w^1$ and $\mathbf{p}_w^2$. It is important to note that $\mathbf{p}_w^1$ and $\mathbf{p}_w^2$ may not correspond to any actual vertex in the data. In that sense, $\mathbf{p}_w^1$ and $\mathbf{p}_w^2$ may be referred to as virtual points. The x, y, and z coordinates of such a virtual point could be defined by the x-coordinate of one data vertex, y-coordinate of another, and z-coordinate of yet another vertex.

To illustrate, let us consider a two-dimensional example shown in Fig. 23, where we have shown three vertices marked A, B, and C. To fit a minimum bounding rectangle, whose sides are parallel to the coordinate axes, we

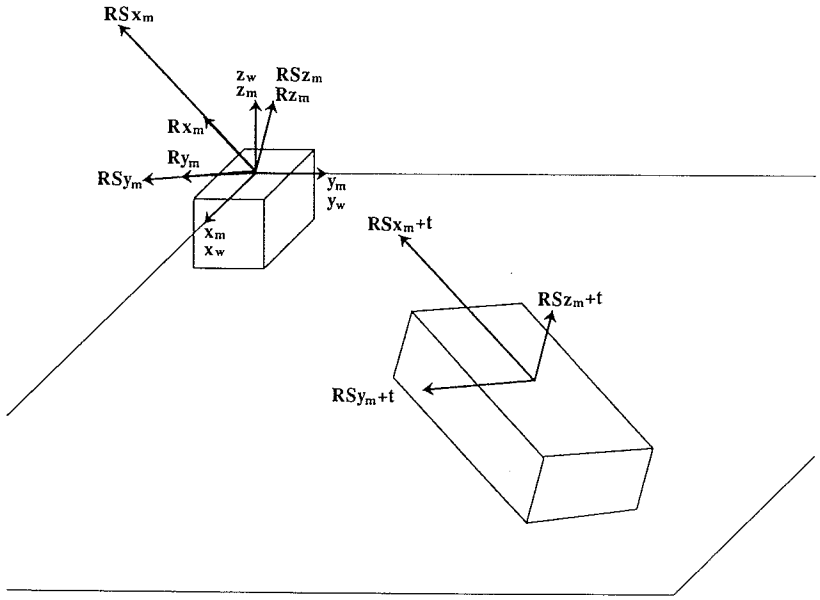


FIG. 22. Coordinate frames computed as steps in the computation of the pose transformation.

take the maximum of the x-coordinates of the vertices and make this maximum the x-coordinate of the max point of the minimum bounding rectangle. We then take the maximum of the y-coordinates of the vertices and make this maximum the y-coordinate of the max point of the minimum bounding rectangle. The max point thus obtained is marked P; clearly, this point does not correspond to any point in the data. To get the min point of the minimum bounding rectangle, we take the minima of the coordinates, and obtain the point Q.

This approach to finding suitable $\mathbf{p}_w^1$ and $\mathbf{p}_w^2$ is necessitated by the fact that, especially in generic shape recognition where data can be very noisy, it frequently is not possible to extract points from the data that can be correlated accurately with their counterparts on the shape models.

A special case of the above procedure arises when the range sensor is directly above a planar surface of an object whose other sides are occluded to the sensor, as was the case in the hypothetical example shown in Fig. 18a. In this case, the object hypotheses would have to be formed from a single planar surface and, for the case of perfect data, the points p_{w_1} and p_{w_2} would not lie on the two opposite corners of a parallelepiped, but on the two opposite corners of a parallelogram. To handle such cases, we associate a default minimum thickness with parallelepiped object hypotheses, this thickness being equal to the depth resolution of the range sensor. If h is this thickness, then the p_{w_2} obtained by the above procedure is modified by decreasing its rotated z-coordinate value by h.

3.3. Computation of the Translation Vector

After the rotation and scale matrices have been computed it is possible to compute the translation vector. Using the rotation and scale matrices, we can construct, at the origin of the world frame, a coordinate frame whose axis directions are parallel to the still-unknown object-centered coordinate frame and whose axis magnitudes are governed by the scale parameters. For example, in Fig. 22 the rotated and scaled model coordinate frame is designated by the ($\mathbf{RSx}_m$, $\mathbf{RSy}_m$, $\mathbf{RSz}_m$) axes. What remains now is the computation of the translation vector $\mathbf{t}$ that would be able to displace this rotated and scaled coordinate frame to its proper location on the object.

To specify the translation vector we need to specify at

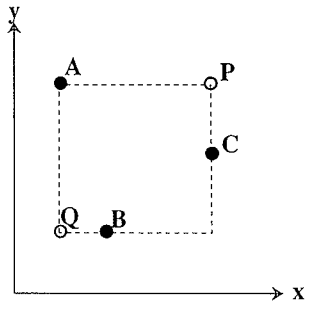


FIG. 23. Two-dimensional example showing how the virtual points P and Q that define the minimum bounding rectangle, with sides parallel to the axes, are obtained from the data points A, B, and C.

least one single point in the data and its model correspondent. Given such a point, we compute the translation vector by

$$\mathbf{t} = \mathbf{p}_w - \mathbf{R} \cdot \mathbf{S} \cdot \mathbf{p}_m. \quad (13)$$

For reasons similar to those mentioned in our discussion on the computation of the scaling parameters, it is usually not possible to identify a unique point in the scene for this purpose. So our strategy is to again derive a unique virtual point from the two virtual points extracted for the computation of scaling parameters. This virtual point is selected in a manner analogous to the definition of the origin of a model shape (see Fig. 20): The point is located at the center of the topmost face of the minimal bounding box for the object in the ($\mathbf{RSx}_m$, $\mathbf{RSy}_m$, $\mathbf{RSz}_m$) coordinate frame. For example, for the scene object shown in Fig. 21, the point selected would be $\mathbf{p}_w$. Thus, in Fig. 22 the final object-centered coordinate frame is designated by the ($\mathbf{RSx}_m + \mathbf{t}$, $\mathbf{RSy}_m + \mathbf{t}$, $\mathbf{RSz}_m + \mathbf{t}$) axes which correspond to the ($\mathbf{x}_o$, $\mathbf{y}_o$, $\mathbf{z}_o$) axes in Fig. 19. This virtual point may not be an actual object point, particularly for the cases of cylinders and irregular object hypotheses, but can always be computed readily and is relatively insensitive to occlusion. Additionally, this point also is very useful as a pickup point for a robot with a suction gripper.

4. COMBINABILITY OF SURFACE SEGMENTS

The primary task of INGEN is to partition the set of surface segments into object hypotheses that form a consistent scene interpretation. The naive approach would be to search through all possible partitions until a satisfactory interpretation is found. However, the number of partitions is enormous even for scenes with only a few surface segments, so a different approach is required. INGEN uses a constructive approach in which objects are incrementally constructed from pairwise combinations of surfaces. The primary advantage of this approach is that many potential surface combinations can be ruled out during a preprocessing stage which results in a considerable reduction in the size of the search space.

There are two types of combination operations: merging and aggregation. Merging is carried out when two surface segments belong potentially to the same face of the same object. On the other hand, aggregation is carried out when two surface segments belong potentially to two separate faces of the same object. Thus, merging involves combining surface segments that are separate because a single object surface was segmented into multiple parts due to noise or occlusion. Aggregation involves combining surface segments that are separate because distinct object faces are present in the scene. While the criteria for merging depend only on the types of surfaces expected to be in the scene, the criteria for aggregation depend on the types of objects expected.

When examining a pair of surfaces under consideration for combination we need only consider whether the surfaces are related in such a way that merging or aggregation is possible. Knowledge about object hypotheses is not necessary in making this determination. Thus, the determination of the possible combinations of surfaces can be done prior to the beginning of the search process. For the purpose of this determination, each pair of surfaces needs only to be considered once. The information is then used to guide the search process.

The information regarding the allowable combinations of surfaces is contained in a data structure known as the combinability graph. In the combinability graph each node represents a surface segment and each arc represents an allowable combination between the two surfaces that it connects. Any pair of surfaces connected in the combinability graph can potentially belong to the same object. The search for objects takes place over this graph.

Initially, to each node of the combinability graph corresponds a separate object hypothesis. The combination of two object hypothesis is carried out by creating a new object hypothesis which has all of the surfaces, edges, and vertices of the two constituent object hypotheses. The attributes for the new hypothesis are computed as described in the previous section.

4.1. Combinability Graph Generation

The combinability graph is generated from information in the hierarchical scene description at the same time as when the initial single-surface object hypotheses are generated. These two processes do not depend on each other so the order in which they are performed is not significant. The combinability graph is generated by enumerating and testing all possible pairs of surfaces. The tests evaluate the suitability of a pair of surfaces for merging or aggregation. If the tests determine that the two surface segments could potentially belong to the same object then an appropriate arc is added to the combinability graph.

4.1.1. *Merging Criteria*

The criteria for merging are based on the types of surfaces expected in the scene. Since INGEN handles scenes with three types of surfaces, planar, cylindrical, and irregular, we need criteria for coplanar, cocylindrical, and coirregular surfaces.

Two surfaces are coplanar and therefore candidates for merging if all of the following criteria are satisfied:

- both surfaces are planar
- the angle between the surface normals of the two planes defined by the surfaces is less than 2.5°

• for at least one of the surfaces, the distance from the centroid of one surface to the plane defined by the other surface is less than 0.2 in.

The second criterion ensures that the surfaces define parallel planes. The third criterion ensures that the planes are not only parallel but actually the same plane.

Two surfaces are cocylindrical and therefore candidates for merging if all of the following criteria are satisfied:

• both surfaces are cylindrical

• the angle between the cylindrical axes of the two cylindrical surfaces, modulo 180°, is less than 15°

• for both surfaces, the angle between the cylindrical axis and the line formed by connecting the centroids of the two surfaces, modulo 180°, is less than 15°

The second criterion ensures that the cylinders are parallel. The third criterion ensures that the cylinders are not only parallel but actually aligned.

Irregular surfaces are roughly planar but have surface irregularities that require that they be treated differently from planar surfaces. These irregularities have two forms. Irregular surfaces can be wrinkled or they can have global irregularities such as bulges or bending at the edges. Thus, coirregularity is similar to coplanarity but with the criteria relaxed to compensate for the irregularity of the surfaces involved.

Two surfaces are coirregular and therefore candidates for merging if all of the following criteria are satisfied:

• one surface is irregular and the other is either irregular or planar

• the angle between the surface normals of the two surfaces is less than 5.0°

• for at least one of the surfaces, the distance from the centroid of one surface to the average plane passing through the other surface is less than 0.4 in.

The second criterion ensures that the surfaces define roughly parallel planes. The third criterion ensures that the planes are not only parallel but actually the same plane.

4.1.2. *Aggregation Criteria*

The criteria for aggregation are based on the types of objects expected in the scene. Since INGEN handles scenes with three types of objects, parallelepiped, cylinder, and irregular, we need criteria for perpendicularity of planar and irregular surfaces and coaxiality of a cylindrical surface with a planar surface corresponding to an end of the cylinder. Irregular objects have essentially the same shape as parallelepipeds but have irregular surfaces instead of planar surfaces. The only relations between parallelepiped surfaces are perpendicularity relations. We disregard the case of parallel surfaces with surface normals pointing in opposite directions because of the nature of the sensors used by INGEN; it is unlikely that any single range sensor would produce data for two surfaces on opposite sides of a parallelepiped. However, if data from multiple sensors with different viewpoints are available then this case must be considered. For cylindrical objects we have a relation which we refer to as the coaxial relation. In this case we have a planar or irregular surface with a surface normal that is roughly coaxial with the cylindrical axis of a cylindrical surface.

Two surfaces are perpendicular if all of the following criteria are satisfied:

• the surfaces are either planar or irregular

• the angle between the surface normals of the two surfaces is between 70 and 100°

• the surface normals diverge; this is computed by comparing the angle between each surface normal and the vector connecting the centroids of the two surfaces

The third criterion ensures that the surfaces are convexly related.

A cylindrical surface is coaxial with a planar or an irregular surface if the following criterion is satisfied:

• the angle between the surface normal of the planar or irregular surface and the cylindrical axis of the cylindrical surface, modulo 180°, is less than 15°

4.1.3. *Adjacency Criteria*

Surfaces which are adjacent in the data but do not meet the merging or aggregation criteria can be considered for combination based on adjacency criteria. These criteria exist primarily to handle the case where a small surface is adjacent to a larger surface but either is too small to be characterized to a high degree of accuracy or has been segmented from the adjacent surface because of surface irregularities.

Two adjacent surfaces can be combined if the following criterion is satisfied:

• the jump discontinuity, if it exists, at the common boundary is less than 0.5 in.

This criterion ensures that there is not a large jump discontinuity between the two surfaces.

4.2. Combinatorics

In order to understand the utility of the combinability graph approach we must examine the combinatorics of the scene interpretation problem. The primary feature used by INGEN is the surface segment, so we will use it as the basis for discussing the complexity of the problem.

The basic task of INGEN is to partition the set of surface segments into objects which form a consistent scene interpretation.

A set with n elements has 2^n subsets; therefore the number of objects that can be hypothesized in a scene containing n surface segments is $2^n - 1$ because the empty set does not specify an object. Of course, at most n of these objects can exist at the same time in a scene interpretation and each object must contain a distinct set of surfaces. These constraints are reflected in the notion of partitioning the set of surfaces into objects. The number of unique partitions of a set with n elements [10] is given by

$$P_n = \frac{1}{n!} \sum_{i=0}^{n} (n - i)^n \binom{n}{i} D_i \tag{14}$$

which represents the number of unique scene interpretations which can be formed from n surface segments. The binomial coefficients are defined by

$$\binom{n}{m} = \frac{n!}{m!(n - m)!} \tag{15}$$

and give the number of combinations possible for n items taken m at a time. D_n is the derangement number,

$$D_n = n! \sum_{i=0}^{n} \frac{(-1)^i}{i!}, \tag{16}$$

which represents the number of derangements (permutations with no fixed points) of a set of size n.

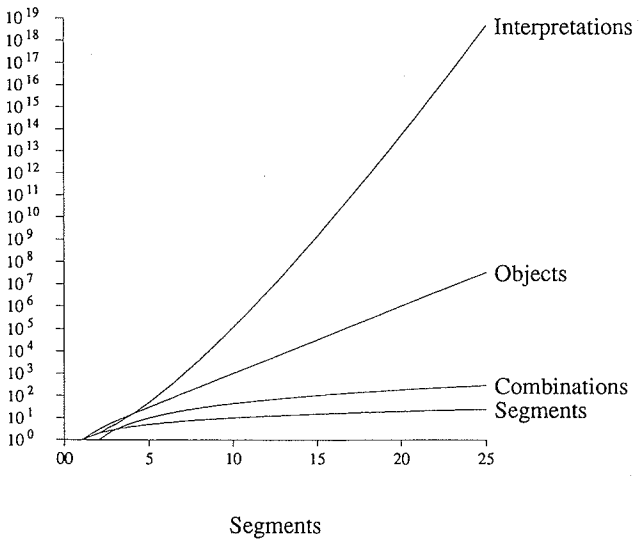


FIG. 24. Combinatorics of the scene interpretation problem.

TABLE 1
Combinatorics of the Scene Interpretation Problem

Surface segments	Combinations	Objects	Interpretations
1	0	1	1
2	1	3	2
3	3	7	5
4	6	15	15
5	10	31	52
6	15	63	203
7	21	127	877
8	28	255	4,140
9	36	511	21,147
10	45	1,023	115,975
11	55	2,047	678,570
12	66	4,095	4,213,597
13	78	8,191	27,644,437
14	91	16,383	190,899,322
15	105	32,767	1,382,958,545
16	120	65,535	10,480,142,147
17	136	131,071	82,864,869,804
18	153	262,143	682,076,806,159
19	171	524,287	5,832,742,205,057
20	190	1,048,575	51,724,158,235,372
21	210	2,097,151	474,869,816,156,751
22	231	4,194,303	4,506,715,738,447,323
23	253	8,388,607	44,152,005,855,084,346
24	276	16,777,215	445,958,869,294,805,289
25	300	33,554,431	4,638,590,332,229,999,353

The combinability graph is constructed by examining every possible pair of surfaces and determining whether or not the pair could correspond to the same object. For a scene with n surfaces there are

$$\binom{n}{2} = \frac{n(n - 1)}{2} \tag{17}$$

possible combinations involving pairs of surfaces.

For our domain we typically have $n < 25$. However, the exponential nature of the problem makes enumerative solutions impractical for scenes with 10 or more surfaces. Figure 24 shows the combinatorics of our problem. The number of surfaces (n) increases along the horizontal axis and the corresponding numbers of surface segments (n), combinations ($n(n - 1)/2$), objects ($2^n - 1$), and scene interpretations (P_n) are plotted using a logarithmic scale on the vertical axis. Table 1 contains the exact values used to construct the plots in Fig. 24. Note that the number of combinations is $O(n^2)$, the number of objects is $O(2^n)$, and the number of interpretations is $O(n^n)$.

These combinatorics assume a search through all possible scene interpretations. This is equivalent to a search over all of the partitions of a complete graph (every node is connected by an arc to every other node) with n nodes. A search over all possible partitions of the combinability

graph instead of the complete graph offers a significant reduction in the size of the search space. In the worst case the combinability graph could be complete but this case is very unlikely because of the physical constraints on the definition of the combinability graph.

The reduction in the size of the search space is based on two factors. First, the combinability graph frequently is made of several disjoint subgraphs. Each of these subgraphs can be handled independently of the others for the purposes of hypothesis combinations. However, they do affect each other during the geometric reasoning process. For example, if a scene has 10 surfaces then there are 115,975 possible scene interpretations when the combinability graph is complete. However, if the combinability graph is partitioned into two complete subgraphs with 5 surfaces each then there are only 52 possible interpretations for each subgraph giving a total of $52 \cdot 52 = 2704$ possible scene interpretations. The second factor is based on the connectivity of the combinability graph. Partitioning a set is equivalent to partitioning a complete graph. However, if the graph is not complete then a smaller number of partitions is possible. The exact size of the reduction is difficult to characterize because it depends on the particularities of the combinations in the graph. However, we have found a large reduction in our experiments. The effects of these factors are shown as we discuss three example scenes.

4.3. Example Scenes

We can see how merging and aggregation are useful by examining three example scenes. The data and range image segmentations for scenes 705, 707, and 711 are shown in Figs. 25, 29, and 33.[7] Each figure contains the registered reflectance image to show what the scene actually looks like, the segmentation of the scene into numbered surface segments, and a 3-D plot of the range data for the scene.

Figs. 26, 30, and 34 show the initial object hypotheses for the scenes. These figures show four views of each scene with the objects represented by wireframe bounding boxes. Included are orthographic projections for the top (x–y plane), front (y–z plane), and side (x–z plane) views and an axonometric projection. Figures 27, 31, and 35 show the final scene interpretations. Although it is a bit early to be presenting final results in our exposition here, we have included these figures in this section so that the final results can be compared to the initial hypotheses and related to the scene data.

The combinability graphs for these scenes are shown in Figs. 28, 32, and 36. To help the reader visually correlate the nodes with their corresponding surface segments, we have placed the nodes of the combinability graph in the same relative positions as the segments in the data. The arcs are labeled with the types of combinations possible for the surfaces: A represents adjacency, P represents coplanarity, I represents coirregularity, C represents cocylindricity, X represents coaxiality, and N represents perpendicularity.

Scene 705 (Figs. 25–28) contains four objects plus the background. The topmost object is a flat which lies across a box and divides it into two segments. The other two objects are a bundle of letters and a package of business reply cards. The scene has been segmented into 15 surfaces. As shown in Fig. 28, 23 of the 105 possible arcs are present in the combinability graph. The combinability graph consists of 4 disjoint subgraphs with 7, 6, 1, and 1 nodes. Thus, the upper bound on the search space is reduced from 1,382,958,545 scene interpretations to $877 \cdot 203 \cdot 1 \cdot 1 = 178{,}031$ possible scene interpretations. Since the subgraphs are not complete the actual number of potential scene interpretations is much smaller. Note that the combinability graph contains only 4 combinations (2–4, 3–6, 1–10, 7–10) which do not belong in the correct interpretation.

This scene contains two situations where merging was essential to finding the correct interpretation. The large box was divided into two widely separated segments because it was occluded by the flat. It also has two more small segments which are due to the rounded edges of the box. The two large surfaces, 2 and 6, were combined based on merging criteria and the two small surfaces, 8 and 9, were combined with surface 6 based on adjacency criteria. Also note that the flat was segmented into two segments, 3 and 4, because of dropouts in the range data caused by the black outline of the mailing label. These two surfaces were combined based on merging criteria. There is no adjacency information in this case because of the dropouts in the range data.

The geometric reasoning process correctly determines that the flat is very thin because it is constrained by the box that supports it and that the box is relatively thick because it is only constrained by the background surface.

Scene 707 (Figs. 29–32) contains three objects plus the background. There are two boxes stacked obliquely and a bundle of letters. The scene has been segmented into 7 surfaces. As shown in Fig. 32, 4 of the 21 possible arcs are present in the combinability graph. The combinability graph consists of 3 disjoint subgraphs with 4, 2, and 1 nodes. Thus, the upper bound on the search space is reduced from 877 scene interpretations to $15 \cdot 2 \cdot 1 = 30$ possible scene interpretations. Since the subgraphs are not complete the actual number of potential scene interpretations is $8 \cdot 2 \cdot 1 = 16$. Note that the combinability

[7] The names of the scenes correspond to those used in the data sets generated by U.S. Postal Service range data acquisition contractors. These scenes were provided by GE/RCA Advanced Technology Labs [27].

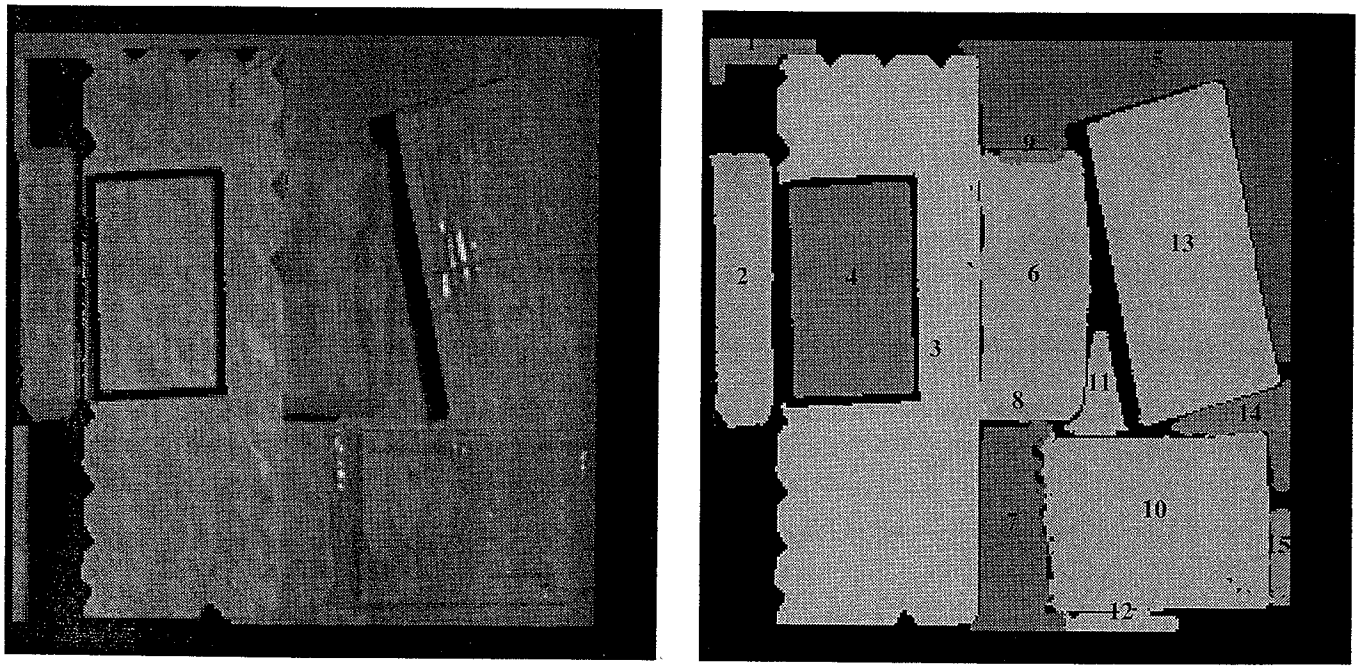


FIG. 25. Range data plot and segmentation for scene 705.

graph contains only 1 combination (4–6) which does not belong in the correct interpretation.

This scene contains two situations where aggregation was essential to finding the correct interpretation. The two boxes each have two perpendicular surfaces visible in the scene. These surfaces are combined based on aggregation criteria.

The geometric reasoning process correctly determines the dimensions of all three objects based on contacts with each other and with the background surface.

Scene 711 (Figs. 33–36) contains four cylinders overlapping each other plus the background. The scene has been segmented into 12 surfaces. As shown in Fig. 36, 12 of the 66 possible arcs are present in the combinability graph. The combinability graph consists of 4 disjoint subgraphs with 4, 3, 3, and 2 nodes. Thus, the upper bound on the search space is reduced from 4,213,597 to $15 \cdot 5 \cdot 5 \cdot 2 = 750$ possible scene interpretations. Since the subgraphs are not complete the actual number of potential scene interpretations is smaller. Note that the combin-

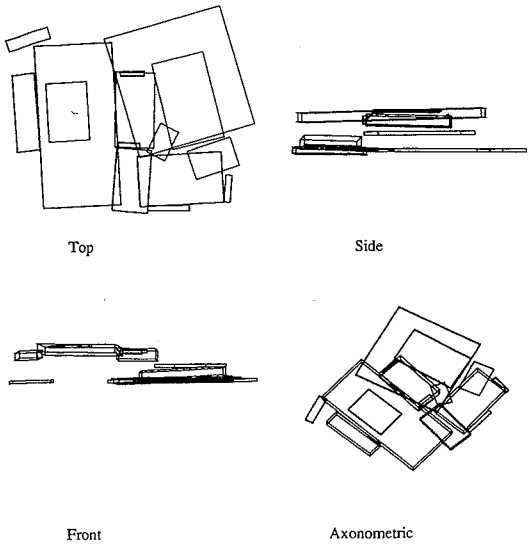


FIG. 26. Hypothesized objects for scene 705.

ability graph contains only 1 combination (3–5) which does not belong in the correct interpretation.

This scene contains three situations where merging was essential to finding the correct interpretation. Three of the cylinders have been segmented into several parts due to occlusion by other cylinders and one was also oversegmented due to noise in the data.

The geometric reasoning process correctly determines that the cylinder on the bottom of the pile must be a relatively flat elliptical cylinder because it is constrained by the background surface. The other cylinders are circular.

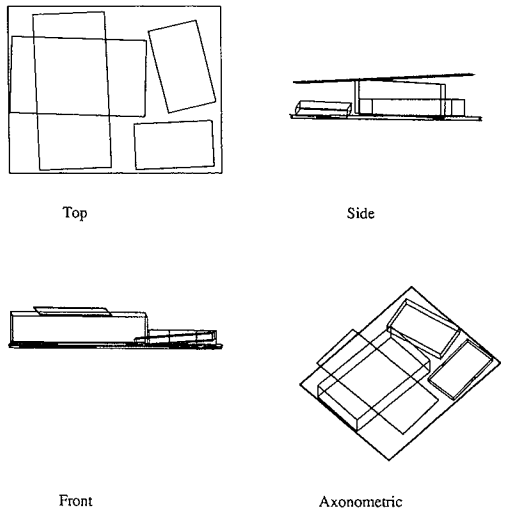


FIG. 27. Final results for scene 705.

5. EVALUATION OF OBJECT HYPOTHESES

As mentioned in Section 3, there are two mechanisms for an object hypothesis to get rejected; one resides in the loop formed by the modules 6, 7, and 8 in Fig. 3 and the other in the loop formed by the modules 6, 9, and 8. In this section we will discuss how the beliefs associated with object hypotheses are computed in the former loop, since a hypothesis with insufficient belief would be rejected. Subsequently, in the next section, we will discuss the second mechanism that is based on the detection of volumetric intersections.

The evidence accumulation formulas presented here allow us to compute the belief with which the data support a given hypothesis, taking into account the pose transform, which includes the scale factors. The belief values should be sensitive to how much of the surface area of the hypothesized object is really supported by what is present in the data. For example, if the surface segments shown in Fig. 37a lead to a parallelepiped hypothesis as shown there, the belief in that hypothesis should be less than that for the case in Fig. 37b where a much larger area of the hypothesis is covered by surface segments. Similar examples could be stated for cylinders and irregulars.

5.1. Basis of Belief

The computation of belief in an object hypothesis depends on how well the data support the object hypothesis on the basis of the following attributes of the object hypothesis:

- type
- position
- orientation
- dimensions

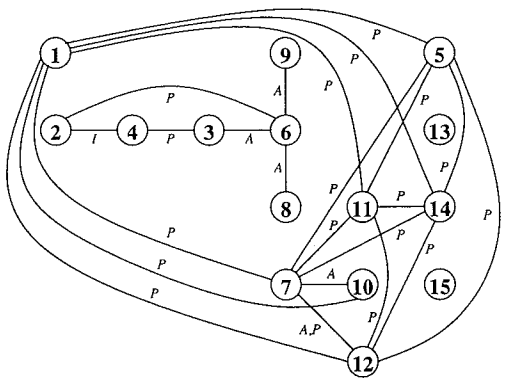


FIG. 28. Combinability graph for scene 705.

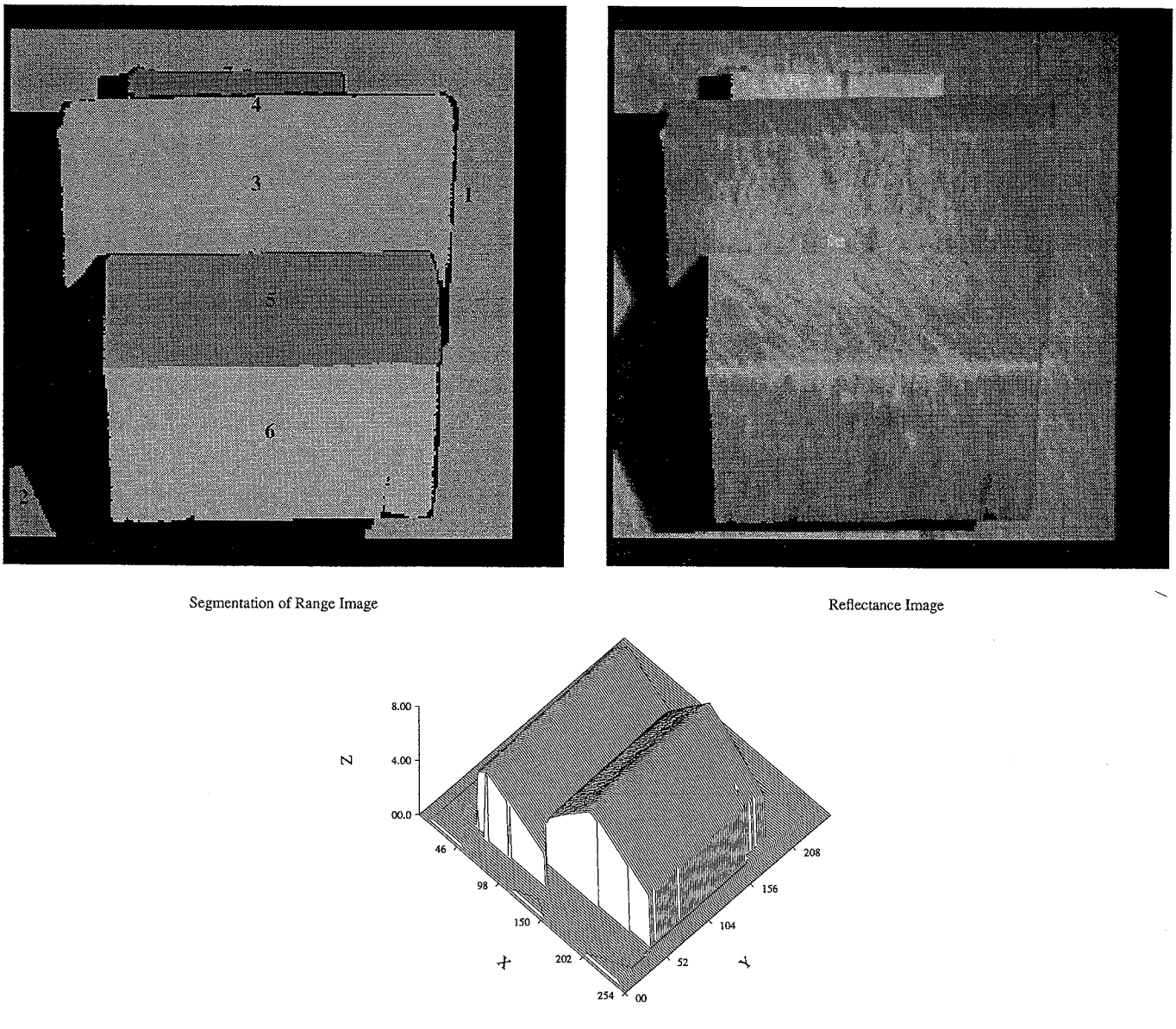


FIG. 29. Range data plot and segmentation for scene 707.

Clearly the correct determination of the type, position, and orientation of an object is critical to the usefulness of any object recognition system. However, in contrast to the more traditional 3D vision systems, INGEN must also determine the dimensions of the object. So it is important that the evaluation process examine the accuracy of the computed dimensions as well as, of course, the accuracy of the type, position, and orientation.

5.2. Evidence Sources Available for Evaluation

The evidence sources available for the evaluation of an object hypothesis are

- surfaces
- edges
- vertices

From a surface we use the surface type which relates to the object type, and the surface normal and/or cylindrical axis which relate to the object orientation. From an edge we use the orientation which relates to the object orientation. From a vertex we use the vertex position which relates to the object position and dimensions. The evaluation of the position and the dimensions of an object depends on the positions of data edges (as defined by their endpoint vertices) relative to the model edges.

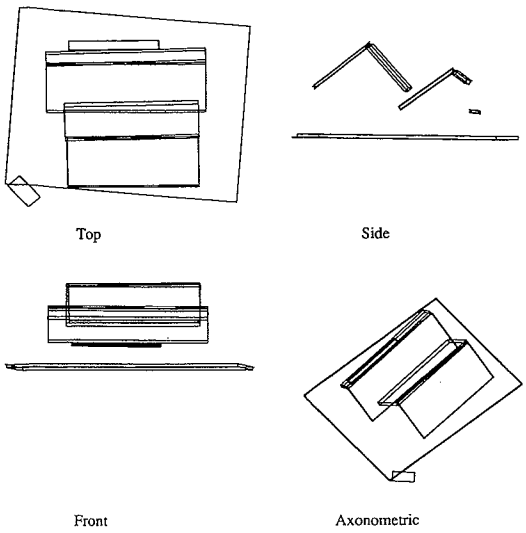


FIG. 30. Hypothesized objects for scene 707.

Experience has shown that it is necessary to weight the evidence provided by an evidence source by the relative "significance" of the source. For example, large surfaces or long edges should contribute more to the belief in a hypothesis than small surfaces or short edges. However, since object hypotheses can have different sizes we must scale these contributions by the dimensions of the object hypothesis. In the formalism we will present, this relative "significance" information is incorporated into a measure of the credibility of the evidence source.

5.3. Evidential Reasoning

The Dempster–Shafer theory of evidence provides a flexible mechanism for reasoning about evidence in problems such as this. For an earlier application of this theory in a related context, the reader is referred to [1]. In INGEN, each data surface and edge in an object hypothesis acts as a source of evidence for the hypothesis. This evidence is represented in the form of confidence functions. The confidence functions are constructed by comparing surface, edge, and vertex attributes for the data and the model. These confidence functions are then used to construct a basic probability assignment (BPA) which satisfies the conditions for use in the Dempster–Shafer theory of evidence. Dempster's rule is used for the combination of evidence.[8] An introduction to the Dempster–Shafer theory of evidence and a discussion of the special case (dichotomous frames of discernment) that we use for hypothesis evaluation are given in Appendix B.

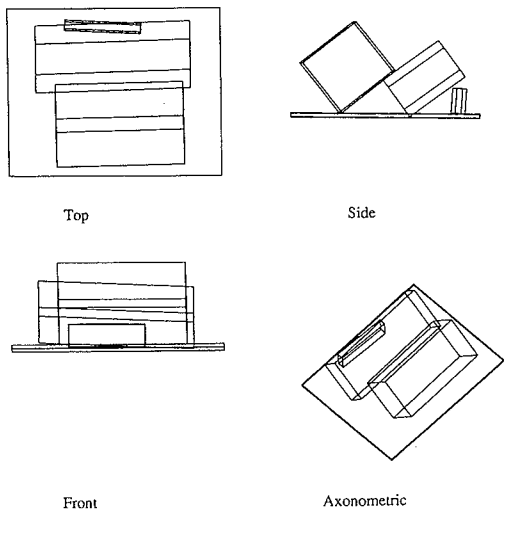


FIG. 31. Final results for scene 707.

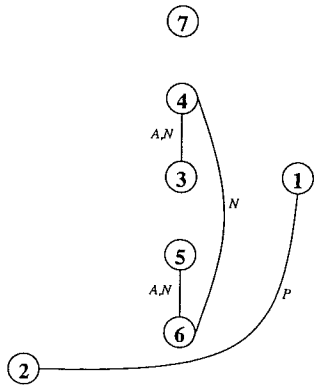
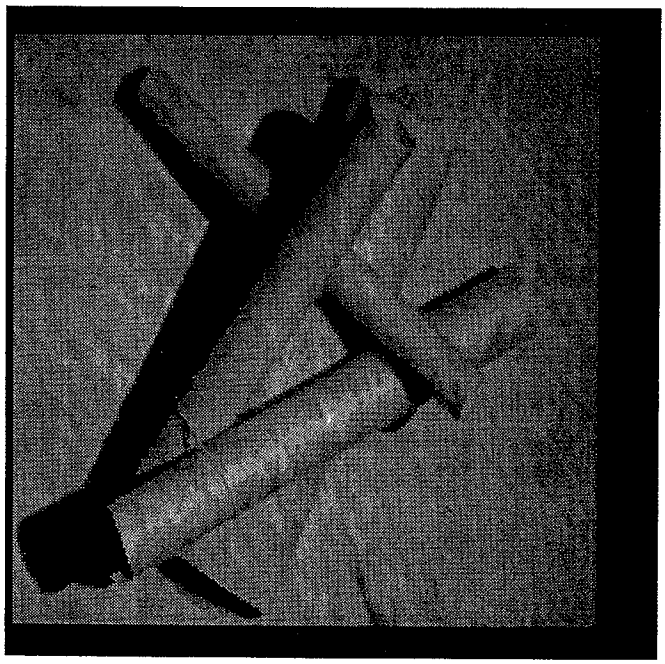


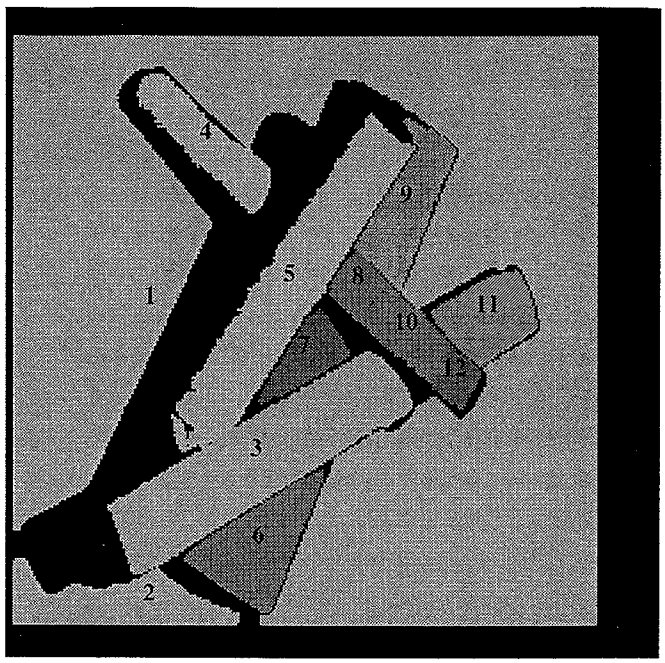
FIG. 32. Combinability graph for scene 707.

Object hypotheses are evaluated individually. The data for an object hypothesis consist of a set of surfaces with their constituent edges and vertices. The hypothesis also has an object model instance which is a copy of the object model which has been transformed and scaled into the scene coordinate frame based on the pose transform associated with the hypothesis. The object model instance has surface, edges, and vertices which can be matched with the data surfaces, edges, and vertices. The evaluation of the object hypothesis produces a measure of the quality of this match. Given the data and the model instance for an object hypothesis, we must determine the degree to which the data fit the model instance. The frame of discernment (FOD) for the object hypothesis is binary and dichotomous: either the data for object O_i fit the model or they do not fit the model. Thus the FOD (Θ) is defined by $\Theta = \{O_i, \neg O_i\}$. All of the BPAs for an object have the same FOD. For a different object we have a different FOD.

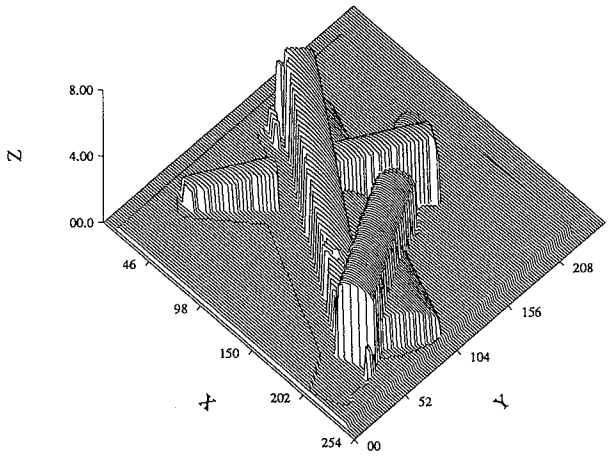
[8] For example, one of the conditions that must be satisfied before Dempster's rule can be used is that the evidence sources be independent. That the independence condition is satisfied in our system is discussed in Section 5.11.



Reflectance Image



Segmentation of Range Image



Range Plot

FIG. 33. Range data plot and segmentation for scene 711.

It is important to note that the comparison between the data and the model takes place in the scene coordinate frame. It is not equivalent to carry out the comparison in the model coordinate frame because the objects have different scale factors along the three axes. In the scene coordinate frame, distance measurements correspond to real distances and are isotropic. In the model coordinate frame, distance measurements will, in general, not correspond to real distances and will be anisotropic if the scale factors are unequal along the three axes. For recognition systems where there is no scaling or only global scaling, the comparisons between the data and the model can be made in either coordinate frame.

Prior to the computation of the belief in an object it is necessary to determine correspondences between data surfaces and edges and model surfaces and edges. (We do not attempt to determine vertex correspondences because there will usually be many data vertices which do not correspond to any model vertex.) These correspondences are determined by matching each data surface or edge with the model surface or edge that is the closest in terms of position and orientation. The correspondences

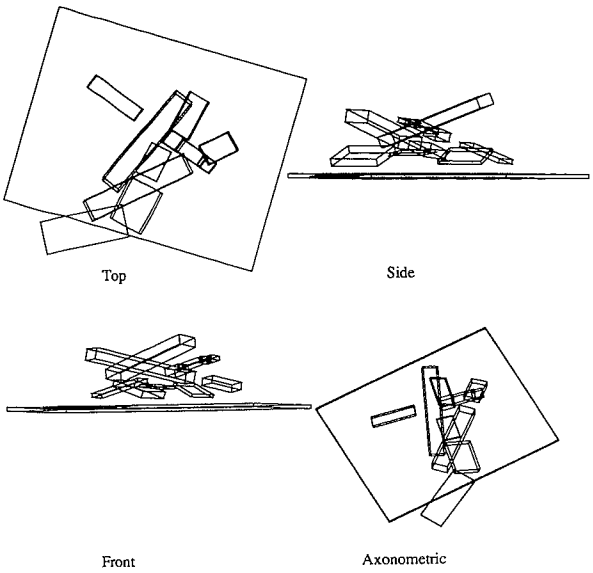


FIG. 34. Hypothesized objects for scene 711.

from data to model entities are typically many-to-one due to oversegmentation of the data entities.

5.4. From Evidence to Confidence Functions

Confidence functions serve as the link between evidence sources and BPAs. They are necessary because sources of evidence rarely supply numbers which satisfy the requirements for a BPA. Formally, an *evidence source* is a function which takes values in the range $[-\infty, \infty]$, although typically, an evidence source will actually produce values over a more limited range. Each source has its own range. The significance placed on the numbers provided by an evidence source also depends on the particular source.

On the other hand, a *confidence function, Conf,* is a function which takes values in the range [0, 1]. The domain of *Conf* is unspecified because the function can have any form as long as the constraints on the range are satisfied. $Conf(X) = 0$ implies that the evidence source has no evidence that the proposition X is true and $Conf(X) = 1$ implies that the evidence source has conclusive evidence that the proposition X is true. Confidence functions may be combined through multiplication. The result will always be a valid confidence function.

How the output of an evidence source is transformed into a confidence function depends on the nature of the evidence source. While in all cases an evidence source is performing a comparison between an attribute of a data entity and the corresponding attribute of a model entity (each comparison producing a number which characterizes how well the attributes compare), different types of comparisons will provide numbers in different ranges. In creating a confidence function we must map these numbers into the range [0, 1] and ensure that the 0 end of the range represents the lowest level of similarity and the 1 end of the range represents the highest level of similarity.

The easiest case to consider occurs when the attributes have scalar values such as surface area or edge length. Assume we have the value A_d from a data attribute and the value A_m from a model attribute. If we can assume

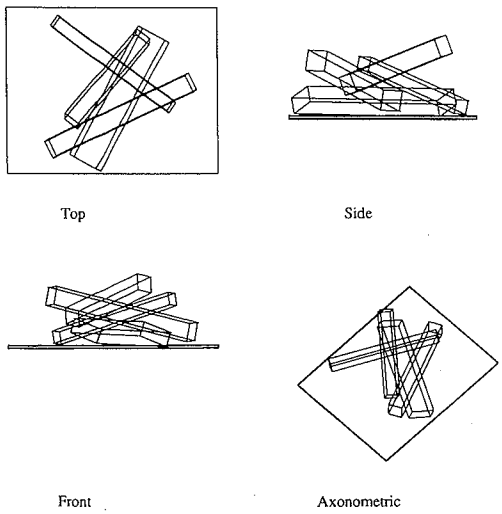


FIG. 35. Final results for scene 711.

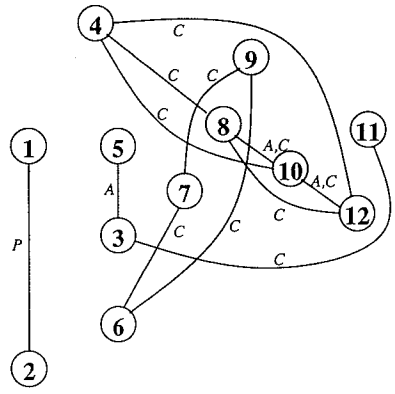


FIG. 36. Combinability graph for scene 711.

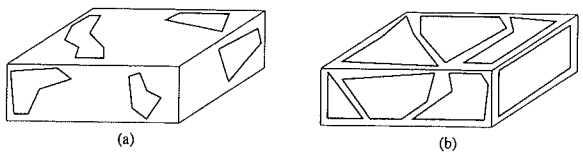


FIG. 37. Object hypotheses with low (a) and high (b) support.

that the model attribute value will always be larger than the data attribute then we can use the quotient as a confidence function

$$Conf_A = \frac{A_d}{A_m}. \tag{18}$$

If, however, we cannot be sure which is larger we can still use the quotient to define a confidence function

$$Conf_A = \min\left(\frac{A_d}{A_m}, \frac{A_m}{A_d}\right) \tag{19}$$

Of course, many other approaches are possible such as the (possibly normalized) difference between the two values. Eqs. (18) and (19) are appropriate when the relative values of the attributes are important as well as the difference in the values.

In some cases the actual values of the attributes are not important and we need only be concerned with the difference between them such as the distance between two points. Assuming that the difference B is always positive, we can use an exponential mapping to define a confidence function

$$Conf_B = e^{-\beta B}, \tag{20}$$

where β is a weighting factor chosen empirically. Note that when the difference is 0 the confidence function takes the value 1 and as the difference approaches ∞ the confidence function approaches 0. This was one of the mappings used in [14].

Another common case involves the comparison of two vectors. The dot product of the data vector D_d and the model vector D_m provides a good measure of how close the two vectors are to pointing in the same direction. However, the dot product of two vectors takes values in the range $[-1, 1]$ and we require values in the range $[0, 1]$. There are two situations to consider. First, in many cases the vectors may be allowed to point in the same direction or in opposite directions such as in the comparison of edge directions. In this case the absolute value of the dot product suffices,

$$Conf_D = |D_d \cdot D_m|, \tag{21}$$

to define the confidence function. In other cases, such as when comparing surface normals, it is required that the vectors point in the same direction; the confidence function is now defined by

$$Conf_D = \frac{(D_d \cdot D_m) + 1}{2}. \tag{22}$$

5.5. From Confidence Functions to Basic Probability Assignments

If we represent by O the proposition that a given data entity supports a particular object hypothesis, we may then use the provided confidence function to compute $Conf(O)$ and $Conf(\neg O)$. If we use $C_{support}$ to denote the number $Conf(O)$, then we may set

$$Conf(\neg O) = 1 - C_{support}. \tag{22}$$

Given $Conf(O)$ and $Conf(\neg O)$ defined in this manner, a simple way to obtain a BPA satisfying all the required properties would be via

$$\begin{aligned} m(O) &= C_{support} \\ m(\neg O) &= 1 - C_{support} \\ m(\Theta) &= 0. \end{aligned} \tag{23}$$

These expressions, although constituting a valid BPA, unfortunately do not suffice when practical considerations are taken account, because they do not take into account the fact that different evidence sources have different credibility. In line with what was said earlier, short edges and small surface segments can be a result of noise and other random phenomena. Due to their very randomness, there can be accidental alignments between such data entities and the corresponding entities in an object hypothesis. For this reason, if an evidence source involves, say, a small surface segment, it must be given a lower credibility compared to a source involving a large surface segment. (How the credibility factor of an evidence source should be computed will be discussed later.) But, given the credibility factor associated with a source, how should the BPA, such as the one shown above, be modified? If *Credibility* represents the credibility of an evidence source—the value of this factor will always be between 0 and 1—we factor in the source credibility into the above BPA in the following manner:

$$\begin{aligned} m(O) &= C_{support} \cdot Credibility \\ m(\neg O) &= (1 - C_{support}) \cdot Credibility \\ m(\Theta) &= 1 - Credibility. \end{aligned} \tag{24}$$

Note that for this to be a valid BPA we must have $m(O) + m(\neg O) + m(\Theta) = 1$. This is always true with the above formulas for any values for the confidence function and the credibility factor. For example, if we have $Conf(O) = 0.75 = C_{support}$ and $Credibility = 0.8$ we have

$$\begin{aligned} m(O) &= 0.75 \cdot 0.8 = 0.6 \\ m(\neg O) &= (1 - 0.75) \cdot 0.8 = 0.2 \\ m(\Theta) &= (1 - 0.8) = 0.2. \end{aligned}$$

This approach to factoring in the credibility of a source is based on the method of discounting discussed by Shafer [30] (pp. 251–255), where the benefit of reducing the deleterious effects of combining conflicting evidence is mentioned. The use of credibility to limit the contribution of each source of evidence also has the effect of ensuring that the belief computation does not become saturated before all of the evidence has been considered. The significance of saturation effects when combining evidence using the Dempster–Shafer theory has been discussed by Safranek, Gottschlich, and Kak [29].

5.6. Determining Feature Correspondences

Given a list of data surfaces and a list of model surfaces for an object hypothesis, we must first determine the proper correspondence between each data surface and a model surface. It is possible for multiple data surfaces to correspond to a single model surface. This is to be expected and does not cause any problems. Each data surface is considered individually. To determine the correspondence we compute the confidence that the data surface corresponds to each of the model surfaces for the object hypothesis. Thus, for a parallelepiped or an irregular there are six model surfaces to consider, and, for a cylinder, there are three. For each data surface, the correspondence with the highest confidence is accepted.

After we determine the correspondence for a data surface we compute the correspondences for the edges which bound the surface. Given a list of data edges and a list of model edges for each data surface and its corresponding model surface, we must determine the proper correspondence between each data edge and a model edge. It is possible for multiple data edges to correspond to a single model edge. This again is to be expected and does not cause any problems. As with surfaces, each data edge is considered individually. To determine the correspondence we compute the confidence that the data edge corresponds to each of the model edges for the model surface. For a rectangular surface there are four edges to consider. For a cylindrical surface there also are four edges to consider. Two of these edges arise from the junction of the cylindrical surface with the two planar end surfaces; the other two edges are virtual edges which arise from the cylindrical surface itself due to self-occlusion, the virtual edges being parallel to the axis of the cylinder. For each data edge, the correspondence with the highest confidence is accepted.

In the following sections we describe how these confidence functions are defined and how basic probability assignments are made.

5.7. Edges

We begin our belief computation with the construction of a BPA representing the belief from the edges for each surface of the object.

The confidence function representing the confidence that data edge D_k corresponds to model edge M_v is given by

$$Conf_{edge}(D_k, M_v) = Conf_{edge}^{orient}(D_k, M_v) Conf_{edge}^{position}(D_k, M_v), \quad (25)$$

where $Conf_{edge}^{orient}$ represents the confidence that the edge orientations correspond and $Conf_{edge}^{position}$ represents the confidence that the edge positions correspond. The confidence based on the edge orientation is given by

$$Conf_{edge}^{orient}(D_k, M_v) = |A_k \cdot A_v|, \quad (26)$$

which measures the difference in the directions of the data and model edges. A is the vector representing the orientation of the edge. The confidence based on the edge position is given by

$$Conf_{edge}^{position}(D_k, M_v) = e^{-\delta\, dist_{edge,edge}(D_k, M_v)}, \quad (27)$$

which measures the distance between the data and model edges. δ is a weighting factor chosen empirically. The function $dist_{edge,edge}(D_k, M_v)$ computes the distance between the data edge and the model edge using the formula

$$dist_{edge,edge}(D_k, M_v) = \tfrac{1}{2}(dist_{point,line}(V_{k_1}, Line_v) + dist_{point,line}(V_{k_2}, Line_v)), \quad (28)$$

where V_{k_1} and V_{k_2} are the endpoint vertices of edge D_k and $Line_v$ is the line equation for the model edge M_v. The function $dist_{point,line}(V, Line)$ computes the perpendicular distance between a vertex V and a line $Line$ Note that $0 \leq dist_{edge,edge}(D_k, M_v) \leq \infty$, so an exponential function is used to map this function into the range [0, 1].

The credibility that we place in the data edge D_k as a source of evidence is given by

$$Credibility_{edge}(D_k, M_v) = \min\left(\frac{L_k}{L_v}, \frac{L_v}{L_k}\right)\frac{L_v}{L_i}, \quad (29)$$

where L_k is the length of the data edge D_k, L_v is the length of the corresponding model edge M_v, and L_i is the sum of the lengths of all of the edges of the model object. Eq. (29) provides a measure of how much of the hypothesized edge is visible in the data and how much of total length of the object's edges the hypothesized edge represents. The measure is normalized so that the effect is the same whether the hypothesized edge is too long or too short.

Now, in accordance with Eq. (24), we can compute the BPA m_{edge_k} as follows:

$$
\begin{aligned}
m_{edge_k}(O_i) &= Conf_{edge}(D_k, M_v)Credibility_{edge}(D_k, M_v) \\
m_{edge_k}(\neg O_i) &= (1 - Conf_{edge}(D_k, M_v)) \\
&\quad Credibility_{edge}(D_k, M_v) \\
m_{edge_k}(\Theta) &= 1 - Credibility_{edge}(D_k, M_v).
\end{aligned} \tag{30}
$$

Once the BPAs for the individual edges have been computed, we can combine them into a single BPA. The BPA representing the belief that the K_j data edges of the jth surface correspond to the ith object's model is

$$
m_{edges_j} = \bigoplus_{1 \le k \le K_j} m_{edge_k}, \tag{31}
$$

where, as explained in Appendix B, the symbol $\oplus$ denotes the invocation of Dempster's rule. That the assumption of independence required by Dempster's rule is not violated will be discussed in Section 5.11.

5.8. Surfaces

The BPA based on the edges for a surface are combined with the BPA derived directly from the surface attributes so that all of the information for a particular surface is contained in a single BPA.

The confidence function associated with matching a data surface D_j with a model surface M_u is given by

$$
Conf_{surf}(D_j, M_u) = Conf^{orient}_{surf}(D_j, M_u)Conf^{shape}_{surf}(D_j, M_u), \tag{32}
$$

where $Conf^{orient}_{surf}$ represents the confidence that the surface orientations correspond and $Conf^{shape}_{surf}$ represents the confidence that the surface shapes (types) correspond. For planar and irregular surfaces the confidence based on orientation is given by

$$
Conf^{orient}_{surf}(D_j, M_u) = \frac{(N_j \cdot N_u) + 1}{2} \tag{33}
$$

which measures the difference in the directions of the surface normals. N is the vector representing the orientation of the surface normal. For cylindrical surfaces the confidence based on orientation is given by

$$
Conf^{orient}_{surf}(D_j, M_u) = |X_j \cdot X_u| \tag{34}
$$

which measures the difference in the directions of the cylindrical axes. X is the vector representing the orientation of the cylindrical axis. For all types of surfaces the confidence based on shape is given by

$$
Conf^{shape}_{surf}(D_j, M_u) = shape(T_j, T_u) \tag{35}
$$

which is based on the confidence function $shape(T_j, T)$ which the mid level processing system provides for each data surface D_j and each possible surface shape $T \in \{planar, irregular, cylindrical\}$. The mid level processing system defines $shape(T_j, T)$ such that

$$
\begin{aligned}
shape(T_j, planar) &+ shape(T_j, irregular) \\
&+ shape(T_j, cylindrical) = 1
\end{aligned} \tag{36}
$$

and

$$
0 \le shape(T_j, T) \le 1. \tag{37}
$$

The credibility factor when the evidence source is the data surface D_j, the model surface being M_u, is given by

$$
Credibility_{surf}(D_j, M_u) = \min\left(\frac{A_j}{A_u}, \frac{A_u}{A_j}\right)\frac{A_u}{A_i}, \tag{38}
$$

where A_j is the area of the data surface, A_u is the area of the corresponding model surface, and A_i is the total surface area of the model object. This provides a measure of how much of the hypothesized surface is visible and how much of the total area of the object the hypothesized surface represents. The measure is normalized so that the effect is the same whether the hypothesized surface is too large or too small.

As with edges, and in accordance with Eq. (24), the confidence and credibility factor are combined into a BPA m_{surf_j}:

$$
\begin{aligned}
m_{surf_j}(O_i) &= Conf_{surf}(D_j, M_u)Credibility_{surf}(D_j, M_u) \\
m_{surf_j}(\neg O_i) &= (1 - Conf_{surf}(D_j, M_u)) \\
&\quad Credibility_{surf}(D_j, M_u) \\
m_{surf_j}(\Theta) &= 1 - Credibility_{surf}(D_j, M_u).
\end{aligned} \tag{39}
$$

Once the BPA based on the surface attributes is computed, it is combined with the BPA derived from the edges. The combined BPA representing the belief that the jth data surface corresponds to the ith object's model is

$$
m_{surface_j} = m_{surf_j} \oplus m_{edges_j}, \tag{40}
$$

where m_{surf} depends on surface attributes only and m_{edges} on edge attributes only. Here again we postpone until Section 5.11 the discussion on the requirement that the sources of evidence be independent.

5.9. Objects

Once the BPAs for all of the object surfaces have been computed, we can combine them into a single BPA for the object. The BPA representing the belief that an object hypothesis O_i, which has J_i data surfaces, corresponds to its model is

$$m_{object_i} = \bigoplus_{1 \le j \le J_i} m_{surface_j}, \tag{41}$$

which simply states that the belief in an object hypothesis is the combination of the belief in each of its surfaces. The discussion of the independence assumption is again deferred to Section 5.11.

5.10. Evaluation of Combined Hypotheses

To evaluate a combination hypothesis, we compare the beliefs in the two original object hypotheses O_1 and O_2 with the belief in the combined hypothesis O. The combined hypothesis is accepted based on the following evaluation:

IF [{$Bel(O) > Bel(\neg O)$} AND {$Bel(O) > Bel(O_1)$}
AND {$Bel(O) > Bel(O_2)$}] THEN
accept hypothesis O
ELSE IF {$Bel(O) > 1.5\ Bel(\neg O)$}
AND [{$Bel(O) > Bel(O_1)$}
OR {$Bel(O) > Bel(O_2)$}] THEN
accept hypothesis O
ELSE
reject hypothesis O
END IF

The first case handles the situation where the new hypothesis is stronger than both of the constituent hypotheses. The second case handles the situation where the new hypothesis is very strong but is only stronger then one of the constituent hypotheses. This sometimes happens when one of the constituent hypotheses is also very strong. If the new hypothesis is accepted then it becomes the new seed hypothesis and the search for further combinations continues. If the new hypothesis is not accepted then the combination is undone (module 8 in Fig. 3) and the system backtracks and continues the search through the combinability graph for other combinations.

5.11. Are The Independence Assumptions for Use of Dempster's Rule Satisfied?

One important issue that must be addressed when discussing evidential reasoning is the issue of independence. Dempster's rule for combining evidence sources can only be applied provided the sources of evidence are independent. We will now provide arguments that establish that in each of the three situations in which Dempster's rule is invoked, the independence assumption is not violated.

Let us first consider the case of combining evidence generated by the data edges. Metaphorically speaking, we may imagine that sitting on each data edge is an expert to whom is available the candidate model object, which corresponds to the object hypothesis under evaluation, together with its associated pose transform, including the scaling factors. This expert examines that model edge which is closest to the data edge in position and orientation. Then the expert issues forth a BPA on the basis of similarity of the positions and the orientations of the data edge, to which the expert is assigned, and the closest model edge, taking into account the credibility factor. Now the question is, can one such expert predict the BPA that will be issued by another expert sitting on some other edge? It is, of course, true that the model knowledge available to all the experts engaged in the evaluation of all the edges in a given object hypothesis is the same. However, it is also true that, in the presence of sensor noise and artifacts generated by the peculiarities of a segmentation algorithm, the position and the orientation of one data edge will not be predictable from the position and the orientation of another data edge, even when the two data edges are adjacent and thus sharing a vertex. Therefore, the judgments made by one expert, regarding the similarity of its data edge to the closest model edge, are independent of the judgments made by any of the other experts.

We must hasten to add that it would be only too easy to come up with a scheme for specifying edge BPAs where the assumption of independence would be violated. Note that in our case our expert's "vision" is limited to the local data edge to which that expert is assigned; in other words the expert cannot see any of the other data edges. Now consider what would happen if we had an expert who looked at all the data edges in an object hypothesis and who then assigned BPAs to each edge. Such BPAs would not be independent because now the expert's judgment regarding what BPA to issue would be influenced by the relations between the data edges, on the one hand, and the relations between the model edges on the other.

The second use of Dempster's rule is to combine evidence from the edges of a surface with evidence from the orientation and shape of the surface. Consider the case of surface orientation, such as the orientation of a cylindrical surface. What edges of a cylindrical surface might be visible would depend on what other objects are occluding the cylinder—something that is entirely unpredictable. Therefore, the edges we extract may or may not be parallel or perpendicular to the axis of the cylinder. (Of course, not to be forgotten is the randomness in edge orientations introduced by noise and other artifacts.) Therefore, in general, it would not be possible to predict the orientation of a surface from the edges associated with that surface. Similarly, the shape information for a surface is completely unrelated to the edge information because it depends entirely on local properties of the surface points. For example, the decision that a surface is cylindrical is based on the surface normal distribution computed from the interior points of a segmented surface; in fact such an estimate is made from only those interior points that are within a predefined neighborhood and not in the vicinity of any of the edges. Therefore, in

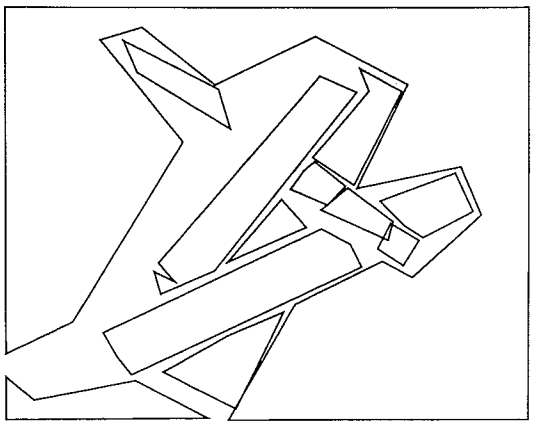


FIG. 38. Data edges for the scene in Fig. 33.

general, a surface BPA will not be predictable from the edge BPAs.

The third use of Dempster's rule is to combine evidence from a set of surfaces into a belief function for the entire object. The reader is probably thinking that since the adjacent surfaces must share common edges, it simply is not possible for the BPAs on the right hand side of Eq. (41) to be independent. Our defense is that, due to the manner in which the range maps of postal objects are segmented, it is more often than not the case that the adjacent surfaces do not share any edges. The important thing to remember is that postal objects often have rounded and crumply edges, as opposed to crisp surface normal discontinuities on industrial objects. When the range maps of these scenes are segmented, we do not enforce the constraint that adjacent surfaces share edges [17]. For example, shown in Fig. 38 are the edges resulting from the segmentation of the scene of Fig. 33.[9] As shown in Fig. 38, the surface segments do not share any edges. Also, the nearby edges from adjacent surfaces do not in all cases possess identical orientations. The point we are trying to make is that in the presence of sensor noise, object surface irregularities, segmentation artifacts, etc., we can assume that the BPAs corresponding to different surfaces would be independent.

Therefore, we can state that we are not violating the requirement that the BPAs be independent before they are combined via Dempster's rule.

6. GEOMETRIC REASONING

As was mentioned in Section 2.4, the task of the Geometric Reasoning Loop in Fig. 3 is twofold: (1) to expand the object hypotheses to their maximum allowable dimensions along directions away from the sensor and toward the work table; and (2) to detect intersections between object hypotheses thus modified. In actuality, both these tasks are accomplished simultaneously, as will be explained in this section. As mentioned earlier, when the Geometric Reasoning Loop finds an object hypothesis to be geometrically inconsistent with the rest of the scene interpretation, in the sense that even in its minimal dimensions the object intersects other object hypotheses, the system backtracks in the search process by undoing previous hypothesis combinations.

When we computed the scale parameters associated with an object hypothesis in Section 3, we calculated the minimum bounding parallelepiped for the data corresponding to the hypothesis. Now, in the Geometric Reasoning module, the system tries to expand this minimum bounding parallelepiped to its maximum possible dimension along a direction away from the sensor. The parallelepiped is expanded (grown) until it contacts some other object hypothesis. After such an expansion, the size of the object hypothesis (meaning its scale parameters) is modified to reflect its enlarged size.

The geometric reasoning algorithm used in INGEN is based on a single operation: finding the maximal extent of an object hypothesis in a particular direction. The determination of the geometric consistency of a scene requires the execution of this algorithm once for each object in the scene. Each object is "grown" along the object-centered axis that points away from the sensor and is most nearly parallel to the sensor viewing direction. This is the direction where the uncertainty about the size of an object hypothesis is the largest. The algorithm could actually be applied in all six directions for each object hypothesis but, we believe, the benefits from the extra computation are small compared to the additional computation cost.

The order in which the object hypotheses are grown is significant only if more than one geometrically consistent scene interpretation is possible. (This occurs in situations where the data are ambiguous; that is, where there is not sufficient information available to completely determine the dimensions of the objects.) The ordering heuristic used by INGEN is to start from the bottom of the pile of objects and work towards the top. Thus, objects higher up in the pile are tested after objects lower in the pile. Although the objects near the bottom of the pile are subject to more severe occlusion than objects near the top of the pile, the former objects are less likely to intersect other objects as they are grown toward the table. Also, the objects higher up in the pile will be grown towards the previously grown objects lower down and thus will benefit from the results of previous object growth.

The algorithm for the overall control of the geometric reasoning process for an entire scene, geometric_reasoning, is shown in Figs. 39 and 40. This procedure is called

[9] The segmentation shown in Fig. 33 is drawn in the parameter space; the parameters correspond to the scanning employed for the construction of the range map. On the other hand, the segmentation shown in Fig. 38 is in the xy plane and is obtained by projecting the segmentation boundaries onto the xy plane. This also explains the overlapping edges in Fig. 38.

```
PROCEDURE geometric_reasoning()
    SortedHypotheses ← list of all object hypotheses sorted from lowest to highest
    Conflicts ← []
    FOR EACH Hypothesis IN SortedHypotheses DO
        ExtentMin ← minimum extent attribute for Hypothesis
        transform SortedHypotheses into model coordinate frame of Hypothesis
        grow Hypothesis until first contact
        Extent ← the distance grown to the first contact
        First ← the first object contacted
        ExtentMax ← Extent transformed into scene coordinate frame
        IF ExtentMax > ExtentMin THEN
            Extent ← ExtentMax
            recompute pose and scale factors for Hypothesis
        ELSE
            add [Hypothesis, First] to Conflicts
        END IF
    END FOR
    IF Conflicts = [] THEN
        RETURN []
    ELSE
        ConflictObjects ← resolve_conflicts(Conflicts)
        RETURN ConflictObjects
    END IF
END PROCEDURE
```

FIG. 39. Algorithm which controls the geometric reasoning process for a scene.

by the procedure search1 which was discussed in Section 2.3 and defined in Fig. 5. Essentially, the system loops through all of the object hypotheses, performing the growing process for each object in turn. As each object is grown, either new maximum dimensions are calculated or an intersection with another object is noted. After all of the intersections (conflicts) are found, the procedure resolve_conflicts (Fig. 40) is called to decide which hypotheses should be uncombined. The rationale for the uncombination decisions are (1) if the growing process did not result in mutual intersections then the object that actually grew into the other should be the first choice for uncombination, or (2) the largest hypothesis in a mutual conflict situation should be the first choice for uncombination, and (3) if the first choice is a single-surface hypothesis then the other hypothesis should be uncombined. If both of the hypotheses are single-surface hypotheses then an unresolvable conflict has been detected. The geometric-reasoning procedure either returns an empty list specifying the fact that the scene interpretation is geometrically consistent or it returns a list of hypotheses which should be uncombined.

The most important aspect of the geometric reasoning process is the intersection detection algorithm which also determines the maximum extent for object hypotheses based on contacts with other objects. Although we refer to this algorithm as a growing process, it is not implemented in that manner. In the remainder of this section we discuss this algorithm.

Before we describe our algorithm we must introduce some terminology that will help to clarify the discussion. The *object* is the minimum bounding parallelepiped for the object hypothesis for which we wish to determine the maximum value of a particular size parameter. The *obstacles* are the minimum bounding parallelepipeds for all of the other object hypotheses in the scene which potentially constrain the object physically. We use this terminology because it is convenient to think of the computation of the maximal extent of an object as the "growing" of the object until it comes in contact with an obstacle.

The algorithm consists of four steps.

1. Find the model transformation for the object; this transform takes the object in the scene into its unit-sized model in the model coordinate frame. (Note that this transform is the inverse of the pose transform discussed in Section 3.)

2. Use the model transformation to transform all obstacles into the model coordinate frame of the object.

3. Find the maximum extent of the object along the positive z axis such that the object is in contact with at least one obstacle in the scene and its volume does not intersect the volume of any obstacle in the scene.

4. Use the model-to-scene transformation to transform the extent measurement back into the scene coordinate frame and thus find its actual value.

As we discuss the geometric reasoning process we will refer to postal scene 711 shown in Fig. 42. As shown in Fig. 42, the scene contains three objects plus the back-

```
PROCEDURE resolve_conflicts(Conflicts)
    ConflictObjects ← []
    FOR EACH [Hypothesis, First] IN Conflicts DO
        IF [First, Hypothesis] is not in Conflicts THEN
            IF Hypothesis is not a single surface hypothesis THEN
                add Hypothesis to ConflictObjects
            ELSE IF First is not a single surface hypothesis THEN
                add First to ConflictObjects
            ELSE
                report unresolvable conflict between Hypothesis and First
            END IF
        ELSE
            Largest ← larger of Hypothesis and First
            Smallest ← smaller of Hypothesis and First
            IF Largest is not a single surface hypothesis THEN
                add Largest to ConflictObjects
            ELSE IF Smallest is not a single surface hypothesis THEN
                add Smallest to ConflictObjects
            ELSE
                report unresolvable conflict between Hypothesis and First
            END IF
        END IF
    END FOR
    RETURN ConflictObjects
END PROCEDURE
```

FIG. 40. Algorithm for making uncombination decisions based on the conflicts detected during geometric reasoning.

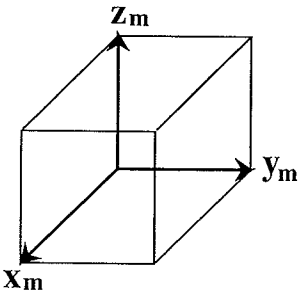


FIG. 41. Model coordinates frame for geometric reasoning.

ground. The topmost object is a letter which is lying on two boxes, one of which is lying on top of the other.

6.1. Step 1

The determination of the pose transformation is made when the attributes of the object hypothesis are computed as discussed in Section 3. However, for the geometric reasoning process it is useful to define a different model coordinate frame. For geometric reasoning, INGEN uses bounding boxes to model all objects but this algorithm could also be applied to more complex object models. The coordinate frame used for geometric reasoning is defined so that the model object is a unit cube located in the positive octant with one vertex at the origin as shown in Fig. 41. In the rest of this section all references to the model coordinate frame refer to the object-centered coordinate frame that we have just defined. As the algorithm is discussed it will be seen how this coordinate frame simplifies the computations. The transformation which relates the model coordinate frame used by the pose transform calculations to the model coordinate frame used for geometric reasoning is precomputed and is a part of the object model.

Figure 43 shows the hypothesized objects for scene 706.

6.2. Step 2

In step two all of the objects and obstacles are transformed into the model coordinate frame of the object of interest. By this we mean that the bounding boxes for all of the objects are transformed into the geometric reasoning model coordinate frame of the object of interest. In the top view of the scene in Fig. 43 the origin of the model coordinate frame for the letter is the top left vertex. The x axis extends toward the right and the y axis toward the bottom of the scene. Figure 44 shows the hypothesized objects of scene 706 after they have been transformed into the model coordinate frame of the letter. The letter is transformed into a unit cube, which is upside down with respect to Fig. 43, and the other objects are transformed accordingly; i.e., the visible top surface of the letter becomes the bottom surface of the unit cube in the model coordinate frame and the other objects appear above the letter.

6.3. Step 3

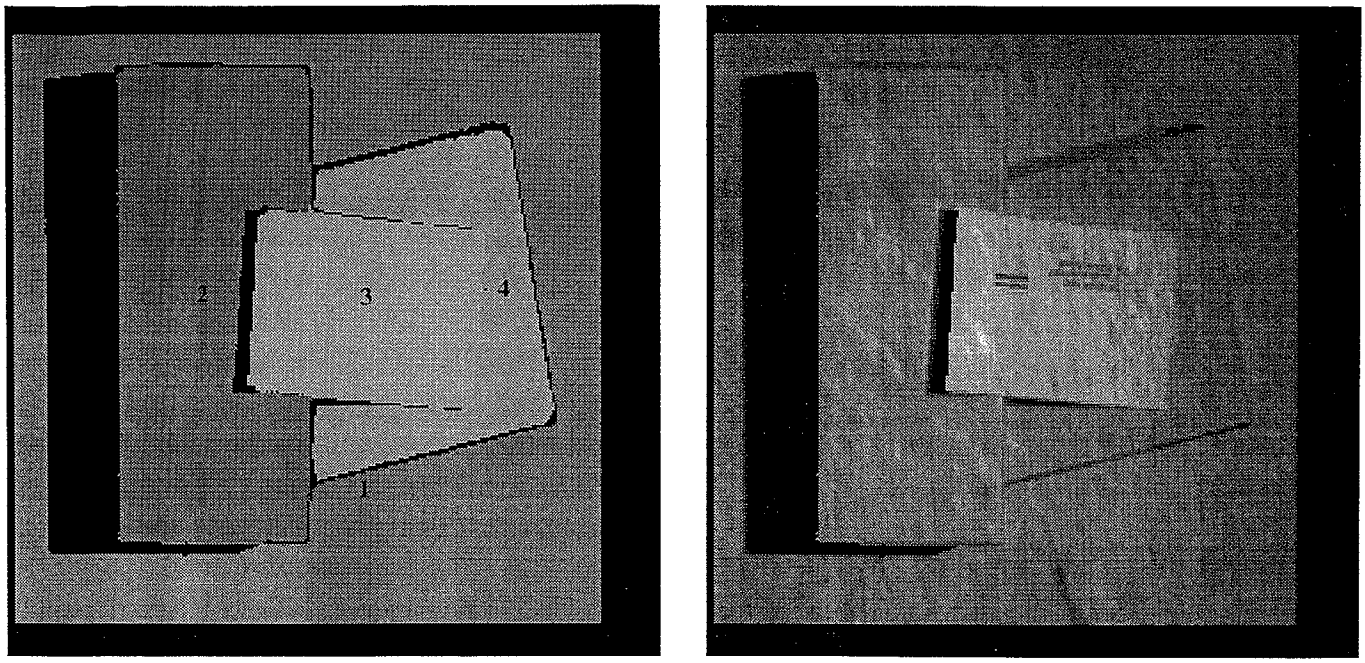
Clearly, step three of this algorithm is the most difficult one. However, the transformation of the obstacles into the model coordinate frame of the object which was carried out in step two simplifies the computations in step three. Our approach borrows some computational techniques from ray casting algorithms in the computer graphics field and from path planning algorithms in the robotics field. The ray casting technique that we borrow is to transform our object and obstacles into a coordinate frame where the object hypothesis is a simple primitive object of fixed size and shape. The path planning technique that we borrow is the use of both parametric and implicit representations of surfaces and edges to aid in the computation of intersections. Roth [28] discusses ray casting techniques and algorithms and Lozano-Pérez [22] discusses the configuration space approach to robot motion planning.

Three types of contact are possible between polyhedral objects and polyhedral obstacles. We use the convention stated in [22] to name these interactions: *Type A*—an object surface contacting an obstacle vertex, *Type B*—an object vertex contacting an obstacle surface, and *Type C*—an object edge contacting an obstacle edge.

The three types of contact are shown in Fig. 45. Note that the objects and the obstacles are positioned in the object's model coordinate frame. Thus, the obstacles which are below the object in the scene appear above the object in the model coordinate frame.

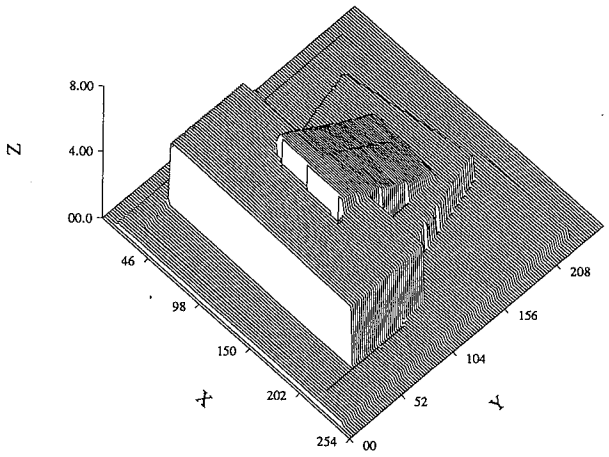
For each object–obstacle interaction we need to check for all three types of contact. The maximum object height (in the model coordinate frame) is determined by the minimum height contact with the obstacle. This process is carried out for each obstacle in the scene and the results are then combined to find the maximum height for the object which is determined by the minimum height contact with any obstacle.

In the following sections we describe computational techniques for dealing with the three types of contact. It is important to note that all of the computations are carried out in the model coordinate frame of the object of interest. We will always be finding the maximum extent of the object along the positive z axis, which we refer to as the height of the object, in the model coordinate frame. Thus, when we refer to the top surface of the object we are referring to the surface which has a surface normal that points in the positive z direction in the model coordinate frame regardless of its acutal location and orientation in the scene.



Segmentation of Range Image

Reflectance Image



Range Plot

FIG. 42. Range data plot and segmentation for scene 706.

6.3.1. *Type A—An Object Surface Contacting an Obstacle Vertex*

For type A contact we need to find the obstacle vertex which will contact the top surface of the object at the lowest point. The procedure is to find all obstacle vertices that could potentially contact the top surface of the object and then find the one with the smallest z component. For parallelepiped obstacles the only vertex that can contact the topmost surface of the object is the one with the smallest z component. For all of the other vertices an edge will necessarily contact the object at a point that is lower than the vertex so these vertices can be ignored. Thus, only one of the eight vertices for each parallelepiped obstacle is tested for constraining the object.

The bounding boxes are all parallelepipeds and their transformation into model coordinates results in the object bounding box becoming a unit cube in the positive octant with a vertex at the origin. Therefore, only those

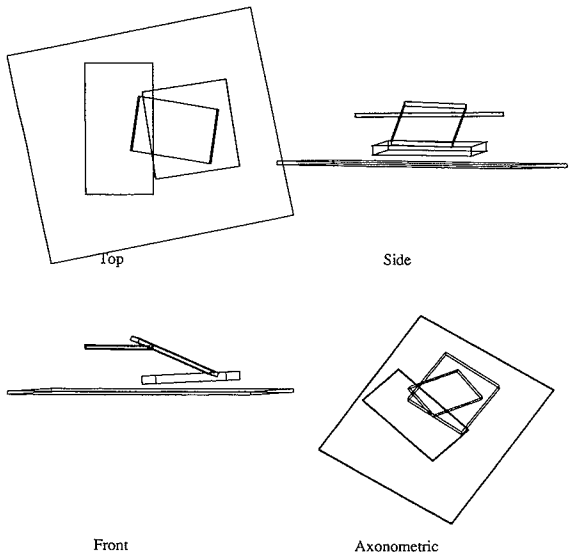


FIG. 43. Hypothesized objects for scene 706.

obstacle vertices which satisfy $0 \leq x \leq 1$ and $0 \leq y \leq 1$ can potentially contact the top surface of the object parallelepiped.

6.3.2. *Type B—An Object Vertex Contacting an Obstacle Surface*

Given that object bounding boxes are parallelepipeds, we need to find the obstacle surface which will contact the object bounding box vertex at the lowest point. The procedure is to find the contact points for all obstacle surfaces that could potentially contact the top vertices of the object bounding box and then find the one with the smallest z component. Potential contact points are the intersection points of the four lines defined by the vertical edges (parallel to the z axis) of the object parallelepiped and the obstacle surfaces. The four lines are defined by: $\{x = 0, y = 0\}$, $\{x = 0, y = 1\}$, $\{x = 1, y = 0\}$, and $\{x = 1, y = 1\}$. We only need consider obstacle surfaces that have their average surface normal pointing in the negative z direction. For all of the other surfaces some other part of the obstacle will necessarily contact the object at a point that is lower than the surface so these surfaces can be ignored. Thus, only three of the six surfaces of each parallelepiped obstacle are tested for constraining the object.

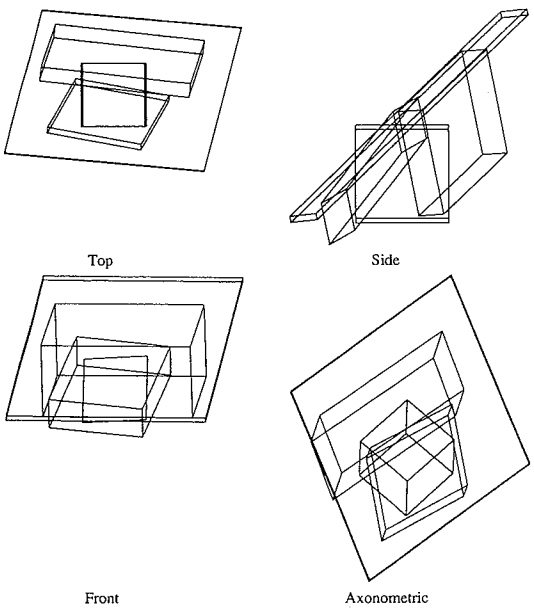


FIG. 44. Objects transformed into the model coordinate frame of the letter.

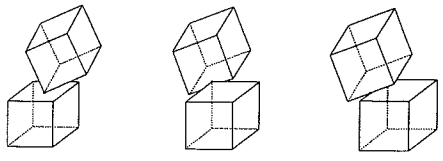


FIG. 45. Type A, B, and C contacts.

All of the planar obstacle faces belong to rectangular parallelepipeds so the faces will be parallelograms when transformed into the model frame of the object. By using the parametric representation for these faces the intersection problem can be solved efficiently. We select one vertex (x_0, y_0, z_0), of the face as its origin and use the two adjacent vertices, (x_1, y_1, z_1) and (x_2, y_2, z_2), to define oblique axes for the parameterization of the surface. Using u and v as parameters, we can write the following parametric equations for the parallelogram face of an obstacle bounding box

$$x = x_0 + uf_1 + vf_2,$$
$$y = y_0 + ug_1 + vg_2,$$
$$z = z_0 + uh_1 + vh_2,$$

where

$$f_1 = x_1 - x_0, \quad g_1 = y_1 - y_0, \quad h_1 = z_1 - z_0$$
$$f_2 = x_2 - x_0, \quad g_2 = y_2 - y_0, \quad h_2 = z_2 - z_0.$$

By substituting the x and y values from the line equation pairs into the x and y parametric equations for the plane we can solve for the two surface parameters u and v:

$$u = \frac{g_2(x - x_0) - f_2(y - y_0)}{f_1 g_2 - f_2 g_1},$$

$$v = \frac{-g_1(x - x_0) + f_1(y - y_0)}{f_1 g_2 - f_2 g_1}.$$

If $0 \leq u \leq 1$ and $0 \leq v \leq 1$ then the intersection point lies within the obstacle face. We then substitute u and v into the z parametric equation to find the z coordinate of the intersection point.

6.3.3. *Type C—An Object Edge Contacting an Obstacle Edge*

For type C contact we need to find the obstacle bounding box edge which will contact the top edge of the object bounding box at the lowest point. The procedure is to find all points at which obstacle edges intersect the vertical surfaces of the object and then find the one with the smallest z component. We can exclude some edges from consideration because of the orientations of the surfaces that they bound. All of the edges of the surface with a surface normal that forms the most positive dot product with the positive z axis are excluded. Also, the edge between the other two surfaces with surface normals that form positive dot products with the positive z axis are excluded. Thus, only seven of the twelve edges of each parallelepiped obstacle need be tested for constraining the object.

For object parallelepipeds the contact points are the intersection points between obstacle edges and the four vertical faces (parallel to the z axis) defined by the four plane equations: $x = 0$, $x = 1$, $y = 0$, and $y = 1$. For planes $x = 0$ and $x = 1$ we are interested in intersection points such that $0 \leq y \leq 1$. For planes $y = 0$ and $y = 1$ we are interested in intersection points such that $0 \leq x \leq 1$.

By using the parametric representation for obstacle edges we can simplify the intersection computations. Edges are defined by their endpoint vertices: (x_0, y_0, z_0) and (x_1, y_1, z_1). Using t as the parameter, the parametric equations for the edge are

$$x = x_0 + tf, \quad y = y_0 + tg, \quad z = z_0 + th,$$

where

$$f = x_1 - x_0, \quad g = y_1 - y_0, \quad h = z_1 - z_0.$$

By substituting the x or y value from the plane equation into the appropriate edge equation we can solve for the edge parameter t:

$$t = \frac{x - x_0}{f}, \quad t = \frac{y - y_0}{g}, \quad t = \frac{z - z_0}{h}.$$

If $0 \leq t \leq 1$ then the intersection point lies within the obstacle edge. We then substitute t into the other parametric equations to find the other coordinates of the intersection point.

6.4. Step 4

The result from step three is a value for the maximum extent of the object along the positive z axis in the model coordinte frame of the object. Note that this coordinate frame depends on the dimensions of the object. A maximum extent value less than one indicates that the old dimension was too large and than an intersection has been found, and a value greater than one indicates that the old dimension was too small and that the object may be extended by an amount proportional to this value. The new object dimension is obtained in step four by simply multiplying the value found in step three by the previous value for the object dimension.

Figure 46 shows the final results produced by INGEN for scene 706. Note that the letter has been recognized correctly. Also note that one of the boxes extends below the other box and the latter appears to be floating in air. There is no way for the system to know how far the box extends under the other one without undertaking more complex geometric reasoning or taking into account the physical stability of the scene interpretation. One approach that would help to solve this problem would be to extend the box under the other one until it contacts known empty space. The same basic approach as discussed here could be used but the representation of empty space would necessarily be more complex than the parallelepipeds that we have used. Mulgaonkar, Cowan, and DeCurtins [24] have investigated some of these possibilities.

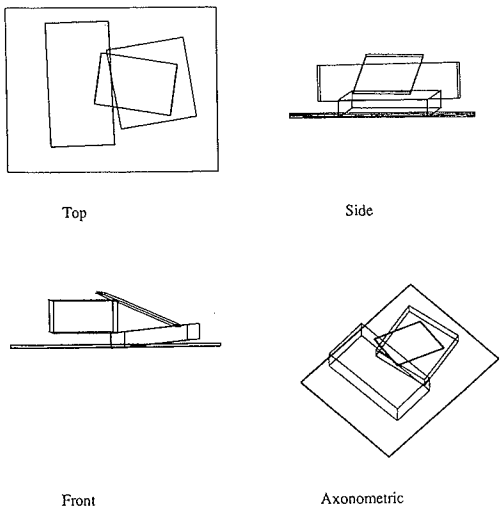


FIG. 46. Final results for scene 706.

7. CONCLUSION

We have discussed the INGEN system which addresses the problem of generic object recognition in the postal domain. There are three important aspects of this system which make it particularly useful for generic ob-

ject recognition. (1) New approaches have been developed for constructing object hypotheses, computing their attributes, and evaluating how well they fit the data. (2) A geometric reasoning process determines constraints on the size of object hypotheses by finding points of contact with other object hypotheses and also detects geometric inconsistencies in the scene interpretation. (3) The recognition process integrates hypothesis evaluation with geometric reasoning to allow backtracking when object hypotheses are rejected due to insufficient support or geometric conflict with other object hypotheses.

8. APPENDIX A: RANGE DATA ACQUISITION AND CHARACTERIZATION

8.1 Range Data Acquisition

Acquisition of range data (module 1 in Fig. 3) is the starting point for the system. Range data are used by the system in the form of three parameterized arrays $x(i, j)$, $y(i, j)$, and $z(i, j)$ to represent points in three dimensional space. These arrays are indexed by the scanning parameters i and j. The size of these arrays depends on the sensor and the application. We have used sizes from 100 $\times$ 100 to 512 $\times$ 512. For the postal application these data usually correspond to a 16 $\times$ 16 $\times$ 16 in. volume in the world.

Figure 47 shows a schematic representation of how a structured light range data sensor is used to acquire range data. A projector projects a vertical plane of light onto the scene. This plane appears as a stripe from the viewpoint of the video camera which is used to produce digitized images of the scene with the stripe. The known geometric relationship between the projector and the camera allows the computation of depth through the use of triangulation formulas. To acquire the range data the plane of light must be scanned across the scene and digitized images collected for a number of stripes. The raw form of the range data is the $d(i, j)$ array. The array parameter i represents the row number in the digitized image while the array parameter j represents the stripe number in the projector, and the value of each $d(i, j)$ is the distance from the left edge of the digitized image to a point where strip j intersects row i. A calibration procedure is used to construct a transformation matrix which is used to transform the $d(i, j)$ data into $x(i, j)$, $y(i, j)$, and $z(i, j)$ data. The calibration procedure is described in [4, 5]. A variety of triangulation formulas for different types of structured light sensing are presented in [36].

INGEN has been operated successfully on data from a variety of range data sensors:

- A gantry-mounted light stripe triangulation sensor constructed at the Purdue Robot Vision Lab. It has a stationary camera and uses rotational scanning of the light stripe. The demonstration shown in Figs. 1 and 2 made use of this sensor.
- A robot-carried light stripe triangulation sensor constructed at the Purdue Robot Vision Lab. It has the camera and the light stripe projector in a fixed relationship and uses the robot to move the scanner to scan the scene.
- A space coded structured light triangulation sensor constructed at the GE/RCA Advanced Technology Labs

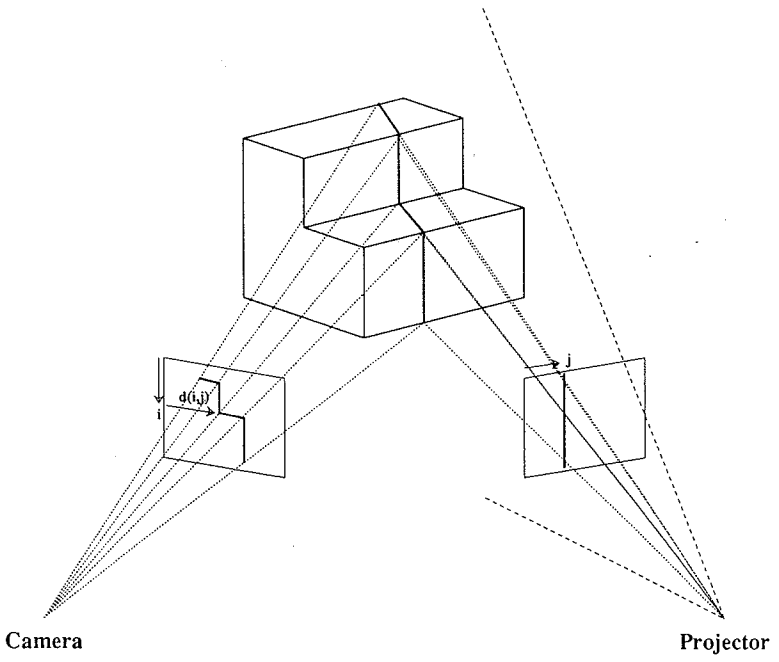


FIG. 47. Structured light range data sensor.

(now David Sarnoff Research Center, a subsidiary of SRI International) [27]. It projects a sequence of 14 patterns on the scene which can simulate the projection of 1024 stripes. The example scenes shown in Figs. 25, 29, 33, and 42 were produced by this sensor.

- A flying laser spot triangulation system constructed at SRI Internationl [26]. It uses moving mirrors to scan a laser spot over the scene and detectors to measure the position of the spot to determine the range.

- Two laser radar sensors constructed at the Environmental Research Institute of Michigan (ERIM) [15]. They both scan an amplitude modulated laser beam over the scene using moving mirrors and detect the phase shift between the projected and the sensed beam. The range is proportional to this phase shift. One sensor uses mirrors for scanning a laser across the entire stationary scene. The other sensor uses mirrors to scan a laser spot in one direction while a conveyer belt moves the objects through the view of the sensor.

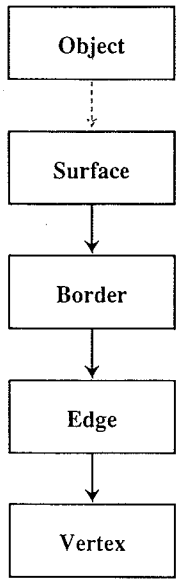


FIG. 48. The hierarchical symbolic scene description.

8.2. Low Level Processing

Low level processing (module 2 in Fig. 3) begins with the computation of local surface normals for each range data point. The surface normals are then used to find surface normal disparity edges where the deviation between adjacent surface normals is high. Two other edge detectors are also used, one to detect range discontinuity edges, where there is a large jump in position between a range point and its neighbors, and the other to detect roof and valley edges, where the local curvature of the surface is high. A range pixel is labeled as an edge point if the value computed by any of the three detection methods for that point exceeds the pre-defined threshold for that detection method. For INGEN the edge thresholds are defined so that the system is biased towards over-segmentation because it is much easier to combine segments that have been erroneously separated than it is to separate segments that have been erroneously labeled as a single segment. After all of the edge pixels in the data have been found a growing and shrinking procedure is used to fill in any small gaps in the edges. These range edges divide the scene into segments which correspond to continuous surfaces. A connected component labeling algorithm is used to label each range pixel with the name of the surface segment that contains it.

8.3 Mid Level Processing

Mid level processing (module 3 in Fig. 3) takes the segmented range data and produces a hierarchical symbolic scene description as shown in Fig. 48. This description consists of surfaces, edges, vertices, and relations. Surfaces are bounded by edges and edges are bounded by vertices. The set of edges which bounds a surface is called the border. These entities are related to each other in a hierarchical fashion. Note that objects can be considered the top of the hierarchy because they are bounded by surfaces.

Surfaces have the following attributes:

1. viewpoint—the viewpoint from which the data was acquired
2. edges—a list of edges that form the exterior boundary of the surface
3. homog—a homogeneous transformation specifying the position and orientation of the surface derived from the major, minor, and normal axes and the position
4. position—the centroid of the data points for the surface
5. orientation—the surface normal for the surface derived from the plane equation
6. types—a list of the three surface types with the confidence in each type
7. type—the surface type with highest confidence: planar, cylindrical, or irregular
8. confidence—the confidence for the surface type
9. shape—the shape of the surface border: circular, irregular, trapezoidal, parallelogram, rectangular, or square
10. area—the area of the surface
11. num_points—the number of datapoints in the surface
12. major_axis—the major axis of the surface
13. minor_axis—the minor axis of the surface
14. cylindrical_axis—the cylindrical axis of the surface

15. length—the length of the surface along the major axis

16. width—the length of the surface along the minor axis

17. height—the length of the surface along the normal axis

18. angle_from_vertical—the angle between the surface normal and the positive z axis

19. angle_major_axis—the angle between the projection of the major axis into the x–y plane and the positive x axis

20. plane_equation—the equation of the best fitting plane for the surface

21. plane_error—the error of the plane fit

22. borders—the borders defining the boundary of the surface

23. xlow—the x, y, z coordinates of the surface point with the smallest x coordinate

24. xhigh—the x, y, z coordinates of the surface point with the largest x coordinate

25. ylow—the x, y, z coordinates of the surface point with the smallest y coordinate

26. yhigh—the x, y, z coordinates of the surface point with the largest y coordinate

27. zlow—the x, y, z coordinates of the surface point with the smallest z coordinate

28. zhigh—the x, y, z coordinates of the surface point with the largest z coordinate

29. highest_point—the x, y, z coordinates of the surface point which is the highest above the supporting surface (same as zhigh for postal data)

The irregular surface type is used to characterize surfaces that are generally planar but have wide variations in the local surface normals. We will not go into the algorithms for classifying surface types, since much has been published on that subject already. For general references, the reader is referred to [3, 35].

Edges have the following attributes:

1. viewpoint—the viewpoint from which the data was acquired

2. vertices—the vertices which form the endpoints of the edge

3. type—the type of the edge as classified by the mid level system: out_of_view, occluded, occluding, concave, convex, valid_to_invalid, or unknown

4. shape—the shape of an edge is always line because the mid level system approximates curved edges by multiple straight line edges

5. length—the length of the edge in inches

6. num_points—the number of datapoints in the edge

7. equation—the equation for the edge in parametric form [[x1, y1, z1], [x2, y2, z2]], where [x1, y1, z1] is the position of one endpoint of the edge and [x2, y2, z2] is a vector pointing from the first endpoint to the other endpoint

8. step—the size (in inches) of the step between the range data on either side of the edge

9. angle—the angle (in degrees) between the range data on either side of the edge

10. borders—the borders to which this edge belongs

Vertices have the following attributes:

1. viewpoint—the viewpoint from which the data was acquired

2. edges—the edges which use this vertex as an endpoint

3. position—the position of the vertex

4. coordinates—the image coordinates of the vertex

Relations have the following attributes:

1. viewpoint—the viewpoint from which the data was acquired

2. surfaces—the two surfaces which define this relation

3. type—the type of relation: convex, concave, occluding, occluded, or unknown

4. angle—the angle between the two surface normals

5. edges—the edges of the first surface which contribute to the relation

6. edges2—the edges of the second surface which contribute to the relation

7. compatibility—the compatibility measure between the two surfaces

8. border_angle—the angle between the two surfaces based only on data near the boundary

9. border_step—the size of the step between the two surfaces based only on data near the boundary

10. num_points—the number of datapoints in the relation

11. num_pts_border_angle—the number of datapoints in the relation that were used to compute the border_angle attribute

12. num_pts_border_step—the number of datapoints in the relation that were used to compute the border_step attribute

The viewpoint attribute for all of these data entities is included to allow for data from multiple viewpoints or multiple sensors. Also, if the viewpoint attribute has the value "model" then the entity is part of the model database rather than part of the scene data.

9. APPENDIX B: DEMPSTER–SHAFER THEORY

In this appendix we provide an introduction to the Dempster–Shafer theory for evidential reasoning and then we describe a simplification of this theory that is applicable to our hypothesis evaluation problem.

9.1 Introduction to Dempster–Shafer Theory

In the Dempster–Shafter (D–S) theory, the set of all the *elementary propositions* likely to occur in a domain of discourse is called the *frame of discernment* (FOD) and frequently denoted by Θ. The $2^{|\Theta|}$ subsets of Θ are called *propositions* and the set of all the propositions is denoted by 2^Θ.

In the D–S theory, *probability masses* are assigned to propositions which are subsets of Θ. The interpretation to be given to the probability mass assigned to any subset of Θ is that the mass is free to move to any element of that subset. Under this interpretation, the probability mass assigned to Θ represents ignorance, since this mass may move to any element of the entire frame of discernment. When a source of evidence assigns probability masses to the propositions represented by the subsets of Θ, the resulting function is called a *basic probability assignment* (BPA). Sometimes this function is also called a *probability mass function*. A *basic probability assignment* is a function m: $2^\Theta \rightarrow [0, 1]$, where

$$\sum_{X \subseteq \Theta} m(X) = 1$$

and $m(\varnothing) = 0$ with $\varnothing$ being the null proposition. We think of $m(X)$ as the measure of the probability mass constrained to stay in X but free to move throughout X.

A *belief function Bel*: $2^\Theta \rightarrow [0, 1]$ is defined by

$$Bel(X) = \sum_{Y \subseteq X} m(Y)$$

which says that to compute our total belief $Bel(X)$ in a proposition X we must add the probability masses for all the propositions that imply X. Note that if $Y \subseteq X$ then Y implies X. As shown by Shafer, there corresponds to each belief function one and only one basic probability assignment and vice versa. We think of $Bel(X)$ as the measure of the total probability mass constrained to stay somewhere in X.

A *commonality function* Q: $2^\Theta \rightarrow [0, 1]$ is defined by

$$Q(X) = \sum_{X \subseteq Y} m(Y).$$

We think of $Q(X)$ as the measure of the total probability mass that can move freely to any point in X. Commonality functions are related to belief functions by

$$Bel(X) = \sum_{Y \subseteq \neg X} (-1)^{|Y|} Q(Y)$$

$$Q(X) = \sum_{Y \subseteq X} (-1)^{|Y|} Bel(\neg Y).$$

A *doubt function Dou*: $2^\Theta \rightarrow [0, 1]$ is related to a belief function by

$$Dou(X) = Bel(\neg X).$$

We think of $Dou(X)$ as the measure of the total probability mass constrained to stay out of X.

A *plausibility function Pl*: $2^\Theta \rightarrow [0, 1]$ is related to belief and doubt functions by

$$Pl(X) = 1 - Dou(X) = 1 - Bel(\neg X).$$

We think of $Pl(X)$ as the measure of the total probability mass that can move into X though it is not necessary that it can all move to a single point.

Given two independent belief functions, Bel_1 and Bel_2, to which correspond the BPAs m_1 and m_2, respectively, we may combine them by using Dempster's rule to yield the belief function Bel to which corresponds the BPA m. This combination is usually denoted by $Bel = Bel_1 \oplus Bel_2$ or equivalently, $m = m_1 \oplus m_2$ and is defined by

$$m(X) = K \sum_{X_i \cap X_j = X} m_1(X_i) m_2(X_j)$$

$$K^{-1} = 1 - \sum_{X_i \cap X_j = \varnothing} m_1(X_i) m_2(X_j).$$

K is a normalization constant that accounts for the fact that in general there will be X_i and X_j such that $X_i \cap X_j = \varnothing$. Multiplying by K ensures that $\sum_{X \subseteq \Theta} m(X) = 1$. This combination may also be computed by the multiplication of commonality functions:

$$Q(X) = K Q_1(X) Q_2(X)$$

$$K^{-1} = \sum_{Y \subseteq \Theta} (-1)^{|Y|+1} Q_1(Y) Q_2(Y).$$

The combination of n BPAs with identical FODs is denoted by

$$m = \bigoplus_{1 \le i \le n} m_i.$$

The commonality functions of the combined BPA are defined by

$$Q(X) = K \prod_{1 \le i \le n} Q_i(X)$$

$$K^{-1} = \sum_{Y \subseteq \Theta} (-1)^{|Y|+1} \prod_{1 \le i \le n} Q_i(Y).$$

9.2. Dichotomous Frames of Discernment

We will now consider a special type of FOD which is useful for the verification of hypotheses. We begin with some definitions.

A frame of discernment Θ is called a *binary frame of discernment* (*BFOD*) if it contains two elements.

A binary frame of discernment Θ is called a *dichotomous frame of discernment* (*DFOD*) if it contains two elements, a proposition and its negation. For example: $\Theta = \{X, \neg X\}$. A BPA on this DFOD has three elements: $m(X)$, $m(\neg X)$, and $m(\Theta)$.

Belief functions and BPAs defined on binary or dichotomous frames of discernment are referred to as binary or dichotomous belief functions and BPAs, respectively.

The relationships between m, *Bel*, *Dou*, Q, and *Pl* are well defined in a DFOD and can be computed in constant time. Table 2 shows the values of all of these functions for X, $\neg X$, Θ, and $\varnothing$ in terms of $m(X)$ and $m(\neg X)$. Note that $m(\Theta) = 1 - m(X) - m(\neg X)$ by definition.

A special case of Dempster's rule is used to combine two dichotomous BPAs with the same DFOD. The result is always a dichotomous BPA with the same DFOD. The equations for each focal element of the combination $m = m_1 \oplus m_2$ are

$$\begin{aligned} m(X) &= K[m_1(X)m_2(X) + m_1(X)m_2(\Theta) + m_1(\Theta)m_2(X)] \\ m(\neg X) &= K[m_1(\neg X)m_2(\neg X) + m_1(\neg X)m_2(\Theta) \\ &\quad + m_1(\Theta)m_2(\neg X)] \\ m(\Theta) &= K[m_1(\Theta)m_2(\Theta)] \\ K^{-1} &= 1 - m_1(X)m_2(\neg X) - m_1(\neg X)m_2(X). \end{aligned}$$

To combine n dichotomous BPAs with identical DFODs we multiply commonality functions. The commonality functions of the combination $m = \oplus_{1 \le i \le n} m_i$ are

$$\begin{aligned} Q(X) &= K \prod_{1 \le i \le n} Q_i(X) \\ &= K \prod_{1 \le i \le n} (1 - m_i(\neg X)) \\ Q(\neg X) &= K \prod_{1 \le i \le n} Q_i(\neg X) \\ &= K \prod_{1 \le i \le n} (1 - m_i(X)) \\ Q(\Theta) &= K \prod_{1 \le i \le n} Q_i(\Theta) \\ &= K \prod_{1 \le i \le n} (1 - m_i(X) - m_i(\neg X)) \\ K^{-1} &= \prod_{1 \le i \le n} Q_i(X)) + \prod_{1 \le i \le n} Q_i(\neg X) - \prod_{1 \le i \le n} Q_i(\Theta) \\ &= \prod_{1 \le i \le n} (1 - m_i(\neg X)) + \prod_{1 \le i \le n} (1 - m_i(X)) \\ &\quad - \prod_{1 \le i \le n} (1 - m_i(X) - m_i(\neg X)). \end{aligned}$$

Therefore, for a DFOD the combined BPA is defined by

$$\begin{aligned} m(X) &= 1 - Q(\neg X) \\ &= 1 - K \prod_{1 \le i \le n} (1 - m_i(X)) \\ m(\neg X) &= 1 - Q(X) \\ &= 1 - K \prod_{1 \le i \le n} (1 - m_i(\neg X)) \\ m(\Theta) &= Q(\Theta) \\ &= 1 - K \prod_{1 \le i \le n} (1 - m_i(X) - m_i(\neg X)). \end{aligned}$$

Note that these functions can be computed in time that is linear in n.

TABLE 2
Relationships between Quantities for a Dichotomous Frame of Discernment

	X	$\neg X$	Θ	$\varnothing$
m	$m(X)$	$m(\neg X)$	$1 - m(X) - m(\neg X)$	0
Bel	$m(X)$	$m(\neg X)$	1	0
Dou	$m(\neg X)$	$m(X)$	0	1
Q	$1 - m(\neg X)$	$1 - m(X)$	$1 - m(X) - m(\neg X)$	1
Pl	$1 - m(\neg X)$	$1 - m(X)$	1	0

ACKNOWLEDGMENTS

We thank Dr. Frederick R. Glickman and Mr. Gary P. Herring of the U.S. Postal Service Office of Advanced Technology for many insightful discussions as the generic object recognition project has progressed from the initial conceptualization through the current implementation. Dr. John Tan, Dr. Gerardo Garcia, Mr. Jeff Shaevel, and Dr. Marc

Bastuscheck of Arthur D. Little, Inc. also deserve thanks for their advice throughout this project. We also thank Kirk D. Smith and Robert L. Cromwell for the development of the low and mid level range data processing, segmentation, and characterization system.

REFERENCES

1. K. M. Andress and A. C. Kak, Evidence accumulation and flow of control in a hierarchical spatial reasoning system, *AI Magazine* **9**(2), Summer 1988, 75–95. For an updated account, see *The PSEIKI Report—Version 3,* Purdue University Technical Report TR-EE 89-35, November 1989.
2. R. Bajcsy and F. Solina, Three dimensional object representation revisited, in *Proceedings of the First International Conference on Computer Vision*, IEEE Computer Society, 1987.
3. P. J. Besl and R. C. Jain, Invariant surface characteristics for 3D object recognition in range images, *Comput. Vision Graphics Image Process.* **33,** 1986, 33–80.
4. C. H. Chen and A. C. Kak, Modeling and calibration of a structured light scanner for 3-D robot vision, in *Proceedings of the 1987 IEEE International Conference on Robotics and Automation,* pp. 807–815.
5. C. H. Chen and A. C. Kak, *3D-POLY: A Robot Vision System for Recognizing Objects in Occluded Environments,* Purdue University Technical Report TR-EE 88-48, December 1988.
6. C. H. Chen and A. C. Kak, A robot vision system for recognizing 3-D objects in low-order polynomial time, *IEEE Trans. Systems Man Cybernet.* **19**(6), 1989, 1535–1563.
7. C. K. Cowan, J. DeCurtins, P. G. Mulgaonkar, and A. R. de St. Vincent, Advanced research in range-image interpretation at SRI, in *Third United States Postal Service Advanced Technology Conference, 1988,* pp. 361–375.
8. C. K. Cowan, P. G. Mulgaonkar, and A. R. de St. Vincent, Detecting global symmetries of mailpieces in range images, in *Third United States Postal Service Advanced Technology Conference, 1988,* pp. 376–390.
9. O. D. Faugeras and M. Hebert, The representation, recognition, and locating of 3-D objects, *Int. J. Robotics Res.* **5**(3), 1986, 27–52.
10. J. E. Graver and M. E. Watkins, *Combinatorics with Emphasis on the Theory of Graphs*, Springer-Verlag, New York, 1977.
11. W. E. L. Grimson and T. Lozano-Pérez, Model-based recognition and localization from sparse range or tactile data, *Int. J. Robotics Res.* **3**(3), 1984, 3–35.
12. W. E. L. Grimson and T. Lozano-Pérez, Localizing overlapping parts by searching the interpretation tree, *IEEE Trans. Pattern Anal. Mach. Intelligence* **PAMI-9**(4), 1987, 469–482.
13. C. Hansen and T. C. Henderson, CAGD-based computer vision, *IEEE Trans. Pattern Anal. Mach. Intelligence* **PAMI-11**(11), 1989, 1181–1193.
14. S. A. Hutchinson and A. C. Kak, Planning sensing strategies in a robot work cell with multi-sensor capabilities, *IEEE Trans. Robotics Automation* **5**(6), 1989, 765–783.
15. C. Jacobus, A. J. Riggs, L. M. Tomko, and K. G. Wesolowicz, Laser radar range imaging sensor for postal applications: A progress report, in *Third United States Postal Service Advanced Technology Conference, 1988,* 6–31.
16. A. C. Kak, K. D. Smith, A. J. Vayda, and R. L. Cromwell, Range image interpretation for postal object recognition, in *Third United States Postal Service Advanced Technology Conference, 1988,* 391–405.
17. A. C. Kak, K. D. Smith, A. J. Vayda, and R. L. Cromwell, *Advanced Research in Range Interpretation Techniques*, Purdue University Technical TR-EE 90-71, December 1990.
18. A. C. Kak, A. J. Vayda, R. L. Cromwell, W. Y. Kim, and C. H. Chen, Knowledge-based robotics, in *Proceedings of the 1987 IEEE International Conference on Robotics and Automation,* pp. 637–646.
19. A. C. Kak, A. J. Vayda, R. L. Cromwell, W. Y. Kim, and C. H. Chen, Knowledge-based robotics, *Int. J. Production Res.* **26**(5), 1988, 707–734
20. A. C. Kak, A. J. Vayda, K. D. Smith, and C. H. Chen, Recognition strategies for 3-D objects in occluded environments, in *Traditional and Non-Traditional Robotic Sensors* (T. C. Henderson, Ed.), NATO Advanced Research Workshop Series F, Springer-Verlag, Berlin, 1990.
21. W. Y. Kim and A. C. Kak, 3-D object recognition using bipartite matching embedded in discrete relaxation, *IEEE Trans. Pattern Anal. Mach. Intelligence*, **PAMI-13**(3), 1991, 224–251.
22. T. Lozano-Pérez, A simple motion-planning algorithm for general robot manipulators, *IEEE J. Robotics Automation* **RA-3**(3), 1987.
23. R. A. McClain and N. S. Kenig, Range image interpretation for mailpiece recognition: An interim report, in *Third United States Postal Service Advanced Technology Conference, 1988,* pp. 943–957.
24. P. G. Mulgaonkar, C. K. Cowan, and J. DeCurtins, Scene description using range data, in *Proceedings of the Workshop on the Interpretation of 3D Scenes,* IEEE Computer Society, 1989, pp. 138–144.
25. P. G. Mulgaonkar and J. DeCurtins, Scene description for object manipulation in unstructured environments, in *Proceedings of the 1989 IEEE International Conference on Robotics and Automation,* 354–359.
26. G. A. Pierce, K. C. Nitz, M. McDowell, SRI three-dimensional range-image acquisition system, in *Third United States Postal Service Advanced Technology Conference, 1988,* pp. 32–42.
27. J. P. Rosenfeld, A range imaging system based on space coding for postal applications, in *Third United States Postal Service Advanced Technology Conference, 1988,* pp. 43–57.
28. S. C. Roth, Ray casting for modeling solids, *Comput. Graphics Image Process.* **18,** 1982.
29. R. J. Safranek, S. Gottschlich, and A. C. Kak, Evidence accumulation using binary frames of discernment for verification vision, *IEEE Trans. Robotics Automation* **RA-6**(4), 1990, 405–417.
30. G. Shafer, *A Mathematical Theory of Evidence*, Princeton Univ. Press, 1976.
31. F. Solina and R. Bajcsy, Shape recovery of mail pieces using deformable models, in *Third United States Postal Service Advanced Technology Conference, 1988,* pp. 988–1002.
32. F. Solina and R. Bajcsy, Recovery of parametric models from range images: The case for superquadrics with global deformations, *IEEE Trans. Pattern Anal. Mach. Intelligence* **PAMI-12**(2), 1990, 131–147.
33. A. J. Vayda and A. C. Kak, Geometric reasoning for pose and size estimation of generic shaped objects, in *Proceedings of the 1990 IEEE International Conference on Robotics and Automation,* pp. 782–789.
34. A. J. Vayda and A. C. Kak, *Reasoning with Geometric Constraints for Generic 3-D Object Recognition in Occluded Environments,* Purdue University Technical Report TR-EE 90-72, December 1990.
35. H. S. Yang and A. C. Kak, Determination of the identity, position, and orientation of the topmost object in a pile, *Comput. Vision, Graphics Image Process.* **36,** 1986, 229–255.
36. H. S. Yang and A. C. Kak, Edge extraction and labelling from structured light 3-D vision data, in *Selected Topics in Signal Processing* (S. Haykin, Ed.), pp. 148–193, Prentice–Hall, Englewood Cliffs, NJ, 1989.