CVGIP: IMAGE UNDERSTANDING
Vol. 58, No. 1, July, pp. 33–48, 1993

CAD-Based Vision: Object Recognition in Cluttered Range Images Using Recognition Strategies

FARSHID ARMAN

Siemens Corporate Research, 755 College Road East, Princeton, New Jersey 08540

AND

J. K. AGGARWAL

Computer and Vision Research Center, Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, Texas 78712

Received August 19, 1991; revised September 15, 1992

This paper addresses the problem of recognizing an object in a given scene using a three-dimensional model of the object. The scene may contain several overlapping objects, arbitrarily positioned and oriented. A laser range scanner is used to collect three-dimensional (3D) data points from the scene. The collected data is segmented into surface patches, and the segments are used to calculate various 3D surface properties. The CAD models are designed using commercially available CADKEY and accessed via the industry standard IGES. The models are analyzed off-line to derive various geometric features, their relationships, and their attributes. A strategy for identifying each model is then automatically generated and stored. The strategy is applied at run-time to complete the task of object recognition. The goal of the generated strategy is to select the model's geometric features in the sequence which may best be used to identify and locate the model in the scene. The generated strategy is guided by several factors, such as the visibility, detectability, the frequency of occurrence, and the topology of the features. The paper concludes with examples of the generated strategies and their application to object recognition in several scenes containing multiple objects.

1. INTRODUCTION

The paper addresses the problem of CAD-based object recognition. The objective is to locate an object in a scene containing several overlapping objects using its geometric CAD model. The solution to this problem involves several steps (see Fig. 1), including steps to match the representations derived from the collected data to that of the given geometric model. Currently, most model-based vision systems use various pruning and searching strategies to reduce the total number of possible matches. However, most of these techniques depend on the expertise of the designer, the class of objects considered, and various search methods. This paper introduces a novel scheme to systematically derive a matching strategy from a geometric model. The given models are 3D CAD descriptions, and the 3D data is collected using a laser range scanner.

The desired object in the scene is identified by matching the model surfaces derived from the given CAD model to the segmented surface patches in the scene. The strategy dictates which model surface to locate first, followed by its neighbors, and the second surface and its neighbors, and so on. If the desired model feature is not found in the scene due to partial occlusion or viewpoint, the strategy may dictate a next-best feature. Some model features may not be detectable because of their geometry, or the effects of low level processing techniques—such as smoothing—applied to the collected data. In such cases, the recognition strategy is used to automatically identify difficult to detect model features and to disregard those features at matching, thus enhancing the performance of the system.

To compile the strategy, the features of the CAD model are used to construct a tree, referred to as the *recognition tree* [3]. The features form the leaves of the tree, and similar features are grouped to form the parent nodes. The topologically related features of the model are connected in the tree as well. Each connection in the tree is assigned a weight representing such factors as the visibility, detectability, the frequency of occurrence, and the topology of the features. Once the recognition tree for a particular CAD model has been compiled off-line, the corresponding recognition strategy is derived on-line by using the leaves of the tree which represent the features of the CAD model and the weights assigned to each tree link.

1049-9660/93 $5.00

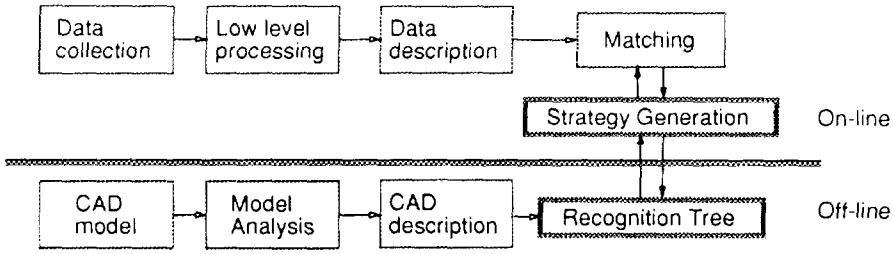

FIG. 1. The modules of the proposed model-based object recognition system.

The object recognition system presented here differs from previous ones in several respects. First, unlike most previous object recognition systems, the CAD system used to design the models is not an experimental one, but, a commercial 3D CAD system, CADKEY, used widely in the industry. Second, IGES, initial graphics exchange specification, has been used as an interface to the CAD system to infer the geometric information necessary for object recognition. IGES is an industry standard developed by the National Institute of Standards and Technology to allow for the *transfer* (or interchange) of CAD models between various commercial CAD systems. Using CADKEY and IGES decreases the dependency of the vision system on any particular CAD modeler and increases its applicability. Third, the model description is used to systematically derive a matching strategy from a geometric model. Such a methodology enables the vision system to recognize a larger set of objects, without requiring any constraints or conditions even when a new model is added to the database. By using the recognition strategy, all possible matching combinations of sensory features and model features need not be considered. This is not the case in tree search and attributed graph approaches to recognition. More importantly, by "precompiling" a recognition strategy, the vision system may perform more efficiently at run-time since less time is spent on model analysis in task execution. Fourth, using the recognition tree, the system is able to predict which CAD features will not be detectable in the scene. During recognition, the system will not expect to find those features, and so it will not spend time during the search process. Finally, the matching strategy does not rely significantly on moment-based and boundary-based features, and unlike many previous approaches to object recognition, our system does not impose an unconditional one-to-one matching of sensory features and model features. Thus, the oversegmentation of surface patches and the partial occlusion of objects are easily tolerated. The presented system has been implemented and successfully tested using 17 models in numerous scenes.

The paper is organized as follows: Section 2 briefly reviews the existing model-based object recognition systems. Section 3 provides an overview of the system presented here. Section 4 describes the off-line modeling analysis module of the system. Section 5 illustrates the details of the recognition strategy compilation. Section 6 presents the application of the recognition strategies to several scenes, and Section 7 presents the concluding remarks.

2. REVIEW OF 3D OBJECT RECOGNITION SYSTEMS

This section briefly reviews the matching module of various model-based object recognition systems. For a complete review of object recognition systems, see [5, 4].

Grimson [19, 20, 22] uses two sets of simple features for matching: linear edge fragments and circular arcs. To account for possible occlusions, null features are introduced and paired with otherwise unmatched features. Tree pruning is performed using two types of constraints: unary constraints, which consider only one feature at a time, and binary constraints, which use feature pairs and consider their interrelationship. Furthermore, Grimson uses several cross-type constraints such as the cross distance constraint and the cross component constraint. The final transformation is obtained by separately considering each of the edge fragments and the arcs for a consistent solution.

Flynn and Jain [16] order the features extracted from the scene by area and type. A series of unary and binary constraints are used to reduce the number of possible matches. The binary constraints are rotation estimates using feature pairs, orientation of feature pairs, visibility [14], and distance of parallel planes. The surviving hypotheses are verified using a synthetic image which is generated using the CAD modeler's description of the model and compared to the collected data.

Fan, Medioni, and Nevatia [12, 13, 11] represent objects and models using attributed graphs. The matching process starts by reducing the possible models based on the number of nodes, the visible 3D area, and the number of planar nodes in each graph. Next, the model graph with the largest set of matched nodes is chosen by comparing edge

adjacency types, curvature, orientation, and distance between centers of inertia.

Previous approaches to automatic generation of recognition algorithms (first introduced by Goad [18]) includes work by Ikeuchi [25, 26] and Hansen and Henderson [23, 24]. Ikeuchi [25, 26] derives a recognition strategy for a given model in the form of an interpretation tree. Similar to the aspect graph representation, the Gaussian sphere is tesselated and each tessel is used as a possible viewpoint; all possible shapes of the object are generated and examined. Similar shapes are grouped to form the leaves of the interpretation tree, referred to as the "attitude group." Recursive divisions of the aspects are generated to form various paths from the root of the interpretation tree, representing the model, to the leaves. For each aspect, a local coordinate system is derived using a set of rules [30] along with the appropriate transformation to the coordinate system of the model. At run-time, the closest surface patch is compared to the possible aspects, several possible candidates are chosen, and the final selection is made using edges of the aspect and the chosen surface patch.

Hansen and Henderson [23, 24] developed a system for the automatic generation of recognition strategies using several feature properties, such as robustness, completeness, consistency, cost, uniqueness, and rarity. This method does not depend on any set of specific features; on the contrary, it selects (ranks) the given features to be used in matching objects to the models based on the above properties. While the work presented in this paper is closest in nature to that of Hansen and Henderson, our scheme allows for each of the feature properties to have various degrees of "importance." In our approach, the topology of the models (the interrelations of the model features) plays an important role in ranking features, and the ranking is adaptive to the scene to account for possible occlusions.

3. SYSTEM OVERVIEW

The vision system presented here may be broadly divided into two components—on-line and off-line, and each component into several modules: data collection, low level image processing, data description, CAD model, model analysis, and CAD description (see Fig. 1). This section presents an outline of the system.

In the first module of the on-line component of the system, 3D data is collected using a laser range scanner from a scene containing several overlapping objects. The collected data is then segmented, partitioning the pixels into several disconnected sets, each representing a homogeneous surface patch [35]. Each segmented surface patch is then classified by surface type as planar, concave, or convex. Surface attributes such as surface normal, or axis of symmetry and surface area are then derived to form the description of the input data.

In the first module of the off-line component of the system, a commercial CAD system, CADKEY, is used to model numerous planar and non-planar (simple quadric) objects. Some of the objects are off-the-shelf items and some have been constructed for experimental purposes. The IGES description of the CAD model is analyzed to derive the features of the model and their relationships. Several surface attributes are then calculated for each CAD surface forming the CAD model description. The CAD surfaces are then used to organize the *recognition tree* (see Fig. 2).To identify a model in the given scene and to derive a recognition strategy, the corresponding recognition tree of the desired model is traversed. By considering the weights assigned to each link and by using the leaves of the tree, a series of filters is issued. A filter is a set of conditions, which depend upon the model feature, used to compare the model features and the features from the scene. The input to each filter is the set of surface patches, and the output of each filter is the sets in which at least one member has satisfied the conditions of the filter. The successful sets in turn become the input to the next successive filter. By issuing a series of filters, the number of possible candidate sets to match the desired model is reduced successively. The set of surface patches which satisfy a minimum number of filters is considered "matched" to the model. The weights assigned to the links of the recognition tree depend on several factors, such as frequency of occurrence of a feature, its detectability, and its location within the model.

4. OBJECT MODELING

Models are the a priori geometrical and topological knowledge that the vision system has about a set of objects. This knowledge is compared with the descriptions of the input data obtained through lower level processes in the matching stage, deriving an interpretation of the input data. In three-dimensional computer vision, models must contain information about the shape of the object. There are two main approaches to model-building: through the "manual" (training) and through a CAD system. In the "manual" approach, multiple viewpoints of the object are integrated in a coherent fashion to provide a 3D description of the object [38, 7, 11]. CAD systems use a set of predefined primitives which allow the user to interactively construct the CAD model of an object.

Many institutions have experimented with CAD systems geared toward the object recognition problem, such as ALPHA_1 [8–10], PADL (part and assembly description language) [39], GEOMAP (geometrical modeling and processing) [32, 33], and VANTAGE [29], designed to be interfaced to other programs including model-based vision

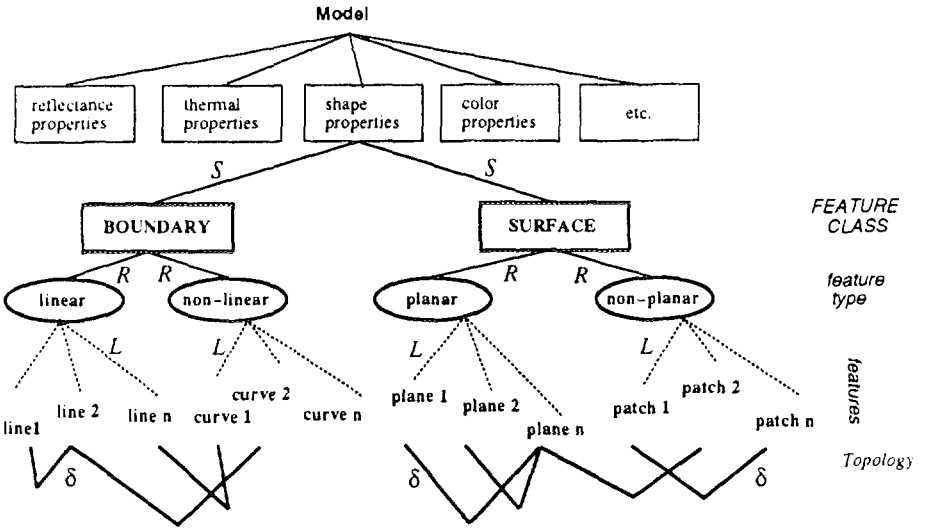

FIG. 2. Recognition tree where only the shape properties have been delineated explicitly. Symbols represent the weights assigned to each link.

systems. These "experimental" CAD systems have been used to model the objects in almost all CAD-based vision systems [27, 25, 31, 12, 28]. In a majority of these cases, the internal representation of the CAD system may freely be accessed, allowing an "easy" derivation of the necessary information. However, the applicability of such systems is very limited since the CAD systems are not widely used by others. An alternative approach in CAD modeling is to use commercially available CAD systems that are widely used in the industry. The major hurdle in using commercial CAD systems, however, is that the purpose of such systems is to design and manufacture parts, not to recognize them. Hence, information that is useful in image understanding tasks, such as relationships among subparts of an object and axes of symmetry, are not stored. Rather, the emphasis in most commercial CAD systems is on the graphical display of objects and easy manipulation of geometric entities. Furthermore, commercial CAD systems are hard to interface to, since they are designed as independent entities and their internal representations are inaccessible. Nevertheless, to integrate model-based vision systems into commercial manufacturing and inspection systems one must utilize existing commercial CAD systems, such as CATIA, AutoCAD, and CADKEY. One method by which computer vision systems may utilize commercial CAD systems is by using IGES [37]. Most commercial CAD systems are able to convert to/from their internal representations from/to the IGES. See Table 1 for a subset of supported IGES entities and their corresponding attributes used in this research.

TABLE 1
A Partial List of Supported Entities and the Corresponding Attributes in IGES

IGES entity	Attributes
Point	Coordinate
Circular curve	Center point, starting, and ending points
Composite curve	Number of components and pointers to their attributes
Line	Start and end points
Parametric spline curve	Spline type, degree of continuity, number of segments, breakpoints, coefficients

4.1. Feature Extraction from the IGES Description

When using IGES the relationships of various entities must be inferred using the available information.[1] In addition, surfaces must be formed from the collection of curves and lines, and surface properties such as curvature and surface area must then be calculated. Furthermore, other information such as line color and thickness must be "filtered out" from the IGES file since this information is not useful in the recognition process. This research recognizes six types of surfaces (see Fig. 3). `Cylinder`, `cone`, `plane`, and `disk` surfaces are defined using combinations of lines and arcs, whereas `peak` surfaces are defined using a combination of two or more arcs. In addition, the system recognizes and extracts `faces` of polyhedrons as a series of connected lines.

[1] IGES will be replaced in near future by a new standard PDES (product data exchange using STEP) currently under development. STEP (standard for the exchange of product model data) is the international standard for CAD data transfer setup by the International Standards Organization. PDES/STEP has many advantages over IGES, such as preserving the topology of entities.

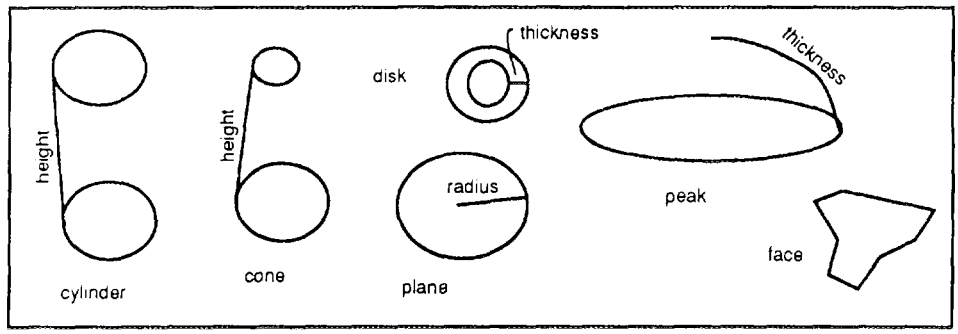

FIG. 3. Six surface classes derived from IGES.

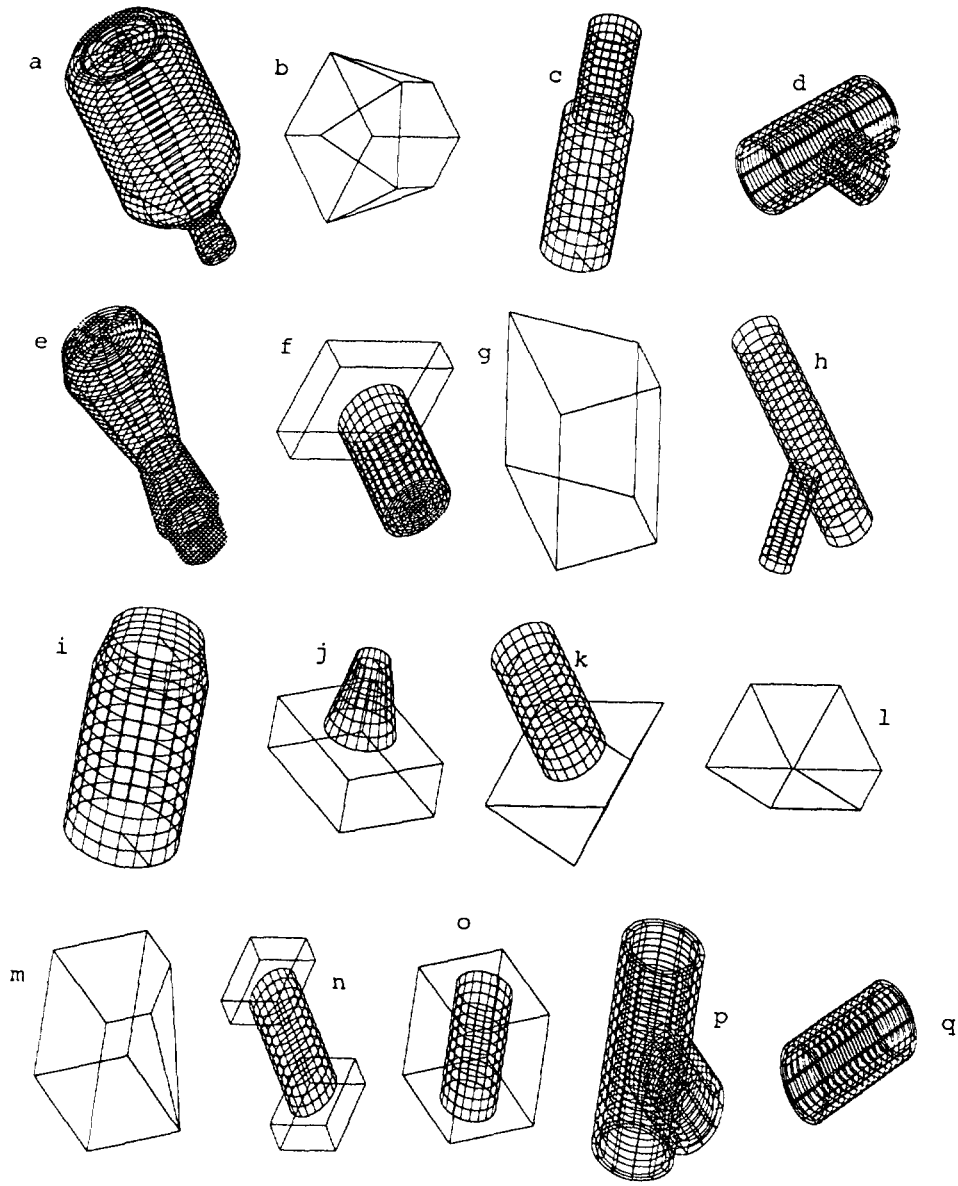

FIG. 4. The database of CADKEY models designed and used in this research.

Once the model has been designed, CADKEY's IGES translator is used to convert the internal representation of the model into IGES format. In this format, the surfaces of the models are defined as meshes, where each mesh consists of a set of 3D parametric cubic splines (see Fig. 4). The choice of representation is dictated by IGES translator of CADKEY and not at the time of design. Given the IGES file, the objective is to analyze the splines and lines, form and classify the surfaces, and derive the appropriate surface properties.

The process of CAD feature extraction consists of several steps. First, the unnecessary information (such as color of the drawing, etc.) is removed from the IGES model description. Second, we examine the entities for repetitious occurrences. We then initiate the process of surface construction. For example, in defining a cone, two splines are used to define each base and a line to define the body of the cone (see Fig. 3); other splines are discarded since they do not provide any additional information necessary for surface type classification or attribute calculation. Finally, the list of connected entities (splines and lines) is analyzed to classify each surface and to calculate the related attributes [1] (see Table 2). At the end of this procedure the desired CAD description is obtained and used in the matching module (see Fig. 5).

5. RECOGNITION STRATEGY COMPILATION

Once the desired descriptions for the CAD model and the collected data are obtained, the matching module is initiated to localize the desired CAD model. As mentioned earlier, a recognition strategy is compiled in the off-line stage of the vision system, and the strategy is applied in the on-line stage to locate the desired model in the given scene (see Fig. 1).

The strategy is to derive an efficient search order of model features. This involves inferring features from the IGES description of the CAD model and ranking the features in an order in which the run-time search patch is shortened. Feature ranking is based upon various characteristics: such as feature size, visibility, its relationships to its neighbors and their types, and distinctive features

TABLE 2
Six Derived Surface Classes and Calculated Attributes, see Fig. 3

Surface type	Calculated attributes
Cylinder	Radius of base, axis vector, surface area
Cone	Radii of the bases, axis vector, surface area
Disk	Radii of inner and outer circle, surface normal, surface area
Peak	Minimum and maximum curvature, surface area (approx.)
Plane	Radius, surface normal, surface area
Face	Surface normal, edge lengths, internal angles surface area

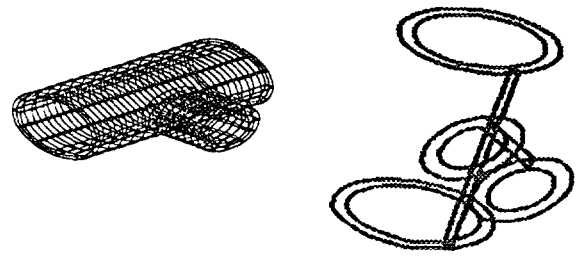

FIG. 5. The original CADKEY model on the left, the derived model used in the vision system on the right.

which might exist in a feature (a cavity on a surface patch, for example), computational cost of approximating the features, the reliability of feature types, and frequency of occurrence. This is achieved using the recognition tree described in this section.

5.1. Recognition Tree Organization

5.1.1. *Motivation*

The organization of the recognition tree is as follows (see Fig. 2): Beginning from the leaves representing the features of the model, similar features are grouped to form a parent node in the tree, referred to as the *Feature Type* node. Similar feature types are grouped to form the next higher level, the *Feature Class* node. The feature class nodes suggest a representation scheme to best describe the feature types. For example, all planar regions in a CAD model may be grouped under the *planar* feature type, which is in turn grouped under the *surface* feature class. As shown in Fig. 2, the tree focuses on shape properties, two alternative description means, *Boundary* and *Surface*, may describe shape properties of the model; the feature class *Surface* has feature types *Planar* and *Nonplanar*. Each *Feature Type* has in turn many *Features* linked to it, each with a set of properties. The models used in this research have surface information only; therefore, the *Surface* branch of the recognition tree has been implemented.

The recognition tree is used to issue a series of filters. A filter is a set of conditions used to compare the model features and the features from the scene. The segmented surface patches are partitioned into a number of sets using jump boundaries and concave edges; these sets are the input to the filters. The filters' output is the sets in which at least one member has satisfied the filter conditions. The conditions depend upon the model feature. For example, filters for cylindrical surfaces compare surface classification and surface area of the scene features with those of the model feature; surface patches with convex surface types and surface areas less than the model feature pass the filter. The output set is in turn input to the next filter. The succession of filters reduces the number of possible candidates in the scene to match the desired model [2,

3]. Filters will be described in more detail in the next subsection.

Many factors must be considered in determining the proper order of filters, or, effectively, the proper ranking of the model features. First and foremost is the choice of feature representation. In the past, there have been numerous efforts in representation schemes of objects [5, 4]. Each representation has advantages and disadvantages, such as the computational cost of deriving the representation from the input data, the accuracy and uniqueness of the description, etc. The second factor to consider in ranking the features is the repetition of a particular feature [24]. Features which appear frequently are much less susceptible to occlusion and hence more "valuable."[2] The third factor is whether a feature is visible, or detectable, using the chosen representation, sensor type, and the applied low level processes, such as noise removal techniques. The last factor to consider is the relationships of the feature with its neighbors. Features with high ranking neighbors should be ranked higher than similar features with low ranking neighbors. This is necessary when the feature in focus is occluded or otherwise undetectable; in such cases the neighbors of the feature may be needed to localize the object. These factors are built into the organization of the recognition tree (see Fig. 2).

5.1.2. *Level One*

The first level of the tree addresses the most fundamental considerations, the possible choices in representing shape properties depicted in the recognition tree as the *Feature Class* nodes. The possible choices considered are surface-based and boundary-based schemes. Of the factors listed above, accuracy and the cost of deriving the feature descriptions are used to choose between the two options. In incorporating this into the recognition tree, each branch on the first level is assigned a weight S defined as

$$S = \frac{\omega_{\text{accuracy}}\alpha_{\text{feature}} + \omega_{\text{cost}}\chi_{\text{feature}} + \dfrac{\omega_{\text{lower level weights}}}{\text{num of branches}} \Sigma_{\text{branches}} R}{\omega_{\text{accuracy}} + \omega_{\text{cost}} + \omega_{\text{lower levels}}}, \tag{1}$$

where ω is a constant weight representing the significance of each of the three terms; R, defined below, is the weight assigned to the lower level *Feature Class–Feature Type* links; *feature* is one of the *Feature Classes* in the recognition tree; α is a measure of the accuracy of a feature; and χ is the value which determines the computational cost of approximating a feature. Two-dimensional manifolds, such as surfaces, are less sensitive to noise than are 1D manifolds, such as boundary segments; therefore, when such features are extracted from the input data, they are more accurate: $\alpha_{\text{Surface}} > \alpha_{\text{Boundary}}$. Furthermore, in most cases the cost of approximating boundary segments is lower than that of approximating surfaces; thus, the former is more advantageous: $\chi_{\text{boundary}} > \chi_{\text{surface}}$. Additionally, the equation above includes the lower level weights, in which the other factors listed in Section 5.1.1 are considered, allowing for a "look ahead" in the early stages of the recognition procedure. This includes the knowledge of several key factors in recognition, such as the number of features that may be found in the chosen branch, and, as it will be shown later, the detectability of each feature.

5.1.3. *Level Two*

The second level of the recognition tree further considers the choice of representation schemes. Shapes described using boundary-based representation may be subdivided into linear and nonlinear segments, and surface-based representations may similarly be subdivided into planar and nonplanar patches. The factors used to decide between the two choices in each case are the number of features and the feature repetition. For example, if there are more nonplanar features in a model, then the nonplanar features should be given a higher ranking than the planar features since a search for planar regions may not be rewarding. Furthermore, feature repetition is also a factor in deciding between planar and nonplanar features. A planar feature repeated many times is valuable as a localization tool, since there is a lower probability of the feature's being occluded. In this example, then, the planar representation should be given a higher priority, and these features have a higher ranking. The weight R assigned to the next level of the recognition tree considers the above factors using the weighted average of three components:

$$R = \frac{\left[\begin{array}{l}\omega_{\text{number of features}}\Psi(\text{number of features}) \\ \quad + \omega_{\text{feature repetition}}\Psi(\text{feature repetition}) \\ \quad + \dfrac{\omega_{\text{lower level weights}}}{\text{number of branches}} \Sigma_{\text{branches}} L\end{array}\right]}{\omega_{\text{number of features}} + \omega_{\text{feature repetition}} + \omega_{\text{lower level weights}}}, \tag{2}$$

where L, defined below, is the "Feature Type-Feature" weight of the lower level summed over the lower branches. Again, ω is the weight assigned to each factor. The function Ψ maps a given x to a *scaled x:*

[2] Unique features are more important once such a feature has been located in a scene. Then, the feature may be used to isolate a particular model in a database of models; however, the problem at hand is to locate a *given* model in a scene. In such cases, a unique feature may easily be occluded; thus, it is less "valuable."

$$\Psi(x) = \begin{cases} 0 & \text{if } x < \psi_1 \\ 1 & \text{if } \psi_1 \le x < \psi_2 \\ 2 & \text{if } \psi_2 \le x < \psi_3 \\ 3 & \text{if } x \ge \psi_3, \end{cases} \tag{3}$$

where ψ_i is a predetermined threshold depending on the class of x; $x \in$ {repetition, frequency}. Such a mapping allows for a uniform "comparative system" at each node, where the branches are compared. If the actual values were used, the comparisons would be among different qualities. A similar mapping is also used in the lower levels of the tree, where absolute properties such as surface area and line lengths are used as ranking factors. At that level, such a mapping allows for a symbolic representation of feature properties. Such a representation in turn permits the filters to operate in a symbolic representation space, as the section below will show. The classes of x are then expanded to {repetition, frequency, length, area}. Table 3 exhibits the value of ψ_i assigned to each class. For example, if the feature is repeated only once, the feature repetition term of R adds no special significance to its corresponding branch; i.e., $\Psi(1) = 0$. On the other hand, a repeatability of 10 is, in many cases, significant; i.e., $\Psi(10) = 3$, adding significantly to R of its branch.

TABLE 3
The Set of Thresholds Used by the System in Eq. (3)

x	ψ_1	ψ_2	ψ_3
Feature repetition	2	5	8
Frequency of features	5	10	20
Line length/arc length (mm)	3	30	300
Surface area (mm^2)	2×10	2×10^3	2×10^5

5.1.4. *Level Three*

The last level of the tree focuses on feature detectability and topology to rank the CAD model's features. Using the Ψ mapping above, a detectability function is defined for each feature type (see Table 4). For simple features, such as a line, the Ψ function is used to check whether the feature is visible under the assumed conditions of the vision system. For example, if line length < 3 mm, then ψ(length) = 0, since the system's resolution combined with noise, smoothing processes, etc., does not allow for the localization of a line shorter than 3 mm. The values of Table 3 are reached by considering the input device resolution and object database. More complex features, such as the surface patch, combine many properties, e.g., surface area, the number of cavities in the surface. The following examples demonstrate a simple and more complex detectability equation:

$$\delta_{\text{line}} = \Psi(\text{length}),$$

$$\delta_{\text{patch}} = \begin{cases} 0, & \text{if } \nu_{\text{patch}} = 0 \\ \dfrac{\left[\omega_{\text{area}}\Psi(\text{area}) + \sum \omega_{\text{cavity}}\Psi(\text{cavity}) + \omega_{\text{visibility}}\nu_{\text{patch}}\right]}{\omega_{\text{area}} + \sum \omega_{\text{cavity}} + \omega_{\text{visibility}}}, & \text{otherwise.} \end{cases}$$

The first equation indicates that the detectability of a line is directly proportional to its length. The second indicates that the detectability of a surface patch is a weighted average of three factors: its surface area, the number of cavities in the surface patch, and the possibility that the surface will be visible (ν_{patch}) considering its geometry. The number of cavities is included in the equation because cavities provide additional distinctive features and make the surface patch more prominent. The visibility function, ν_{patch}, considers the geometry of each feature independently of its relations to other features in the model. Table 5 exhibits the definitions of ν_{patch} for each surface patch type. Using the detectability functions, the weights L for the last level of the recognition tree are defined as

$$L = \begin{cases} 0, & \text{if } \delta_{\text{feature}} = 0 \\ \left[\delta_{\text{feature}} + \left[\dfrac{\sum \delta_{\text{neighbor}}\omega_{\text{neighbor}}}{\sum \omega_{\text{neighbor}}}\right]\right] \Big/ 2.0, & \text{otherwise,} \end{cases} \tag{4}$$

where *neighbor* is a physically connected feature within the same *Feature Class*.

5.1.5. *Topology of the Model*

Features of the CAD model that are physical neighbors are connected in the recognition tree to reflect the topology. Each connection is assigned a weight δ^*(neighbor, feature) representing the detectability of the neighbor.[3] If δ^*(feature,neighbor) = 0, then the features are connected, bypassing the common neighbor (see Fig. 6). Such links are referred to as virtual links.

5.1.6. *Recognition Tree Construction*

The construction of the recognition tree starts with extracting the CAD features and inferring the topology of the features. Then, the weights of the recognition tree, as defined above, are calculated bottom-to-top. Table 6

[3] In the current implementation δ^* (neighbor, feature) = δ(neighbor) as defined in Table 4.

TABLE 4
The Detectability Functions Used for All Features in the Recognition Tree

Feature	Detectability definition, δ
Line	Ψ(length)
Curve	Ψ(arc length)
Top of cavity	Ψ(area of top)
Bottom of cavity	$\Psi\left(\frac{\text{area of top}}{\text{depth}}\right)$ (deeper cavities $\Rightarrow$ lower detectability)
Side of cavity	$\Psi\left(\frac{\text{area of top}}{\text{depth}}\right)$ (wider cavities $\Rightarrow$ higher detectability)
Cavity	$\frac{\omega_{top}\delta(top) + \omega_{bottom}\delta(bottom) + \omega_{side}\delta(side)}{\omega_{top} + \omega_{bottom} + \omega_{side}}$
Plane/patch	$\begin{cases} 0, & \text{if } \nu_{patch} = 0 \\ \frac{\left[\omega_{area}\Psi(area) + \sum \omega_{cavity}\Psi(cavity) + \omega_{visibility}\nu_{patch}\right]}{\omega_{area} + \sum \omega_{cavity} + \omega_{visibility}}, & \text{otherwise} \end{cases}$

contains the ω's used in the equations presented above. These values were determined manually by analyzing the complete database of the CAD models. The *Surface* branch of the recognition tree has been constructed automatically for all models in the database (Fig. 4).

Figure 7 shows an example of a recognition tree. Additional examples are presented in Section 6. In Fig. 7, analysis of the CAD model has yielded two nonplanar and three planar surfaces. The numbers on each edge of the tree are the calculated weights—R and L (see Fig. 2). The numbers inside the boxes (leaf nodes) are the surface numbers, and the numbers below each box are the calculated attributes: surface classification, surface area, visibility ν, and detectability δ (shown as d) [1, 3]. The topology of the surfaces is preserved in the recognition tree by connecting the two leaf nodes by solid lines.

TABLE 5
The Visibility Function Defined for Each Surface Class (See Fig. 3 and Table 2)

Surface class	ν_{patch}
Cylindrical	Ψ(height)
Cone	Ψ(height)
Disk	Ψ(thickness)
Peak	Ψ(thickness)
Plane	Ψ(radius)
Face	Ψ(area)

TABLE 6
The Set of Weights Used in Compiling the Recognition Tree

Weight, ω	Value	Weight, ω	Value
ω_{area}	3	$\omega_{neighbor}$	3
ω_{cavity}	3	$\omega_{number\ of\ features}$	2
$\omega_{visibility}$	5	$\omega_{feature\ repetition}$	4
ω_{top}	5	$\omega_{lower\ level\ weights}$	5
ω_{bottom}	2	$\omega_{accuracy}$	2
ω_{side}	2	ω_{cost}	3

Note. The weights are constant for all CAD models in the data base.

5.2. Recognition Strategy Generation

The recognition tree is used to generate a strategy for locating the desired model in the given scene. The strategy is in the form of a series of filters that are issued using the recognition tree. A filter is a series of constraints that check whether the properties of the surface patch from the scene are similar to those of the model features. Spe-

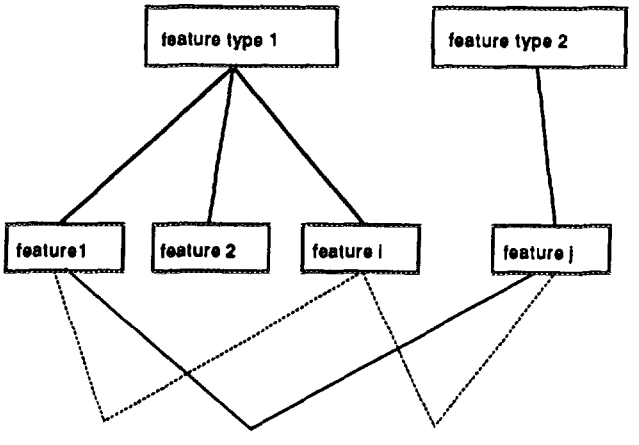

FIG. 6. Virtual links in the recognition tree. The detectability of feature i is zero; therefore, its neighbors, feature 1 and feature j, are connected, bypassing feature i.

cifically, there are two classes of filters. Unary filters examine unary constraints such as matching surface classifications (convex, concave, planar) and surface area. Since objects may be partially occluded and since data points are from a single view, the surface area of a surface patch from the scene is always expected to be less than the corresponding surface area of the model. For planar surface types, the number of edges of the face and the angles between the edges of the face are also used as filter conditions [2]. Binary filters check for constraints between two neighboring surfaces (binary constraints). The angle between the representative vectors of the two surfaces in the model is compared to the corresponding two surfaces in the scene. In addition, if the surfaces are nonplanar, the ratios of the approximated curvatures are used as a constraint. The curvature values are not used directly, since the approximated values are very sensitive to noise and are often inaccurate [15]. Rather, our experiments have shown that the ratios of curvatures of two surfaces (applied to cylindrical surface types only) are an effective constraint when used in addition to other constraints. Following segmentation, the connected surface patches are grouped into several sets. When a filter is issued, each member of each set is compared to the filter conditions. If at least one member passes the constraints (two members for binary filters), then the set is allowed to continue. Otherwise, the set is deactivated.

Given the recognition tree of a given model, the recognition strategy is derived by the following algorithm:

```
1:  for all features                  choose features according to weights
2:   FIND best feature                use unary filters
3:   for all neighbors                choose neighbors according to weights
4:    FIND best neighbor              use unary filters
5:    examine relationship            use binary filters
     next neighbor
   next feature
```

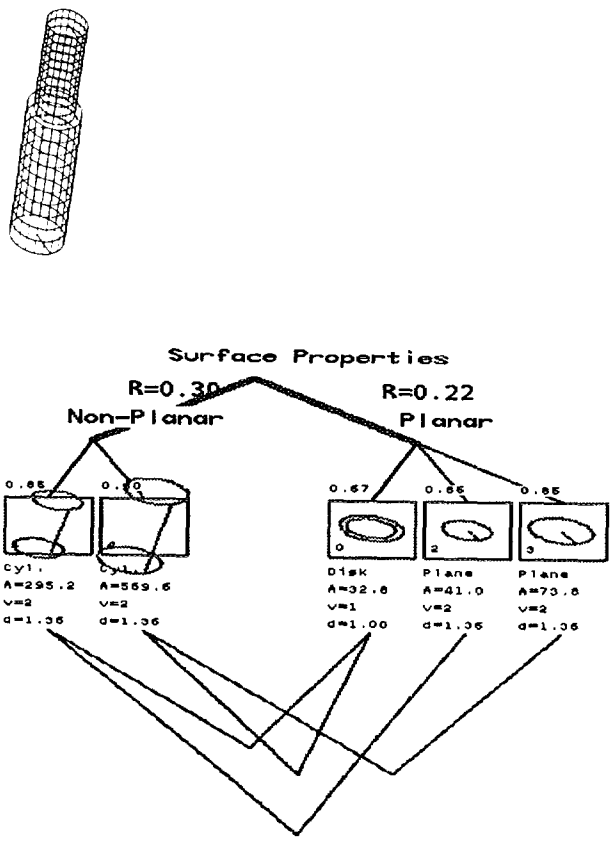

FIG. 7. The recognition tree generated for the CAD model on the top.

In Step 1, features are chosen in the order of the weight L (weight assigned to each leaf); in case of ties, surface areas are used. In Step 2, unary filters are used, and the sets which satisfy the filter conditions are passed on to the next step. In Step 3, the best neighbor of the first feature is chosen based upon L, and a second unary filter is issued (Step 4). The sets from the previous steps are tested against the constraints of this filter as before. Next, in Step 5, binary constraints are issued to check for the relations of the model feature and its chosen neighbor against the corresponding surface patches that passed the earlier two filters. Again, only the sets that satisfy the filter conditions are allowed to continue. This process continues for all other neighbors in order of the assigned L. At any stage, if none of the sets pass a particular filter (i.e., the model feature is not found in the scene), then the system breaks from the loop; in other words, the next best neighbor or the next best feature is chosen, and the search continues. As the system progresses through the succession of recognition filters, the number of candidate sets diminishes rapidly. If more than one set remains after all filters have been exhausted, then the set which has passed more binary filters is considered the successful match. If no sets have passed at least one binary filter, then no matches exist.

6. EXPERIMENTAL RESULTS

The range images are obtained from a Technical Arts laser range scanner [36] equipped with a translation stage

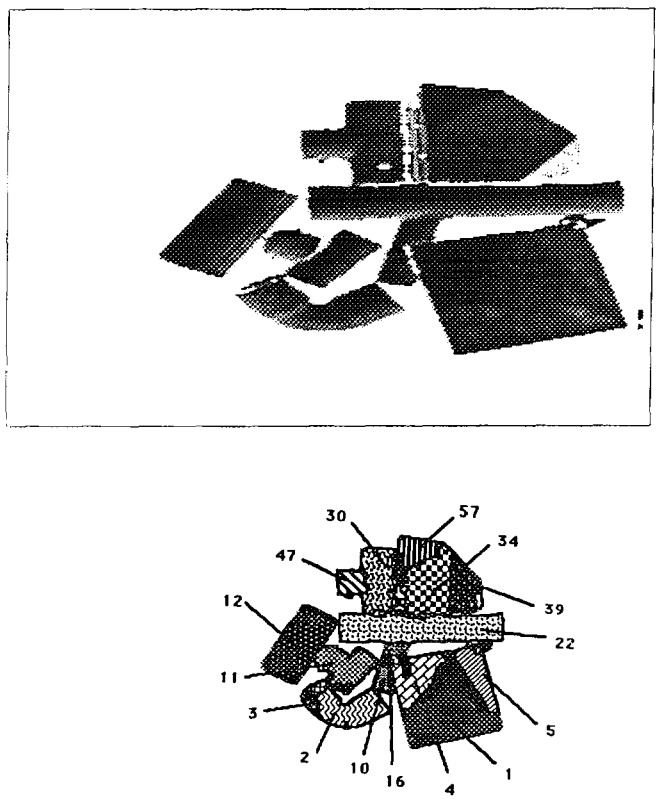

FIG. 8. Scene 1. The input image on the top (shown here using Lambertian shading), the segmented output on the bottom, and the calculated attributes are presented in Table 7.

where the objects are placed and scanned. The data points are partitioned into a set of homogeneous surface patches [35]. Once the segmentation is achieved, the desired description for each surface patch is then derived. The specific attributes for each surface patch are the surface area, surface type (convex, concave, planar), axis of symmetry (if nonplanar), surface normal (if planar), approximation of the radius of curvature (if nonplanar), and a list of neighboring patches [6, 1].

This section presents several applications of the above strategies on several scenes (Figs. 8 through 13). More than 40 experiments have been performed using the database of CAD models (Fig. 4) in numerous scenes [1]. The experiments involve objects that are partially occluded by other objects in the scene, objects that are on the top of the pile of objects, and cases where the surfaces were oversegmented. Figures 8, 10, and 12 depict several representative scenes. In each figure, the input scene is shown at the top, and the segmentation results are displayed on the bottom. Figures 9, 11, and 13 show several applications of the recognition trees. In instances when the desired CAD model did not exist in the scene, the strategy successfully exited, noting the absence of the desired model from the scene. For example, model h in Fig. 4

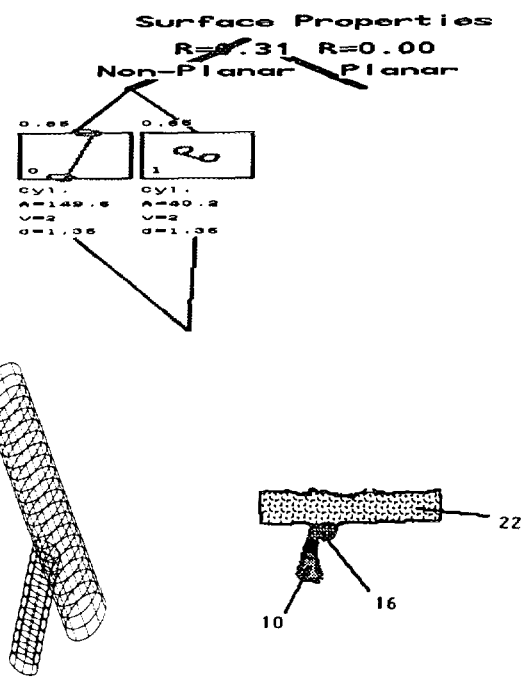

FIG. 9. Application of the recognition tree (at the top) to scene 1, the model is at the bottom left, and the identified surfaces are displayed at the bottom right.

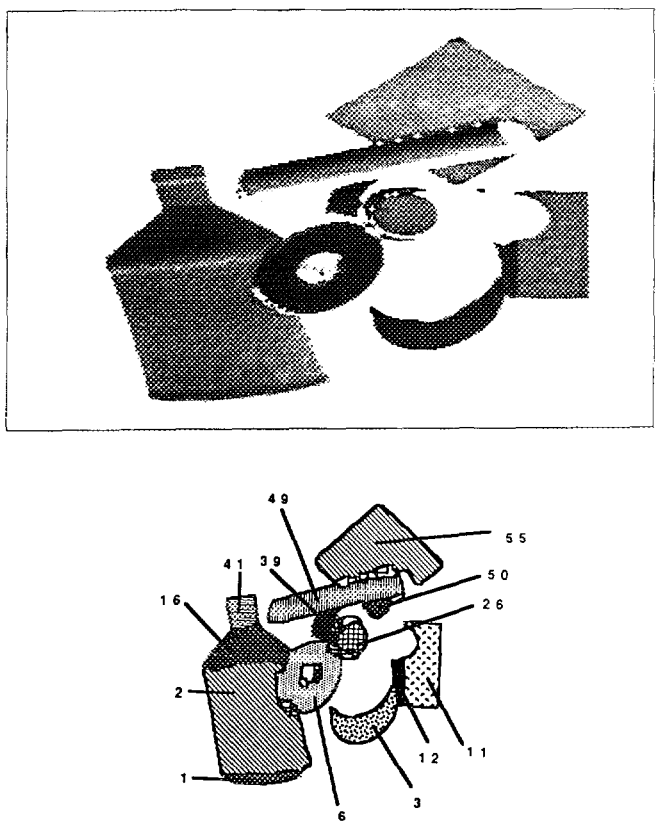

FIG. 10. Scene 2. The input image on the top, the segmented output on the bottom, and the calculated attributes are presented in Table 8.

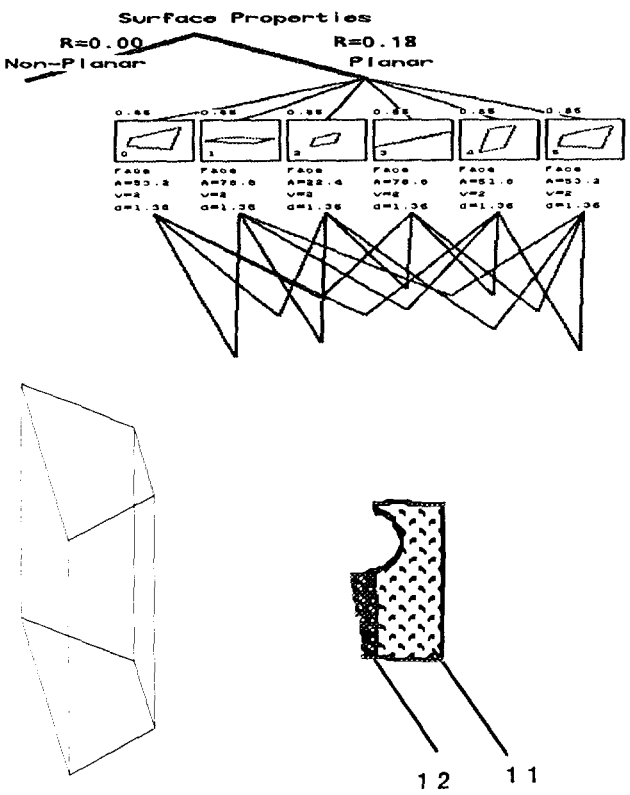

FIG. 11. Application of the recognition tree (at the top) to scene 2, the model is at the bottom left, and the identified surfaces are displayed at the bottom right.

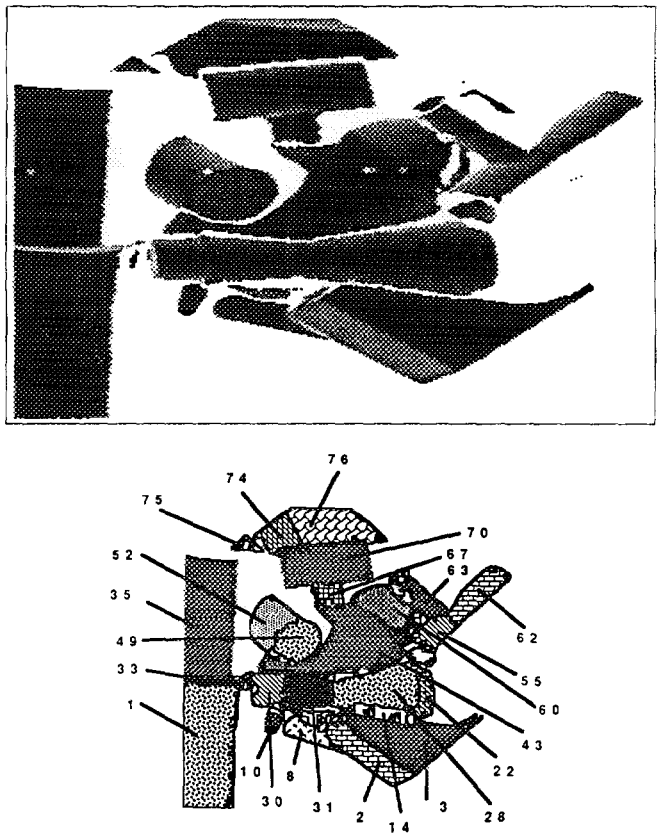

FIG. 12. Scene 3. The input image on the top, the segmented output on the bottom, and the calculated attributes are presented in Table 9.

does not exist in Fig. 10 (surface number 49 belongs to a bolt which does not exist in the model database; surfaces 39 and 26 form model e of Fig. 4). In cases where the segmentation results were incorrect or the approximation of the attributes did not fall within the expected error thresholds the recognition failed. For instance, in Fig. 8 model d of Fig. 4 was incorrectly identified as surfaces 47 and 30 instead of surface 11 (surfaces 47 and 30 belong to model p) (see Tables 7, 8, and 9).

The scene in Fig. 8 contains several pipes and polyhedrons. Two of the pipes have been oversegmented, and the third pipe (surfaces 10, 16, and 22) is oversegmented and partially occluded by one of the polyhedrons. In Fig. 9, the recognition tree for the desired model is displayed.

TABLE 7
The Calculated Attributes for Surfaces of the Scene in Fig. 8

Label	Classification	Surface area	Representative vector
1	Planar	92.63	$-0.07i - 1.00k$
2	Convex	57.79	$-0.07i - 1.00k$
4	Planar	23.28	$-0.23i - 0.46j - 0.66k$
5	Planar	40.78	$33i - 0.37j - 0.86k$
10	Convex	13.55	$15i - 0.95j + 0.10k$
11	Convex	26.31	$33i - 0.37j - 0.86k$
12	Convex	33.48	$53i - 0.76j + 0.04k$
16	Convex	6.25	$17i - 0.98j + 0.15k$
22	Convex	67.22	$92i + 0.39j + 0.01k$
30	Convex	11.63	$14i + 0.99j + 0.03k$
34	Planar	20.19	$01i + 0.08j - 1.00k$
39	Planar	26.15	$06i + 0.07j - 1.00k$
47	Convex	5.52	$98i - 0.19j + 0.05k$
57	Planar	14.26	$-0.29i - 0.65j - 0.70k$

TABLE 8
The Calculated Attributes for Surfaces of the Scene in Fig. 10

Label	Classification	Surface area	Representative vector
1	Convex	58.94	$0.17i + 0.98j$
2	Convex	103.19	$0.17i + 0.99j$
3	Planar	85.22	$0.38i + 0.53j + 0.76k$
8	Planar	63.03	$-0.34i - 0.51j - 0.79k$
11	Planar	22.20	$-0.01i + 0.05j - 1.00k$
12	Planar	12.66	$0.96i - 0.27k$
15	Convex	3.98	$-0.01i + 0.06j - 1.00k$
18	Convex	33.32	$0.17i + 0.98j + 0.09k$
28	Planar	16.24	$-0.01j - 1.00k$
39	Concave	7.81	$-0.01j - 1.00k$
41	Convex	19.53	$0.14i + 0.99j + 0.02k$
49	Convex	47.66	$0.65j - 0.76k$
50	Convex	9.49	$-0.01j - 1.00k$
55	Planar	86.34	$-0.08i + 0.41j - 0.91k$

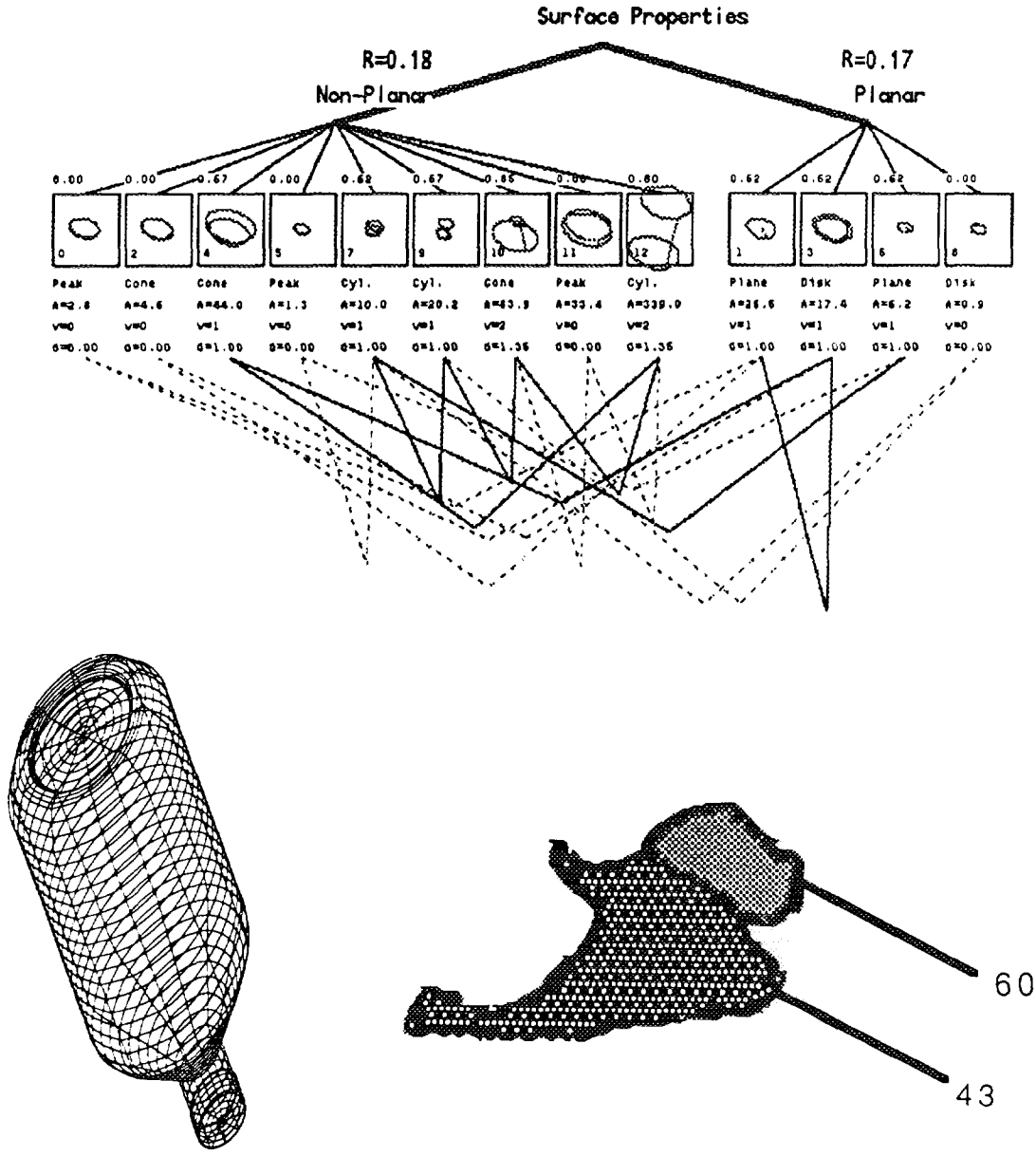

FIG. 13. Application of the recognition tree (at the top) to scene 3, the model is at the bottom left, and the identified surfaces are displayed at the bottom right.

The strategy first issues a filter (algorithm step 2) to locate model surface 0. (The two model surfaces have equal weight; however, model surface 0 has a larger area.) Five sets of segmented surfaces (A = {30,47}, B = {12}, C = {2,3}, D = {11}, and F = {10,16,22}) are activated; sets G = {57,34,39} and H = {4,5,1} are not activated since the planar branch of the tree has zero weight (R = 0), and sets G and H contain only planar patches. All active sets pass the surface area and surface type unary filters issued by the recognition tree. Using the only neighbor of model surface 0, similar unary filters are issued for model surface 1 (algorithm step 4), and similarly all five remaining active sets pass the unary filters. The strategy next issues binary filters (algorithm step 5). The ratio of radii between the model surfaces 0 and 1 are examined, eliminating sets B and D from further consideration. The next binary filter examines the angle between the representative vectors of model surfaces 0 and 1. Only set F passes the filter—surfaces 10 and 22 and surfaces 16 and 22. The last binary filter issued by the recognition strategy examines the connectivity of the surfaces in the model versus the segmented surface patches. The recognition

TABLE 9
The Calculated Attributes for Surfaces of the Scene in Fig. 12

Label	Classification	Surface area	Representative vector
1	Convex	201.98	0.16**i** + 0.96**j** − 0.21**k**
2	Planar	57.06	−0.24**i** + 0.73**j** − 0.64**k**
3	Planar	89.18	0.10**i** − 1.00**k**
8	Convex	28.40	0.04**i** − 0.94**k**
10	Convex	8.99	0.20**i** + 0.01**j** − 0.94**k**
11	Convex	15.47	0.10**i** + 0.01**j** − 0.98**k**
14	Planar	9.17	−0.09**i** + 0.86**j** − 0.31**k**
22	Convex	12.72	0.96**i** − 0.07**j** − 0.26**k**
28	Convex	37.59	0.94**i** + 0.27**j** − 0.25**k**
30	Convex	21.70	0.96**i** + 0.02**j** − 0.22**k**
31	Convex	34.47	0.96**i** + 0.12**j** − 0.27**k**
33	Concave	5.83	0.31**i** + 0.93**j** − 0.18**k**
35	Convex	194.69	−0.07**i** + 0.96**j** − 0.26**k**
36	Planar	3.00	−0.07**i** − 0.92**j** − 0.38**k**
43	Convex	59.01	0.16**i** + 0.95**j** − 0.28**k**
49	Planar	23.26	0.20**i** + 0.68**j** + 0.71**k**
51	Concave	14.13	0.25**i** + 0.61**j** − 0.71**k**
52	Convex	54.17	0.23**i** + 0.61**j** + 0.75**k**
55	Convex	30.40	0.03**i** + 0.92**j** − 0.39**k**
60	Convex	23.49	0.28**i** + 0.94**j** − 0.21**k**
62	Convex	31.48	0.45**i** + 0.82**j** − 0.36**k**
63	Convex	33.95	−0.68**i** + 0.61**j** − 0.41**k**
67	Convex	9.87	−0.41**i** − 0.56**j** − 0.72**k**
70	Convex	81.00	0.23**i** + 0.61**j** − 0.75**k**
74	Planar	39.95	−0.27**i** + 0.26**j** − 0.93**k**
75	Planar	13.02	0.23**i** + 0.08**j** − 0.97**k**
76	Planar	74.92	0.23**i** + 0.08**j** − 0.97**k**

strategy ends since there are no more neighbors left for surface model 0 (exit algorithm loop 3), and no more features are present (exit algorithm loop 1). In the final match, segmented surfaces 22 and 16 are matched to model surfaces 0 and 1, respectively. Surface 10 is not matched; however, it belongs to set F and is identified as part of the desired model. In future implementations, feedback to the segmentation module could be used to merge surfaces 10 and 16 and further refine the recognition system. The execution time of the strategy for this image is less than 2 s. All tests were performed in the order of seconds; the longest test lasted 5 s (Fig. 11).

Figure 11 shows the recognition tree for a polyhedron. The weight assigned to the nonplanar branch is zero since all detected surfaces are planar. This is advantageous during recognition strategy generation. During recognition, all segmented surfaces in the scene that are not classified as planar are efficiently recognized as noncandidates and not considered any further.

Figure 12 shows a scene of numerous overlapping objects. Figure 13 shows the recognition tree for the desired model containing several virtual links. For instance, model surface 11, classified as a peak (see Fig. 3), is connected to both surfaces 10 and 12. As Table 5 shows, $\nu_{\text{peak}} = \Psi(\text{thickness})$, and for this particular surface, $\nu_{\text{peak}} = 0$. Therefore, using Table 4 and Eq. (4), the weight assigned to this surface is zero, and surfaces 10 and 12 are considered immediate neighbors. Similar links have been formed between other surfaces, such as surfaces 7 and 9 (bypassing surface 8). During recognition strategy generation, the bypassed surfaces do not need to be located by the system, which further enhances the performance of the matching stage. Without the use of virtual links, the system would have unnecessarily searched for several additional surfaces in the scene. In Fig. 13, the desired motor oil bottle has been located; however, segmented surface 11 is not recognized as part of the successful set of surface patches since it has been separated by the shampoo bottle. Future work on this system could consider merging sets of surface patches.

7. CONCLUSION

In this paper we have addressed the problem of is CAD-based object recognition. Given a CAD model, the corresponding object in the scene is located. In general, the scenes contain several overlapping objects in arbitrary orientation and arrangement. A laser scanner is used to collect 3D data from the scene. Once segmented into a set of surface patches, various surface attributes are calculated. These attributes are then used to match to the given CAD model in the matching stage. In an earlier, off-line process, the model's surfaces have been automatically derived using the IGES description of the CAD model, and various surface attributes and the relationships among the surfaces have been inferred. Using the derived description, a recognition strategy has been compiled once and stored. At recognition time, the strategy is then used to guide the search. The desired object in the scene is identified by matching the surface patches derived from the given CAD model to the segmented surface patches in the scene. The strategy dictates which model feature to locate first, followed by its neighbors, and the second feature and its neighbors, and so on. The generated strategy is guided by several factors, such as feature detectability and the relationships of the features. Furthermore, in cases where the model features may not be detectable due to their geometry or to the effects of low level processing techniques, the recognition strategy is used to disregard the feature at the matching stage, enhancing the performance of the system. The strategy is not sensitive to partial occlusion of objects and oversegmentation of surface patches.

The object recognition system presented here significantly differs from previous systems in several respects. First, unlike many previous object recognition systems, the CAD system used to design the models is not an experimental one; rather, a commercial 3D CAD system has been used to design the models. Second, IGES has been used as an interface to the CAD system to infer the

geometric information necessary for object recognition. Using CADKEY and IGES decreases the dependency of the vision system on any particular CAD modeler and increases its applicability. Third, the model description, derived automatically from IGES, is used to systematically derive a matching strategy from a geometric model. By using the recognition strategy, not all the possible matching combinations of sensory features and model features are necessarily considered, increasing the efficiency of the system. And, by "precompiling" a recognition strategy, the vision system may perform more efficiently at run-time since less time is spent on model analysis in task execution. Finally, the matching strategy is not significantly dependent on moment-based and boundary-based features, and unlike many previous approaches to object recognition, our system does not impose an unconditional one-to-one matching of sensory features and model features. Thus, the recognition system is not sensitive to the partial occlusion of objects, and the oversegmentation of surface patches is easily tolerated. The effectiveness of the system is demonstrated on numerous examples.

ACKNOWLEDGMENTS

We thank CADKEY Inc. for providing the CAD package and Mr. Y. Chang for his comments on this paper. This research is supported in part by Army Research Office under Contract DAAL-03-91-G-0050.

REFERENCES

1. F. Arman, *CAD-Based Object Recognition In 3-D Range Images Using Pre-Compiled Recognition Strategy Programs*, Ph.D. thesis, Department of Electrical and Computer Engineering, University of Texas at Austin, 1992.
2. F. Arman and J. K. Aggarwal, Object recognition in dense range images using a CAD system as a model base, in *IEEE Conference on Robotics and Automation, Cincinnati, OH, May 13–18, 1990*, pp. 1858–1863.
3. F. Arman and J. K. Aggarwal, Automatic generation of recognition strategies using CAD models, in *IEEE Workshop on Directions in Automated CAD-Based Vision, Maui, Hawaii, Jun 2–3, 1991*, pp. 124–133.
4. F. Arman and J. K. Aggarwal, Model-based object recognition in dense range images—A review, in *ACM Comput. Surveys*, March, 1993.
5. P. J. Besl and R. C. Jain, Three-dimensional object recognition, *ACM Comput. Surveys* **17**(1), 1985, 75–145.
6. P. J. Besl and R. C. Jain, Invariant surface characteristics for 3D object recognition in range images, *Comput. Vision Graphics Image Process.* **33**(1), 1986, 33–80.
7. B. Bhanu, Representation and shape matching of 3-D objects, *IEEE Trans. Pattern Anal. Mach. Intell.* **6**, 1984, 340–351.
8. B. Bhanu and C. C. Ho, Geometric design based 3-D models for machine vision, in *8th International Conference on Pattern Recognition, Paris, France, Oct. 27–31, 1986*, pp. 107–110.
9. B. Bhanu and C. C. Ho, 3-D modeling for computer vision, *Pattern Recognit. Lett.* **5**, 1987, 349–356.
10. B. Bhanu and C. C. Ho, CAD-based 3-D object representation for robot vision, *COMPUTER*, 1987, 19–35. **Aug.**
11. T. J. Fan, *Describing and Recognizing 3-D Objects Using Surface Properties*, Springer-Verlag, New York, 1990.
12. T. J. Fan, G. Medioni, and R. Nevatia, Segmented descriptions of 3-D surfaces, *IEEE Int. J. Rob. Autom.* 3(6), 1987, 527–538.
13. T. J. Fan, G. Medioni, and R. Nevatia, Recognizing 3-D objects using surface descriptions, *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(11), 1989, 1140–1157.
14. R. B. Fisher, Determining back-facing curved model surfaces by analysis at the boundary, in *Third International Conference on Computer Vision, Osaka, Japan, Dec. 4–7, 1990*, pp. 296–303.
15. P. J. Flynn and A. K. Jain, On reliable curvature estimation, in *Proceedings, Computer Vision and Pattern Recognition, San Diego, CA, Jun 4–8, 1989*, pp. 110–116.
16. P. J. Flynn and A. K. Jain, Bonzai: 3-D object recognition using constrained search, in *Third International Conference on Computer Vision, Osaka, Japan, Dec. 4–7, 1990*, pp. 263–267.
17. P. J. Flynn and A. K. Jain, 3-D object recognition using invariant feature indexing of interpretation tables, in *IEEE Workshop on Directions in Automated CAD-Based Vision, 1991*.
18. C. Goad, Special purpose automatic programming for 3-D model-based vision, in *Proceedings, DARPA Image Understanding Workshop, Cambridge, MA, Apr. 6–8, 1983*, pp. 94–104.
19. W. E. L. Grimson, On the recognition of parameterized objects, in *4th International Symposium on Robotics Research, Santa Cruz, CA., Aug. 1987*.
20. W. E. L. Grimson, On the recognition of curved objects, *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(6), 1989, 632–642.
21. W. E. L. Grimson and D. P. Huttenlocker, On the sensitivity of geometric hashing, in *Proceedings, Third International Conference on Computer Vision, Osaka, Japan, Dec. 4–7, 1990*, pp. 334–338.
22. W. E. L. Grimson and T. Lozeno-Perez, Localizing overlapping parts by searching the interpretation tree, *IEEE Trans. Pattern Anal. Mach. Intell.* **9**(4), 1987, 469–482.
23. C. Hansen and T. C. Henderson, Towards automatic generation of recognition strategies, in *Proceedings, International Conference on Computer Vision, Tampa, FL, Dec. 5–8, 1988*, pp. 275–279.
24. C. Hansen and T. C. Henderson, CAGD-based computer vision, *IEEE Trans. Pattern Anal. Mach. Intell.* **11**(11), 1989, 1181–1193.
25. K. Ikeuchi, Generating an interpretation tree from a CAD model for 3-D-object recognition in bin-picking tasks, *Int. J. Computer Vision* **1**, 1987, 145–165.
26. K. Ikeuchi, Pre-compiling a geometrical model into an interpretation tree for object recognition in bin-picking tasks, in *DARPA Image Understanding Workshop, Los Angeles, CA, Feb. 23–25, 1987*, pp. 321–339.
27. A. K. Jain and R. Hoffman, Evidence-based recognition of 3-D objects, *IEEE Trans. Pattern Anal. Mach. Intell.* **10**(6), 1988, 783–802.
28. A. C. Kak, A. J. Vayada, R. L. Cormwell, W. Y. Kim, and C. H. Chen, Knowledge-based robotics, in *Proceedings, IEEE International Conference on Robotics and Automation, 1987*, pp. 637–644.
29. T. Kanade, P. Balakamar, J. C. Robert, R. Hoffman, and K. Ikeuchi, Vantage: A framebased geometric modeler with explicit symbolic representation of 3-D and 2-D information, in *Proceedings, International Symposium and Exposition on Robots, Sydney, Australia, Nov. 6–8, 1988*, pp. 1405–1420.
30. S. B. Kang and K. Ikeuchi, Determining 3-D object pose using the complex extended gaussian image, in *Proceedings, Computer Vision and Pattern Recognition, Maui, Hawaii, Jun 3–6, 1991*, pp. 580–585.

31. W. Y. Kim and A. C. Kak, 3-D object recognition using bipartite matching embedded in discrete relaxation, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(3), 1991, 224–251.

32. F. Kimura and M. Hosaka, Program package geomap, in *Proceedings, Geometric Model Project Meeting, 1978.*

33. K. Koshikawa and Y. Shirai, A 3-D modeler for vision research, in *Proceedings, International Conference on Advanced Robotics, 1985*, pp. 185–190.

34. Y. Lamdan and H. J. Wolfson, Geometric hashing: A general and efficient model-based recognition scheme, in *Proceedings, Third International Conference on Computer Vision, 1988*, pp. 238–249.

35. B. Sabata, F. Arman, and J. K. Aggarwal, Segmentation of 3-D range images using pyramidal data structures, *Comput. Vision Graphics Image Process: Image Understanding* **57**(3), 1993, 373–387.

36. Technical Arts Corporation, *Technical Arts 100X Users Manual and Application Programming Guide.*

37. U.S. Department of Commerce, *Initial Graphics Exchange Specification, Version 4.0*, National Bureau of Standards, Washington, DC, 1988.

38. B. C. Vemuri and J. K. Aggarwal, 3-D model construction from multiple views using range and intensity data, in *Proceedings, Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, June 22–26, 1986*, pp. 435–437.

39. H. B. Voelcker, A. A. G. Requicha, E. Hartquist, W. Fisher, W. Hunt, G. Armstrong, T. Check, R. Moote, and J. McSweeny, The PADL-1.0/2 system for defining and displaying solid objects. *ACM Comput. Graphics* **12**(3), 1978.