CAGD-Based Computer Vision

Chuck Hansen and Tom Henderson[1]

Department of Computer Science
The University of Utah
Salt Lake City, Utah 84112

Abstract

This paper explores the connection between Computer Aided Geometric Design and computer vision. A method for the automatic generation of recognition strategies based on the geometric properties of shape has been devised and implemented. This uses a novel technique developed for quantifying the following properties of features which compose models used in computer vision: robustness, completeness, consistency, cost, and uniqueness. By utilizing this information, the automatic synthesis of a specialized recognition scheme, called a Strategy Tree, is accomplished. Strategy Trees describe, in a systematic and robust manner, the search process used for recognition and localization of particular objects in the given scene. They consist of selected features which satisfy system constraints and Corroborating Evidence Subtrees which are used in the formation of hypotheses. Verification techniques, used to substantiate or refute these hypotheses, are explored. Experiments utilizing 3-D data are presented.

1. Introduction

We describe a systematic approach for both the generation of representations and recognition strategies based on Computer Aided Geometric Design (CAGD) models. Figure 1 shows such an integrated system which is composed of several components: a CAGD system, a milling system, a recognition system and a manipulation system. Recent work by Ho has focused on the generation of computer vision models directly from the CAGD model[1, 2]. In this paper, the automatic generation of recognition strategies based on the CAGD model is studied.

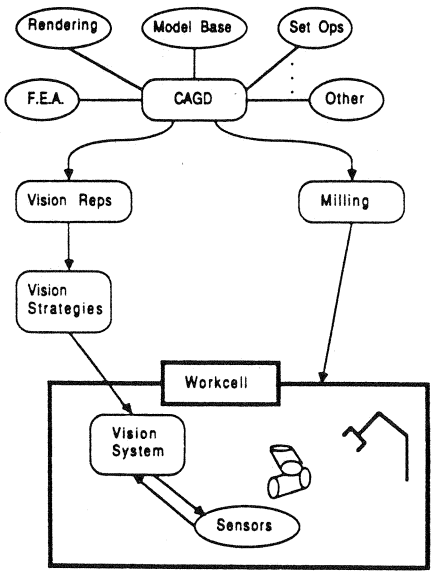


Figure 1. Integrated Automation Environment

The work described here investigates the use of geometric knowledge in constructing *strategy trees*. These trees provide a robust mechanism for recognition and localization of three dimensional objects (occluded as well as non-occluded) in typical manufacturing scenes. The run time matching of 3-D models to a scene can be expensive. If the search technique is optimized, cost can be decreased, thereby improving run time performance. One way to accomplish such optimization is by the off-line examination and evaluation of the 3-D model.

1.1. Related Work

One of the first researchers to study the automatic synthesis of general recognition strategies was Goad[3]. His work differs from that described here in that he obtained 3-D interpretations of 2-D intensity images rather than 3-D sensor data. The only features used were straight edges from intensity images and the search trees were generated from a template and ordered by hand rather than automatically. His system didn't consider partial occlusion. However, this was a major contribution since it was one of the first attempts to automate the generation of recognition schemes.

Another influential project was the 3DPO system by Bolles and Horaud[4]. This work is the 3-D generalization of the Local Feature Focus method[5]. Their system annotates a CAD model producing what is called the extended CAD model. From this model, feature analysis is performed to determine unique features from which to base hypotheses. The focus feature in their system is the dihedral arc. When the recognition system finds a dihedral arc, it looks for nearby features which are used to discriminate between model arcs with similar attributes. From these, an object's pose is hypothesized and subsequently verified. The work here work closely parallels the 3DPO system. However, focus features were hand chosen in 3DPO as were the local features used for discrimination.

Recently, Ikeuchi has explored the use of *interpretation trees* for representation of recognition strategies[6]. His system uses the concept of visible faces to generate generic representative views, called aspects. From this set of aspects, an interpretation tree is formed which discriminates among the different aspects. There doesn't appear to be any algorithmic approach for the application of the rules to discriminate between the aspects. The branching on the tree seems to be a function of the particular aspects chosen rather than being based on the geometric information in the model.

The system developed in this paper incorporates ideas from all of the systems described above. However, the system isn't dependent on a certain class of features but rather can be extended to include many classes of features. The system also performs automatic selection of features based on a set of constraints: feature filters. These features are used to form a *strategy tree* which provides a scheme for hypothesis formation, corroborating evidence gathering and object verification.

---

[1]This work was supported in part by NSF Grants MCS-8221750, DCR-8506393, and DMC-8502115

Our main goal is the automatic synthesis of recognition system specifications for CAD-based 3-Dimensional computer vision[7]. Given a CAD model of an object, a specific, tailor-made system to recognize and locate the object is synthesized. To attain this goal, the following problems have been solved:

1. **Geometric Knowledge Representation**,
2. **Automatic Feature Selection**, and
3. **Strategy Tree Synthesis**.

2. Geometric Knowledge Representation

The use of geometric data is central to a strong recognition paradigm. The Alpha_1 B-spline model[8] allows the modeling of freeform sculptured surfaces. To obtain the geometric features of interest for 3-D recognition, techniques for the transformation to a computer vision representation have been developed.

In the experimental system developed here, a modified winged-edge model[9] is used as the interface between CAD and vision, where relationships between features are explicit in the model. It is extended for inclusion of non-planar surfaces. In addition to special mechanisms for matching, access to the geometric knowledge of the object is required for the automatic generation of strategy trees. From this modified winged-edge description, an index on feature attributes can be generated which can quickly and efficiently access the geometric knowledge contained in the model.

3. Automatic Feature Selection

Several kinds of knowledge are required for 3-D feature selection. Geometric knowledge permits the selection of a complete and consistent set of features, while the sensor knowledge provides information on the robustness and reliability with which such features can be extracted. On the other hand, domain specific information about the task can be used to select feature extraction algorithms based on their complexity, robustness, etc.

The part to be recognized or manipulated is examined off-line for significant geometric features which can be reliably detected and which constrain the object's pose as much as possible. Moreover, such a set of features must *cover* the object from any possible viewing angle. In solving the feature selection problem, a technique is given for synthesizing recognition systems. This produces much more efficient, robust, reliable and comprehensible systems.

The feature selection process can be viewed as a set of *filters* applied to the complete original set of features of an object. Filters select and rank features; order of application is important. Conceptually, the filters remove features from the input, in order of application, which do not meet the filter's criteria. The goal here is to automate and optimize this filtering process. The filters select features based on the following qualities:

- **rare** - histogram the features; rare features are useful for quickly identifying the object; these features make good root nodes in a search tree.
- **robust** - measure of how well the features can be detected; error and reliability.
- **inexpensive** - measure of complexity (space and time) for computing feature.
- **complete** - does set of features cover all possible views of the object.
- **consistent** - how completely does feature characterize object pose; (i.e., how many degrees of freedom are unresolved); how well does the feature differentiate between objects; measure of likelihood of correctly identifying the object.

There are two types of feature robustness a system can quantify: the robustness of a feature itself and the robustness of the extraction techniques which are applied to obtain the feature. Furthermore, features should be dependable with respect to artifacts in the scene.

Three dimensional models define the entire object, yet, during scene analysis only a single view is available, or possibly multiple views, but not a complete view. How then, can the model be matched with the sensed data from the scene? One solution is the use of aspect graphs. An aspect graph is a representation of an object's topology; thus it captures all viewpoints of an object[10]. The *aspect* is the topological appearance of the object from a particular viewpoint. Slight changes in the viewpoint change the size of features, edges and faces, but do not cause them to appear or disappear. When a slight change in viewpoint causes a feature to appear or disappear, an *event* takes place. An aspect graph, or visual potential graph, is formed by representing *aspects* as nodes and *events* between aspects as paths between corresponding nodes. Several researchers have developed algorithms for the construction of aspect graphs, however, the size of the graphs poses computation limitations to their use[11, 12].

We use a discrete approximation by placing a tessellated sphere around the model, where each of the polygons represents a different viewpoint. Tessellation cells which contain the same features are merged into the same aspect. When no more tessels can be merged, the minimal aspect set for the model/sensor pair is reached.

Although features may fulfill the requirements of the above filters for a specific workcell and task configuration, they may not discriminate between views of the object or between different objects. A feature set is considered consistent if it possesses the necessary geometric information to distinguish between aspects. Symmetric objects pose problems for this type filter since multiple aspects appear similar to the system. The consistency filter forces the set of features to be strong enough to form a hypothesis.

When used in combination, these filters provide the mechanism with which to build a strategy tree. The task requirements may be such that the result of these filters is the null set of features. (Alternatively, the features can be ranked; see below.) This can be dependent on the order in which the filters are applied to the complete feature set. For example, if the filter for rare features determines that a 1/4 inch dihedral edge is the *best* feature and is applied prior to the robustness filter, that dihedral might not be accepted by the robustness filter since it is so small. Thus, the set of features would be null after the application of the robustness filter. Whereas, if the robustness filter is applied first, it wouldn't accept such features and when the rare filter is applied to the features accepted by the robustness filter, it would determine a different set of features as being *best*. The order of application is to be determined by knowledge of both the task to be accomplished and experience. The order was specified manually in the experiments described here.

4. Strategy Tree Synthesis

Once a robust, complete and consistent set of features has been selected, a search strategy is automatically generated. Such a strategy takes into account the strongest features and how their presence in a scene constrains the remaining search. The features and the corresponding detection algorithms are welded, as optimally as possible, into a search process for object identification and pose determination. The automatic synthesis of search strategies is a great step forward toward the goal of automated manufacturing. Generation of strategies is constrained, not only by the feature selection process, but by the actual task to be accomplished. Thus, strategies for a specific task might not be as strong when applied to a different task; strategies are task specific.

Another benefit of the tree structure is the inherent parallelism of trees. This occurs whenever there is a branch; thus, trees with greater breadth will, in general, have higher inherent parallelism. The sequentiality of trees refers to the depth of paths in the tree.

Strategy trees are shallow trees with many branches in the first two levels. Thus, there is a great deal of inherent parallelism in these trees.

The matching strategy consists of two phases: the hypothesis generation phase and the hypothesis verification phase. This recognition technique is known as hypothesize and verify. The hypothesis generation phase is controlled by the strategy tree and the verification phase substantiates or refutes the hypotheses generated from the strategy tree.

4.1. Description of Strategy Trees

A strategy tree consists of three major parts:

1. The Root - Which represents the object to be recognized.
2. Level 1 Features - Which are the strongest set of view independent features chosen for their ability to permit rapid identification of the object and its pose.
3. Corroborating Evidence Subtrees, CES - Whose purpose is twofold: they direct the search for corroborating evidence that supports the hypothesis of the level 1 features and they direct the search for geometric information to completely determine the pose prior to hypothesis generation.

4.2. Construction of Strategy Trees

A method is now needed for extracting the features of interest from the aspects. The level 1 nodes of the strategy tree are built from these features. Recall, that an aspect is a feature accumulator which forms a topologically equivalent set of features from multiple viewpoints. The Aspect Coverage Algorithm, shown in Figure 2, is used to form level 1 nodes by extracting the best, unique features from the aspects.

Algorithm Define A_i to be the set of all features contained in the ith aspect, where $0 < i \leq$ number-of-aspects. Define the operation, $-$, to denote set difference. Define, f, to be a level 1 node containing a set of unique features, possibly a singleton set, which permit rapid identification of the object and its pose.

For each A_i,
$D = \bigcap D_{ij}$ where $D_{ij} = A_i - A_j$ $(i < j)$
if $D \neq \emptyset$, then
chose f from **D**
if $D = \emptyset$ and no $D_{ij} = \emptyset$, then
select f to be the union of 1 element from each D_{ij}
if $D_{ij} = \emptyset$ for some j, then
$A_i \subset A_j$ so do nothing

Figure 2. The Aspect Coverage Algorithm to Choose Level 1

When **D** not the empty set, it means there is at least one feature which is contained in all the aspects. Thus, that feature is used as a level 1 node. In the case where **D** is null but all the $\mathbf{D_{ij}}$s are not empty, there is a combination of features which uniquely spans the aspects. Thus, a set of features for the level 1 node is used. In the last case, where the $\mathbf{D_{ij}}$ is null for some j, then **D** will also be null. Additionally, it is known that the aspect, $\mathbf{A_i}$ is completely contained in aspect $\mathbf{A_j}$. $\mathbf{A_i}$ must be a subset of $\mathbf{A_j}$ because the set difference is null and if the two aspects, $\mathbf{A_i}$ and $\mathbf{A_j}$, contained the exact same elements, they would have been merged. Since $\mathbf{A_i}$ is contained in $\mathbf{A_j}$, a level 1 node is not created at this point. Rather, this aspect will be covered by the level 1 node generated from aspect $\mathbf{A_j}$.

Once the level 1 nodes are built, it is necessary to generate the CES, Corroborating Evidence Subtrees. The CESs simply substantiate that a hypothesis should be generated based on a feature matching a level 1 node. Sufficient evidence must be found that a correct hypothesis is being made before a hypothesis for the verification phase to validate is generated.

Occlusion becomes a factor during the determination of the CES strategy. Since dihedral edges and arcs provide the most consistent information, they are used for level 1 nodes more often than regions or curved surfaces. Edges and arcs are composed of a starting point, an ending point, and the connecting edge or arc. When forming a strategy to handle occlusion for these features, both ends of the feature must be considered since it can't be known *a priori* which end is occluded. Generally, four cases are considered when forming the subtrees for local feature corroboration: (1) detected feature is not occluded, (2) one end of detected feature is occluded, (3) other end of detected feature is occluded, or (4) both ends of detected feature are occluded. For some features, such as faces or regions of constant curvature, there is no concept of direction; hence, the end conditions check can be replaced with adjacency information.

There are several rules which are implemented to control the construction of the CES level. These rules are feature dependent and are expandable should other classes of features be included in the system (e.g., *generalized cylinders*).

A CES is generated for every feature in the model which has similar attributes as the level 1 node. For example, suppose the level 1 node is a dihedral edge of included angle 30° and a dihedral edge in the scene is detected with an included angle close to 30°. A CES is generated for all 30° angles in the model. In other words, an attempt is made to determine which 30° dihedral was detected. The use of corroborating evidence focuses the search strategy by pruning unattractive paths at an early stage of the search.

4.3. Usage of Strategy Trees

The strategy tree *guides* the search through possible solutions. When a level 1 node is matched in the strategy tree and it is supported by the Corroborating Evidence Subtrees, then a hypothesis is generated. The hypothesis is passed to an object verifier which determines whether the hypothesis is valid within some confidence level.

In the above method, occlusion must be detected in the range data. Three simple cases suffice to determine whether occlusion is present or not. These tests are performed at the boundary of the detected features (i.e., dihedral edge - endpoints, surface/face - bounding edges).

Two forms of verification have been examined: structural and pixel correlation. Structural verification refers to verifying spatial relations among the features which should be present in the scene. This is similar to relational graph matching in 2-D. Pixel correlation refers to the verification technique of matching predicted depth, pixel by pixel, in a generated image and the sensed *image*. This corresponds to template matching in 2-D. Either of these methods provides for verification. This follows the hypothesis verification techniques used by others[5, 4, 13]. One of three states is assigned to the match of the hypothesized feature or pixel with the observed feature or pixel: *positive evidence*, *neutral evidence*, *negative evidence*. If these measures are accumulated for the predicted range image or structural features, the hypothesis can be quantified and accepted or rejected accordingly. This quantification provides a measure of confidence in the hypothesis.

5. Experiments and Discussion

The concepts which have been outlined above have been implemented in an experimental system. This section describes the sensing and computational environment. The synthesis of strategy trees is demonstrated with an example polyhedron. The equipment used for the experiments consisted of a Technical Arts 100A White

Scanner, DEC VAX class processors and an HP Bobcat. The images used in the experiments are part of the the Utah Range Database which was compiled for standardization of research on range images for the research community[14]. Feature computation was coded on a VAX 750 in C. The automatic generation of strategy trees and the matcher were coded on an HP Bobcat in HP Common Lisp.

5.1. Geometric Design

A polyhedron, called poly_1, was designed using the Alpha_1 design system. Of course, this simple polyhedron doesn't exploit the freeform power of Alpha_1 but suffices as an example of how the system functions. Starting with a primitive object, a parallelpiped, planes of intersection are defined with which to remove portions of the primitive. For poly_1, two portions are removed, and this is accomplished with the set difference of two planes and the primitive.

The construction of the hierarchical winged-edge model from the CAD model is quite simple. Form an object consisting of faces which consist of edges which consist of vertices. Figure 3 shows the edge numbering of the model. Table 1 lists the dihedral edges for poly_1. Table 2 lists the faces for poly_1. These are used by the feature selection process as well as in the generation of strategy trees. Note the grouping of the edges in Table 1 denoted by the horizontal lines. Due to noise in the data and error in the feature extraction methods, the system can't discriminate on angle value alone. Thus, dihedrals are grouped together if they are within 5 degrees of each other.

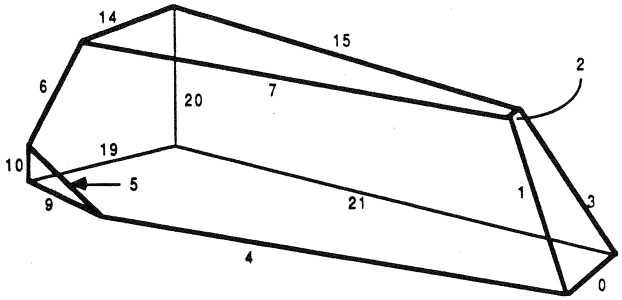


Figure 3. Wireframe Drawing for Poly_1

5.2. Feature Selection

The next step in the process is the evaluation of features. The *filters* are applied to the complete set of features. For the rare filter, a feature histogram is used to determine which features don't occur often in the model. Table 3 shows the histogram for the angle of all dihedral edges.

Robustness must be determined with respect to both the sensor and the suite of algorithms used. Through experimentation, it has been determined, for the sensor configuration used here, that an edge length under 1.0 inch can't be reliably detected. Similarly, if a face is below a certain size, its surface area can't be reliably detected, nor can the pointwise normals or the dihedral edges which form the face.

Dihedral edges were selected as the most consistent feature since they solve 5 DOFs. For this reason, the consistent filter ranks dihedral edges as the best level 1 feature.

The ability of the strategy tree to provide a path for recognition given an arbitrary object orientation is assured through the use of the aspect generation. Thus, the complete filter must be sure that at least one feature from every aspect (there are 26 aspects for poly_1) is included as a level 1 node. It is desirable to use features which are visible from the greatest number of different viewpoints.

edge	*angle*	length	*adjacent faces*
4	45.8	5.99	1 6
0	42.6	1.4	0 6
6	76.09	1.52	1 4
7	134.19	5.84	1 3
1	132.48	2.45	0 1
5	132.5	1.53	1 2
2	137.33	0.28	0 3
21	90	7.1	5 6
15	90	5.5	3 5
19	90	3.41	4 6
14	90	2.25	3 4
3	90	2.18	0 5
20	90	1.48	4 5
9	90	1.45	2 6
10	90	0.5	2 4

Table 1. Edge Attributes in Poly_1

face	*area*	normal	*poly type*
0	1.8225	-0.678 0.000 0.735	convex
1	13.9725	-0.240 0.676 0.697	convex
2	0.3625	0.000 1.000 0.000	convex
3	6.9438	0.000 0.000 1.000	convex
4	4.4643	1.000 0.000 0.000	convex
5	9.2925	0.000 -1.000 0.000	convex
6	18.5328	0.000 0.000 -1.000	convex

Table 2. Face Attributes in Poly_1

angle in degrees							
0-35	*35-55*	*55-65*	*65-82.5*	*82.5-100*	*100-125*	*125-145*	*145-360*
0	2	0	1	8	0	4	0

Table 3. Histogram of Dihedral Edges

Feature cost has not been incorporated at this time. It is clear that algorithmic cost could be included via Logical Sensors and this is an area of future research.

5.3. Strategy Tree Synthesis

Since strategy tree synthesis is automated, it is desirable to minimize the possibility of the null feature set and non-optimal level 1 nodes. This is accomplished by *ranking* the features with the filters. Thus, each filter produces a ranked list of the current feature set. As the strategy tree is built, the application of filters now means to choose the feature with the highest rank from that set.

The order in which the filters are applied was determined through experimentation. It has been found that if the complete filter is applied first, the desired coverage is assured. From this filter, a set of aspects is produced which contains visible features. Level 1 features are selected for the strategy tree such that all aspects are represented by a level 1 node. However, one feature might be visible from multiple aspects. Using the histogram, form a set of the features which are contained in the greatest number of aspects (highest histogram value), possibly a singleton set. From this set of features, use the rare filter to determine which of these features are unique. From the ranked set of unique features, use the robustness filter to rank the robustness of each of these features. Select the feature which is most robust. If this feature is robust enough, then use it as a level 1 node. If it isn't robust, repeat the algorithm for the next lowest histogram value. When a level 1 node is generated, remove, from the set of aspects, all the aspects which contain this feature. Recompute the histogram with the remaining aspects and

repeat. Either a set of level 1 nodes has been generated which spans the entire set of aspects or there are aspects remaining which contain only non-robust features. In the latter case, a *weaker* level 1 node must be formed for each of these aspects. This level 1 node will contain a feature which is not the most consistent type of feature. In this case, rather than having a dihedral edge as a level 1 node, the *back up* strategy is to match a face. At this point, the CES can be built.

One corroborating evidence subtree is generated for each dihedral edge which has attributes similar to the level 1 node. For example, for the level 1 node, edge 7, a CES must be formed for each of the edges in the 125-145 range. The reason for this is that when a 135° edge is located it should match one of these edges, but which one isn't known until corroborating evidence is gathered.

The next branch in each CES is determined by looking at the ends of the dihedral edge to determine if they are occluded. Recall that occlusion is determined by the end type of a particular edge. Shadow is assumed to be occluded, jump edge depends on whether it is an occluded jump or a non-occluded jump edge. All others are non-occluded.

In the non-occluded case, use the rules described above for the type feature which forms the particular level 1 node. In the example, most level 1 features are dihedral edges so the dihedral rules are used. The rules are applied in the following order:

1. Attempt to find a dihedral edge close to the endpoint of the current edge. If found, use this to quickly form a hypothesis.
2. Attempt to find the local 2-D corners. If found, these can help determine which hypothesis should be formed. For example, if a 135° edge is located, the adjacent 2-D corner can help to determine which, if any, of the 125-145 edges have been located.
3. Use the areas of the adjacent faces and relations between them to generate a hypothesis.

Figure 4 shows part of the strategy tree for poly_1. The edges are represented by their edge number in the model. Note that there is a CES for each dihedral edge which is similar to the level 1 node. These are derived from Table 2. For level 1 node 7, edges 1, 5, and 2 all have similar dihedral angles. Thus, there is a CES for each of these edges as well as edge 7. Note that the same CES can appear under multiple level 1 nodes. When matching, the rules on attribute similarity are used to invoke these CESs. Figure 5 shows the Corroborating Evidence Subtree for the dihedral edge 7. Note that there are 4 possible branches shown for clarity. The non-occlusion branch is composed of an OR of the partial occlusion cases. Thus, during run-time, the results of the partial occlusion are used by the non-occlusion branch.

5.4. Recognition

Now that the off-line procedure is completed, the usage of strategy trees can be demonstrated with an example of matching. A range image is obtained and low-level 3-D feature extraction performed on that data. From this data, the pointwise intrinsic features are computed for the object: surface normals and surface curvature. Since this is a polyhedral object, the planar face finder is used to develop a surface representation. Two dihedral edges are located using the dihedral edge finder. These edges correspond to edge 7 and edge 1 in the model.

Now the strategy tree shown in Figure 4 can be used. The level 1 features in the strategy tree are the dihedral edges: 7, 19, 14, 3, 4, 1, 0, 21, 6, 20, and 9. The dihedral edges located in the scene are shown in Table 4. (The corresponding model edges are included to help the reader.) The system has not matched the dihedral edges at this point.

The detected edge A is too short for reliability so it is not used. The detected edge D has an angle which doesn't match the model so it is not used in the matching process. Detected edges B and C both have angles with in the 130-140 range. These edges match 2 different level 1 nodes each: model edge 7 and model edge 1. The first determination in the strategy tree is to check for similarity. If a detected edge is larger than a model edge, the match fails. Detected edge C fails to match the level 1 node: edge 1, because the length is too long. Next the check for occlusion takes place. Detected edge B is non-occluded at both endpoints and detected edge C is occluded at one endpoint. Since it has been determined that detected edge B is a non-occluded edge, the attributes must be close to the model for a match to succeed. For this reason, edge B fails to match the level

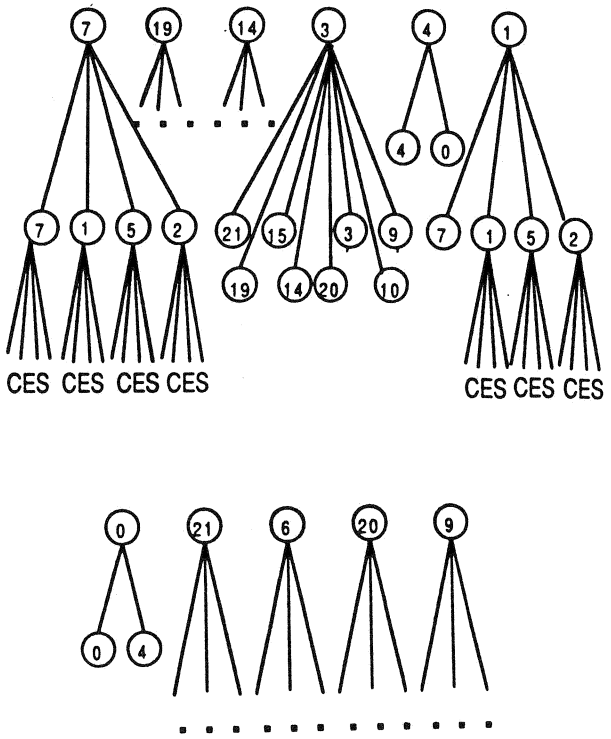


Figure 4. Strategy Tree for Poly_1

1 node: edge 7. Thus, only one Corroborating Evidence Subtree is invoked for each of the level 1 nodes which have been matched: edge 7 and edge 1.

The CES strategy first looks for an adjancent dihedral. In both cases, a dihedral is found. For the level 1 node: edge 7, the dihedral used as corroborative evidence is detected edge B. Whereas for the level 1 node: edge 1, the dihedral used as evidence is detected edge C. These two dihedrals are sufficent to solve all 6 DOFs and each of these forms a hypothesis at this point.

Since both the hypotheses are the same, the verifier only needs to check one. An image is formed with the hypothesized transform applied to the model and the perspective transform of the sensor applied to that result. For every pixel in the image, the z-depth is determined. Pixelwise evidence gathering can now be performed.

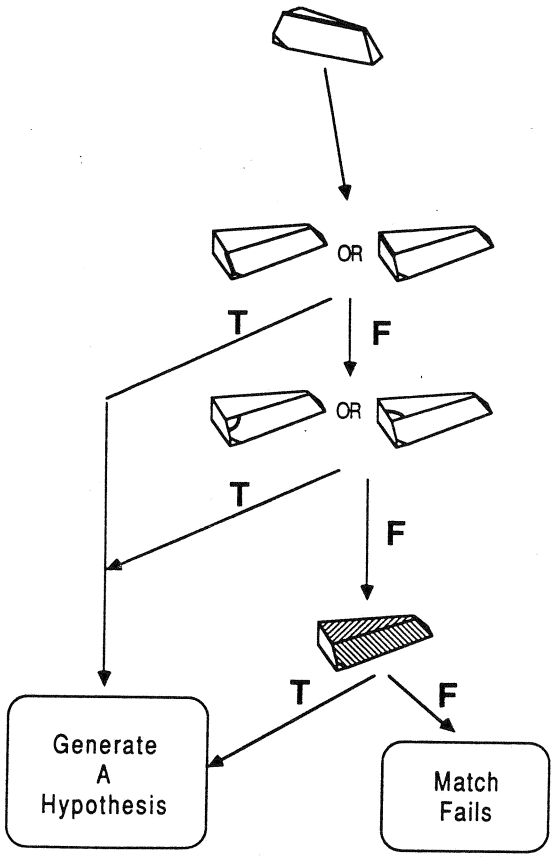


Figure 5. Corroborating Evidence Subtree for Edge 7

detected edge	*Edges Located*		*Model Edge*
edge name	*angle*	*length*	*edge number*
A	138.393°	0.2339	2
B	136.546°	2.4619	1
C	139.558°	5.7732	7
D	150.477°	0.8748	5

Table 4. Dihedral Edges Located in Scene

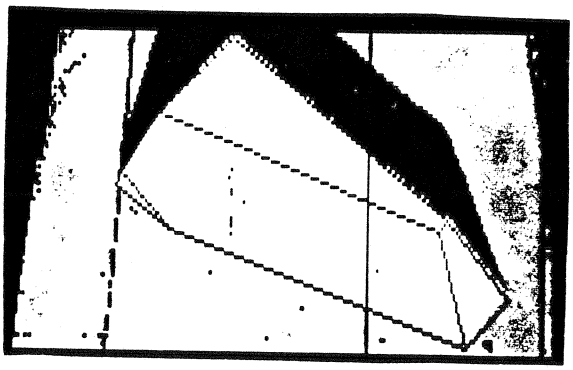


Figure 6. Model Overlayed on Sensed Image

The positive, negative and neutral evidence is combined to verify or refute the match. For the hypothesized transform, the hypothesis is correct in this case. This is shown in Figure 6.

Although the example is a polyhedral object, extensions to non-polyhedral objects are underway. If occlusion occurs in the scene, more CESs would be invoked to corroborate possible matches. The use of this approach with multiple objects merely requires running the recognizers in parallel.

6. Future Work

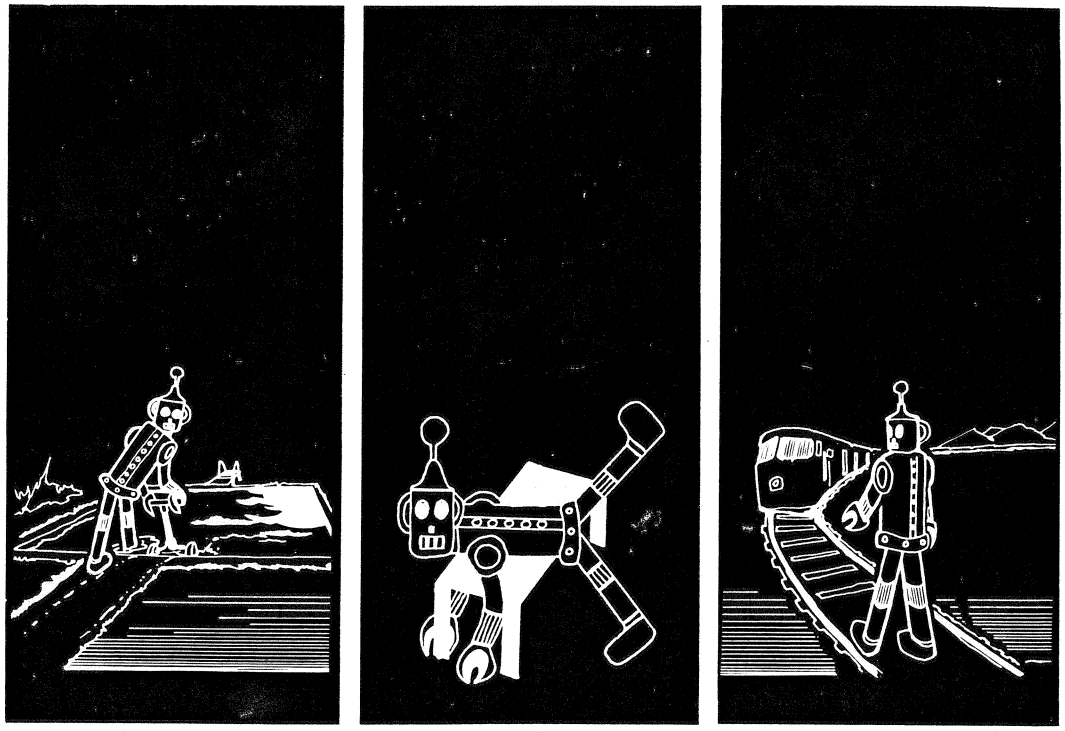
The application of these concepts to other representations, such as generalized cylinders, should be explored, as well as the use of diverse 3-D geometric features. Another problem is the use of a more general knowledge-based framework for the synthesis of recognition schemes.

References

1. Bhanu, B. and C.C. Ho, "CAGD-Based 3-D Object Representations for Computer Vision", *IEEE Computer*,Vol. 20No. 8August 1987, pp. 19-36.
2. Ho, C.C., "CAGD-Based 3-D Object Representations for Computer Vision", Master's thesis, University of Utah, June 1987.
3. Goad, C., "Special Purpose, Automatic Programming for 3D Model-Based Vision", *Proceedings of the DARPA Image Understanding Workshop*, DARPA, 1983, pp. 94-104.
4. Bolles, R.C. and P. Horaud, "3DPO: A Three-Dimensional Part Orientation System", *Robotics Research*,Vol. 5No. 3 1986, pp. 3-26.
5. Bolles, R.C. and R.A. Cain, "Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method", *Robotics Research*,Vol. 1No. 3 1982, pp. 57-82.
6. Ikeuchi, K., "Model-Based Interpretation of Range Imagery", *Proceedings of the DARPA Image Understanding Workshop*, DARPA, 1987, pp. 321-339.
7. Hansen, C.D., *CAGD-Based Computer Vision: The Automatic Generation of Recognition Strategies,* PhD dissertation, The University of Utah, July 1987.
8. Cohen, E., T. Lyche and R.F. Riesenfeld, "Discrete B-Splines and Subdivision Techniques in Computer Aided Geometric design and Computer Graphics", *Computer Graphics and Image Processing*,Vol. 14No. 2October 1980, pp. 87-111.
9. Baumgart, B.G., "Geometric Modeling for Computer Vision", AIM 249, Stanford, October 1974.
10. Koenderink, J.J. and A.J. Van Doorn, "The Singularities of the Visual Mapping", *Biological Cybernetics*,Vol. 24 1976, pp. 51-59.
11. Kent, E.W., M.O. Schneir and T.-H. Hong, "Building Representations from Fusions of Multiple Views", *Proceedings of the IEEE Conference on Robotics and Automation*, San Francisco, CA, April 1986, pp. 1634-1639.
12. Plantinga, H. and C. Dyer, "The Aspect Representation", Tech. report cstr-683, University of Wisconsin, 1987.
13. Knoll, T. and R. Jain, "Recognizing Partially Visible Objects using Feature Indexed Hypotheses", *J. of Robotics and Automation*,Vol. RA-2No. 1 1986, pp. 3-13.
14. Hansen, C. and Thomas C. Henderson, "The UTAH Range Database", Computer Science UUCS-86-113, University of Utah, April 1986.

PROCEEDINGS of the IEEE Computer Society

Workshop on Computer Vision



Computer Society Order Number 779
Library of Congress Number 87-83133
IEEE Catalog Number TH0210-5
ISBN 0-8186-0779-3
SAN 264-620X

November 30-December 2, 1987
Fontainebleau Hilton,
Miami Beach, Florida

THE COMPUTER SOCIETY OF THE IEEE

IEEE THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

COMPUTER SOCIETY PRESS