Scene Understanding by Rule Evaluation

Walter F. Bischof and Terry Caelli

Abstract—We consider how machine learning can be used to help solve the problem of identifying objects or structures composed of parts in complex scenes. We first discuss a conditional rule generation technique (CRG) that is designed to describe structures using part attributes and their relations. We then show how the resultant rules can be used for region labeling and examine constraint propagation techniques for improving rule-based object classification.

Index Terms—Conditional rule generation, machine learning, object recognition, scene understanding, visual learning.

1 Introduction

ALTHOUGH the literature abounds with techniques for the recognition of *isolated* 2D patterns and 3D objects, the problem of efficiently detecting and recognizing such structures in complex signals or scenes has not received as much attention, both in the computer vision and the machine learning literature. In computer vision, most recognition procedures, apart from those based on cross-correlation techniques, assume that patterns to be classified are isolated. Similarly, many machine learning techniques, from inductive logic programming [1] through decision trees [2] to neural networks, are typically unable to accomplish efficient recognition of structures embedded in complex scenes. Further, most approaches to visual pattern and object recognition do not adequately deal with the problem of recognition under distortions or from partial data, as is the case in 3D object recognition in the context on multiple, and possibly overlapping, objects.

For recognition problems involving the comparisons of patterns composed of many parts, the literature has typically focused on methods of reducing the run-time comparisons using different constraint satisfaction algorithms. The problem of matching parts and their relations is an example of the subgraph isomorphism problem, and solution algorithms typically involve tree search procedures where tree pruning is based on the comparison of model and image data parts and their relational attributes. Different models differ with respect to the methods used for generating attribute (feature) trees, the degree of precompilation of the search procedures from training examples, and the degree of generalization inherent in each system. The more common approaches to classification of isolated patterns and objects include geometric hashing [3], feature indexing [4], interpretation trees [5], and constraint propagation trees [6]. These approaches have focused almost exclusively on the selection and ordering of model attributes for the efficient interpretation of *isolated* objects, and have not addressed issues of generalization in an adequate way.

More recently, a number of authors have endeavored to use techniques from machine learning to increase the robustness and efficiency to these methods, i.e., to improve their ability to generalize from training or known object data, and to improve their efficiency in searching scene data. The former involves using standard

- *W.F. Bischof is with the Department of Psychology, University of Alberta, Edmonton, Alberta, T6G 2E9, Canada. E-mail: wfb@psych.ualberta.ca.*
- *T. Caelli is with the School of Computing, Curtin University of Technology, GPO Box U 1987, Perth, WA, Australia. E-mail: tmc@cs.curtain.edu.au.*

Manuscript received 7 July 1994; revised 22 Aug. 1997. Recommended for acceptance by H. Keshavan.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 105707.

generalization methods from machine learning, while the latter involves the development of methods for describing patterns or objects by minimum-length descriptions and the development of appropriate constraint propagation techniques. For example, evidence-based methods have been used recently in 3D object recognition [7], [8], where object model views and parts are used to automatically generate rules (attribute bounds) that provide evidence for different models. In such systems, generalizations to new views or to distortions are defined in terms of the rule attribute bounds and evidence weights. However, they typically have not encoded relational information and have not been applied to complex scenes. The aim of this paper is to address both of these issues from the viewpoint of automated rule generation and the application of rules to the interpretation of complex scenes.

2 Conditional Rule Generation: CRG

The idea of Conditional Rule Generation (CRG) is to generate classification rules that include structural pattern information to the extent that is required for classifying correctly a set of training patterns. CRG searches for the occurrence of unary and binary features states between connected pattern components and creates a tree of hierarchically organized rules for classifying new patterns. Generation of a rule tree proceeds in the following manner.

First, the unary features of all parts of all patterns are collected into a unary feature space U in which each point represents a single pattern part. The feature space U is partitioned into a number of clusters U_i. Some of these clusters may be unique with respect to class membership (e.g., U_3 in Fig. 1) and provide a classification rule: If a pattern contains a part p_r whose unary features $u(p_r)$ satisfy the bounds of a unique cluster U_i, then the pattern can be assigned a unique classification. The nonunique clusters contain parts from multiple-pattern classes and have to be analyzed further. For every part of a nonunique cluster (e.g., U_2 in Fig. 1), we collect the binary features of this part with all other parts in the pattern to form a (conditional) binary feature space UB_i. The binary feature space is clustered into a number of clusters UB_{ij}. Again, some clusters may be unique (e.g., cluster UB_{11} in Fig. 1) and provide a classification rule: If a pattern contains a part p_r whose unary features satisfy the bounds of cluster U_i, and there is an other part p_s, such that the binary features $b(p_r, p_s)$ of the pair $\langle p_r, p_s \rangle$ satisfy the bounds of a unique cluster UB_{ij}, then the pattern can be assigned a unique classification. For nonunique clusters, the unary features of the second part p_s are used to construct another unary feature space UBU_{ij} that is again clustered to produce clusters UBU_{ijk}. This expansion of the cluster tree continues at additional levels *UBUB*, *UBUBU*, … involving additional pattern parts until all clusters are completely resolved. Some clusters may, however, never be resolved. In this case, the cluster tree has to be refined by either reclustering one of the features spaces or by splitting one of the clusters.

Analysis of nonunique clusters can proceed by further expanding the cluster tree, analyzing unary and binary attributes of additional pattern parts. Alternatively, the derived clusters in the tree can be refined or broken into smaller, more discriminating feature bounds or rules as described below.

One successful approach to cluster-tree refinement involves entropy-based splitting procedures. Consider splitting the elements of an unresolved cluster C along a (unary or binary) feature dimension F. The elements of C are first sorted by their feature value $f(c)$, and then all possible cut points T midway between successive feature values in the sorted sequence are evaluated. For each cut point T, the elements of C are partitioned into two sets, $P_1 = \{c \mid f(c) \le T\}$ with n_1 elements and $P_2 = \{c \mid f(c) > T\}$ with n_2 elements. We define the normalized partition entropy $H_P(T)$ as $H_P(T) = (n_1 H(P_1) + n_2 H(P_2))/(n_1 + n_2)$. The cut point T_F that minimizes $H_P(T_F)$ is considered the best point for splitting cluster C along feature dimension F. The best split of cluster C is considered the one along the feature dimension F that minimizes H_P. For an unresolved leaf cluster C_L, one can split C_L or any cluster in the parent chain of C_L. Among these clusters, the one that minimizes H_P is considered the optimal point for refining the cluster tree.

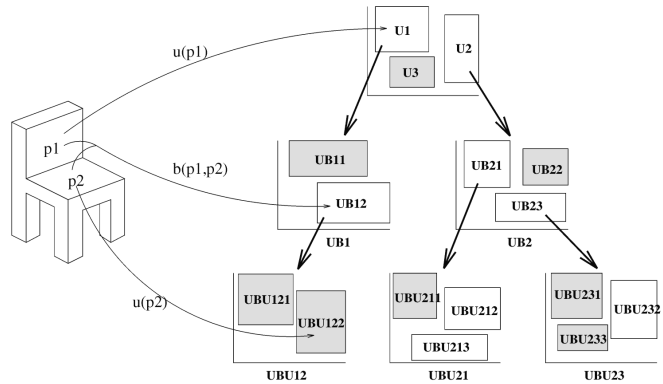

Fig. 1. Cluster tree generated by the Conditional Rule Generation procedure (CRG). The unresolved unary clusters (*U*1 and *U*2)—with element from more than one class—are expanded to the binary feature spaces *UB*1 and *UB*2, from where clustering and expansion continues until either all rules are resolved or the predetermined maximum rule length is reached, in which case rule splitting occurs.

A completely resolved cluster tree provides a set of deterministic rules for classification of patterns. Every cluster element in the cluster tree corresponds to a sequence $U_i - B_{ij} - U_j - B_{jk} - \ldots$ of unary and binary features associated with a noncyclic chain of pattern parts. CRG thus produces classification rules for (small) pattern fragments and their associated unary and binary features. For each feature space in the cluster tree, a standard decision tree [2] is produced. CRG thus produces a tree of decision trees that is indexed by sequences of pattern parts. The dynamic expansion of cluster trees constitutes a major advantage of CRG over decision trees: CRG can expand trees to the level optimized for a given data set whereas decision trees operate on fixed sets of features that have to be chosen a priori (see [9] for more details).

3 Scene Labeling and Recognition: SURE

CRG generates classification rules for (small) pattern fragments based on their unary and binary features. When the classification rules are applied to some pattern, one obtains one or more (classification) evidence vectors for each pattern fragment, and the evidence vectors have to be combined into a single evidence vector for the whole pattern. This is more or less straightforward for single (isolated) patterns, but difficulties arise in scenes composed of multiple patterns where it is unclear whether a sequence $p_i - p_j - \ldots - p_n$ of pattern parts belongs to the same pattern or whether it is "crossing the boundary" between different patterns. In the former case, the CRG rule can be expected to produce correct classifications, whereas in the latter case, classification may be arbitrary. This problem has been studied by Grimson [6], Lowe [10], and others in the context of model-based vision. Here, we discuss a solution in the context of a rule-based system that makes only

weak and general assumptions about the structure of scene and objects. Our solution is based on the analysis of the relationships within (intra) and between (inter) instantiated rules. The solution method, termed SURE (Scene Understanding using Rule Evaluation), is based on the *sequential* evaluation of constraints described in the following sections.

3.1 Initial Rule Evaluation

The first stage in SURE involves direct activation of the CRG rules in a parallel, iterative, deepening method. Starting from each scene part, all possible sequences of parts, termed *chains*, are generated and classified using the CRG rules. Expansion of each chain $S = <s_1, s_2, \ldots, s_n>$ terminates if one of the following conditions occurs:

1) the chain cannot be expanded without creating a cycle,
2) all CRG rules instantiated by S are completely resolved, or
3) the binary features $\vec{b}(s_n, s_{n+1})$ do not satisfy the features bounds of any CRG rule.

If a chain S cannot be expanded, the evidence vectors of all rules instantiated by S are averaged to obtain the evidence vector $\vec{E}(S)$ of the chain S. Further, the set S_p of all chains that start at p is used to obtain an initial evidence vector for part p:

$$\vec{E}(p) = \frac{1}{\#(S_p)} \sum_{S \in S_p} \vec{E}(S) \tag{1}$$

where $\#(S)$ denotes the cardinality of the set S. Classification of scene parts based on (1) has one major problem. Chains that are contained completely within a single "object" are likely to be classified correctly. However, chains that "cross" two or more objects are likely to be classified in a arbitrary way, and they thus distort classifications. To the extent that such "crossing" chains can be detected and eliminated, the part classification (1) can be improved.

In the following, we present four rules aimed at detecting and eliminating "crossing" chains. Two of the rules are deterministic constraint rules, and the other two are probabilistic compatibility rules. Further, two of the rules use dependencies within chains (intrachain rules), and the other two use dependencies between chains (interchain rules). We discuss now, in turn, each of the four rules.

3.2 Chain Permutation Constraint

Every CRG rule encodes a set of model chains

$$\{M_k = < m_{k1}, m_{k2}, \cdots, m_{kn} >, 1 \le k \le K\}$$

(see Section 2). When a chain $S = < s_1 s_2 \cdots s_n >$ instantiates such a rule, each image part s_i indexes a set of model parts $\mathcal{M}(s_i) = \{m_{ki}, 1 \le k \le K\}$. The chain permutation constraint is based on the assumption that rule instantiations are invariant to permutations, i.e., if two chains are permutations of each other (e.g., $S_1 = <A, B, C>$ and $S_2 = <B, A, C>$), their parts must index the same set of model parts, independent of instantiated rules. For example, the model parts corresponding to scene part A given chain S_1 must be the same as those corresponding to A given chain S_2, i.e., $\mathcal{M}(A \mid S_1) = \mathcal{M}(A \mid S_2)$. This constraint can be applied independent of whether the chains S_1 and S_2 are "crossing" or not. Further, the chain permutation constraint can be applied repeatedly over sets of chain permutations using standard constraint propagation.

3.3 Single Classification Constraint

The single classification constraint is based on the assumption that *at least one* chain among all chains starting at a scene part does not cross an object boundary and that at least one instantiated rule indexes the correct model parts. Given this, if there is any scene part that initiates a single chain S_i and this chain instantiates a single classification rule, then the model parts indexed by S_i can be used to constrain all chains that touch S_i. More formally, let $S_i = < s_{i1}, s_{i2}, \ldots, s_{in_i} >$ be a single classification chain. Then, for all chains $S_j = < s_{j1}, s_{j2}, \ldots, s_{jn_j} >$ that touch S_i, the common parts must index the same set of model parts. That is, for $s_{ik} \in S_i$, $s_{jl} \in S_j$, $s_{ik} \equiv s_{jl} \Rightarrow \mathcal{M}(s_{ik}) \equiv \mathcal{M}(s_{jl})$. With the single classification constraint, S_i constrains all chains S_j that *touch* S_i, but the reverse does not hold, since touching chains may be "crossing" and should therefore not be used to constrain other chains.

As with the chain permutation constraint, the single classification constraint can be propagated through the network of chains. Any chain that ends up with an empty set of model parts $\mathcal{M}(s_i)$ for one of its components s_i is considered inconsistent and eliminated. Further, the evidence vector of each chain is recomputed from the set of all indexed model parts (rather than from the set of all model parts of the instantiated rule as described in Section 2).

These two deterministic constraints are very powerful in terms of eliminating inconsistent (crossing) chains. Their usefulness breaks down, however, for cases where the assumptions formulated earlier are not met for a given training and test data set.

3.4 Interchain Compatibility Analysis

The interchain compatibility analysis is based on the following general idea: The less compatible the evidence vector of a chain S_i is with the evidence vectors of all chains that S_i touches, the more likely it is that S_i crosses an object boundary. In this case, S_i is given a low weight in the computation of (1). More formally, let $S_i = < s_{i1}, s_{i2}, \ldots, s_{in_i} >$ and $S_j = < s_{j1} s_{j2} \ldots s_{jn_j} >$ be touching chains, and let T_{ij} be the set of common parts, i.e., $T_{ij} = \{p \mid \exists k\, p = s_{ik} \text{ and } \exists l\, p = s_{jl}\}$ with $\#(T_{ij}) > 0$. The compatibility of S_i and S_j, $C(S_i, S_j)$ is defined as

$$C(S_i S_j) = \frac{1}{\#(T_{ij})} \sum_{p \in T_{ij}} \frac{\#\left(\mathcal{M}(p|S_i) \cap \mathcal{M}(p|S_j)\right)}{\#\left(\mathcal{M}(p|S_i) \cup \mathcal{M}(p|S_j)\right)} \tag{2}$$

The overall compatibility of a chain S_i is then defined with respect to the set S_T of chains that touch S_i, i.e., $S_T = \{S_j \mid \#(T_{ij}) > 0\}$:

$$w_{inter}(S_i) = \frac{1}{\#(S_T)} \sum_{S \in S_T} C(S_i, S) \tag{3}$$

Using the interchain compatibility, we can now modify the original averaging for the part evidence vectors in (1) to

$$\vec{E}(p) = \frac{\Sigma_{S \in S_p} w_{inter}(S) \vec{E}(S)}{\Sigma_{S \in S_p} w_{inter}(S)} \tag{4}$$

where S_p is defined as in (1).

3.5 Intrachain Compatibility Analysis

The last rule for detecting boundary-crossing chains is based on the following idea. If a chain $S_i = < s_{i1}, s_{i2}, \ldots, s_{in} >$ does not cross boundaries of objects then the evidence vectors $\vec{E}(s_{i1})$, $\vec{E}(s_{i2})$, $\ldots$, $\vec{E}(s_{in})$ computed by (4) are likely to be similar, and dissimilarity of the evidence vectors suggests that S_i may be a "crossing" chain. Similarity of any pair of evidence vectors can be measured by their dot product, and similarity of all intrachain

evidence vectors is captured by the following measure:[1]

$$w_{intra}(S) = \frac{1}{n(n-1)} \sum_{k=1}^{n} \sum_{\substack{l \neq k \\ l=1}}^{n} \vec{E}(s_{ik}) \cdot \vec{E}(s_{il}) \quad (5)$$

With the incorporation of the intrachain compatibility analysis, the part evidence vectors are computed using the following iterative (relaxation) scheme:

$$\vec{E}^{(t+1)}(p) = \Phi\left[\frac{1}{Z} \sum_{S \in S_p} w_{inter}(S) w_{inter}^{(t)}(S) \vec{E}(S) \right] \quad (6)$$

with the normalizing factor $Z = \sum_{S \in S_p} w_{inter}(S) w_{intra}^{(t)}(S)$ and the logistic function $\Phi(z) = [1 + \exp[-20\,(z - 0.5)]]^{-1}$. Iterative computation of (6) is required, since recomputation of $\vec{E}(p)$ affects the intrachain compatibility $w_{intra}(S)$. As indicated above, the four rules presented in this section are evaluated sequentially, and the final part classification is given by the iterative scheme (6).

4 Examples

Although many problems in 3D object recognition require the registration of depth information, there remain some that can be sufficiently represented by simple multiview images of each object. This is the case with the *blocks* example shown in Fig. 2. It consists of configurations of blocks that are learned in isolation (Fig. 2, top and middle row) and have to be recognized in complex arrangements (Fig. 2, bottom row). The training set consisted of five classes of block configurations, each with three training examples with one of each being shown in Fig. 2 (c1-c5). The complex scenes consisted of up to 20 blocks, two of which are shown in Fig. 2 (s1 and s2).

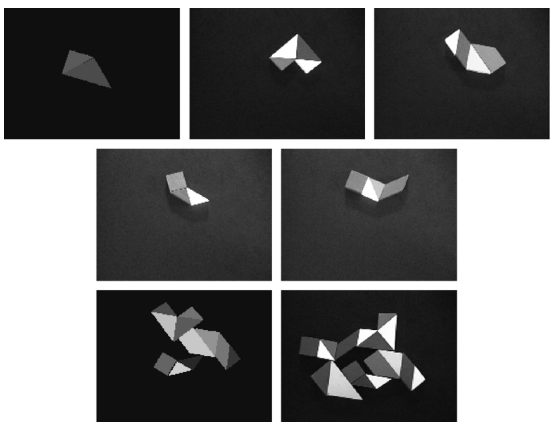

Fig. 2. (c1-c5): Example of one training pattern for each of the five classes of patterns in the blocks example. (s1-s2): Complex arrangements of patterns of learned training patterns.

Images of the training and test images were captured with a color camera and segmented using a form of K-means clustering on position (x, y) and color (r, g, b) attributes [11]. Small clusters were merged with larger neighbor clusters in order to eliminate spurious image regions. The following unary features were extracted for each image region: size (in pixels), compactness ($perimeter^2 / area$), and the normalized color signals $R/(R + G + B)$, $G/(R + G + B)$, and $B/(R + G + B)$. For pairs of image regions, the following binary features were computed: absolute distance of region centers, minimum distance between the regions, distance of region centers normalized by the sum of the region areas, and length of shared boundaries normalized by total boundary length.

For the training data, CRG analyzed 240 different chains and produced 25 rules: 11 *U*-rules, 3 *UB*-rules, and 11 *UBU*-rules. Classification performance was tested with the two scenes, s1 and s2, shown in Fig. 2. For scene s1, 16 out of 17 parts were classified correctly, and, for the 17 parts of scene s2, 15 were classified correctly and one part was not classified. Relative merits of the rules discussed in Section 3 are illustrated in Table 1, which shows number of chains, the number of correct classifications, and the average entropy of all classification vectors after the (sequential) application of the four rules. It is clear from these results that the rules proposed in Section 3 are capable of improving the classification of scene parts through elimination of crossing chains, and, hence, improve region and object classification in complex scenes.

TABLE 1
Number of Chains (NS), Number of Correct Classifications (NC), and Average Entropy of Classification Vectors (AE) After Application of Each Rule Described in Section 3, for the Two Complex Scenes Shown in Fig. 2

	Scene s1			Scene s2		
	NS	NC	AE	NS	NC	AE
after initial rule evaluation	81	13	0.26	93	11	0.33
after chain permutation constraint	55	15	0.26	66	13	0.32
after single classification constraint	32	16	0.10	36	15	0.21
after inter-chain compatibility analysis	32	16	0.10	36	15	0.18
after intra-chain compatibility analysis	32	16	0.0	36	15	0.0

The *blocks* example is essentially a 2D application even though the objects were 3D objects. The *objects* example, on the other hand, consists of full 3D range images. Cylindrical range images of six isolated objects and complex scenes were obtained with a CyberWare scanner where objects are placed on a rotating table and illuminated by a vertical laser beam. Range was determined by normal triangulation techniques (given camera and laser calibration) with an accuracy between 0.1–0.4 mm. From a Delauney triangulation of the sampled surface points, a description of the local surface geometry was obtained using the covariance method proposed in [12].

The object surfaces were segmented into regions using local covariance descriptors. Each region was described by the unary features avg(d_1) and avg(d_2), the average logarithm of the two eigenvalues of the local second-order covariance matrix. Pairs of regions were described by the following binary features: the difference avg(d_1) − avg(d_2), the product of the average convexity of the two regions, the distance between the region centers, and the ratio of the two region areas.

The training set of the *objects* example consisted of five objects, a bottle, a cup, a planter, a pot, and a spray, as shown in Fig. 3. The cylindrical range images of each object were segmented and unary, and binary features were extracted as described above, and CRG rules were generated as described in Section 2. The generated rules were applied to three different scenes containing multiple objects (see Fig. 4). The three scenes contained 25, 32, and 31 parts, respectively. Of these parts, 18, 23, and 28 parts were classified correctly.

5 Discussion

In the previous sections, we have presented CRG, a method for learning structural descriptions in the form of evidence rules on attribute bounds of ordered predicates. We have also introduced SURE, a method for constraining the application of classification rules in scenes composed of multiple objects.

CRG shares with standard decision trees [2] similar methods for the search and expansion of decision trees. CRG differs from these methods in several respects. First, it is designed to develop

1. We thank Dennis Moore for suggesting this measure.

rules for unique identification of classes with respect to their "structural," i.e., linked unary and binary feature representation, whereas decision trees use unstructured sets of attributes. Second, CRG expands cluster trees adaptively and optimized for a given dataset, whereas decision trees operate on fixed sets of attributes.

In the present paper, we have addressed two major issues. First, given that CRG represents structural descriptions in terms of sets of independent pattern chains, we have studied how interdependence of these chains can be analyzed. Second, and more pertinent to this paper, we have studied how these interdependencies can be used to group pattern parts or image regions into groups that are likely to be associated with a single object.

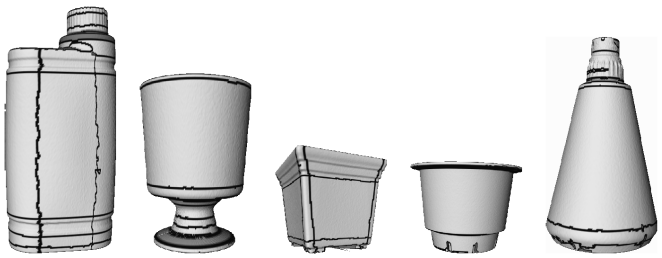

Fig. 3. Five objects used in the *objects* example. A full 360-degree cylindrical range image was obtained for each object. The black lines correspond to boundaries between segmented surface regions.

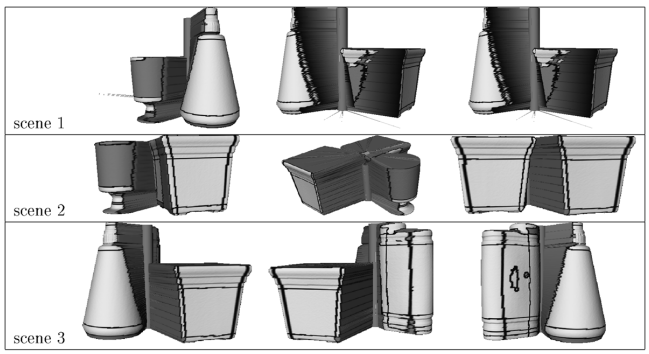

Fig. 4. Three different views of each of the three test scenes. Note that the different views are for illustration purposes only. A full 360-degree cylindrical range image was obtained for each scene.

Some work has been previously done on grouping of image features and parts within the context of model-based vision, where model *backprojections* can be used to prune false groupings or incorrect interpretations. Here, we have attempted to achieve the same within the framework of a rule-based interpretation system and by relying only on very general and weak assumptions about image and scene structure. We have achieved this by utilizing constraints within and between instantiated rules, using both deterministic and probabilistic constraints. The SURE method we have described can be used alone, as in the examples described here, or it can be used as an additional mechanism for hypothesis pruning in model-based vision systems.

ACKNOWLEDGMENTS

This project was funded by a grant from the Australian Research Committee. WFB was supported by grant OGP38251 from the Canadian Natural Sciences and Engineering Council.

REFERENCES

[1] S. Muggleton, *Inductive Logic Programming*. New York: Academic Press, 1992.

[2] J.R. Quinlan, *C4.5 Programs for Machine Learning*. San Mateo, Calif.: Morgan Kaufmann, 1993.

[3] F. Tsai, "Geometric Hashing With Line Features," *Pattern Recognition*, vol. 27, pp. 377–389, 1994.

[4] P. Flynn and A.K. Jain, "3D Object Recognition Using Invariant Feature Indexing of Interpretation Tables," *Computer Vision, Graphics, and Image Processing*, vol. 55, pp. 119–129, 1992.

[5] C. Hanson and T. Henderson, "Gagd-Based Computer Vision," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, pp. 1,181–1,193, 1989.

[6] W.E.L. Grimson, *Object Recognition by Computer*. Cambridge, Mass.: MIT Press, 1990.

[7] A.K. Jain and D. Hoffman, "Evidence-Based Recognition of Objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 783–802, 1988.

[8] T. Caelli and A. Dreier, "Variations on the Evidenced-Based Object Recognition Theme," *Pattern Recognition*, vol. 27, pp. 185–204, 1994.

[9] W.F. Bischof and T. Caelli, "Visual Learning of Patterns and Objects," *IEEE Trans. Systems, Man and Cybernetics*, in press.

[10] D.G. Lowe, *Perceptual Organization and Visual Recognition*. Boston: Kluwer Academic Publishers, 1985.

[11] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, N.J.: Prentice Hall, 1988.

[12] T. Caelli, E. Osman, and G. West, "On Learning to Recognize 3D Shape Defects," *Proc. Third Int'l Conf. Visual Form* (IWVF3), 1997.