INTEGRATING LOW-LEVEL FEATURES COMPUTATION WITH INDUCTIVE LEARNING TECHNIQUES FOR TEXTURE RECOGNITION

PETER W. PACHOWICZ
Center for Artificial Intelligence
George Mason University
4400 University Drive
Fairfax, VA 22030, USA
ppach@gmuvax2.gmu.edu

Received 7 November 1989
Revised 30 January 1990

This paper presents a method for applying inductive learning techniques to texture description and recognition. Local features of texture are computed by two well-known methods, Laws' masks and co-occurrence matrices. Then, a three-level generalization of local features is applied to create texture description rules. The first level generalization, the scaling interface, has been implemented to transform the numeric data of local texture features into their higher symbolic representation as numerical ranges. This scaling interface tests data consistency as well. The creation of description rules incorporating the inductive incremental learning algorithm is the second generalization step. The SG-TRUNC method of rule reduction is applied as the next hierarchical generalization level. This machine learning approach to texture description and recognition is compared with the classic pattern recognition methodology. The results from the recognition phase are presented from six classes of textures, characterized by smoothly changing illumination and/or texture resolution. The average recognition rate was 91% for the inductive learning approach, and all classes of textures were recognized. In comparison, the traditional k-NN pattern recognition method did not recognize one class of texture, and the average recognition rate was 83%. The proposed methodology smooths the recognition rates through the hierarchy of generalization levels, i.e. the next generalization step increases these rates for classes that were less easily recognized, and decreases these rates for classes that were more easily recognized.

Keywords: Inductive learning; Texture recognition; Texture description rules.

1. INTRODUCTION AND MOTIVATION

The aim of this work is to develop a method for applying inductive learning to low-level texture description and recognition. We present such a method and test its effectiveness applied to texture recognition as compared with a classic pattern recognition (PR) technique (where PR has been applied mainly to imagine the complexity of our textures, along with computation methods of features). An automatic scaling of quantitative data of texture features (e.g. responses to the local convolution masks) is used to transfer numeric data into symbolic intervals. Later, these symbolic values are processed by symbolic machine learning techniques. This scaling is the first generalization step, and it is limited in this work to the static performance, i.e. the numeric range of a texture feature is divided into a constant number of intervals and a single interval cannot be scaled on the next

International Journal of Pattern Recognition and Artificial Intelligence Vol. 4, No. 2 (1990) 147–165

(hierarchically lower) level of higher resolution. The dynamic approach to the management of scaling processes, incremental extension of the number of classes, spacing of teaching events, as well as the increase/decrease of the number of attributes of single events is studied separately in the second phase. Our approach to the integration of low-level numeric computation and high-level symbolic processing is an initial step in the creation of a vision system that will be capable of adapting to a dynamic environment and new vision tasks. Such a system will acquire knowledge from its environment, modifying its internal structure, and will improve its recognition performance over time.

Traditional adaptability and learning in engineering systems apply control engineering[1] and/or pattern recognition[2] techniques. These approaches work well using quantitative information. But typical processes of high-level vision are based on symbolic computation. The border line between numeric and symbolic computation has not been established. There are methods, which successfully apply the iconic approach alone to invariant recognition and data fusion.[3] However, the application of such methods to the object inspection task is difficult. For example, the inspection task uses object models or object variances that are acquired through explanation, technical diagrams and tolerances, rather than from presentation of various real objects during a teaching phase.

Our approach to robot vision incorporates machine learning (ML) methodologies applied to object location, recognition, inspection and scene understanding tasks, and it is based on the creation of system adaptability functions. The learning capabilities of the system use context dependence to adapt to the dynamic environment. In our approach, models used to represent data and system knowledge must possess a dynamic structure that can be modified by the system itself based on its own experience. Incorporating ML and cognitive processes such as induction, deduction, discovery and analogy, we plan to combine elements of visual recognition and visual imagery[4] within a unified system. Currently, we are investigating vision adaptation mechanisms that allow a system to recognize objects in new external situations. Such situations include 3-D rotation, changed resolution, quantitative and qualitative noise, light reflectivity and mutual positional relations between objects such as occlusion. The system has to perform the following vision tasks: object location, recognition and inspection. We assume that prior to the system work, it either possesses an initial model of objects, or acquires the optimal model from the environment, based on background meta-knowledge. The adaptive system must be able to manage its internal structure and data (dynamic memory) in order to increase/decrease the number of distinctive features, add new objects and modify process flow through the network.

A typical task that must be based on such an adaptive approach is texture-based object recognition for robot navigation in natural terrain. The difficulties of robust recognition of 3-D objects is caused by a great variability of images projected onto the sensory array. This variability is caused not only by the 3-D rotation of an object, but also by changes of resolution, positioning of light sources (e.g. light reflection, shadows), mutual positional relations with other objects at the scene (e.g. touching, occlusion), etc. In the traditional approach to object recognition, the system is associated with a set of characteristic object

features, which are used for recognizing particular objects. Such a set of features is constant[5] in that the perceptual system is unable to verify the usefulness of the object model. Thus, the classification is correct as far as the accuracy of the off-line created recognition model is concerned.

Recent progress in visual recognition of objects has generally been based on the extension of traditional visual recognition processes. Marr's[6] 2.5-D sketch theory introduces directional information to object shapes reconstructed from contour, color, texture, stereo, motion, etc. Later, this data is integrated into a perceptual system. Currently, this concept is being applied to the creation of the MIT Vision Machine.[7] The recognition of an object and its position, which is insensitive to 3-D rotation, has been proposed by Ikeuchi and Kanade.[8] In the teaching phase, the system creates internal multi-view object representations based on the CAD model. The recognition phase matches currently visible features of an object (where features can possess a 2.5-D description) with predicted features of the object's model. In these approaches, the recognition systems have been created without adaptive capabilities. One can say that these systems are closed and are unable to verify and modify the object model. These typical limitations of object recognition systems have been discussed by Bhanu[9] within the context of automated target recognition. As a consequence, a new approach has been suggested that integrates ML with target recognition in order to create an adaptable target recognition system.[10] This new approach integrates two ML techniques with a knowledge-based reasoning system. The first technique, explanation-based learning, provides the ability to build generalized descriptions of object classes. The second technique, conceptual clustering, allows the system to make decisions based on simple symbolic descriptions rather than numerical similarity measures. When created, the system will be capable of modifying its internal data concerning objects on the basis of its experience.

Our texture-based object recognition system is made dynamic by applying an inductive incremental learning methodology as a kernel of the system. This introductory work has applied inductive learning without the use of incremental mode to develop and test the scaling method, which is both an early generalization process and an interface for numeric and symbolic data integration. In Sec. 2 of this paper, we present this methodology, while in Sec. 3 we discuss the results. An improvement of the recognition rate by the application of a third level generalization is demonstrated in Sec. 4. Finally, in Sec. 5 we summarize our work.

2. METHODOLOGY

In this section we explain the methodology for applying ML techniques to texture description and recognition. We also compare ML and PR approaches. The general architecture, presented in Fig. 1, is separated into the learning phase and the recognition phase. We present subsections that elaborate the preparation of image data, detection of texture features, automatic conversion of numeric-to-symbolic data, description and recognition of texture, and evaluation of results.

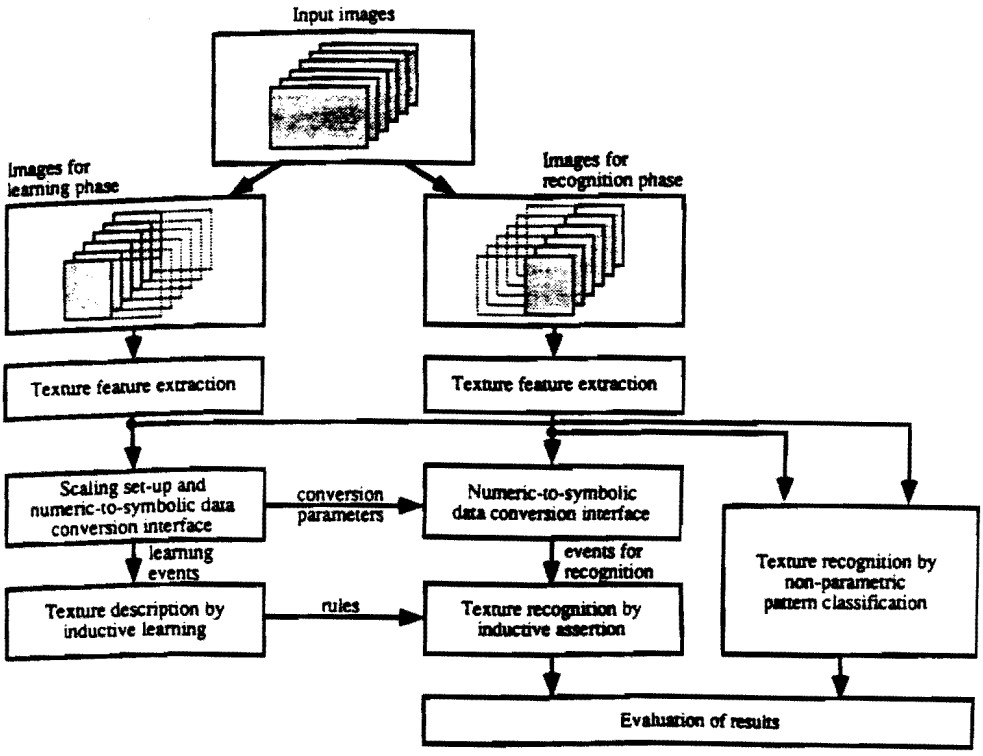


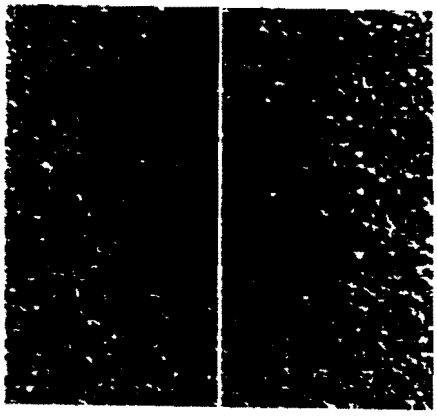
Fig. 1. General architecture of experimental phases.

2.1. Image Data Preparation

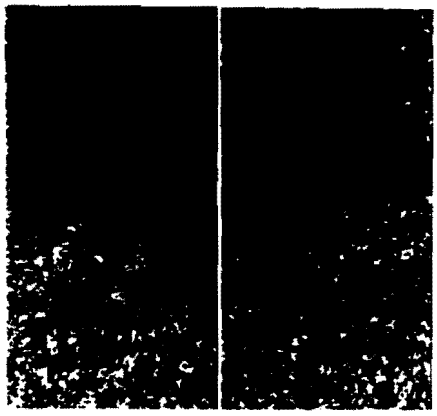
Six input images were obtained from a black-and-white camera, where each pixel of a 512 × 512 image was coded onto 256 gray levels. These images contain six classes of texture: pressed cork, lawn, woolen cloth, water, pigskin and fur. They were non-uniformly illuminated. Two of them (lawn and water) have significant smoothly changing texture resolution, caused by the screw direction of the image projection. Later, the intensities of all images were decreased to 64 gray levels. This limitation was imposed by the application of texture-feature detection by co-occurrence matrices (i.e. a large number of image gray levels causes the dimension of the co-occurrence matrix to be very large). These input images were used to create two files of images. In this way, the left-hand side of the input images (subimages of 512 × 256 pixels) were used in the learning phase, while the right-hand sides (subimages of 512 × 256 pixels) were used as data for the recognition phase. These images are shown in Fig. 2.

2.2. Texture-Feature Detection

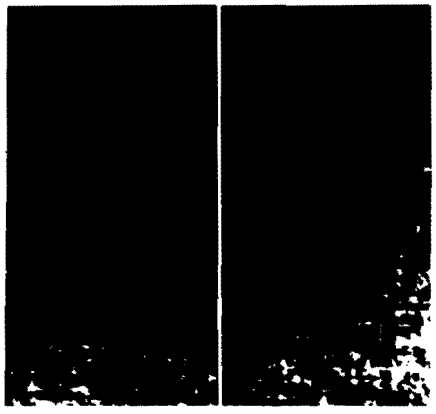
We applied the same methodology to the extraction of local texture features both for the ML and PR approaches to texture recognition. Two traditional methods, Laws' masks and co-occurrence matrices, were used for texture-feature extraction and sample data preparation.[12] The following steps of this data computation are presented in Fig. 3.



(a) pressed cork

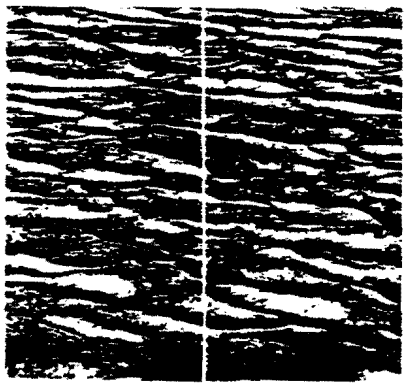


(b) lawn

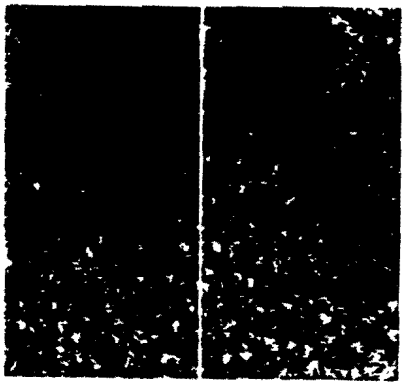


(c) woolen cloth

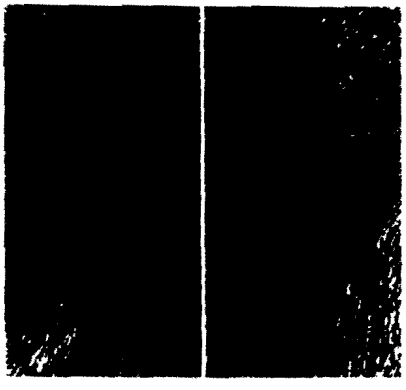
Fig. 2. Texture images separated into left-hand learning subimage and right-hand testing subimage, all printed in five gray levels.



(d) water



(e) pigskin



(f) fur

Fig. 2. Cont'd.

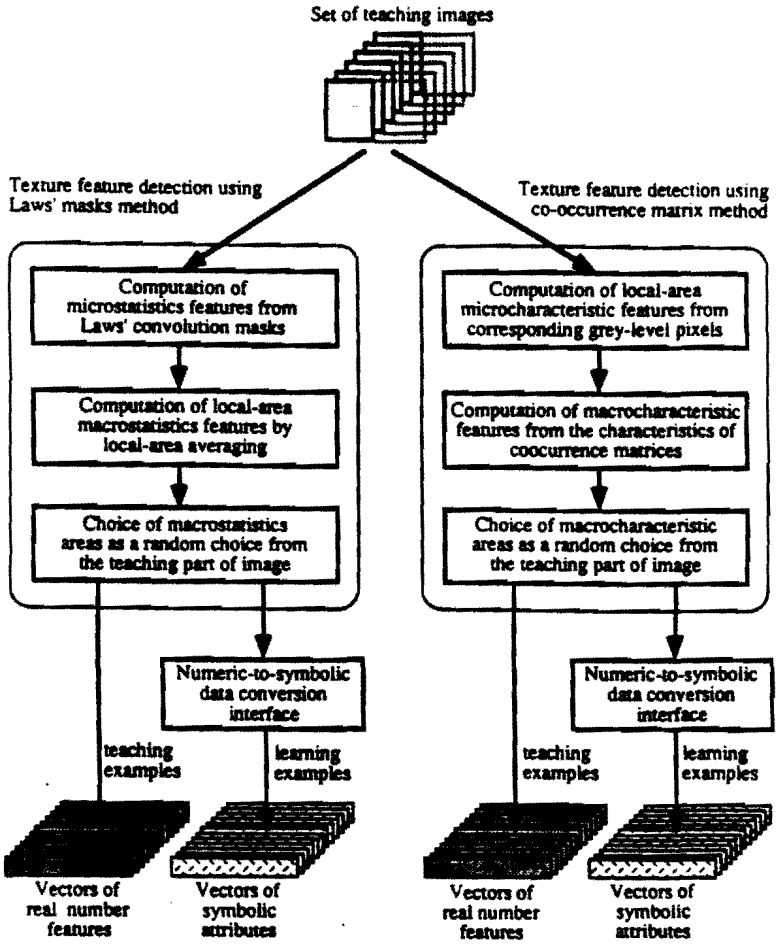


Fig. 3. Data computation processes precede texture description and recognition phases.

2.2.1. Local feature detection with the Laws' masks method

1. The computation of a single vector of microstatistical features was executed for a central pixel using one 3 × 3 and seven 5 × 5 convolution masks[13] shown in Fig. 4.
2. Based on the vectors of microstatistical features, we computed local-area macrostatistics over a larger window of 20 × 20 pixels, and a vector of macrostatistical properties was computed as the average absolute value of each pixel feature over the window.
3. The third step provides a random choice of macrostatistics over the teaching image of 512 × 256 pixels. A set of 200 random macrostatistical vectors was computed as learning samples for each class (image) of texture.

2.2.2. Local feature detection by the co-occurrence matrix method

1. A vector of four microcharacteristic features was extracted, in the form of gray values

of pixels located within a distance d, and direction β, from the central pixel (see Fig. 5), and associated with the gray value of the corresponding central pixel.

2. For each microcharacteristic feature and a class of texture, we computed local-area diagonal co-occurrence matrices over a larger window (20 × 20 pixels). Each co-occurrence matrix was then used to calculate both the angular second momentum (also called the matrix uniformity) and the contrast as a measure of the spread of values away from the main diagonal.[14] Finally, we obtained a vector of texture parameters, where each position within the vector represents local-area macrocharacteristics for a given distance d, and direction β.
3. This step of texture-feature detection using the co-occurrence matrix method provides a random choice of macrostatistic vectors over the teaching image (512 × 256 pixels), where a set of 200 teaching examples was selected for each class of texture.

1	−4	6	−4	1
−4	16	−24	16	−4
6	−24	36	−24	6
−4	16	−24	16	−4
1	−4	6	−4	1

4	−6	4
−6	8	−6
4	−6	4

−1	−4	−6	−4	−1
−2	−8	−12	−8	−2
0	0	0	0	0
2	8	12	8	2
1	4	6	4	1

−1	−2	0	2	1
−4	−8	0	8	4
−6	−12	0	12	6
−4	−8	0	8	4
−1	−2	0	2	1

−1	0	2	0	−1
−2	0	4	0	−2
0	0	0	0	0
2	0	−4	0	2
1	0	−2	0	1

−1	−2	0	2	1
0	0	0	0	0
2	4	0	−4	−2
0	0	0	0	0
−1	−2	0	2	1

−1	0	2	0	−1
−4	0	8	0	−4
−6	0	12	0	−6
−4	0	8	0	−4
−1	0	2	0	−1

−1	−4	−6	−4	−1
0	0	0	0	0
2	8	12	8	2
0	0	0	0	0
−1	−4	−6	−4	−1

Fig. 4. Laws' masks used to extract local microstatistical features of texture.

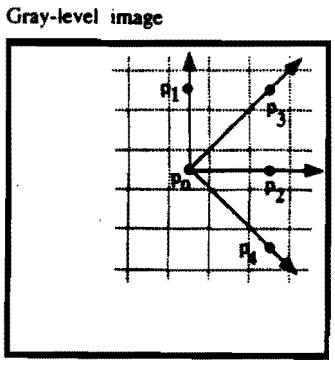


Fig. 5. Local microcharacteristic features for given direction and distance.

2.3. Interface of Numeric-to-Symbolic Data Conversion

Inductive learning, as applied to texture description and recognition, requires the creation of a special interface of numeric-to-symbolic data conversion. To fulfill this requirement, we converted numerical texture features into their symbolic intervals. As a consequence, the "static" conversion or scaling process is an early generalization of numerical examples from the feature space into their more general symbolic representation as a cubic cell of feature space intervals. The static character of the scaling assumes that we have a constant number of texture classes, as well as the attribute number of a sample event. The scaling was determined by an *a priori* given set of events, which cannot be changed or extended by adding data from the environment, such as that obtained during on-line system experience. On the other hand, future use of dynamic scaling assumes that a system will be able to extend the number of texture classes, to add new events that are characteristic of a single class, and to change the number of attributes or modify them.

Let us assume that for the static scaling, $V = \{V^1, \ldots, V^j, \ldots, V^k\}$ is a set of numerical events for a single class of texture, and $V \ni v = (v_1, \ldots, v_i, \ldots, v_m)$ is a learning event expressed as a vector of m numerical attributes. Then, for each ith attribute, we compute the following scaling parameters: $v\min_i$, $v\max_i$ and Δ_i, which are the minimum value, maximum value and data interval of each ith attribute respectively. These parameters were found for all elements of the set V:

$$v\min_i = \min_{j \in [1,k]} \{v_i \in V^j\} \qquad v\max_i = \max_{j \in [1,k]} \{v_i \ni V^j\} \tag{1}$$

$$\Delta_i = (v\max_i - v\min_i)/(n\text{int} - 1) \tag{2}$$

where

$n$int is the number of intervals accepted by an inductive learning algorithm.

We applied the 0-level conversion of numeric attributes to their symbolic representation and computed $u = (u_1, \ldots, u_i, \ldots, u_m)$ with the following formula:

$$\text{if } v_i \in [vmin_i + (n-1)^*\Delta_i;\ vmin_i + n^*\Delta_i) \quad \text{then } u_i = n \tag{3}$$

where n was equal to 50 in our experiments. Then, the system checked the consistency of the created data. For those events that were inconsistent for two or more classes, the system created an additional class of inconsistent events and the scaling process was repeated on the lower level as shown in Fig. 6. The application of lower-level rescaling is caused by the requirement that the texture classes be separated. Considering this task, the system predicted that the non-limited recursive rescaling would create a very complicated hierarchical structure, with high resolution, that makes it difficult to execute object recognition. Therefore, we set up a criterion for the scaling applied to the xth level of the hierarchy. It indicates that the scaling of the lower level must be applied if more than 5% of learning events for a given class are inconsistent. That is, they are removed from the given xth level of the scaling hierarchy and placed into the additionally created class of inconsistent events. For our textures, the scaling processes were not applied to the lower levels, because the criterion was not satisfied.

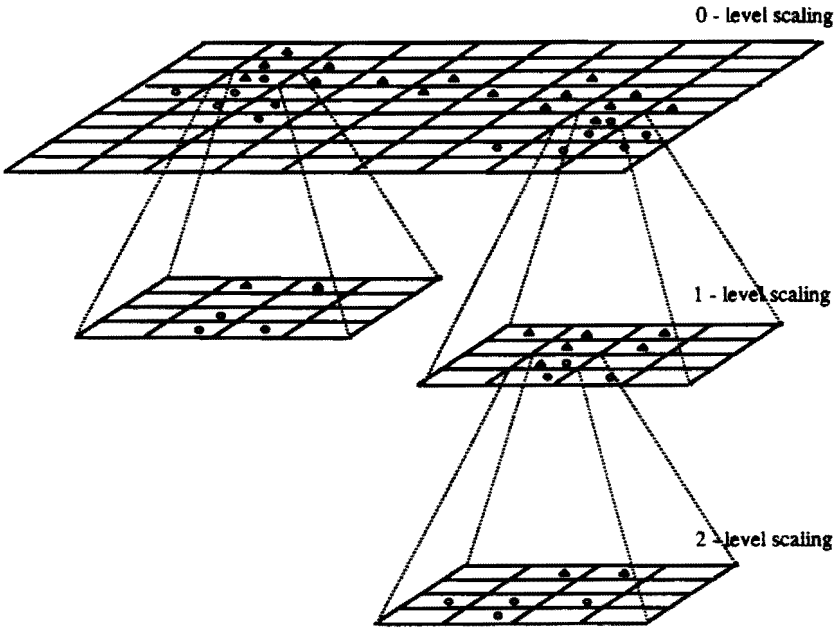


Fig. 6. Multi-level scaling (in the case of two attributes and two classes of objects).

2.4. Texture Description Processes

2.4.1. Machine learning approach

The inductive incremental learning program AQ14[11,15] was applied to learn the texture attributional descriptions from examples. The AQ program performs a heuristic search through a space of symbolic expressions, and its goal is to find the most preferred

expression according to a specified criterion. The input to the AQ program consists of a string of learning events, and each event is a vector of attribute values. The set of events obtained for one class is called a set of positive examples. With respect to this particular class, all other events are negative examples. The program finds an optimal cover of all the positive examples. This cover cannot include any negative examples. The process repeated for each class of learning events produces decision rules to discriminate all classes of texture images amongst themselves. The conditional part of a rule is defined as a cover, and it is a disjunction of complexes (using the OR operator), where a complex is decomposed into selectors (using the AND operator). A selector is a value or a disjunction of values within a selector, e.g.

$$\begin{aligned}
&\text{rule:} && [\text{Transport} = \text{car}] \Leftarrow [\text{Weather} = \text{bad}] \vee [\text{Temp} < 60] \\
&\text{complex:} && [\text{Weather} = \text{bad}] \leftarrow [\text{Weather_type} = \text{cloudy}] \ \& \ [\text{Temp} > 60] \ \& \\
& && \qquad [\text{Wind_dir} = \text{South} \vee \text{West}] \\
&\text{selector:} && [\text{Weather_type} = \text{cloudy}]
\end{aligned} \tag{4}$$

The AQ14 inductive incremental learning program can work in two modes producing intersecting or disjoint covers. Rule induction in the intersecting mode produces covers that can logically intersect with those of other classes over "don't care" areas of the event space. On the other hand, rule induction in the disjoint mode produces covers that do not intersect at all with covers of other classes. As a consequence, rules produced in the intersecting mode are more general than rules produced in the disjoint mode.

In our experiments with texture recognition, we used both disjoint and intersecting modes, mainly to compare recognition results. The input data was composed of six sets of learning events according to six texture classes and an additional set of inconsistent events. A single event was composed of eight attributes representing one of the two approaches to texture characteristics obtained using Laws' masks or co-occurrence matrices. Each attribute was coded onto 50 levels. The output of the AQ14 algorithm consisted of the discrimination rules which were transferred to the texture recognition phase.

2.4.2. Pattern recognition approach

We considered several traditional parametric and non-parametric PR methods for texture description and recognition. The parametric methods, e.g. risk minimization using Bayes decision method, were excluded after testing the feature space. The creation of parametric models of feature distribution was not satisfactory because the distribution of teaching data was irregular and difficult to estimate from parametric curves. Therefore, we chose the well-known k-NN non-parametric statistical pattern recognition method.[2]

During the learning phase, teaching examples are cumulated into their classes. During the recognition phase, the set of k-nearest teaching examples from a testing event is selected from the set of all teaching examples. Thus, the classification decision is created indicating this class, for which most of the k-nearest teaching examples was selected. The main advantage of this method is its handling of irregularity and complexity of the

teaching data. However, the requirement of storing all teaching data (or selected data from the most representative samples of the feature domain distribution) for their use during the recognition phase is its main disadvantage. This disadvantage limits the use of such methods when applying the dynamic recognition system mentioned in Sec. 1. For the first stage of our experiments, we used the k-NN method to provide a simple comparison of "static" ML and PR approaches to texture recognition and to imagine the texture complexity and effectiveness of feature extraction methods. According to this method we transferred all sets of acquired teaching numerical examples directly to the recognition module (Figs. 1 and 3).

2.5. Decision-Making Within The Recognition Phase

In the next phase of the experiment we tested the inductive descriptions of texture classes, which were generated by the AQ14 program. The recognition process was applied to the right-hand side of the input images that had not been seen by the system before. The same methods as in the learning phase were applied to texture-feature detection. One hundred examples were obtained for each class of texture and for each of two texture description methods (Laws' masks and co-occurrence matrices). These examples were scaled using parameters calculated during the learning phase within the data conversion interface.

2.5.1. Recognition by rule induction

We applied a software tool (ATEST) developed for rule base testing[16] to support texture recognition by rule induction. The ATEST program evaluates the overall performance of the rule base. In our case, the program worked on the separated sets of events, where each set was obtained for a single class of texture as described above. Each event was classified into one of six classes of texture. There were three possible classification decisions for a single event, i.e. an event belongs only to the correct class (unique-classification), an event does not belong to the correct class (misclassification) and an event belongs to several classes where one of them is the correct class (multiple-classification). The final recognition decision is made based on counting the unique- and multiple-classification events for a single image. Such classification is called *first rank*, and is our main measure of the effectiveness of recognition.

2.5.2. Recognition by pattern classification

A simple comparison of applied inductive learning with the traditional k-NN pattern recognition method was made. This comparison used the values $k = 10$ and 30 for the k-NN method.

3. COMPARISON OF RESULTS

The results obtained from the recognition experiments are presented separately for the inductive learning approach (Table 1) and for the pattern recognition approach (Table 2). Below, we briefly discuss these results:

Table 1. Recognition results for inductive learning approach (AQ14)-intersection mode.

	Texture description method					
	Laws' masks			Co-occurrence matrices		
	Number of generated complexes	Recognition		Number of generated complexes	Recognition	
		First rank	Unique		First rank	Unique
Class 1	38	**72%**	37%	26	**88%**	78%
Class 2	35	**74%**	45%	31	**78%**	57%
Class 3	23	**81%**	57%	24	**90%**	78%
Class 4	8	**93%**	91%	6	**99%**	96%
Class 5	31	**73%**	45%	46	**57%**	35%
Class 6	23	**87%**	64%	13	**84%**	78%
Average recognition		**80%**	56%		**83%**	70%

Table 2. Recognition results for k-NN pattern recognition approach.

	Texture description method			
	Laws' masks		Co-occurrence matrices	
	Recognition for $k = 10$	Recognition for $k = 30$	Recognition for $k = 10$	Recognition for $k = 30$
Class 1	53%	49%	88%	90%
Class 2	73%	68%	75%	77%
Class 3	88%	91%	95%	97%
Class 4	99%	97%	98%	96%
Class 5	40%	38%	49%	46%
Class 6	69%	57%	97%	93%
Average recognition	**70%**	**66%**	**83%**	**83%**

- The same average recognition effectiveness was observed both for the ML and the PR approaches when the texture features were obtained from the co-occurrence matrix method. For the Laws' masks method, the ML approach to texture recognition was better than the PR approach.

- The maximum recognition effectiveness was significantly decreased by the low recognition rate obtained for the Class 5 texture. With a minimum threshold of 50%, the inductive learning approach recognized all textures, but the k-NN method did not recognize Class 5.
- We observed that neither Laws' masks method nor the co-occurrence matrix method for texture-feature extraction was consistent when compared amongst themselves for each class of texture. The Laws' masks method was generally worse considering both the number of generated complexes of the rules and the recognition results. But in the case of the fifth class, the number of complexes in the rule was significantly lower and the recognition rate was higher.

The recognition results shown in Table 1 were obtained for the intersection cover mode of the inductive learning algorithm. In this case, the generation of rules for the intersecting mode was much faster than for the disjoint cover mode. The average recognition effectiveness was also better. The results show that for approximately the same number of complexes generated in the intersecting and disjoint modes, the recognition results were better for the disjoint mode. This tendency is presented in Table 3 for Class 1 texture. On the other hand, the recognition rate for Class 5 is also included, to show the tremendous decrease in recognition rate for the disjoint mode, where a large number of complexes was generated.

Table 3. Results comparison for two modes of rule generation—disjoint cover mode (DC) and intersection cover mode (IC).

	Texture description method							
	Laws' masks method				Co-occurrence matrices method			
	DC mode		IC mode		DC mode		IC mode	
	Number of complexes	Recog. result	Number of complexes	Recog. result	Number of complexes	Recog. result	Number of complexes	Recog. result
Class 1	38	85%	38	72%	26	89%	26	88%
Class 5	118	54%	31	73%	111	34%	46	57%

4. MODIFICATION OF INDUCTIVE DESCRIPTION VIA SG-TRUNC RULE REDUCTION METHOD

The promising recognition results obtained from the ML approach motivate the investigation of applying a rule optimization methodology executed after the learning and before the recognition phases. We used a method of rule optimization that is based on the two-tiered description of imprecise concepts introduced by Michalski et al.[11] and Michalski.[17] A simple two-tiered concept description generates both the Base Concept Representation (BCR) of typical properties of a concept, as well as the Inferential Concept

Interpretation (ICI) of allowed concept modifications. The SG-TRUNC method was used to obtain a BCR through a sequence of generalization and specialization operations.[18] Initially, the SG-TRUNC method performs generalization to remove selectors from the complexes. After such removal, a complex is more general, i.e. it covers more examples. Then, a specialization operation removes the number of complexes. In this way, the description covers less examples.

The rule optimization processes are based on rule characteristics. These characteristics are composed of two coefficients, the t-weight and the u-weight. The t-weight is the total number of examples covered by a complex, while the u-weight is the number of examples covered by the same complex and no other. The SG-TRUNC method preserves those complexes that have high t- and high u-weights and modifies those complexes with low t- and u-weights. The degree of rule optimization is controlled by two real parameters, both in the range from 0. to 1.0. The first parameter controls the removal of selectors and the second one controls the reduction of complexes.[18] Increases in parameter values cause greater rule modification.

We already used the SG-TRUNC method as contained in the AQ16 algorithm. Relatively low parameter values were applied, both equal to 0.05, to control the removal of selectors and complexes. This means that the optimization of rules was low. The obtained recognition rates are presented in Table 4 and can be compared with the results in Table 1.

It is seen that the number of complexes has been reduced significantly. The number of selectors has been reduced as well. The recognition rules, both for the Laws' masks method and the co-occurrence matrix method, are much better. The secondary effect of this optimization is the increase of recognition speed.

Table 4. Recognition results for combined inductive learning and rule truncation method (AQ16)—intersection cover mode.

	Texture description method					
	Laws' masks			Co-occurrence matrices		
	Number of generated complexes	Recognition		Number of generated complexes	Recognition	
		First rank	Unique		First rank	Unique
Class 1	6	**96%**	1%	5	**90%**	42%
Class 2	6	**88%**	6%	3	**77%**	34%
Class 3	3	**90%**	25%	3	**86%**	61%
Class 4	1	**92%**	80%	1	**94%**	92%
Class 5	7	**83%**	2%	12	**75%**	32%
Class 6	5	**98%**	39%	4	**96%**	61%
Average recognition		**91%**	35%		**86%**	53%

The average recognition rate increased to 91% in the case of the Laws' masks method of feature extraction and to 86% for the co-occurrence matrix method. The recognition rate was significantly increased (up to 83% and 75% respectively) for the recognition of the fifth class of texture, while this texture was not recognized by the PR approach. In this way, the minimum recognition rate for both methods was improved. Moreover, the variation of recognition rates over texture classes has been reduced, i.e. the recognition rates have been soothed. The smoothing effect has been computed for both methods of texture-feature extraction and the method of texture description (ML-optimized rules, ML-rules and PR k-NN method) as an averaged deviation from the average recognition rate:

$$c = 1/N \sum_{i/1}^{N} |\underline{x} - x_i|. \tag{5}$$

The summary of results, presented in Table 5, compares the effectiveness of the applied approach to the texture recognition problem. Table 5 illustrates the smoothing effect of the recognition rates by the comparison of the averaged deviation (5). This smoothing effect consequently increases these rates for classes that are less easily recognized, and decreases these rates for classes that are more easily recognized. The lowest averaged deviation was obtained for the ML approach executed with rule optimization, while the highest averaged deviation was obtained for the above described PR approach. Applied methodology gave us a three-fold decrease of the deviation coefficient for the Laws' masks method.

The application of the SG-TRUNC method to rule optimization is also good. It is seen when studying recognition rates of unique-classification events. These recognition rates dropped for all classes of texture, which means that rules are more general.

Table 5. Summary of results.

	Texture description method					
	Laws' masks			Co-occurrence matrices		
	ML approach optimal rules	ML approach rules	PR approach k-NN method	ML approach optimal rules	ML approach rules	PR approach k-NN method
Average recognition rate	**91%**	**80%**	**70%**	**86%**	**83%**	**83%**
Highest recognition rate	98%	93%	99%	96%	99%	78%
Lowest recognition rate	83%	72%	40%	75%	57%	49%
Averaged deviation	**4.0**	**7.0**	**16.3**	**7.0**	**10.0**	**14.6**

5. CONCLUSIONS AND FUTURE WORK

The main aim of this work was to test the inductive learning approach for texture recognition, where textures were characterized by well-known low-level feature extraction methods. Three hierarchical levels of the generalization processes were applied: scaling, inductive learning and rule optimization. We showed that the scaling method can be applied as an interface for numeric-to-symbolic data conversion; it allows the use of symbolic computation not only by high-level vision but also on the lower levels of the recognition hierarchy. A comparison with the simple k-NN pattern recognition method was provided to present the complexity and levels of difficulty of our textures and their accurate recognition. This work has proved that the ML (inductive learning) approach can be applied successfully to typical pattern recognition problems. The obtained recognition results for each of the texture classes and the average recognition rate (91%) are quite satisfactory at this stage of our work. Class 5 of the texture was recognized by the ML approach with an 83% rate, whereas it was not recognized by the PR approach.

Based on these results and experiences, the following requirements will be necessary for our future efforts to develop a dynamic adaptable system for texture recognition that can be applied to mobile robot navigation.

- The future system must integrate several texture-feature extraction methods (instead of only one method) including both numeric and symbolic feature extraction. A feedback connection between feature extraction and texture (objects) recognition modules will find an optimal method and tune its parameters.
- Lower level numerical computation will be integrated with symbolic inductive learning using a scaling interface. However, such an interface must provide hierarchical multiresolution scaling at different levels of the feature space (top-down) and early generalization of learning events (bottom-up).
- A dynamic environment needs an incremental inductive learning module, which must be directly integrated with a dynamic memory system for the control of the scaling module. Such a memory will allow the input of new texture classes or their merging to update the texture description by new events and to change the number of attributes of a single event.

The need for these studies is motivated by the requirement to develop an intelligent system with learning capabilities used to support vision adaptability functions. This adaptability is necessary in most vision applications and will be tested for robot navigation in an outdoor terrain. We expect that a symbolic approach to system adaptability can be applied in the domain of numerical computation, with the use of a scaling interface (as an early generalization operation executed under higher control). In this way, low-level symbolic computation can be integrated with numeric transformations into a hybrid system of texture-feature extraction and recognition.

ACKNOWLEDGEMENTS

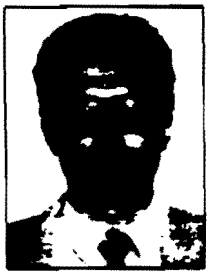
The author wishes to thank Professor Ryszard Michalski for constructive discussions and comments, and J. Bala, H. de Garis, K. Kaufman and J. Zhang for discussion and technical help.

This research was done in the Artificial Intelligence Center of George Mason University. Research activities of the Center are sponsored in part by the Defense Advanced Research Projects Agency under grant No. N00014-87-K-0874, administrated by the Office of Naval Research; and in part by the Office of Naval Research under grant No. N00014-88-K0226 and grant No. N00014-88-K-0397.

REFERENCES

1. K. S. Narendra, Ed., *Adaptive and Learning Systems*, Plenum Press, 1986.
2. O. R. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, 1973.
3. H. Wechsler and L. Zimmerman, "2-D invariant object recognition using distributed associative memory", *IEEE Trans. Pattern Anal. Mach. Intell.* **10**, 6 (1988) 811–821.
4. S. Pinker, *Visual Cognition*, MIT Press, 1985.
5. R. C. Bolles and R. A. Cain, "Recognizing and locating partially visible objects: The local-feature-focus method", in *Robot Vision*, Ed. A. Pough, Springer-Verlag, 1983, pp. 44–81.
6. D. Marr, *Vision*, Freeman, San Francisco, 1982.
7. T. Poggio, J. Little, E. Gamble, W. Gillett, D. Geiger, D. Weinshall, M. Villalba, N. Larson, T. Cass, H. Buelhoff, M. Drumheller, P. Oppenheimer, W. Yang and A. Hurlbert, "The MIT Vision Machine", *Proc. DARPA Image Understanding Workshop*, Cambridge, 1988, pp. 177–198.
8. K. Ikeuchi and T. Kanade, "Modeling sensors and applying sensor model to automatic generation of object recognition program", *Proc. DARPA Image Understanding Workshop*, Cambridge, 1988, pp. 697–710.
9. B. Bhanu, "Automatic target recognition: State of the art survey", *IEEE Trans. Aerospace Electron. Syst.* **22**, 4 (1986) 364–379.
10. B. Bhanu and J. C. Ming, "TRIPLE: A multi-strategy machine learning approach to target recognition", *Proc. DARPA Image Understanding Workshop*, Cambridge, 1988, pp. 537–547.
11. R. S. Michalski, I. Mozetic, J. Hong and N. Lavrac, "The AQ15 Inductive Learning System: An Overview and Experiments", ISG 86-23, UIUCDCS-R-86-1260, Department of Computer Science, University of Illinois, Urbana, 1986.
12. L. Van Gool, P. Dewaele and A. Oosterlinck, "Texture analysis Anno 1983", *Comput. Vision Graph. Image Process.* **29** (1985) 336–357.
13. K. I. Laws, "Textured Image Segmentation", Ph. D. Thesis, Dept. of Electrical Engineering, University of Southern California, Los Angeles, 1980.
14. S. J. Roan, J. K. Aggarwal and W. N. Martin, "Multiple resolution imagery and texture analysis", *Pattern Recogn.* **20**, 1 (1987) 17–31.
15. J. Hong, I. Mozetic and R. S. Michalski, "AQ15: Incremental Learning of Attribute-Based Descriptions from Examples, the Method and User's Guide", ISG 86-5, UIUCDCS-F-86-949, Department of Computer Science, University of Illinois, Urbana, 1986.

16. R. E. Reinke, "Knowledge Acquisition and Refinement Tools for the ADVICE META-EXPERT System", ISG 84-4, UIUCDCS-F-84-921, Department of Computer Science, University of Illinois, Urbana, 1984.
17. R. S. Michalski, "Two-tiered concept meaning, inferential matching and conceptual cohesiveness", in *Similarity and Analogy*, Eds. S. Vosniadou and A. Orton, Cambridge University Press, 1987.
18. J. Zhang and R. S. Michalski, "Rule optimization via SG-TRUNC method", *Proc. Fourth European Working Session on Learning*, Montpellier, Dec. 1989, Morgan Kaufmann, pp. 251–262.



Peter W. Pachowicz received the M.S. in computer and electrical engineering and the Ph.D. in computer science and engineering from the University of Mining and Metallurgy, Kracow, Poland, in 1981 and 1984, respectively. Since 1984 he has been a faculty member at the Institute of Control Engineering, University of Mining and Metallurgy, where he worked on fast and cheap processing of images in industrial applications. In 1986, he received the Alexander von Humboldt Research Fellowship to study self-adaptation capabilities of robot vision systems. From 1986 to 1988 he worked with the Cognition Systems Group of the Computer Science Department, University of Hamburg in West Germany. In 1989, he joined both the AI Center and the Computer Science Department of George Mason University. His research approaches are usually practically oriented. His areas of interest include intelligent autonomous systems, robot vision, adaptive systems and the application of AI. His present effort at the AI Center is related to the integration and application of high-level AI (i.e. machine learning) within an engineering domain.