# Evolutionary Feature Synthesis for Object Recognition

Yingqiang Lin and Bir Bhanu, *Fellow, IEEE*

***Abstract*—Features represent the characteristics of objects and selecting or synthesizing effective composite features are the key to the performance of object recognition. In this paper, we propose a coevolutionary genetic programming (CGP) approach to learn composite features for object recognition. The knowledge about the problem domain is incorporated in primitive features that are used in the synthesis of composite features by CGP using domain-independent primitive operators. The motivation for using CGP is to overcome the limitations of human experts who consider only a small number of conventional combinations of primitive features during synthesis. CGP, on the other hand, can try a very large number of unconventional combinations and these unconventional combinations yield exceptionally good results in some cases. Our experimental results with real synthetic aperture radar (SAR) images show that CGP can discover good composite features to distinguish objects from clutter and to distinguish among objects belonging to several classes. The comparison with other classical classification algorithms is favorable to the CGP-based approach proposed in this paper.**

***Index Terms*—Composite feature, feature synthesis, genetic programming, object recognition, synthetic aperture radar images, vehicle recognition.**

## I. Introduction

In THIS paper, we investigate the effectiveness of domain knowledge in improving the efficiency of evolutionary search and the efficacy of genetic programming in synthesizing composite features for object recognition. The basic task of object recognition is to identify the kinds of objects in an image, and sometimes the task may include estimating the pose of the recognized objects. One of the key approaches to object recognition is based on features extracted from images. These features capture the characteristics of the object and are fed into a classifier to perform recognition. The quality of object recognition is heavily dependent on the effectiveness of the features. However, it is difficult to extract good features from real images due to various factors, including noise. More importantly, there are many features that can be extracted. What are the appropriate features or how to select an appropriate set of features from the available features? If it is very difficult or even impossible to extract effective features from images, how to synthesize useful features based on the available ones? To make use of knowledge about a specific domain and improve the quality of synthesized features, how to incorporate domain knowledge in the feature synthesis? The answers to these questions are largely dependent on the instinct, knowledge, experience, and the bias of human experts.

In this paper, the effectiveness of coevolutionary genetic programming (CGP) [1] in generating composite operator vectors for object recognition is investigated. Genetic programming (GP) is an evolutionary computational paradigm [1] that is an extension of genetic algorithm and works with a population of individuals. An individual in a population can be any complicated data structure such as linked lists, trees and graphs, etc. CGP is an extension of GP in which several populations are maintained and employed to evolve solutions cooperatively. A population maintained by CGP is called a subpopulation and it is responsible for evolving a part of a solution. A complete solution is obtained by combining the partial solutions from all the subpopulations. In this paper, individuals in subpopulations are composite operators, which are the elements of a composite operator vector. A composite operator is represented by a binary tree whose internal nodes are the pre-specified domain-independent primitive operators and leaf nodes are primitive features. It is a way of combining primitive features. The advantage of using a tree structure is that it is powerful enough in expressing the ways of combining primitive features and unlike a graph, it has no loops and this guarantees that the execution of individuals represented by trees will terminate and not be trapped in an infinite loop. The primitive features can be simple features directly extracted or complicated features designed by human experts based on the characteristics of objects to be recognized in a particular kind of imagery (e.g., SAR images). The primitive features are real-valued attributes in this paper. With each element evolved by a subpopulation of CGP, a composite operator vector is cooperatively evolved by all the subpopulations. By applying composite operators, corresponding to each subpopulation, to the primitive features extracted from images, composite feature vectors are obtained. These composite feature vectors are fed into a classifier for recognition. It is worth noting that the primitive operators and primitive features are decoupled from the CGP mechanism that generates composite features, so they can be tailored to particular recognition tasks without affecting the other parts of the system. Thus, the method and the recognition system are flexible and can be applied to a wide variety of images.

Section II explains the motivation for using CGP as a tool for learning composite features. It also surveys the related works. Section III provides the overall structure of the learning and recognition system and gives the technical details used in this paper. Experimental results are presented in Section IV. Section V concludes the paper and proposes possible future research directions.

Manuscript received August 31, 2003; revised February 27, 2004 and April 8, 2004. This work was supported by Grant F33615-99-C-1440. The contents of the information do not necessarily reflect the position or policy of the U.S. Government. This paper was recommended by Guest Editor Y. Jin.

The authors are with the Center for Research in Intelligent Systems, University of California, Riverside, CA 92521 USA (e-mail: yqlin@cris.ucr.edu; bhanu@cris.ucr.edu).

Digital Object Identifier 10.1109/TSMCC.2004.841912

## II. Motivation and Related Research

### A. Motivation

The recognition accuracy of an automatic object recognition system is determined by the quality of the feature set used. Usually, it is the human experts who design the features to be used in recognition. Designing a set of effective features requires human ingenuity and insight into the characteristics of the objects to be recognized and in general, it is very difficult to identify a set of features that characterize a complex set of objects. Typically, many types of features are explored before a recognition system can be built to perform the desired recognition task. There are many features available and these features may be correlated, making the designing and selection of appropriate features a very time consuming and expensive process. Sometimes, it is very difficult to figure out and extract simple features that are effective in recognition directly from images. However, human experts generally know what kinds of features are useful for a particular kind of imagery. These simple features can be selected as primitive features. At this time, synthesizing composite features that are effective to the current recognition task from these primitive features becomes extremely important.

The process of synthesizing composite features can often be dissected into some primitive operations on the primitive features. It is usually the human experts who, replying on their knowledge and rich experience, figure out a smart way to combine these primitive operations to yield good composite features. The task of finding good composite features is equivalent to finding good points in the composite feature space. However, the ways of combining primitive features are almost infinite, leading to a huge composite feature space. It is obvious that a smart search strategy is a must in order to find good composite features in such a huge space. The human experts can only try a very limited number of combinations due to slow speed of human being and usually only the conventional combinations are tried due to the limited knowledge, experience and even the bias of human experts. CGP, on the other hand, may try many unconventional combinations and in some cases it is these unconventional combinations that yield exceptionally good recognition performance. Also, the inherent parallelism of CGP and the concept of subpopulations (search by many individuals) facilitate its implementation on multiprocessor supercomputers to further increase the search speed and allow a much larger portion of the search space to be explored by CGP than that explored by human experts, thus, greatly enhancing the chance of finding good composite features. As a result, CGP is a very useful tool in comparison to human experts in the feature design and synthesis.

### B. Related Research

In general, feature selection and feature synthesis are two kinds of feature transformations. In feature selection, original features are not changed and some original features are selected to form a subset of features to be used by classifiers. Genetic algorithm is widely used in feature selection [11]. In feature synthesis, a transformation, linear or nonlinear, is applied to the original features to generate new features. Weighted summation is a kind of linear transformation on the original features, and the weights of features can be determined by genetic algorithm. In multilayer neural networks, each node of a neural network takes the weighted sum of the outputs of its child nodes as input [10]. The weights are determined by backpropagation algorithm during training. The output of a node is determined by the input and the activation function of the node. It can be viewed as a nonlinear transformation on the original features. The CGP-based feature synthesis is another kind of nonlinear transformation on the original features, which are the primitive features in this paper.

GP [1] has been used in image processing, object detection and recognition. Harris and Buxton [2] apply GP to the production of high performance edge detectors for one-dimensional (1-D) signals and image profiles. The method is also extended to the development of practical edge detectors for use in image processing and machine vision. Ebner and Zell [3] use GP to automate the process of chaining a series of well-known image processing operators to perform image processing. Poli [4] uses GP to develop effective image filters to enhance and detect features of interest or to build pixel-classification-based segmentation algorithm. Bhanu and Lin [5] use GP to generate composite operators for object detection. The primitive operators and primitive features used in their system are very basic and domain-independent, so their object detection system can be applied to a wide variety of images. Their experimental results show that GP is a viable way of synthesizing composite features from primitive features for object detection and ROI (region-of-interest) extraction. Howard *et al.* [6] apply GP to automatic detection of ships in low resolution SAR imagery using an approach that evolves detectors. The detectors are algebraic formulae involving the values at pixels belonging to a small region surrounding the pixel undergoing the test and the detectors evolved by GP compare favorably in accuracy to those obtained using a neural network. Roberts and Howard [7] use GP to develop automatic object detectors in infrared images. They present a multistage approach to address feature detection and object segregation and the detectors developed by GP do not require images to be preprocessed. Stanhope and Daida [8] use GP paradigms for the generation of rules for target/clutter classification and rules for the identification of objects. GP determines relevant features from previously defined features to form a selected feature set. It evolves logical expressions based on the comparison of the selected features to both real-valued constants and other features in the selected feature set to create a classifier. Krawiec and Bhanu [9] present a method for the automatic synthesis of recognition procedures chaining elementary operations for computer vision and pattern recognition tasks based on cooperative coevolution and linear GP. Each subpopulation evolves a part of the recognition procedure and all the subpopulations coevolve the whole recognition procedure by selecting the best individual from each subpopulation and chaining them together. Their experimental results show that linear GP is effective in synthesizing a recognition procedure from elementary image processing operations. They also show that coevolutionary linear GP is superior to regular single-population linear GP that is equivalent to genetic algorithms.

Unlike the work of Stanhope and Daida [8], the primitive operators in this paper are not logical operators, but operators on

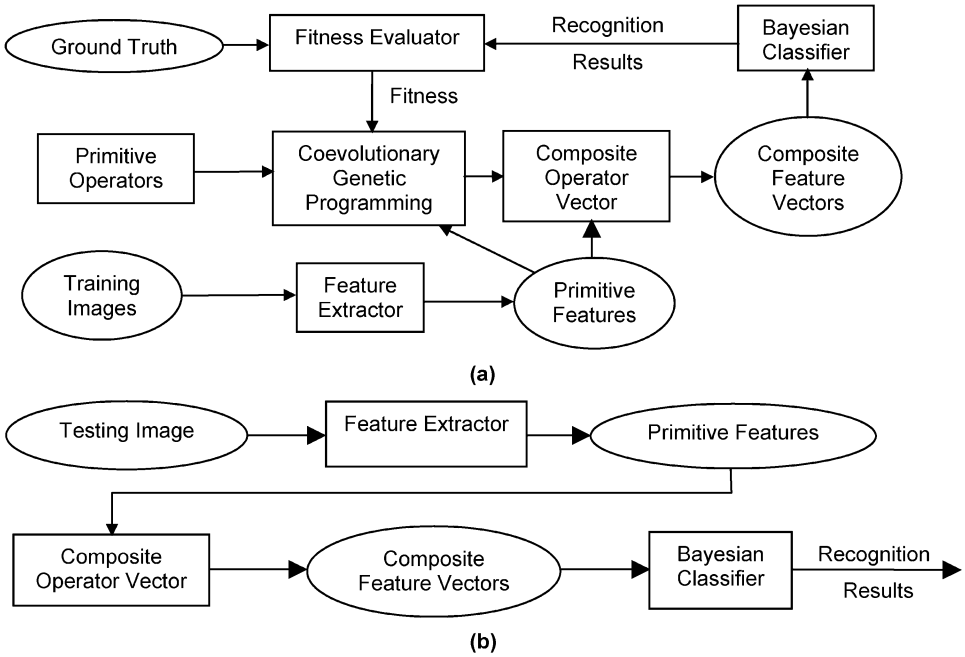


Fig. 1. System diagram for object recognition using co-evolutionary GP. (a) Training module—learning composite operator vectors and Bayesian classifier. (b) Testing module—applying learned composite operator vector and Bayesian classifier.

real numbers and the composite operators are binary trees of primitive operators on real numbers, not binary trees of logical operators. In [8], GP is used to evolve logical expressions and the final outcome of the logical expression determines the type of the object under consideration (for example, 1 means target and 0 means clutter). In this paper, CGP is used to evolve composite feature vectors to be used by a Bayesian classifier [10] and each subpopulation is responsible for evolving a specific composite feature in a composite feature vector. The classifier evolved by GP in [8] is a logical expression represented by the binary tree with the best classification rate in the population, but the classifier evolved by CGP in this paper is a Bayesian classifier determined by the composite feature vectors obtained from the training images. Unlike the work of Krawiec and Bhanu [9], composite operators in this paper are binary trees of primitive operators and primitive features, whereas the recognition procedures in [9] are linked lists of simple image processing operations.

## III. Technical Approach

In the CGP-based approach proposed in this paper, individuals are composite operators represented by binary trees with primitive operators as internal nodes and primitive features as leaf nodes. The search space is the set of all possible composite operators. The search space is huge and it is extremely difficult to find good composite operators from this vast space unless one has a smart search strategy. The system consists of training and testing modules, which are shown in Fig. 1(a) and (b), respectively. During training, CGP runs on training images and evolves composite operators to obtain composite features. Since a Bayesian classifier is derived from the composite feature vectors obtained from training images, both the composite operator vector and the classifier are learned by CGP.

### A. Design Considerations

To apply GP, there are five major design considerations, which involve determining the set of terminals, the set of primitive operators, the fitness measure, the parameters for controlling the run and the criterion for terminating a run.

*1) The Set of Terminals:* The set of terminals are 20 primitive features used in [11]. The first ten of them are designed by the Massachusetts Institute of Technology (MIT) Lincoln Lab to capture the particular characteristics of synthetic aperture radar (SAR) imagery and are found useful for object detection [12]. The other ten features are common features used widely in image processing and computer vision. To extract some primitive features, the constant false alarm rate (CFAR) image of an original image is needed. The CFAR image is generated by applying a two-parameter CFAR detector on an original image. A pixel value in the CFAR image measures the extent that the corresponding pixel in the original image stands out from those pixels on the border of the guard area around it. The guard area is a rectangular area with the pixel under measurement at the center. The CFAR value of a pixel $X$ in the original image is computed according to the following rule:

$$\mathrm{CFAR}(X) = \frac{X_t - \hat{u}_c}{\hat{\sigma}_c} \tag{1}$$

where $X_t$ is the amplitude of the pixel under consideration, $\hat{u}_c$ is the estimated mean of the amplitude of pixels on the border of the guard area, $\hat{\sigma}_c$ is the estimated standard deviation of the amplitude. For the detailed description of CFAR detector, please refer to [12]. In the following, we describe 20 primitive features in detail and compare the feature values of an object with those of natural clutter to demonstrate that the primitive features capture the characteristics of objects against natural clutter.

*a) The Standard Deviation Feature (Feature 1):* The standard deviation of an image is a statistical measurement of the

fluctuation of pixel intensities. If we use $P(r,a)$ to represent the pixel intensity from range $r$ and azimuth $a$, the standard deviation can be calculated as follows:

$$\sigma = \sqrt{\frac{S_2 - \frac{S_1^2}{N}}{N-1}} \quad \text{where}$$
$$S_1 = \sum_{r,a \in \text{region}} 10\log_{10} P(r,a)$$
$$S_2 = \sum_{r,a \in \text{region}} [10\log_{10} P(r,a)]^2 \tag{2}$$

and $N$ is the number of pixels in the region. Objects usually exhibit larger standard deviation than natural clutters, as illustrated by Fig. 2.

*b) The Fractal Dimension Feature (Feature 2):* The fractal dimension of the pixels in the region of interest provides information about the spatial distribution of the brightest scatterers of the detected object. It complements the standard deviation feature, which depends only on the intensities of the scatterers, not on their spatial locations.

The first step in applying the fractal dimension concept to a radar image is to select an appropriately sized region of interest, then convert the pixel values in the region of interest to binary values. One method of performing this conversion is to select the $N$ brightest pixels in the region of interest and convert their values to 1, while converting the rest of pixel values to 0. Based on these $N$ brightest pixels, the fractal dimension is defined by the following formula:

$$\dim = -\frac{\log M_1 - \log M_2}{\log 1 - \log 2} = \frac{\log M_1 - \log M_2}{\log 2} \tag{3}$$

where $M_1$ represents the minimum number of $1 \times 1$-pixel boxes required to cover $N$ brightest pixels in the region of interest (This number is obviously equal to $N$) and $M_2$ represents the minimum number of $2 \times 2$-pixel boxes required to cover $N$ brightest pixels.

The bright pixels for a natural clutter tend to be widely separated, thus produce a small value for the fractal dimension, while the bright pixels for an object tend to be closely bunched, thus a large value for the fractal dimension is expected, which is illustrated by Fig. 3. Fig. 3(a) shows an object image. In Fig. 3(b), the 50 brightest pixels from the object image are tightly clustered, and 22 $2 \times 2$-pixel boxes are needed to cover them, which results in a fractal dimension of 1.2. Fig. 3(c) shows a natural clutter image. In Fig. 3(d), the 50 brightest pixels from this natural clutter are relatively isolated, and 46 $2 \times 2$-pixel boxes are needed to cover them, which results in a fractal dimension of 0.29.

*c) Weighted-Rank Fill Ratio Feature (Feature 3):* This textural feature measures the percentage of the total energy contained in the brightest scatterers of a detected object. The weighted-rank fill ratio is defined as follows:

$$\eta = \frac{\sum_{k \text{ brightest pixels}} P(r,a)}{\sum_{\text{all pixels}} P(r,a)} \tag{4}$$

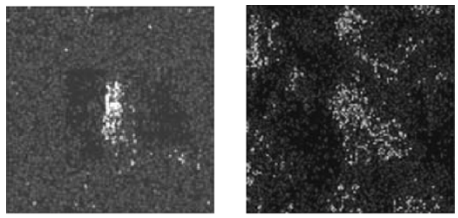


Fig. 2. Example of the standard deviation feature. (a) Typical object image with standard deviation 5.2832. (b) Typical natural clutter image with standard deviation 4.5187.

where $k$ is equal to 50 in our experiments. This feature attempts to exploit the fact that power returns from most objects tend to be concentrated in a few bright scatterers, whereas power returns form natural-clutter false alarms tend to be diffused. The weighted-rank fill ratio values of the object in Fig. 2(a) and the clutter in Fig. 2(b) are 0.3861 and 0.2321, respectively.

*d) Size-Related Feature (Features 4–6):* Three size-related features utilize the binary image created by the morphological operations on the CFAR detection result. The morphological operations are applied in the order of clean (remove isolated pixels), bridge (connect unconnected components if they are close to each other, at most three pixels apart) and close (dilation followed by erosion). The resulting largest component is called morphological blob.

1) The mass feature is the number of pixels in the morphological blob.
2) The diameter is the length of the diagonal of the smallest rectangle that encloses the blob.
3) The square-normalized rotational inertia is the second mechanical moment of the blob around its center of mass, normalized by the inertia of an equal mass square.

In the experiments, we find the size features are not effective, since the size and the shape of the detected morphological blob can be arbitrary. For the clutter, there is also no ground to assert that the resulting morphological blob will exhibit a certain amount of coherence. The experimental results in Fig. 4 show the arbitrariness of the morphological blobs for objects as well as clutters.

*e) The Contrast-Based Features (Features 7–9):* The CFAR statistics is computed for each pixel in an object-sized blob to create a CFAR image. Then the three features can be derived as follows.

1) The maximum CFAR feature is the maximum value of pixels in the CFAR image contained within an object-sized blob.
2) The mean CFAR feature is the average pixel value of pixels in the CFAR image taken over an object-sized blob.
3) The percent bright CFAR feature is the percentage of pixels within an object-sized blob that exceed a certain CFAR value.

The maximum CFAR feature, the mean CFAR feature and the percent bright CFAR feature values of the object in Fig. 2(a) are 55.69, 5.53, and 0.15, respectively, and these feature values of the clutter in Fig. 2(b) are 10.32, 2.37, and 0.042, respectively.

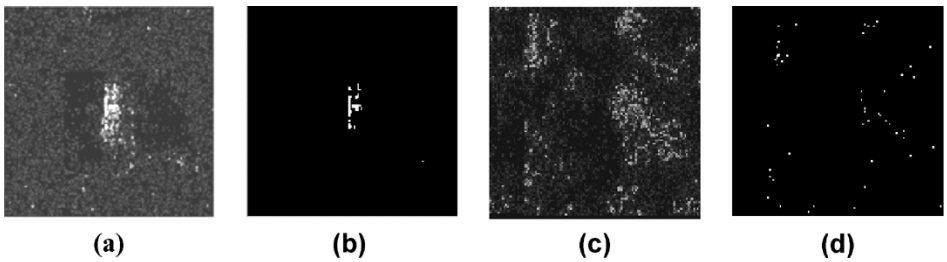


Fig. 3. Example of the fractal dimension feature. (a) Object image. (b) 50 brightest pixels in (a). (c) Natural clutter image. (d) 50 brightest pixels in (c).

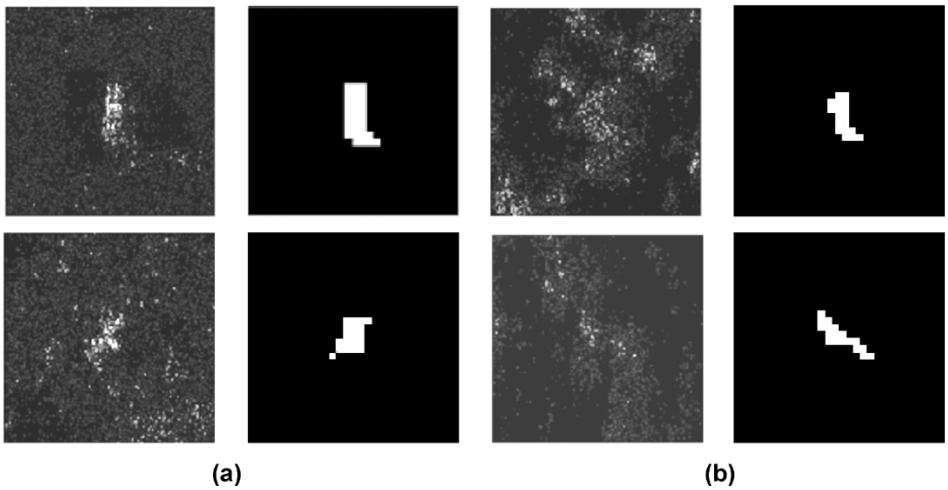


Fig. 4. Examples of images used to compute size features (4–6) for (a) object and (b) clutter. (a) Figures on the left-hand side represent the object images and the figures on the right-hand side represent their corresponding morphological blobs. (b) Figures on the left-hand side represent the clutter images and the figures on the right-hand side represent their corresponding morphological blobs.

We can see that CFAR feature values for an object are much larger than those for a natural clutter.

*f) The Count Feature (Feature 10):* The count feature is very simple; it counts the number of pixels that exceed the threshold $T$ and normalize this value by the total possible number of pixels in an object blob. The threshold $T$ is set to the quantity corresponding to the 98th percentile of the surrounding clutter. The count feature values of the object in Fig. 2(a) and the clutter in Fig. 2(b) are 0.6 and 0.1376, respectively. We can see that the count feature value for an object is much larger than that for a natural clutter false alarm. This is reasonable because the intensity values of the pixels belonging to an object stand out from the surrounding clutter.

The following ten features (four projection features, three distance features, and three moment features) are common features used in image processing and object recognition. They are extracted from binary images generated from the CFAR detection. In these images, foreground pixels (pixels with value 1) are potential object pixels.

*g) Projection Features (Features 11–14):* Four projection features are extracted from each binary image.

1) *Horizontal projection feature:* project the foreground pixels on a horizontal line (x axis of image) and compute the distance between the leftmost point and the rightmost point.
2) *Vertical projection feature:* project the foreground pixels on a vertical line (y axis of image) and compute the distance between the uppermost point and the lowermost point.
3) *Major diagonal projection feature:* project the foreground pixels on the major diagonal line and compute the distance between the upper leftmost and the lower rightmost points.
4) *Minor diagonal projection feature:* project the foreground pixels on the minor diagonal line and compute the distance between the lower leftmost and the upper rightmost points.

The average values of horizontal, vertical, major, and minor diagonal projection features of all the 1048 clutter SAR images ($120 \times 120$) we collected are approximately 60.0, 60.0, 90.0, and 90.0, respectively. Their corresponding values for the 1048 object SAR images ($120 \times 120$) are 34.5, 29.5, 46.7, and 47.8, respectively. It can be seen that the feature values of the clutters are larger than those of the objects. This result is reasonable, since the bright pixels of a natural clutter tend to be widely separated. This has already been shown by the fractal dimension feature value.

*h) Distance Features (Features 15–17):* Three distance features are extracted from each binary image. Before computing distance features, the centroid of all the foreground pixels in a binary image is computed.

1) *Minimum distance:* compute the distance from each foreground pixel to the centroid and select the minimum one.
2) *Maximum distance:* compute the distance from each foreground pixel to the centroid and select the maximum one.
3) *Average distance:* compute the distance from each foreground pixel to the centroid and get the average value of all these distances.

TABLE I
TWELVE PRIMITIVE OPERATORS

| Primitive Operator | Description | Primitive Operator | Description |
|---|---|---|---|
| ADD (a, b) | Add a and b. | ADDC (a, c) | Add constant value c to a. |
| SUB (a, b) | Subtract b from a. | SUBC (a, c) | Subtract constant value c from a. |
| MUL (a, b) | Multiply a and b. | MUL (a, c) | Multiply a with constant value c. |
| DIV (a, b) | Divide a by b. | DIVC (a, c) | Divide a by constant value c. |
| MAX2 (a, b) | Get the larger of a and b. | MIN2 (a, b) | Get the smaller of a and b. |
| SQRT (a) | Return $\sqrt{a}$ if a ≥ 0; otherwise, return $-\sqrt{-a}$. | LOG (a) | Return log(a) if a ≥ 0; otherwise, return – log(-a). |

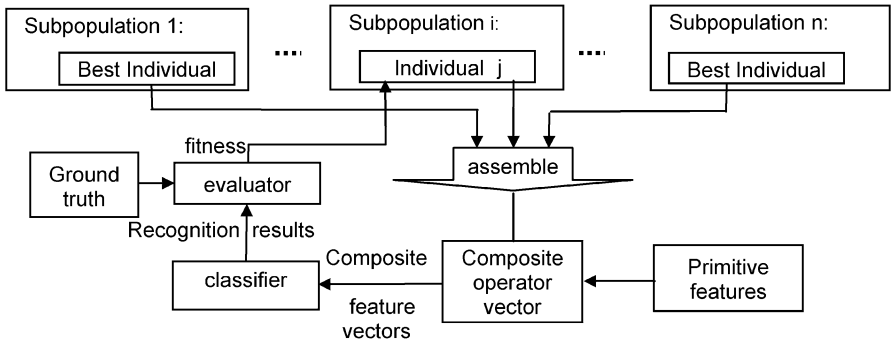


Fig. 5. Computation of fitness of $j$th composite operator of $i$th subpopulation.

The average values of minimum, maximum and average distance features of all the 1048 clutter SAR images are approximately 40.0, 70.0, and 60.0, respectively. Their corresponding values of the 1048 object SAR images are 3.8, 26.7, and 11.5, respectively. It can be seen that the feature values of the clutters are larger than those of the objects. This result is reasonable, since the bright pixels of a natural clutter tend to be widely separated.

*i) Moment Features (Features 18–20):* Three moment features are extracted from each binary image. All three moments are central moments, so before computing moment features, the centroid of all the foreground pixels in a binary image is computed. The central moments can be expressed as:

$$\mu_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}(x-\bar{x})^p(y-\bar{y})^q\,dx\,dy \tag{5}$$

where $(\bar{x},\bar{y})$ is the centroid and $p$ and $q$ are integers.

Moments $\mu_{20}, \mu_{02}$, and $\mu_{22}$ are also called horizontal, vertical and diagonal second-order moment features, respectively. The average values of horizontal, vertical and diagonal second-order moment features of all the 1048 clutter SAR images are approximately 910.0, 910.0, and 374 020.0, respectively. Their corresponding values of the 1048 object SAR images are 80.5, 46.7, and 4021.6, respectively. It can be seen that the feature values for the clutters are much larger than those for the objects. This result is reasonable, since the bright pixels of a natural clutter tend to be widely separated.

*2) The Set of Primitive Operators:* A primitive operator takes one or two real numbers, performs a simple operation on them and outputs the result. Currently, 12 primitive operators shown in Table I are used, where a and b are real numbers and input to an operator and c is a constant real number stored in an operator.

*3) The Fitness Measure:* The fitness of a composite operator vector is computed in the following way: apply each composite operator of the composite operator vector on the primitive features of training images to obtain composite feature vectors of training images and feed them to a Bayesian classifier. Note that not all the primitive features are necessarily used in feature synthesis. Only the primitive features that appear in the leaf nodes of the composite operator are used to generate composite features. The recognition rate of the classifier is the fitness of the composite operator vector. To evaluate a composite operator evolved in a subpopulation (see Fig. 5), the composite operator is combined with the current best composite operators in other subpopulations to form a complete composite operator vector where composite operator from the $i$th subpopulation occupies the $i$th position in the vector and the fitness of the vector is defined as the fitness of the composite operator under evaluation. The fitness values of other composite operators in the vector are not affected. When subpopulations are initially generated, the composite operators in each subpopulation are evaluated individually without being combined with composite operators from other subpopulations. In each generation, the composite operators in the first subpopulation are evaluated first, then the composite operators in the second subpopulation, and so on.

*4) Parameters and Termination:* The key parameters are the number of subpopulations $N$, the population size $M$, the number of generations $G$, the crossover and mutation rates, and the fitness threshold. GP stops whenever it finishes the specified number of generations or the performance of the Bayesian classifier is above the fitness threshold. After termination, CGP selects the best composite operator of each subpopulation to form the learned composite operator vector to be used in testing.

### B. Selection, Crossover, and Mutation

The CGP searches through the space of composite operator vectors to generate new composite operator vectors. The search is performed by selection, crossover and mutation operations. The initial subpopulations are randomly generated. Although subpopulations are cooperatively evolved (the fitness of a composite operator in a subpopulation is not solely determined by itself, but affected by the composite operators from other subpopulations), selection is performed only on composite operators within a subpopulation and crossover is not allowed between two composite operators from different subpopulations.

- **Selection**: The selection operation involves selecting composite operators from the current subpopulation. In this paper, tournament selection is used and the tournament size is 5. The higher the fitness value, the more likely the composite operator is selected to survive.
- **Crossover**: Two composite operators, called parents, are selected on the basis of their fitness values. The higher the fitness value, the more likely the composite operator is selected for crossover. One internal node in each of these two parents is randomly selected, and the two subtrees rooted at these two nodes are exchanged between the parents to generate two new composite operators, called offspring. It is easy to see that the size of one offspring (i.e., the number of nodes in the binary tree representing the offspring) may be greater than both parents if crossover is implemented in such a simple way. To prevent code bloat, we specify a maximum size of a composite operator (called max-operator-size). If the size of one offspring exceeds the max-operator-size, the crossover is performed again. If the size of an offspring still exceeds the max-operator-size after the crossover is performed ten times, GP selects two subtrees of same size (i.e., the same number nodes) from two parents and swaps the subtrees between the parents. These two subtrees can always be found, since a leaf node can be viewed as a subtree of size 1.
- **Mutation:** To avoid premature convergence, mutation is introduced to randomly change the structure of some composite operators to maintain the diversity of subpopulations. Candidates for mutation are randomly selected and the mutated composite operators replace the old ones in the subpopulations. There are three mutations invoked with equal probability:

1) Randomly select a node of the composite operator and replace the subtree rooted at this node by another randomly generated binary tree.
2) Randomly select a node of the composite operator and replace the primitive operator stored in the node with another primitive operator randomly selected from the primitive operators of the same number of input as the replaced one.
3) Randomly select two subtrees of the composite operator and swap them. Of course, neither of the two subtrees can be a subtree of the other.

### C. Generational Coevolutionary GP

Generational CGP is used to evolve composite operators. The GP operations are applied in the order of crossover, mutation and selection. The composite operators in the initial subpopulations are randomly generated. A composite operator is generated in two steps. In the first step, the number of internal nodes of the tree representing the composite operator is randomly determined as long as this number is smaller than half of max-operator-size. Suppose the tree has $n$ internal nodes. The tree is generated from top to bottom by a tree generation algorithm. The root node is generated first and the primitive operator stored in the root node is randomly selected. The selected primitive operator determines the number of children the root node has. If it has only one child, the algorithm is recursively invoked to generate a tree of $n-1$ internal nodes; if it has two children, the algorithm is recursively invoked to generate two trees of $\lfloor (n-1)/2 \rfloor$ and $\lceil (n-1)/2 \rceil$ internal nodes, respectively. In the second step, after all the internal nodes are generated, the leaf nodes containing primitive features are attached to those internal nodes that are temporarily the leaf nodes before the real leaf nodes are attached. The number of leaf nodes attached to an internal node is determined by the primitive operator stored in the internal node. In addition, an elitism replacement method is adopted to keep the best composite operator from generation to generation.

- **Generational CGP:**

0. randomly generate $N$ subpopulations of size $M$ and evaluate each composite operator in each subpopulation individually.
1. for $\mathrm{gen} = 1$ to G do
2. for $i = 1$ to $N$ do
3. keep the best composite operator in subpopulation $P_i$.
4. perform crossover on the composite operators in $P_i$ until the crossover rate is satisfied and keep all the offspring from crossover.
5. perform mutation on the composite operators in $P_i$ and the offspring from crossover with the probability of mutation rate.
6. perform selection on $P_i$ to select some composite operators and combine them with the composite operators from crossover to get a new subpopulation $P_i'$ of the same size as $P_i$.
7. evaluate each composite operator $C_j$ in $P_i'$.

To evaluate $C_j$, select the current best composite operator in each of the other subpopulations, combine $C_j$ with those $N-1$ best composite operators to form a composite operator vecter where composite operator from the $k$th subpopulation occupy the $k$th position in the vector ($k = 1, \ldots, N$). Run the composite operator vector on the primitive features of the training images to get composite feature vectors and use them to build a Bayesian classifier. Feed the composite feature vectors into the Bayesian classifier and let the recognition rate be the fitness of the composite operator vector and the fitness of $C_j$.

8. perform elitism replacement.

let the best composite operator from $P_i$ replace the worst composite operator in $P_i'$ and let $P_i = P_i'$

9. Form the current best composite operator vector consisting of the best composite operators from corresponding subpopulations and evaluate it. If its fitness is above the fitness threshold, goto 10.

endfor // loop 2 iterates on each subpopulation. after a new subpopulation is generated, the best composite feature vector is changed and we need to find the best composite feature vector and evaluate it to determine if CGP can be terminated

endfor // loop 1 iterates on each generation.

10. select the best composite operator from each subpopulation to form the learned composite operator vector and output it.

TABLE II
PARAMETERS OF CGP USED THROUGHOUT THE EXPERIMENTS

| | | | |
|---|---|---|---|
| Sub-population size | 50 | Crossover rate | 0.6 |
| Number of generation | 50 | Mutation rate | 0.05 |
| Fitness threshold | 1.0 | Tournament size | 5 |

### D. Bayesian Classifier

For each class $C_i$ ($i = 1, 2, 3$, or $1, 2, 3, 4, 5$ in this paper), a Bayesian classifier is generated based on CGP-learned composite features. A Bayesian classifier consists of a mean feature vector and a covariance matrix of feature vectors of class $C_i$. Suppose $f_{i1}, f_{i2}, \ldots, f_{in}$ are the feature vectors extracted from $n$ training images of class $C_i$, then the mean feature vector and the covariance matrix are computed by

$$\mu_i = \frac{1}{n}\sum_{j=1}^{n} f_{ij} \quad C_i = \frac{1}{n}\sum_{j=1}^{n}(f_{ij} - \mu_i)(f_{ij} - \mu_i)^T, \quad i = 1, 2, 3, 4 \text{ or } 5. \quad (6)$$

During testing, for a feature vector $f$ from a testing image, we compute distance $d_i$ ($d_i = (f - \mu_i)^T C_i^{-1}(f - \mu_i)$) and assign the object in the testing image to the class corresponding to the smallest distance. Here, we assume that the prior probability of each class is equal. For details on Bayesian classifier, please refer to [10].

## IV. EXPERIMENTS

Various experiments are performed to test the efficacy of GP in generating composite features for object recognition. In this paper, we show some selected examples. All the images used in the experiments are real synthetic aperture radar (SAR) images. These images are divided into training and testing images. Twenty primitive features are extracted from each SAR image. CGP runs on primitive features from training images to generate a composite operator vector and a Bayesian classifier. The composite operator vector and the Bayesian classifier are tested against the testing images. It is worth noting that the ground-truth is used only during training. The experiments are categorized into three classes: 1) distinguishing man-made objects from natural clutters; 2) distinguishing between three kinds of man-made objects; and 3) distinguishing between five kinds of man-made objects. For the purpose of objective comparison, CGP is invoked ten times for each experiment with the same set of parameters and the same set of training images. Only the average performances are used for comparison. Some of the parameters of CGP used throughout the experiments are shown in Table II. The max-operator-size is 10 in experiment 1 and 20 in experiments 2 and 3. The real number c stored in primitive operators ADDC, SUBC, MULC, and DIVC can be any real number from $-20$ to 20. When mutation is performed on these primitive operators, the value c stored in these primitive operators may be changed.

### A. Distinguish Object From Clutter

- **Data:** The data used here are the same as those used in [11]. From MSTAR (Moving and Stationary Target Acquisition and Recognition) public real SAR images, 1048 SAR images containing objects and 1048 SAR images containing natural clutters are generated. These images have size $120 \times 120$ and are called object images and clutter images, respectively. An example object image and clutter image are shown in Fig. 6, where white spots indicate scatterers with high magnitude. 300 object images and 300 clutter images are randomly selected as training images and the rest is used in testing.

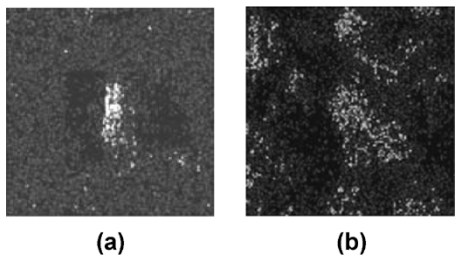


Fig. 6. Example object and clutter SAR images. (a) Object image. (b) Natural clutter image.

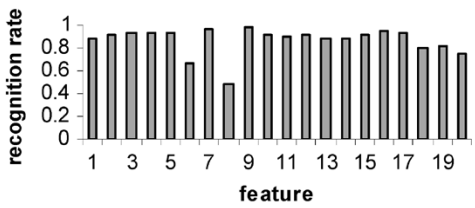


Fig. 7. Recognition rates of 20 primitive features.

- **Experiment 1:** First, the effectiveness of each primitive feature in discriminating the objects from the clutters is examined. Each kind of primitive features from the training images is used to train a Bayesian classifier and the classifier is tested against the same kind of primitive features from the testing images. The results are shown in Fig. 7 and Table III. Feature contrast brightness of blob (9) is the best one with recognition rate 0.98.

  To show the efficacy of CGP in synthesizing effective composite features, we consider three cases: only the worst two primitive features [blob inertia (6) and mean values of pixels within blob (8)] are used by CGP; five bad primitive features [blob inertia (6), mean values of pixels within blob (8), moments $\mu_{20}$ (18), $\mu_{02}$ (19) and $\mu_{22}$ (20) of scatters] are used by CGP; ten common features (primitive features 11 to 20) not specifically designed to process SAR images are used by CGP during feature synthesis. The number of subpopulations is 3, which means the dimension of the composite feature vectors is 3. CGP is invoked ten times with the same parameters. The average recognition performance over ten runs is shown in Table IV (first row) and Fig. 8, where 2f means only features (6) and (8) are used as primitive features (case 1), 5f means features 6, 8, 18, 19, and 20 are used (case 2) and 10f means only ten common features are used in feature synthesis (case 3). In Fig. 8, the horizontal coordinates are the number of primitive features used in synthesis and the vertical coordinates are the recognition

TABLE III
RECOGNITION RATES OF 20 PRIMITIVE FEATURES

| Feature Number | Primitive Feature | Recognition Rate | Feature Number | Primitive Feature | Recognition Rate |
|---|---|---|---|---|---|
| 1 | Standard deviation | 0.88 | 11 | Horizontal projection | 0.90 |
| 2 | Fractal dimension | 0.91 | 12 | Vertical projection | 0.91 |
| 3 | Weight-rank fill ratio | 0.94 | 13 | Major diagonal projection | 0.89 |
| 4 | Blob mass | 0.94 | 14 | Minor diagonal projection | 0.88 |
| 5 | Blob diameter | 0.94 | 15 | Minimum distance | 0.92 |
| 6 | Blob inertia | 0.66 | 16 | Maximum distance | 0.95 |
| 7 | Maximum CFAR | 0.97 | 17 | Mean distance | 0.94 |
| 8 | Mean CFAR | 0.49 | 18 | Moment $\mu_{20}$ | 0.80 |
| 9 | Percent bright CFAR | 0.98 | 19 | Moment $\mu_{02}$ | 0.81 |
| 10 | Count | 0.92 | 20 | Moment $\mu_{22}$ | 0.75 |

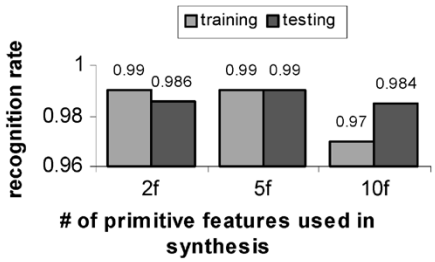


Fig. 8. Experimental results with three subpopulations.

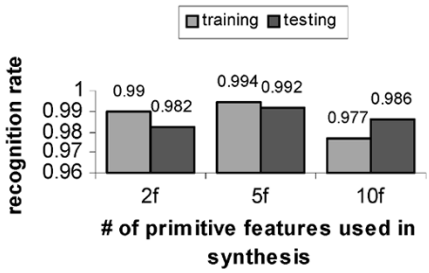


Fig. 9. Experimental results with five subpopulations.

rates. The bins on the left show the training results and those on the right show the testing results. The numbers above the bins are the average recognition rates over ten runs. Then the number of subpopulation is increased to five. The same two, five, and ten primitive features are used by CGP to evolve composite features. The average recognition performance over ten runs is shown in Table IV (second row) and Fig. 9. The performance of synthesized composite features is worse than the feature set selected by GA in [11]. It is reasonable, since in [11], a MDL-based GA is applied to select a set of features from all the 20 primitive features to distinguish objects from clutter, and effective features are always selected by GA. In this paper, we deliberately let CGP synthesize composite features from ten common features not specifically designed for SAR images or from two or five worst primitive features selected from 20 primitive features.

From Figs. 8 and 9, it can be seen that composite feature vectors synthesized by CGP are very effective. They are much better than the primitive features upon which they are built. Actually, if both features 6 and 8 from the training images jointly form two-dimensional (2-D) primitive feature vectors to train a Bayesian classifier for recognition, the recognition rates on training and testing data are 0.625 and 0.668, respectively; if features 6, 8, 18, 19, and 20 jointly form 5-dimensional primitive feature vectors, the recognition rates on training and testing data are 0.908 and 0.947, respectively; if all the ten common primitive features are used, the recognition rates on training and testing data are 0.963 and 0.978, respectively. These results are shown in Table IV (third row), where 2f, 5f, and 10f indicate both the primitive features used and the dimension of primitive feature vectors. The average recognition rates of composite feature vectors are better than all of the above results and this is the value of using CGP for feature synthesis. Fig. 10 shows the composite operator vector evolved by CGP maintaining three subpopulations in the sixth run when five primitive features are used, where PF$i$ means the primitive feature $i$ and so on. In Fig. 10, the least effective feature (feature 8) is used by the effective composite operator evolved by CGP. This phenomenon is not uncommon, since a feature is not isolated from other features and the interaction of features (covariance) is complicated. Sometimes, a feature is not effective if it is used alone, but when it is used in combination with other features, the high recognition rate may be achieved. At this time, all these features form an effective feature set.

From Figs. 8 and 9, it can be seen that composite feature vectors synthesized by CGP are very effective. They are much better than the primitive features upon which they are built. Actually, if both features 6 and 8 from the training images jointly form two-dimensional (2-D) primitive feature vectors to train a Bayesian classifier for recognition, the recognition rates on training and testing data are 0.625 and 0.668, respectively; if features 6, 8, 18, 19, and 20 jointly form 5-dimensional primitive feature vectors, the recognition rates on training and testing data are 0.908 and 0.947, respectively; if all the ten common primitive features are used, the recognition rates on training and testing data are 0.963 and 0.978, respectively. These results are shown in Table IV (third row), where 2f, 5f, and 10f indicate both the primitive features used and the dimension of primitive feature vectors. The average recognition rates of composite feature vectors are better than all of the above results and this is the value of using CGP for feature synthesis. Fig. 10 shows the composite operator vector evolved by CGP maintaining three subpopulations in the sixth run when five primitive features are

TABLE IV
PERFORMANCE OF COMPOSITE AND PRIMITIVE FEATURES ON OBJECT/CLUTTER DISCRIMINATION

| | Recognition Rate | | | | | |
|---|---|---|---|---|---|---|
| | 2f | | 5f | | 10f | |
| | Training | Testing | Training | Testing | Training | Testing |
| 3-dimensional composite feature vector | 0.989 | 0.986 | 0.991 | 0.989 | 0.971 | 0.984 |
| 5-dimensional composite feature vector | 0.99 | 0.982 | 0.994 | 0.992 | 0.977 | 0.986 |
| primitive feature vector | 0.625 | 0.668 | 0.908 | 0.947 | 0.963 | 0.978 |

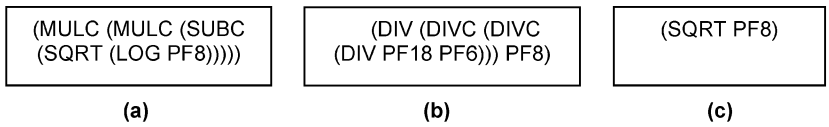


Fig. 10. Composite operator vector learned by CGP. (a) Composite operator 1. (b) Composite operator 2. (c) Composite operator 3.

used, where PF$i$ means the primitive feature $i$ and so on. In Fig. 10, the least effective feature (feature 8) is used by the effective composite operator evolved by CGP. This phenomenon is not uncommon, since a feature is not isolated from other features and the interaction of features (covariance) is complicated. Sometimes, a feature is not effective if it is used alone, but when it is used in combination with other features, the high recognition rate may be achieved. At this time, all these features form an effective feature set.

### B. Recognize Objects

- **Data:** Five objects (BRDM2 truck, D7 bulldozer, T62 tank, ZIL131 truck, and ZSU anti-aircraft gun) are used in the experiments. For each object, 210 real SAR images under 15°-depression angle and various azimuth angles between 0° and 359° are collected from moving and stationary target acquisition and recognition (MSTAR) public data. Fig. 11 shows one optical and four SAR images of each object. From Fig. 11, we can see that it is not easy to distinguish SAR images of different objects. Since SAR images are very sensitive to azimuth angles and training images should represent the characteristics of an object under various azimuth angles, 210 SAR images of each object are sorted in the ascending order of their azimuth angles and the first, fourth, seventh, tenth SAR images and so on are selected for training. Thus, for each object, 70 SAR images are used in training and the rest is used in testing.

- **Experiment 2—Discriminate three objects:** CGP synthesizes composite features to recognize three objects: BRDM2, D7, and T62. First, the effectiveness of each primitive feature in discriminating these three objects is examined. The results are shown in Table V and Fig. 12. Feature mean values of pixels within blob (8) is the best primitive feature with recognition rate 0.73. Three series of experiments are performed in which CGP maintains three, five, and eight subpopulations to evolve three-dimensional (3-D), five-dimensional (5-D), and eight-dimensional (8-D) composite feature vectors, respectively. The primitive features used in the experiments are all the 20 primitive features and ten common primitive features (primitive features 11 to 20). The average recognition rates of 3-D, 5-D, and 8-D composite feature vectors over ten runs are shown in Table VI and Figs. 13–15, where 10f and 20f mean primitive features 11 to 20 and all the 20 primitive features, respectively. The bins on the left show the training results and those on the right show the testing results. The numbers above the bins are the average recognition rates over ten runs.

  From Figs. 13–15, it can be seen that the learned composite feature vectors are more effective than primitive features. If all the 20 primitive features from the training images are used to form 20-dimensional primitive feature vectors to train a Bayesian classifier for recognition, the recognition rates on training and testing data are 0.995 and 0.96, respectively. This result is a little bit better than the average performance shown in Figs. 13 (0.94) and 14 (0.96), but the dimension of the feature vector is 20. However, the dimensions of composite feature vectors in Figs. 13 and 14 are just 3 and 5 respectively. If the dimension of composite feature vector is increased to 8, the CGP results are better. If the last ten primitive features are used, the recognition rates on training and testing data are 0.86 and 0.81, respectively. From these results, we can see that the effectiveness of the primitive features has an important impact on that of the composite features synthesized by CGP. In general, with more effective primitive features, CGP can synthesize more effective composite features. Fig. 16 shows the composite operator vector evolved by CGP with five subpopulations in the tenth run using 20 primitive features. The size of the first and second composite operators is 20. The size of the third one and last one are 9 and 15, respectively. The fourth composite operator is just primitive feature 11. The primitive features used by the learned composite operator vector are primitive features 2, 3, 4, 5, 6, 7, 8, 11, 12, 14, 18, 19, 20. If all these 13 primitive features form 13-dimensional (13-D) primitive feature vectors for recognition, the recognition rate is 0.96.

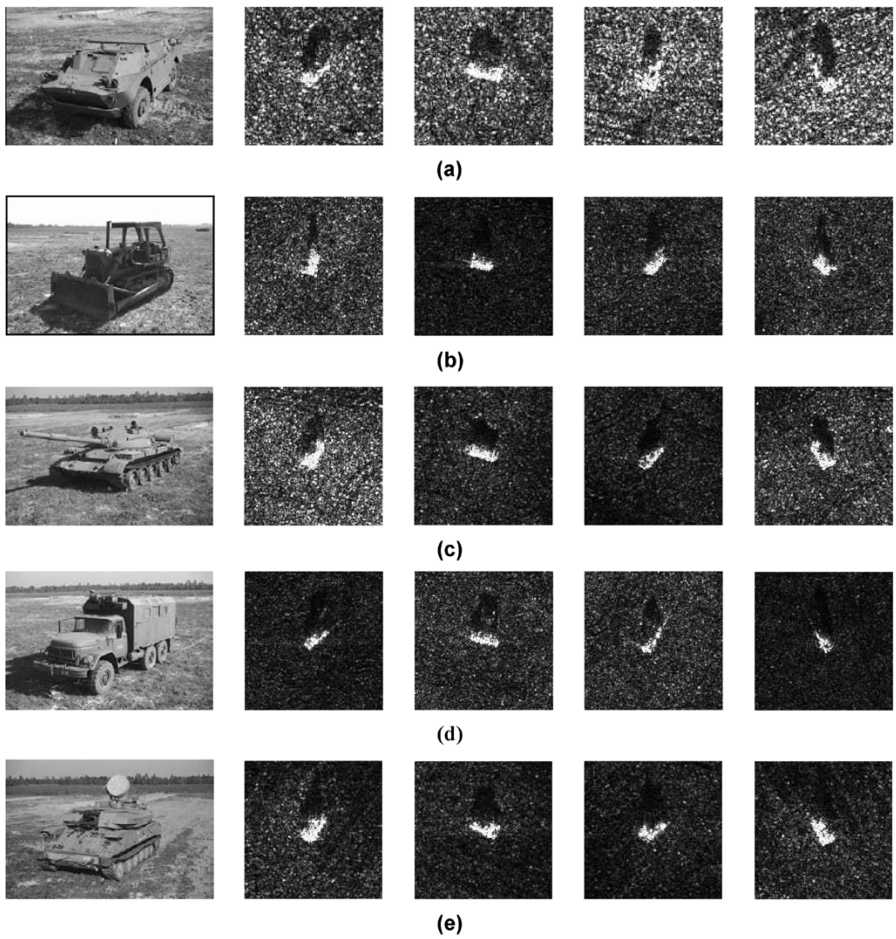


Fig. 11. Five objects used in recognition. (a) Optical and SAR images of BRDM2. (b) Optical and SAR images of D7. (c) Optical and SAR images of T62. (d) Optical and SAR images of ZIL. (e) Optical and SAR images of ZSU.

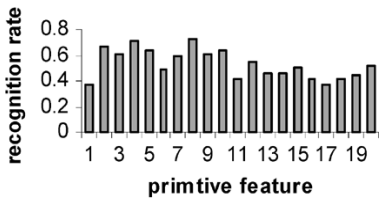


Fig. 12. Recognition rates of 20 primitive features.

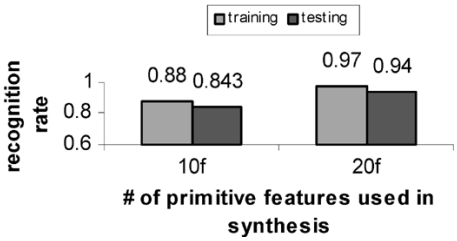


Fig. 13. Recognition rates with three subpopulations.

- **Experiment 3—Discriminate five objects:** With more objects added, the recognition becomes more difficult. This can be seen from Table VII and Fig. 17, which show the effectiveness of each primitive feature in discriminating these five objects. Feature blob mass (4) is the best primitive feature with recognition rate 0.49. If all the 20 primitive features from the training images are used jointly to form 20-dimensional primitive feature vectors to train a Bayesian classifier for recognition, the recognition rates on training and testing are 0.91 and 0.81, respectively; if only the ten common primitive features

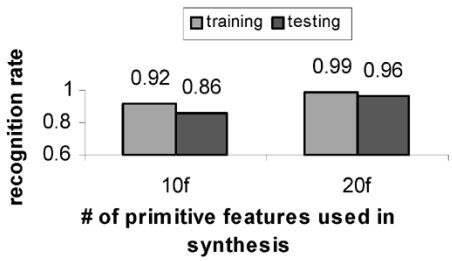


Fig. 14. Recognition rates with five subpopulations.

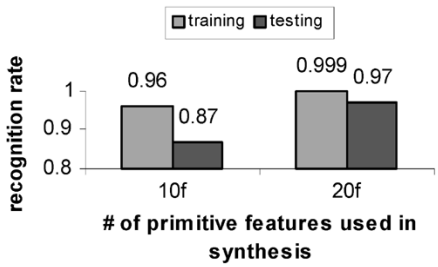


Fig. 15. Recognition rates with eight subpopulations.

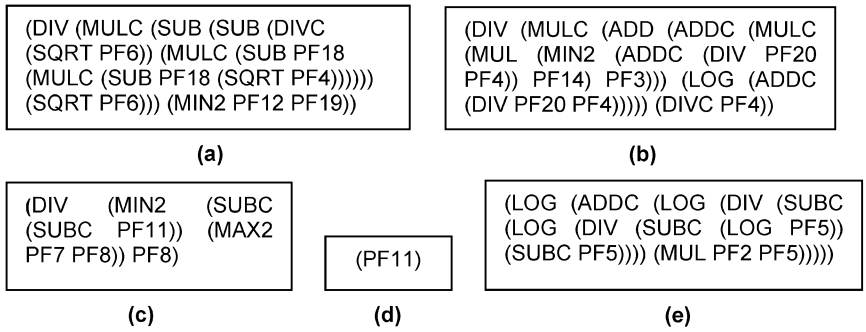


Fig. 16. Composite operator vector learned by CGP with five subpopulations. (a) Composite operator 1. (b) Composite operator 2. (c) Composite operator 3. (d) Composite operator 4. (e) Composite operator 5.

TABLE V
RECOGNITION RATES OF 20 PRIMITIVE FEATURES

| Feature Number | Primitive Feature | Recognition Rate | Feature Number | Primitive Feature | Recognition Rate |
|---|---|---|---|---|---|
| 1 | Standard deviation | 0.376 | 11 | Horizontal projection | 0.414 |
| 2 | Fractal dimension | 0.662 | 12 | Vertical projection | 0.545 |
| 3 | Weight-rank fill ratio | 0.607 | 13 | Major diagonal projection | 0.460 |
| 4 | Mass | 0.717 | 14 | Minor diagonal projection | 0.455 |
| 5 | Diameter | 0.643 | 15 | Minimum distance | 0.505 |
| 6 | rotational inertia | 0.495 | 16 | Maximum distance | 0.417 |
| 7 | Maximum CFAR | 0.588 | 17 | Mean distance | 0.376 |
| 8 | Mean CFAR | 0.726 | 18 | Moment $\mu_{20}$ | 0.421 |
| 9 | Percent bright CFAR | 0.607 | 19 | Moment $\mu_{02}$ | 0.443 |
| 10 | Count | 0.633 | 20 | Moment $\mu_{22}$ | 0.512 |

TABLE VI
PERFORMANCE OF COMPOSITE AND PRIMITIVE FEATURES ON THREE-OBJECT DISCRIMINATION

| Runs | Recognition Rate | | | |
|---|---|---|---|---|
| | 10f | | 20f | |
| | Training | Testing | Training | Testing |
| 3-dimensional composite feature vector | 0.880 | 0.843 | 0.969 | 0.943 |
| 5-dimensional composite feature vector | 0.921 | 0.857 | 0.990 | 0.961 |
| 8-dimensional composite feature vector | 0.962 | 0.870 | 0.999 | 0.970 |
| primitive feature vector | 0.863 | 0.812 | 0.995 | 0.962 |

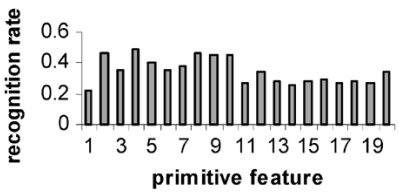


Fig. 17. Recognition rates of 20 primitive features.

are used, the recognition rates on training and testing data are 0.74 and 0.62, respectively. Composite features built on the primitive features 11 to 20 are not very effective, since these ten primitive features are common features and are not designed with the characteristics of SAR images taken into consideration.

Two series of experiments are performed in which CGP maintains five and eight subpopulations to evolve 5-D and 8-D composite feature vectors for recognition. The primitive features used in the experiments are 20 primitive features and ten common primitive features. The max-operator-size is 20. The average recognition rates of 5-D and 8-D composite feature vectors over ten runs are shown in Table VIII and Fig. 18. The left two bins in columns 10f and 20f correspond to five subpopulations and the right two bins correspond to eight subpopulations. The bins showing the training results are to the left of those showing the testing results. The numbers above the bins are the average recognition rates over ten runs.

From Fig. 18, we can see that when the dimension of the composite feature vector is 8, the performance of the composite features is good and it is better than using all 20 (0.81) or ten(0.62) primitive features upon which the composite features are built. When the dimension of the composite feature vector is 5, the recognition is not satisfactory when using just ten common features as building blocks. Also, when the dimension is 5, the average performance is a little bit worse than using all 20 or ten primitive features, but the dimension of the composite feature vector

TABLE VII
RECOGNITION RATES OF 20 PRIMITIVE FEATURES

| Feature Number | Primitive Feature | Recognition Rate | Feature Number | Primitive Feature | Recognition Rate |
|---|---|---|---|---|---|
| 1 | Standard deviation | 0.224 | 11 | Horizontal projection | 0.273 |
| 2 | Fractal dimension | 0.473 | 12 | Vertical projection | 0.343 |
| 3 | Weight-rank fill ratio | 0.361 | 13 | Major diagonal projection | 0.281 |
| 4 | Mass | 0.486 | 14 | Minor diagonal projection | 0.265 |
| 5 | Diameter | 0.404 | 15 | Minimum distance | 0.277 |
| 6 | rotational inertia | 0.346 | 16 | Maximum distance | 0.294 |
| 7 | Maximum CFAR | 0.379 | 17 | Mean distance | 0.266 |
| 8 | Mean CFAR | 0.471 | 18 | Moment $\mu_{20}$ | 0.277 |
| 9 | Percent bright CFAR | 0.449 | 19 | Moment $\mu_{02}$ | 0.267 |
| 10 | Count | 0.453 | 20 | Moment $\mu_{22}$ | 0.34 |

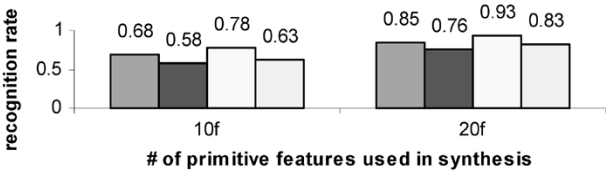


Fig. 18. Recognition rates with five (left two bins) and eight (right two bins) subpopulations.

TABLE VIII
PERFORMANCE OF COMPOSITE AND PRIMITIVE FEATURES ON FIVE-OBJECT DISCRIMINATION

| Runs | Recognition Rate | | | |
|---|---|---|---|---|
| | 10f | | 20f | |
| | Training | Testing | Training | Testing |
| 5-dimensional composite feature vector | 0.681 | 0.579 | 0.854 | 0.764 |
| 8-dimensional composite feature vector | 0.778 | 0.630 | 0.926 | 0.825 |
| primitive feature vector | 0.737 | 0.623 | 0.914 | 0.812 |

is just one-fourth or half of the number of primitive features, saving a lot of computational burden in recognition. When all the 20 primitive features are used and CGP has eight subpopulations, the composite operators in the best composite operator vector evolved have sizes 19, 1, 16, 19, 15, 7, 16, and 6, respectively and they are shown in Fig. 19. The primitive features used by the synthesized composite operator vector are primitive features 2, 3, 4, 5, 8, 9, 10, 11, 12, 13, 14, 15, 16, 18, 19, and 20. If all these 16 primitive features from the training images directly form 16-dimensional primitive feature vectors to train a Bayesian classifier for recognition, the recognition rate is 0.80 on the testing images, which is lower than the average performance of the composite feature vector shown in Fig. 18.

### C. Comparison With Other Classification Algorithms

In Sections IV-A and B, the effectiveness of CGP-learned composite features is reported and compared with that of original primitive features. The comparison shows that CGP-learned composite features are more effective in object recognition. In this section, the performance of CGP-based approach proposed in this paper is compared with other four classical classification algorithms: multilayer feedforward neural networks trained with: a) backpropagation algorithm; b) stochastic backpropagation algorithm; c) stochastic backpropagation algorithm with momentum; and d) C4.5 classification algorithm. For a detailed description of these algorithms, please refer to [13] and [14].

Multilayer feed forward neural networks used in this paper have three layers—the output layer, the hidden layer and the input layer. The output layer has only one output node and the hidden layer has three, five, or eight nodes. A node of input layer contains a primitive feature and the number of nodes in the input layer is equal to the number of primitive features used in the recognition. The activation function of nodes in the output and hidden layers is

$$f(x) = a\frac{e^{\mathrm{bx}} - e^{-\mathrm{bx}}}{e^{\mathrm{bx}} + e^{-\mathrm{bx}}} \quad \text{where} \quad a = 1.716 \quad \text{and} \quad b = \tfrac{2}{3}. \quad (7)$$

The inputs to the neural networks are normalized primitive features. The primitive features from training images and testing images are normalized separately. The normalization is performed by the following formula:

$$\mathrm{nf}_{ij} = \frac{f_{ij} - \mu_i}{\sigma_i} \quad i = 1, 2, \ldots \text{ or } 20, \quad j = 1, 2, \ldots, n_i \quad (8)$$

where $f_{ij}(j = 1, 2, \ldots, n_i)$ are the feature values of original primitive feature $i$ ($i = 1, 2, \ldots$ or 20) and $\mathrm{nf}_{ij}(j = 1, 2, \ldots, n_i)$ are the corresponding normalized feature values, $n_i$ is the number of training or testing images, $\mu_i$ and $\sigma_i$ are the mean and standard deviation of these $n_i$ feature values. The reason for feature normalization is that the values of some primitive features are very large, making the value of $e^{\mathrm{bx}}$ overflow. For the details, please see [13], [14]. The weight values of connections between nodes of different layers are initialized with small randomly generated real numbers in the range of $[-1.0, 1.0]$. The learning rate $\eta$ of backpropagation algorithms is 0.1 and the momentum $\alpha$ of stochastic backpropagation algorithm with momentum is 0.5. Backpropagation algorithms stop when they finish 300 weight-update loops or when the recognition rate on training data is above 0.9, whichever occurs first. The best recognition rate and its associated weight values are kept from loop to loop, and the trained neural network (the one with the best recognition rate) is applied to the testing data. In each experiment, backpropagation algorithms are invoked ten times with the same parameters and input data to train a neural network. For the purpose of objective comparison, only the average results over ten runs are reported. The original backpropagation algorithm sometimes constructs a neural network with very bad performance (below

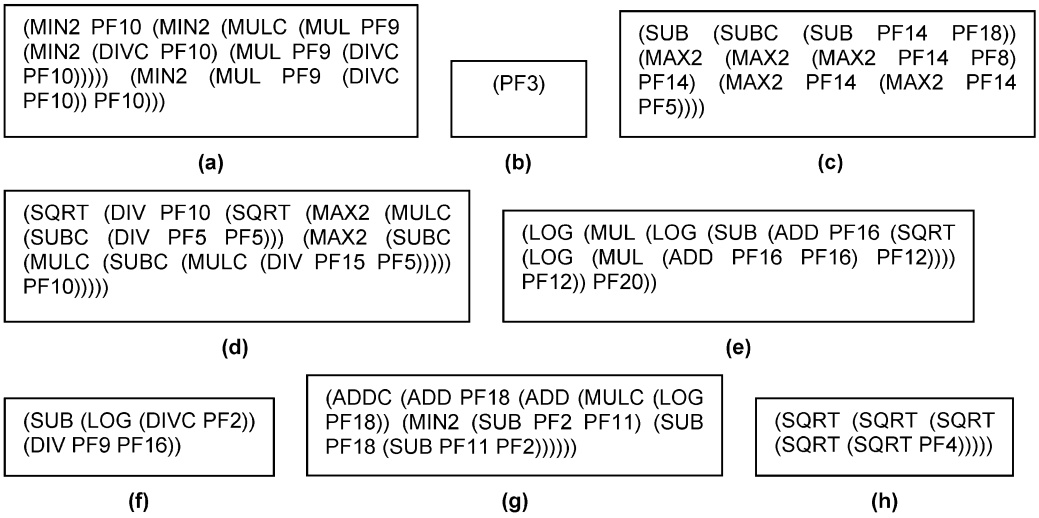


Fig. 19. Composite operator vector learned by CGP. (a) Composite operator 1. (b) Composite operator 2. (c) Composite operator 3. (d) Composite operator 4. (e) Composite operator 5. (f) Composite operator 6. (g) Composite operator 7. (k) Composite operator 8.

TABLE IX
AVERAGE RECOGNITION PERFORMANCE OF MULTILAYER NEURAL NETWORKS TRAINED BY BACKPROPAGATION ALGORITHM (THREE OBJECTS)

| Number of hidden nodes | Recognition rate | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Backpropagation | | | | Backpropagation - Stochastic | | | | Backpropagation - Momentum | | | |
| | 10f | | 20f | | 10f | | 20f | | 10f | | 20f | |
| | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test |
| 3 | 0.548 | 0.526 | 0.634 | 0.623 | 0.672 | 0.667 | 0.792 | 0.801 | 0.676 | 0.670 | 0.831 | 0.829 |
| 5 | 0.546 | 0.532 | 0.633 | 0.626 | 0.669 | 0.664 | 0.801 | 0.787 | 0.677 | 0.656 | 0.823 | 0.817 |
| 8 | 0.570 | 0.552 | 0.645 | 0.644 | 0.664 | 0.649 | 0.765 | 0.761 | 0.674 | 0.658 | 0.751 | 0.745 |

0.1) due to the gradient descent convergence to a poor local minimum point. We do not use the results from these runs in the calculation of average performance and invoke the backpropagation algorithm to perform training again.

The input to C4.5 algorithm is the original set of primitive features, not the normalized ones. For a particular primitive feature, if it has at most ten unique feature values among the feature values extracted from training images, it is treated as a discrete feature; otherwise, it is treated as a continuous feature [14]. Since C4.5 is a deterministic algorithm, it is invoked only once in each experiments.

Four experiments are performed: distinguishing between three objects using all the 20 primitive features or ten common primitive features; distinguishing between five objects using all the 20 primitive features or ten common primitive features. As previously stated, in each experiment, the backpropagation is invoked ten times to train ten neural networks, the average recognition rates of trained multilayer neural networks with three, five, and eight hidden layers are shown in Tables IX and X, where 10f means using the primitive features 11 to 20 and 20f means using all the primitive features. Tables IX and X show the performance on distinguishing three and five objects, respectively.

From the above tables, it can be seen that the CGP-based approach proposed in this paper outperforms backpropagation and C4.5 algorithms and C4.5 algorithm is more effective than backpropagation algorithms (Table XI). Stochastic backpropagation and stochastic backpropagation with momentum outperform the original backpropagation algorithm, since the original backpropagation algorithm is more likely to converge to some local minimum points, yielding a neural network with poor performance. According to our experiments, three hidden nodes are enough, increasing the number of hidden nodes to 5 or 8 does not increase the performance significantly. In fact, sometimes it decreases the recognition performance.

### *D. Discussions*

The above experiments (experiments 1, 2, and 3) demonstrate the following.

- It is important to introduce domain knowledge by defining the primitive features into the feature synthesis for object recognition. In these experiments, we compare the effectiveness of composite features built on the primitive features encoding domain knowledge (the characteristics of SAR imagery in this paper) and the effectiveness of composite features built only on common and domain-independent primitive features. The comparison shows that with primitive features encoding domain knowledge, more effective composite features can be generated in the feature synthesis. It is also observed from the experiments that with primitive features encoding domain knowledge, CGP can evolve effective composite features within a fewer number of generations, thus, improving the efficiency of CGP search.
- In general, the effectiveness of composite features learned by CGP is dependent on the effectiveness of primitive features. With more effective primitive features available, more effective composite features can be generated by

TABLE X
AVERAGE RECOGNITION PERFORMANCE OF MULTILAYER NEURAL NETWORKS TRAINED BY BACKPROPAGATION ALGORITHM (FIVE OBJECTS)

| Number of hidden nodes | Recognition rate | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Backpropagation | | | | Backpropagation - Stochastic | | | | Backpropagation - Momentum | | | |
| | 10f | | 20f | | 10f | | 20f | | 10f | | 20f | |
| | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test | Train | Test |
| 5 | 0.274 | 0.267 | 0.346 | 0.340 | 0.302 | 0.300 | 0.370 | 0.366 | 0.319 | 0.304 | 0.376 | 0.367 |
| 8 | 0.292 | 0.290 | 0.330 | 0.325 | 0.296 | 0.296 | 0.366 | 0.366 | 0.331 | 0.328 | 0.369 | 0.368 |

TABLE XI
RECOGNITION PERFORMANCE OF C4.5 CLASSIFICATION ALGORITHM

| 3 objects | | | | 5 objects | | | |
|---|---|---|---|---|---|---|---|
| 10f | | 20f | | 10f | | 20f | |
| Training | Testing | Training | Testing | Training | Testing | Training | Testing |
| 0.962 | 0.679 | 0.995 | 0.917 | 0.754 | 0.344 | 0.917 | 0.686 |

CGP. But this does not mean that ineffective primitive features are never used by CGP in the feature synthesis. As Fig. 10 shows, the ineffective feature (primitive feature 8) is used to synthesize effective composite features. The reason for the use of one or more ineffective primitive features in the synthesis of effective composite features is due to the interaction between primitive features. Although some features are ineffective when used alone, they can be elements of a primitive feature set that is the build block for effective composite features.

- CGP is a viable tool to synthesize effective composite features from primitive features for object recognition. In general, the synthesized composite features are more effective than the primitive features from which they are built, although there are a few exceptions in our experiments. In experiment 1, the learned composite features outperform the primitive features or any combination of primitive features upon which they are evolved, although the improvement in recognition rate is not significant when all the ten common primitive features are used to synthesize composite features, since the performance of these ten primitive features, when used together, is already very good (0.96 in training and 0.98 in testing). In experiment 2, when 8-D composite feature vectors are evolved or when only ten common primitive features are used in feature synthesize, the synthesized composite features are more effective. But when all the 20 primitive features are used and the dimension of composite feature vectors is five or three, the performance of primitive features is a little bit higher. However, the dimension of primitive feature vectors is 20, much higher than that of composite feature vectors, which is five or three. In experiment 3, when 8-D composite feature vectors are evolved, the synthesized composite features produce better recognition results. But if the dimension of composite feature vectors is 5, the 10- or 20-dimensional primitive feature vectors yield better performance. Since there is much randomness involved in CGP, we can still conclude that CGP may evolve composite features that are more effective than the primitive ones upon which they are evolved. More importantly, to achieve the same or similar recognition rate, the number of composite features needed is smaller than the number of primitive features needed (one-fourth or half), reducing the computational expense during run-time recognition. Thus, the composite features outperform the primitive ones.

## V. CONCLUSION

This paper investigates synthesizing composite features for object recognition. Our experimental results using real SAR images show that CGP can evolve composite features that are more effective than the primitive features upon which they are built, although sometimes the improvement in recognition rate may not be significant. To achieve the same recognition performance of primitive features, fewer composite features are needed and this reduces the computational burden during recognition. From the experimental results, it can be seen that primitive features that provide domain knowledge for the evolutionary process have a substantial impact on the goodness of the synthesized composite features. Although the effectiveness of synthesized composite features is not solely dependent on the effectiveness of primitive features, on the average, if primitive features do not capture the characteristics of the objects to be recognized, it is difficult for CGP to synthesize effective composite features. Thus, it is still very important to design effective primitive features. We cannot entirely rely on CGP to generate good features. However, designing effective primitive features needs human ingenuity. If human experts lack insight into the characteristics of the objects to be detected and recognized, they may not figure out effective primitive features. Currently, there is only one object in an image or a ROI during recognition, so all the features come from the same object. If there are multiple overlapped objects in an image or a ROI, the recognition becomes much more difficult. Some of the features of an object may not be available due to occlusion and we need to distinguish features from different objects before these features are fed into a classifier. Recognizing multiple overlapped objects is a challenging future research topic.

## REFERENCES

[1] J. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*. Cambridge, MA: MIT Press, 1994.
[2] C. Harris and B. Buxton, "Evolving edge detectors with genetic programming," in *Proc. Genetic Programming, 1st Annu. Conf.*, 1996, pp. 309–314.

[3] M. Ebner and A. Zell, "Evolving a task specific image operator," in *Proc. Evolutionary Image Analysis, Signal Processing and Telecommunications, 1st European Workshops, EvoIASP'99 and EuroEcTel'99*, 1999, pp. 74–89.
[4] R. Poli, "Genetic programming for feature detection and image segmentation," in *Evolut. Comput.*, T. C. Forgarty, Ed., 1996, pp. 110–125.
[5] B. Bhanu and Y. Lin, "Object detection in multi-modal images using genetic programming," *Appl. Soft Comput.*, vol. 4, no. 2, pp. 175–201, May 2004.
[6] D. Howard, S. C. Roberts, and R. Brankin, "Target detection in SAR imagery by genetic programming," in *Adv. Eng. Software*, May 1999, vol. 30, pp. 303–311.
[7] S. C. Roberts and D. Howard, "Evolution of vehicle detectors for infrared line scan imagery," in *Proc. Evolutionary Image Analysis, Signal Processing and Telecommunications, 1st Eur. Workshops, EvoIASP'99 and EuroEcTel'99*, 1999, pp. 110–125.
[8] S. A. Stanhope and J. M. Daida, "Genetic programming for automatic target classification and recognition in synthetic aperture radar imagery," in *Proc. Conf. Evolutionary Programming VII*, 1998, pp. 735–744.
[9] K. Krawiec and B. Bhanu, "Visual learning by evolutionary feature synthesis," in *Proc. Int. Conf. Machine Learning*, Washington, DC, Aug. 2003.
[10] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. New York: Academic, 1999.
[11] B. Bhanu and Y. Lin, "Genetic algorithm based feature selection for target detection in SAR images," *Image Vis. Comput.*, vol. 21, no. 7, pp. 591–608, 2003.
[12] D. Kreithen, S. Halversen, and G. Owirka, "Discriminating targets from clutter," *Lincoln Lab. J.*, vol. 6, no. 1, pp. 25–52, 1993.
[13] R. Duda, P. Hart, and D. Stork, *Pattern Recognition*, 2nd ed. New York: A Wiley-Interscience, 2001.
[14] T. Mitchell, *Machine Learning*. New York: McGraw-Hill, 1997.

**Yingqiang Lin** received the B.S. and M.S. degrees in computer science from Fudan University, Shanghai, China, in 1991 and 1994, respectively, and the Ph.D. degree in computer science from the University of California, Riverside, in June 2003.

Currently, he is a Postdoctoral Researcher at the University of California, Riverside, working in the area of computer graphics. He is a co-author of a book on *Synthesis of Pattern Recognition Systems* (New York: Springer, 2005). His research interests include image processing, pattern recognition, and machine learning.

**Bir Bhanu** (S'72–M'82–SM'87–F'95) received the S.M. and E.E. degrees in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, the Ph.D. degree in electrical engineering from the Image Processing Institute, University of Southern California, Los Angeles, and the M.B.A. degree from the University of California, Irvine.

Dr. Bhanu was the founding Professor of electrical engineering and served its first Chair at the University of California, Riverside (UCR). He has been the Cooperative Professor of Computer Science and Engineering and Director of Visualization and Intelligent Systems Laboratory (VISLab) since 1991. He has also served as the founding Director of an interdisciplinary Center for Research in Intelligent Systems (CRIS) at UCR since 1998. Previously, he was a Senior Honeywell Fellow at Honeywell Inc., Minneapolis, MN. He was on the faculty of the Department of Computer Science, University of Utah, Salt Lake City, and has worked at Ford Aerospace and Communications Corporation, Newport Beach, CA; INRIA, Rocquencourt, France; and IBM San Jose Research Laboratory, San Jose, CA. He has been the principal investigator of various programs for the Defense Advanced Research Projects Agency, the National Aeronautics and Space Administration, the National Science Foundation, the Air Force Office of Scientific Research, the Army Research Office, and other agencies and industries in the areas of learning and vision, image understanding, pattern recognition, target recognition, biometrics, navigation, image databases, and machine vision applications. He is the co-author of the books *Evolutionary Synthesis of Pattern Recognition Systems* (New York: Springer, 2005), *Computational Algorithms for Fingerprint Recognition* (Boston, MA: Kluwer, 2004), *Genetic Learning for Adaptive Image Segmentation* (Kluwer, 1994), and *Qualitative Motion Understanding* (Kluwer, 1992), and the co-editor of the book *Computer Vision Beyond the Visible Spectrum* (New York: Springer, 2004).

Dr. Bhanu received two outstanding paper awards from the Pattern Recognition Society and has received industrial and university awards for research excellence, outstanding contributions, and team efforts. He has been on the editorial board of various journals and has edited special issues of several IEEE TRANSACTIONS and other journals. He holds 11 U.S. and international patents and over 230 reviewed technical publications in the areas of his interest. He was General Chair for IEEE Workshops on Applications of Computer Vision, Chair for the DARPA Image Understanding Workshop, General Chair for the IEEE Conference on Computer Vision and Pattern Recognition, and Program Chair for the IEEE Workshops on Computer Vision Beyond the Visible Spectrum. He is a Fellow of the American Association for the Advancement of Science, the International Association of Pattern Recognition, and the International Society for Optical Engineering.