Correspondence

Object Detection via Feature Synthesis Using MDL-Based Genetic Programming

Yingqiang Lin and Bir Bhanu, *Fellow, IEEE*

***Abstract*—In this paper, we use genetic programming (GP) to synthesize composite operators and composite features from combinations of primitive operations and primitive features for object detection. The motivation for using GP is to overcome the human experts' limitations of focusing only on conventional combinations of primitive image processing operations in the feature synthesis. GP attempts many unconventional combinations that in some cases yield exceptionally good results. To improve the efficiency of GP and prevent its well-known code bloat problem without imposing severe restriction on the GP search, we design a new fitness function based on minimum description length principle to incorporate both the pixel labeling error and the size of a composite operator into the fitness evaluation process. To further improve the efficiency of GP, smart crossover, smart mutation and a public library ideas are incorporated to identify and keep the effective components of composite operators. Our experiments, which are performed on selected training regions of a training image to reduce the training time, show that compared to normal GP, our GP algorithm finds effective composite operators more quickly and the learned composite operators can be applied to the whole training image and other similar testing images. Also, compared to a traditional region-of-interest extraction algorithm, the composite operators learned by GP are more effective and efficient for object detection.**

***Index Terms*—Feature learning, minimum description length (MDL), primitive feature image, primitive operator, synthetic aperture radar (SAR) image.**

I. Introduction

Automatic object detection is one of the important steps in image processing and computer vision [1]. The major task of object detection is to locate objects in images and extract the regions containing them. The extracted regions are called regions-of-interest (ROIs). The quality of object detection is highly dependent on the effectiveness of the features used in detection. Finding or designing appropriate features to capture the characteristics of objects and building the feature-based representation of objects are the key to the success of detection. However, there are many features that can be extracted from images. What are the appropriate features or how to synthesize features useful for detection from the features extracted from images? Usually, it is not easy for human experts to figure out a set of features to characterize complex objects, and sometimes, simple features (also called primitive features in this paper) directly extracted from images may not be effective in object detection. At this point, synthesizing composite features, useful for the current detection task, from those simple ones becomes imperative.

In this paper, we use genetic programming (GP) to synthesize composite features, which are the output of composite operators, to perform object detection. Genetic algorithms (GAs) and GP are search algorithms inspired by the mechanism used by nature to evolve species [2], [3]. GP is an extension of GA. GA manipulates a population of individuals represented by fixed-length bit strings through selection, crossover, and mutation operations. In GP, individuals can be much more complicated structures such as trees and graphs. Thus, to apply GA and GP, we must decide how to represent potential solutions by bit strings or complicated tree or graph structures, define the fitness of individuals and the crossover and mutation operations on them so that GA and GP can evaluate and adapt individuals to improve their performance. In this paper, composite operators are individuals in the population and they are represented by binary trees whose internal nodes are the pre-specified primitive operators and the leaf nodes are the primitive feature images (generated from the original image). It can be viewed as a way of combining primitive operations on images. The basic approach is to apply a composite operator on primitive feature images, segment the composite operator output, called composite feature image, to obtain a binary image and use the binary image to extract the region containing the object from the original image. We design a new fitness function based on the minimum description length (MDL) principle [4] that incorporates the size of composite operators into the fitness evaluation process to address the well-known code bloat problem of GP without imposing severe restriction on the GP search. Our experiments show that the MDL-based fitness function balances the above two conflicting factors well. To improve the efficiency of GP, we also design smart crossover, smart mutation and a public library to identify and keep the effective components of composite operators based on the analysis of the interactions among the nodes of composite operators.

Manuscript received August 5, 2003; revised May 20, 2004 and September 30, 2004. This work was supported in part by Air Force Research Laboratory 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 Associate Editor J. Peng.

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/TSMCB.2005.846656

II. Motivation and Related Research

GP has a well-known code bloat problem in which the sizes of individuals become larger and larger. In traditional GP with individuals represented by tree structures, a crossover operation is performed by swapping randomly selected subtrees between parents, and one of mutation operations is performed by substituting a randomly selected subtree with another randomly generated tree. 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 and mutation are performed in this simple way. Thus, as GP proceeds, the size of composite operators in the population becomes larger and larger. It takes a long time to execute a large composite operator, greatly reducing the speed of GP. Also, large-size composite operators may overfit training data by approximating the noise in images. Finally, large composite operators take up a lot of computer memory.

Usually in traditional GP, a limit on the size of composite operators is established when performing crossover or mutation. If the size of an offspring exceeds the size limit, the crossover or mutation operation is performed again until the sizes of both offspring are within the limit. Although this simple method prevents the code bloat, the size limit may greatly restrict the search performed by GP, since after randomly selecting a crossover point in one composite operator, GP cannot select some nodes of the other composite operator as crossover point in order to guarantee that both offspring do not exceed the size limit. Also, the size limit restricts the size of trees used to replace subtrees in mutation. However, the composite operator space is huge, and to find effective composite operators, GP must search extensively. Restricting the search greatly reduces the efficiency of GP, making it less likely to find good composite operators. Finally, with little knowledge of the composite operator space and the object characteristics, it is very difficult to determine the appropriate size limit to prevent code bloat and

TABLE I
SEVENTEEN PRIMITIVE OPERATORS

Operator	Description
ADD (A, B)	Add images A and B.
SUB (A, B)	Sub image B from A.
MUL (A, B)	Multiply images A and B.
DIV (A, B)	Divide image A by image B (If the pixel in B has value 0, the corresponding pixel in the resultant image takes the maximum pixel value in A).
MAX2 (A, B)	The pixel in the resultant image takes the larger pixel value of images A and B.
MIN2 (A, B)	The pixel in the resultant image takes the smaller pixel value of images A and B.
ADDC (A)	Increase each pixel value by c.
SUBC (A)	Decrease each pixel value by c.
MULC (A)	Multiply each pixel value by c.
DIVC (A)	Divide each pixel value by c.
SQRT (A)	For each pixel with value v, if v ≥ 0, change its value to $\sqrt{v}$. Otherwise, to $-\sqrt{-v}$.
LOG (A)	For each pixel with value v, if v ≥ 0, change its value to ln(v). Otherwise, to –ln(-v).
MAX (A)	Replace the pixel value by the maximum pixel value in a 3×3, 5×5 or 7×7 neighborhood.
MIN (A)	Replace the pixel value by the minimum pixel value in a 3×3, 5×5 or 7×7 neighborhood.
MED (A)	Replace the pixel value by the median pixel value in a 3×3, 5×5 or 7×7 neighborhood.
MEAN (A)	Replace the pixel value by the average pixel value of a 3×3, 5×5 or 7×7 neighborhood.
STDV (A)	Replace the pixel value by the standard deviation of pixel values in a 3×3, 5×5 or 7×7neighborhood.

overfitting while still allowing the resulting composite operators to capture the characteristics of objects. One may suggest that let GP perform crossover twice and keep the offspring of smaller size in each crossover. This method can enforce the size limit and prevent the size of offspring composite operators from growing large. Suppose the sizes of parents are s1 and s2, the sizes of offspring are s3 and s4, and $s1 < s2$ and $s3 < s4$. There are three possibilities: if $s2 < s4$, then $s1 > s3$; if $s2 = s4$, then $s1 = s3$; if $s2 > s4$, then $s1 < s3$ and $s3 < s4 < s2$. It can be seen that the smaller offspring has a size smaller than that of the larger parent. So after two crossovers and keeping the smaller offspring in each crossover, the size of two offspring cannot exceed the size limit and gradually the size of individuals in the population will be reduced. But GP now only searches the space of these small composite operators, which may not capture the characteristics of the objects to be detected. To avoid restricting the GP search without causing code bloat is the key to the success of GP search.

The minimum description length principle has been used in computer vision, pattern recognition and machine learning to find simple effective solutions and avoid overfitting in the training. Quinlan and Rivest [5] explore the use of the minimum description length principle for the construction of decision trees. The MDL principle defines the best decision tree to be the one that yields the minimum combined length of the decision tree itself plus the description of the misclassified data items. Their experimental results show that MDL provides a unified framework for both growing and pruning the decision tree, and these trees seem to compare favorably with those created by other techniques such as C4.5 algorithm. Gao *et al.* [6] use MDL principle to determine the best model granularity such as the sampling interval between the adjacent sampled points along the curve of Chinese characters or the number of nodes in the hidden layer of a three layer feed-forward neural network. Their experiments show that in these two quite different settings the theoretical value determined using the MDL principle coincides with the best value found experimentally. In this paper, a fitness function is designed based on MDL principle [4] to take the size of a composite operator into the fitness evaluation process. According to MDL principle, large composite operators effective on training regions may not have good fitness and will be culled out by selection, taking off the restriction on crossover and mutation while preventing composite operators from growing too large.

III. TECHNICAL APPROACH

To apply GP to evolve composite operators, we must define: primitive features and primitive operators to form composite operators, the fitness function to evaluate synthesized composite operators, the search operators such as selection, crossover and mutation to explore the composite operator space, the parameters and the termination criterion to control the GP run.

1) The Set of Primitive Features and Primitive Operators: There are sixteen primitive features [7]: the first one is the original image (PFIM0); the others are mean (PFIM1-3), deviation (PFIM4-6), maximum (PFIM7-9), minimum (PFIM10-12), and median (PFIM13-15) images obtained by applying templates of sizes 3×3, 5×5, and 7×7, respectively. These images are the input to composite operators. GP determines which operations are applied on them and how to combine the results. Note that although primitive features are very simple in this paper, in general, they can be any complicated features. They are primitive [7] compared to the composite features synthesized by GP.

A primitive operator takes one or two input images, performs a primitive operation on them and stores the result in a resultant image. Currently, 17 primitive operators are used by GP to compose composite operators, as shown in Table I, where A and B are input images of the same size and c is a constant stored in some primitive operators. For operators such as ADD, SUB, MUL, etc. that take two images as input, the operations are performed on a pixel-by-pixel basis. In the operators MAX, MIN, MED, MEAN, and STDV, 3×3, 5×5, and 7×7 neighborhood are used with equal probability. The neighborhood size of these operators in a node of a binary tree is determined randomly by GP when the binary tree is initialized, and this neighborhood does not change until the primitive operator the node contains is changed or the node itself is deleted in the later mutation operations. Thus, two nodes may contain the same primitive operator with different sized neighborhoods in two different composite operators. Even in the same composite operator, the same primitive operator may appear more than once with different size of neighborhood as a result of initialization, crossover, and mutation during the evolutionary process. It is worth noting that the primitive features and primitive operators are basic, easy to compute and domain-independent (not specific to a kind of imagery), so our method can be applied to a wide variety of images.

2) MDL-Based Fitness Function: To address the well-known code bloat problem and prevent severe restriction on the GP search, a MDL-based fitness function is designed to incorporate the composite operator size into the fitness evaluation. The fitness of a composite operator is defined as the sum of the description length of the composite operator and the description length of the segmented training regions with respect to this composite operator as a predictor for the label (object or background) of each pixel in the training regions. Both lengths are measured in bits. The tradeoff between the simplicity and complexity of composite operators is that if the size of a composite operator is too

small, it may not capture the characteristics of objects, on the other hand, if the size is too large, the composite operator may overfit the training image, thus performing poorly on the unseen testing images. With the MDL-based fitness function, the composite operator with the minimum combined description lengths of both the operator itself and image-to-operator error is the best composite operator and may perform best on unseen testing images. Based on the MDL principle, we propose the following *fitness function* for GP to maximize:

$$\mathrm{F}(\mathrm{CO}_i) = -(r \times \log(\mathrm{N_{po}}) \times \mathrm{Size}(\mathrm{CO}_i) + (1-r) \times (n_o + n_b) \times (\log(W_{\mathrm{im}}) + \log(H_{\mathrm{im}}))) \quad (1)$$

where $\mathrm{CO_i}$ is the ith composite operator in the population, $\mathrm{N_{po}}$ is the number of primitive operators (including primitive feature images) available for GP to synthesize composite operators, $\mathrm{Size}(\mathrm{CO}_i)$ is the size of the composite operator which is the number of nodes in the binary tree representing it, $\mathrm{n_o}$ and $\mathrm{n_b}$ are the number of object and background pixels misclassified, $\mathrm{W_{im}}$ and $\mathrm{H_{im}}$ are the width and height of the training image and r is a parameter determining the relative importance of the composite operator size and the detection rate, which is 0.7 in this paper. The value 0.7 is selected experimentally. In our experiments, we find 0.7 is an appropriate value to balance the composite operator size and its performance. Note that the first term $(\log(\mathrm{N_{po}}) \times \mathrm{Size}(\mathrm{CO_i}))$ of the fitness function is the description length of a composite operator. The description length is the number of bits needed to encode a composite operator, and the size of a composite operator is the number of nodes in the composite operator. Note that the description length is linearly related to the size of a composite operator.

We now give a brief explanation of this fitness function. Suppose a sender and a receiver both have the training image and the training regions and they agree in advance that the composite operators can be use to locate the object in the image, that is, to determine the label (object or background) of each pixel in the training regions. But only the sender knows the ground truth (the label of each pixel). Now, the sender wants to tell the receiver which pixels belong to the object and which pixels belong to the background. One simple approach to do this is to send a bit sequence of n (n is the number of pixels in the training regions) bits where 1 represents the object and 0 represents background, provided that both the sender and receiver know the order of the training regions and they agree that the pixels are scanned in the top-to-bottom and left-to-right fashion. However, n is usually very large, thus the communication burden is very heavy. To reduce the number of bits to be transmitted, the sender can send the composite operator to the receiver. Then the receiver applies the composite operator on the training regions to get segmented training regions. When sending the composite operator, the sender can send its nodes in a pre-traversal order. Given $\mathrm{N_{po}}$ primitive operators (including primitive features), $\log(\mathrm{N_{po}})$ bits are needed to encode each node. Thus the cost of sending composite operator is $\log(\mathrm{N_{po}}) \times \mathrm{Size}(\mathrm{CO_i})$. However, some pixels may be misclassified by the composite operator. In order for the receiver to get the truth, the sender needs to tell the receiver which pixels are misclassified. Each pixel is represented by its coordinate in the image. If the width and height of the image are $\mathrm{W_{im}}$ and $\mathrm{H_{im}}$ respectively, then $\log(\mathrm{W_{im}}) + \log(\mathrm{H_{im}})$ bits are needed to encode each pixel. Thus, the cost of sending the misclassified pixels is $(\mathrm{n_o} + \mathrm{n_b}) \times (\log(\mathrm{W_{im}}) + \log(\mathrm{H_{im}}))$. If the composite operator is very effective and its size is not too large, then only few pixels are misclassified and the number of bits to send is much smaller than n.

We define the *goodness* of a composite operator $\mathrm{CO_i}$ as

$$\mathrm{goodness}(\mathrm{CO}_i) = n(G \cap G')/n(G \cup G') \quad (2)$$

where G and G' are foregrounds in the ground-truth image and the resultant image of a composite operator respectively and $n(X)$ denote the number of pixels within the intersection of region X and the training region(s). It measures how the ground-truth and the detection results are overlapped. If G is the output image of another node, rather than the root node, of a composite operator, it measures the goodness of that particular node. Note that a composite operator has a higher fitness than another composite operator does not necessarily mean that it has a higher goodness, since its size may be much smaller than the composite operator with lower fitness.

3) Operators, Parameters, and Termination: The selection operation selects composite operators from the current population. We use tournament selection with tournament size 5. In traditional GP, a crossover or mutation point is randomly selected in each of two parent composite operators, leading to the disruption of effective components in the later stage of GP search. In this paper, smart crossover and smart mutation are used to identify effective components of a composite operator so as to prevent disrupting them by carefully choosing the crossover and mutation points. The identified effective components are kept in a public library of size 100 for later reuse. If the library is full, a new effective component replaces the worst one in the library if the new one is better than the replaced one. To perform *smart crossover* and *smart mutation* [8], GP evaluates the performance (goodness) of the internal nodes of a composite operator and analyzes the interactions between them. All this information is used to identify effective components so as to carefully select crossover and mutation points to avoid breaking them. For the details on smart crossover and smart mutation, the reader is referred to [8].

In this paper, the traditional GP with random operators (random crossover and random mutation) and driven by goodness function defined in (2) is called normal GP. The GP with smart operators (smart crossover and smart mutation) and driven by MDL-based fitness function proposed in this paper is called smart GP.

The key parameters are the population size M, the number of generation N, the crossover rate, the mutation rate and the goodness threshold. The GP stops whenever it finishes the pre-specified number of generations or whenever the best composite operator in the population has goodness value greater than the goodness threshold.

4) Steady-State and Generational Genetic Programming: As in [8], generational genetic programming and steady-state genetic programming are used to synthesize composite operators. The major difference is that in generational GP, the offspring from crossover are kept aside and do not participate in the crossover operations on the current population. The current population is not changed during crossover. But in steady-state GP, the offspring from crossover are evaluated and replace the worst individuals in the population immediately, and they participate in the following crossover operations on the current population. In smart GP, MDL-based fitness function is used, both smart and random GP operators are invoked and a public library is set up to store the effective components. Also, we adopt an elitism replacement method to keep the best composite operator from generation to generation.

IV. Experimental Results

Various experiments were performed to test the efficacy of genetic programming in extracting regions of interest from real synthetic aperture radar (SAR) images. The size of SAR images is 128×128, except the tank SAR images whose size is 80×80. In this paper, GP is applied only to a region or regions carefully selected from the training image to generate the composite operators. The generated composite operator is then applied to the whole training image and some other testing images to evaluate it. The advantage of performing training on a small selected region is that it can greatly reduce the training time, making it practical for the GP system to be used as a subsystem of

TABLE II
PERFORMANCE OF THE BEST COMPOSITE OPERATORS FROM NORMAL AND SMART GPS

	Normal GP					Smart GP				
	Road	Lake	River	Field	Tank	Road	Lake	River	Field	Tank
GP	G	S	S	G	G	G	S	S	G	G
F_{BI}						-2303.6	-1585.5	-2480.8	-7936.2	-807.2
F_{BF}						-325.4	-158.9	-404.6	-1999.4	-190.8
G_{BI}	0.60	0.62	0.59	0.52	0.65	0.45	0.55	0.23	0.39	0.54
G_{BF}	0.94	0.99	0.89	0.78	0.88	0.94	0.97	0.90	0.79	0.89
PF	0.90*, 0.90, 0.93	0.95*, 0.97	0.72*, 0.83	0.88*, 0.81	0.88*, 0.84	0.91*, 0.91, 0.93	0.94*, 0.98	0.71*, 0.86	0.90*, 0.84	0.89*, 0.84
Size	27	28	30	9	28	18	13	13	15	5
Time	5	15	33	8	3	2.6	1	19	12	2

GP – the type of GP used to synthesize composite operators. G: generational GP, S: steady-state GP.
F_{BI} – fitness of the best composite operator in the initial population (on training region).
F_{BF} – fitness of the best composite operator in the final population (on training region).
G_{BI} – goodness of the best composite operator in the initial population (on training region).
G_{BF} – goodness of the best composite operator in the final population (on training region).
PF – performance, the goodness of the ROI extracted by the best composite operator from training and testing images. * indicates the goodness of ROI extracted from the training image.
Size – size of the best composite operator.
Time – average running time (measured in seconds) of the best composite operator on training and testing images.

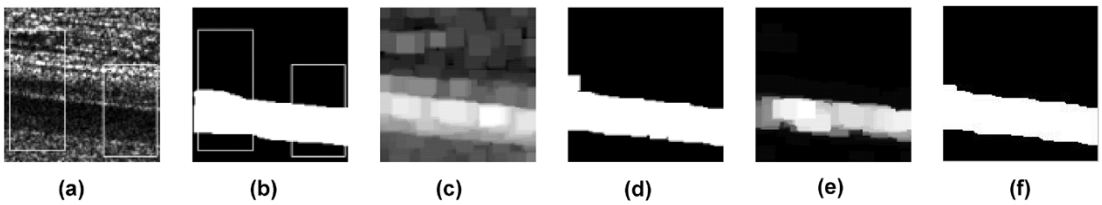

Fig. 1. Training SAR image containing road. (a) Paved road versus field. (b) Ground-truth. (c) Feature image (normal GP). (d) ROI extracted (normal GP). (e) Feature image (smart GP). (f) ROI extracted (smart GP).

other larger learning systems so as to improve the efficiency of GP by adapting its parameters dynamically. In each experiment, both normal GP and smart GP are applied. For the purpose of objective comparison, we invoke normal GP and smart GP with the same set of parameters and training regions ten times and report the results from the run in which GP finds the best composite operator among the best composite operators found in all ten runs. The parameters in the experiments are: population size (100), the number of generations (70), the goodness threshold value (1.0), the crossover rate (0.6), the mutation rate (0.05), and the segmentation threshold (0). For normal GP, the size limit of a composite operator is 30. Table II shows the performance of the best composite operators learned by GP on various SAR images.

In the following experiments, there is only one training image. Using a single training image may cause overfitting sometimes. For example, in the experiments with a complicated SAR image containing road, lake, field, tree, and shadow in [7], it is difficult for the learned program to extract road if only one image containing a road is used as the training image, since road and lake look somewhat similar in SAR images. To learn composite operators to discriminate lake and road, another image containing both road and lake is added in the training set.

1) Road Extraction: The training image contains horizontal paved road and field, as shown in Fig. 1(a). Two training regions are from (5, 19) to (50, 119) and from (82, 48) to (126, 124). Fig. 1(b) shows the ground-truth. The white region corresponds to the road and only the portion of ground-truth in the training regions is used in the fitness evaluation. Fig. 1(c)–(f) shows the performance of the learned composite operators on the training image. Testing images contain unpaved road versus field and vertical paved road versus grass, as shown in Fig. 2(a) and (f). Fig. 2 also shows the performance of the learned composite operators on testing images.

The best composite operator has 18 nodes and its depth is 13. It has three leaf nodes all containing 7×7 median image, which contains less speckles due to the median filter's effectiveness in eliminating speckle noise. It is shown in Fig. 3, where PFIM15 represents 7×7 median image. Compared to smart GP, the best composite operator from normal GP has 27 nodes and its depth is 16. Note that the best composite operator shown in Fig. 3 does not use MED primitive operator. MED is very effective in speckle noise elimination, so it is frequently selected by GP to build effective composite operators, but Fig. 3 shows that without it, GP may still generate effective composite operators. The interaction among primitive operators and primitive features is very complicated, indicating the high complexity of the search space structure and the difficulty of the feature synthesis process. Also, some combinations of other primitive operators and primitive feature images may approximate the function of MED primitive operator.

Fig. 4 shows how the average fitness of the best composite operators and the average fitness of the populations over all ten runs change as GP proceeds. In Fig. 4, the population fitness (average fitness of all the composite operators in the population) is much lower than that of the best composite operator even at the end of GP search. It is reasonable, since the selection of crossover points is not restricted by a hard size limit on composite operators. The difference between the sizes of the composite operators in the population is large and so are their fitness values. The population fitness is not important since only the best composite operator is used in testing. If GP finds one effective composite operator, the GP learning is successful. The large difference between the fitness of the best composite operator and that of the population indicates that the diversity of the population is maintained during GP search, which is very helpful in preventing premature convergence.

2) Lake Extraction: Two SAR images contain lake. The training image, shown in Fig. 5(a), contains a lake and field, and the testing image, shown in Fig. 6(a) contains a lake and grass. The training region is from (85, 85) to (127, 127). Fig. 5(b) shows the ground-truth.

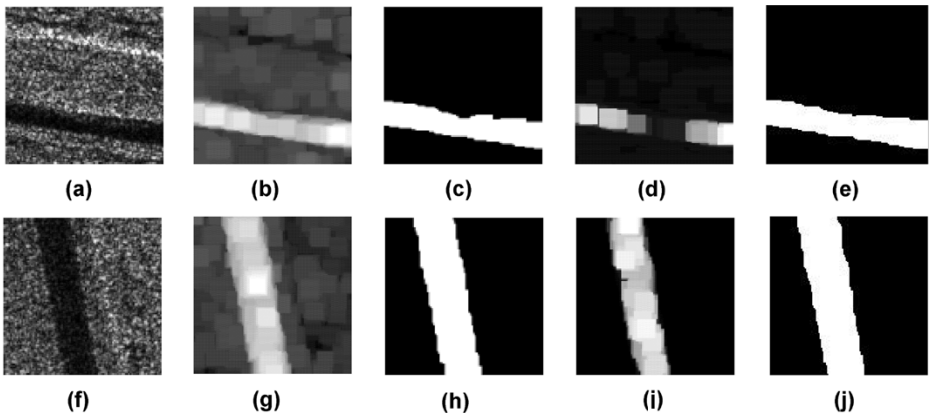

Fig. 2. Testing SAR images containing road. (a) Unpaved road versus field. (b) Feature image (normal GP). (c) ROI extracted (normal GP). (d) Feature image (smart GP). (e) ROI extracted (smart GP). (f) Paved road versus grass. (g) Feature image (normal GP). (h) ROI extracted (normal GP). (i) Feature image (smart GP). (j) ROI extracted (smart GP).

```
(MAX (MAX (MAX (MAX (MAX (SUBC
(MUL (DIVC (ADDC (MAX (MAX (MAX
(ADDC PFIM15)))))) (DIV PFIM15 (STDV
PFIM15)))))))))
```

Fig. 3. Learned composite operator tree in LISP notation.

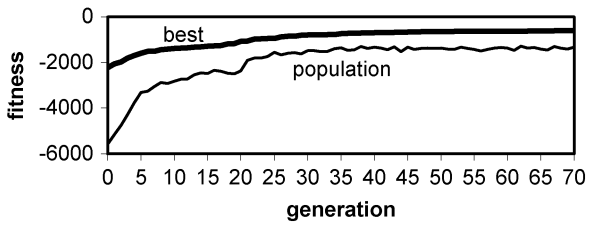

Fig. 4. Fitness versus generation (road versus field).

3) River Extraction: Two SAR images contain river and field. Fig. 7(a) and (b) shows the original training image and the ground-truth provided by the user. The white region in Fig. 7(b) corresponds to the river to be extracted. The training regions are from (68, 31) to (126, 103) and from (2, 8) to (28, 74). The testing SAR image is shown in Fig. 8(a). In both training and testing images, there are some islands along with the river around them that are not extracted, since these islands look similar to the field.

The best composite operator has 13 nodes and its depth is 12. It has one leaf node containing 3×3 mean image. Among 13 nodes, seven of them are MED operators effective in eliminating speckle noises. It is shown in Fig. 9. Compared to smart GP, the best composite operator from normal GP has 30 nodes with depth 23. Fig. 10 shows how the average fitness of the best composite operators and the average fitness of the populations over all ten runs change as GP searches the composite operator space.

4) Field Extraction: Two SAR images contain field and grass. Fig. 11(a) and (b) show the original training image and the ground-truth. The training regions are from (17, 3) to (75, 61) and from (79, 62) to (124, 122). Extracting field from a SAR image containing field and grass is considered as the most difficult task among the five experiments, since the grass and field are similar to each other and some small regions between grasses are actually fields.

From the experiment on the field extraction (see Figs. 11 and 12), we can see that the proposed algorithm has difficulties in dealing with textures and objects with great variations, and the reason lies in the fact that only domain-independent primitive operators and primitive features are used in the feature synthesis. The predefined primitive operators and primitive features have a significant impact on the performance of the learned composite operators. If texture-specific primitive operators and primitive features are included for the synthesis of composite operators, GP may learn effective composite operators in dealing with textures.

5) Tank Extraction: GP is applied to synthesize features for the detection of T72 tanks. Their SAR images are taken under different depression and azimuth angles and the size of the images is 80×80. The training image contains T72 tank under depression angle 17° and azimuth angle 135°, which is shown in Fig. 13(a). The training region is from (19, 17) to (68, 66). The ground-truth is shown in Fig. 13(b). The testing SAR image contains a T72 tank under depression angle 20° and azimuth angle 225°, which is shown in Fig. 14(a). The testing results are shown Fig. 14.

The best composite operator has five nodes and a depth of 4. It has one leaf node containing 3×3 maximum image. Two internal nodes are MED operator, which is useful in eliminating speckle noises in SAR images. It is shown in Fig. 15. Compared to smart GP, the best composite operator from normal GP has 28 nodes and its depth is 17. Fig. 16 shows how the average fitness of the best composite operators and the average fitness of the populations over all ten runs change as GP proceeds.

A. Comparison Between Normal GP and Smart GP

The comparison is based on the goodness of composite operators synthesized by normal and smart GPs. The reason for using goodness as a comparison metric is that a composite operator having higher fitness than another composite operator does not mean it always has a higher performance than the composite operator with lower fitness, since its size may be much smaller. For objective comparison, only the average performance over all ten runs is used for comparison.

Fig. 17 shows how the average goodness of the best composites operators improves as normal GP and smart GP proceed. The thick line stands for the goodness of smart GP and the thin line stands for the goodness of normal GP. It shows that if normal GP already achieves very good performance such as in the lake and tank cases, then it is difficult for smart GP to significantly improve the performance, since there is not much room for improvement. At this time, smart GP may achieve similar or a little better performance than normal GP. If

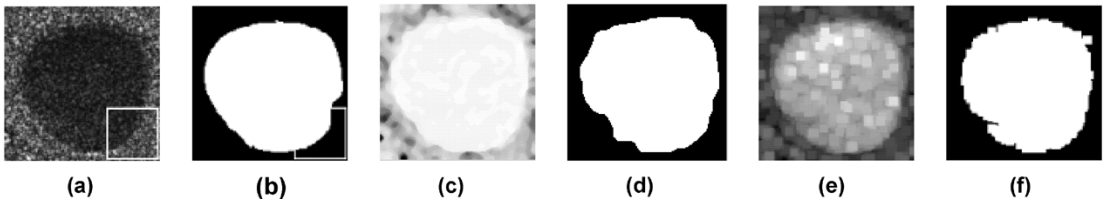

Fig. 5. Training SAR image containing lake. (a) Lake versus field. (b) Ground-truth. (c) Feature image (normal GP). (d) ROI extracted (normal GP). (e) Feature image (smart GP). (f) ROI extracted (smart GP).

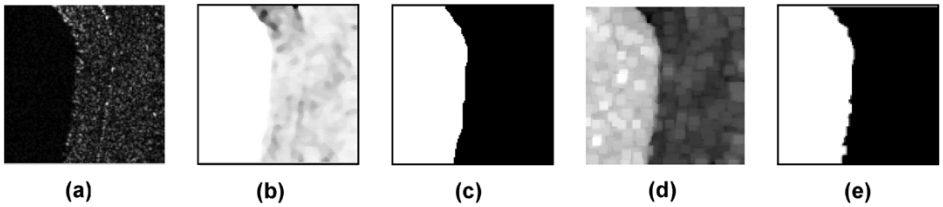

Fig. 6. Testing SAR image containing lake. (a) Lake versus grass. (b) Feature image (normal GP). (c) ROI extracted (normal GP). (d) Feature image (smart GP). (e) ROI extracted (smart GP).

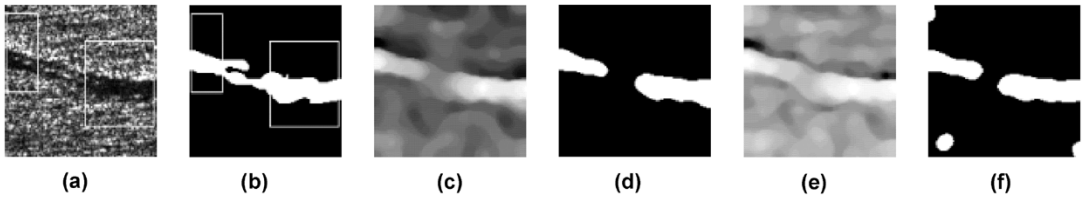

Fig. 7. Training SAR image containing river. (a) River versus field. (b) Ground-truth. (c) Feature image (normal GP). (d) ROI extracted (normal GP). (e) Feature image (smart GP). (f) ROI extracted (smart GP).

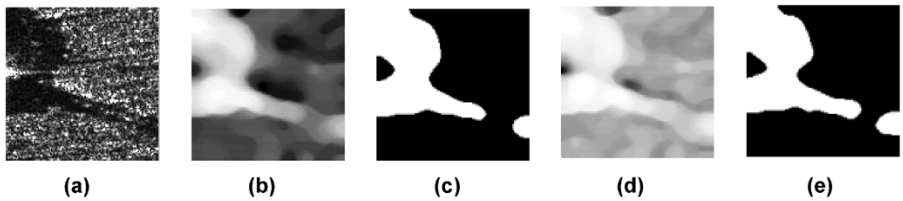

Fig. 8. Testing SAR image containing river. (a) River versus field. (b) Feature image (normal GP). (c) ROI extracted (normal GP). (d) Feature image (smart GP). (e) ROI extracted (smart GP).

```
(DIVC (SUBC (SUBC (MED (MED
(MIN (MED (MED (MED (MED (MED
(SUBC PFIM1)))))))))))
```

Fig. 9. Learned composite operator tree in LISP notation.

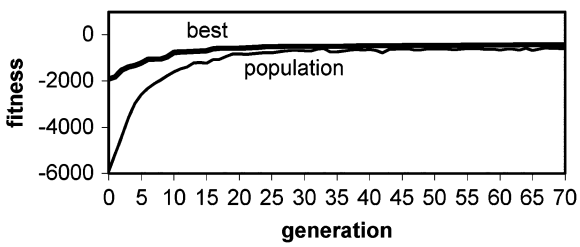

Fig. 10. Fitness versus generation (river versus field).

primitive operators and primitive features are not suitable to the tasks to be solved, both normal and smart GP may not generate effective composite operators such as in the field case, since primitive operators and primitive features have significant impact on the effectiveness of learned composite operators. The smart GP operators have an undesired side effect of restricting the GP search by biasing the selection of crossover and mutation points to keep the effective components generated during GP search. If effective components are generated and assembled together in some composite operators in the first 20 generations, then it is beneficial to apply the smart GP operators in the remaining generations to generate effective composite operators. Otherwise, it may be harmful to apply smart GP. From Fig. 17, it can be seen that on the average, smart GP finds good composite operators more quickly.

Table III shows the average goodness and standard deviation of the best composite operator in the initial and final populations. Table IV shows the average size of the best composite operators from normal GP and smart GP. It also shows the average performances of the best composite operators on the whole training image and other testing image(s). The standard deviations of the size and performance are also provided. It can be seen that although smart GP does not always generate composite operators with better performance, on the average, the best composite operators learned by smart GP have better performance and smaller size than those from normal GP, reducing the computational expense during testing.

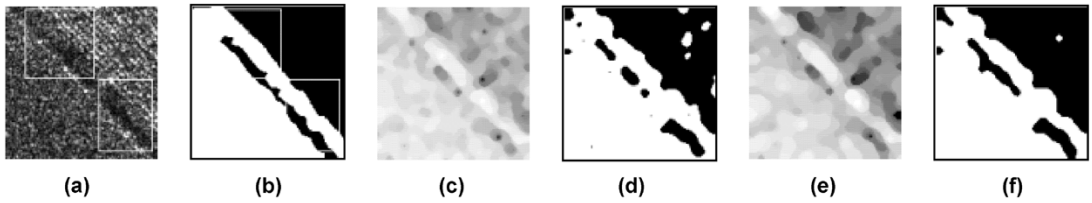

Fig. 11. Training SAR image containing field. (a) Field versus grass. (b) Ground-truth. (c) Feature image (normal GP). (d) ROI extracted (normal GP). (e) Feature image (smart GP). (f) ROI extracted (smart GP).

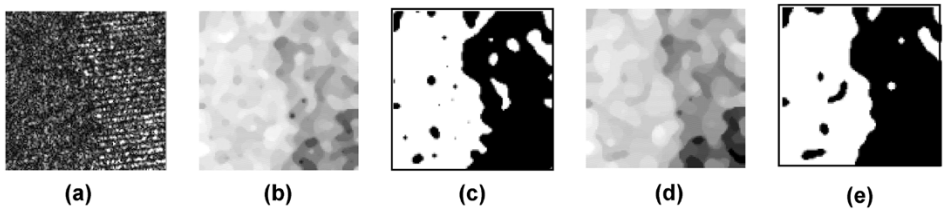

Fig. 12. Testing SAR image containing field. (a) Field versus grass. (b) Feature image (normal GP). (c) ROI extracted (normal GP). (d) Feature image (smart GP). (e) ROI extracted (smart GP).

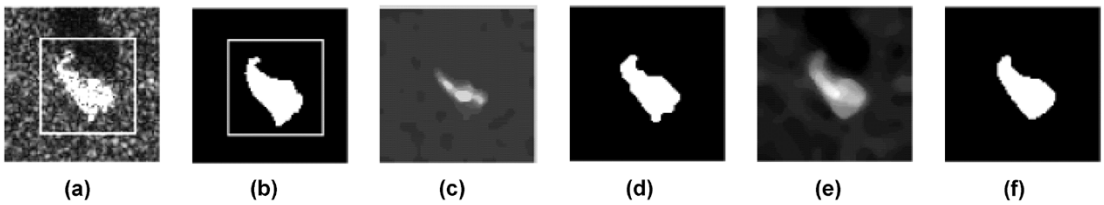

Fig. 13. Training SAR image containing tank. (a) T72 tank. (b) Ground-truth. (c) Feature image (normal GP). (d) ROI extracted (normal GP). (e) Feature image (smart GP). (f) ROI extracted (smart GP).

Fig. 14. Testing SAR image containing tank. (a) T72 tank. (b) Feature image (normal GP). (c) ROI extracted (normal GP). (d) Feature image (smart GP). (e) ROI extracted (smart GP).

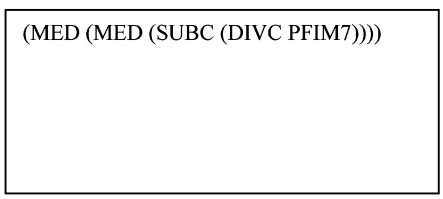

Fig. 15. Learned composite operator tree in LISP notation.

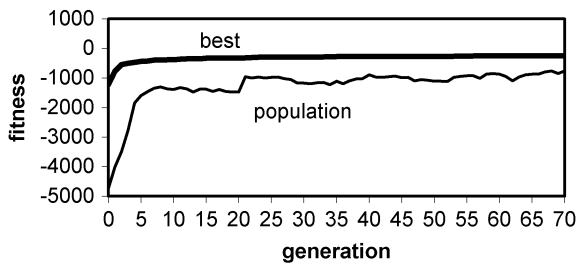

Fig. 16. Fitness versus generation (T72 tank).

Table V shows the average and standard deviation of running time of normal GP and smart GP. By intuition, the running time of smart GP should be much longer than that of normal GP, since in normal GP, only the output image of the root node is evaluated and smart GP evaluates the output image of each node of a composite operator. From Table V, it can be seen that the difference between the running times is not as much as expected. In the experiments with lake and tank images, the running time of smart GP is much shorter. The reason lies in the code bloat problem of GP. In normal GP, a size limit of composite operators (in this paper, it is 30) is specified. At the later stage of the GP search, most of the composite operators have size equal or close to the size limit. In smart GP, the MDL-based fitness function takes the size of composite operators into the fitness evaluation. The difference between the sizes of composite operators is large, even at the later stage of the GP search. Although a few composite operators have a size larger than the size limit in normal GP, many of them have size smaller than the size limit. If the size limit set in normal GP is large, it can be expected that the running time of the normal GP will be longer than that of smart GP. Also, in the above experiments, the goodness threshold value is set to 1.0 to force GP to finish the pre-specified number of generations. If the goodness threshold value is smaller than 1.0, smart GP may run fewer

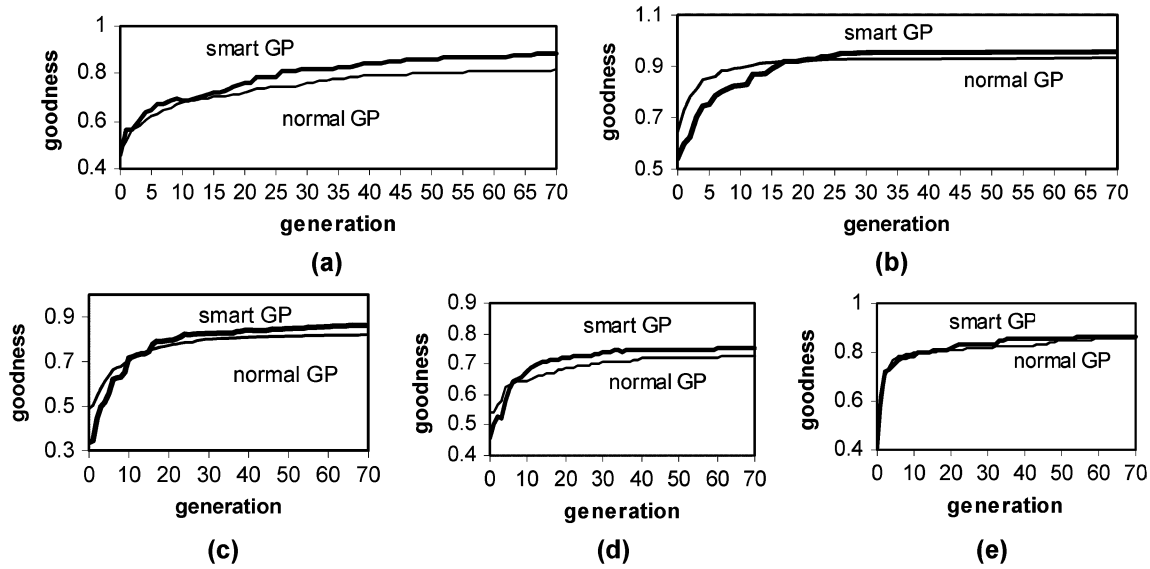

Fig. 17. The average goodness of the best composite operators versus generation. (a) Road. (b) Lake. (c) River. (d) Field. (e) Tank.

TABLE III
AVERAGE GOODNESS OF THE BEST COMPOSITE OPERATORS FROM NORMAL AND SMART GPS

		Normal GP					Smart GP				
		Road	Lake	River	Field	Tank	Road	Lake	River	Field	Tank
Initial	mean	0.473	0.645	0.490	0.539	0.490	0.467	0.544	0.337	0.459	0.411
	stdv	0.086	0.138	0.178	0.028	0.166	0.139	0.194	0.198	0.096	0.229
Final	mean	0.817	0.933	0.822	0.729	0.850	0.881	0.955	0.864	0.752	0.865
	stdv	0.078	0.091	0.040	0.070	0.024	0.042	0.017	0.029	0.031	0.009

TABLE IV
AVERAGE SIZE AND PERFORMANCE OF THE BEST COMPOSITE OPERATORS FROM NORMAL AND SMART GPS

		Normal GP					Smart GP				
		Road	Lake	River	Field	Tank	Road	Lake	River	Field	Tank
Size	mean	29.4	28.4	27.6	20.2	24.6	24.6	11.8	16.8	14.9	5.7
	stdv	1.07	1.74	4.43	8.93	6.17	4.58	5.65	7.19	9.98	1.9
Training	mean	0.789	0.891	0.583	0.794	0.829	0.860	0.916	0.650	0.839	0.849
	stdv	0.080	0.128	0.112	0.101	0.035	0.038	0.021	0.049	0.039	0.025
Testing	mean	0.620, 0.797	0.913	0.754	0.675	0.766	0.831, 0.914	0.972	0.836	0.784	0.821
	stdv	0.274, 0.151	0.161	0.129	0.124	0.042	0.115, 0.025	0.009	0.023	0.033	0.012

TABLE V
AVERAGE AND STANDARD DEVIATION OF RUNNING TIME (SECONDS) OF NORMAL GP AND SMART GP

	Road	Lake	River	Field	Tank
Normal GP	6915 (5348)	2577 (1213)	7951 (8006)	3606 (2679)	2686 (2163)
Smart GP	10249 (8993)	770 (724)	11035 (10310)	5251 (5506)	649 (589)

TABLE VI
AVERAGE GOODNESS OF THE BEST COMPOSITE OPERATORS FROM SMART GPS WITH AND WITHOUT THE PUBLIC LIBRARY

		Smart GP (Lib Size 100)		Smart GP (No Lib)	
		Road	Tank	Road	Tank
Training image	mean	0.860	0.849	0.800	0.820
	stdv	0.038	0.025	0.106	0.052
Testing image	Mean	0.831, 0.914	0.821	0.640, 0.812	0.782
	stdv	0.115, 0.025	0.012	0.113, 0.083	0.050

generations, since it finds effective composite operators more quickly, thus reducing its running time.

Table VI shows the average performance (goodness) of the best composite operators from smart GP with and without the public library on training and testing images containing road and tank over ten runs. From Table VI, it can be seen that with the public library to keep the effective components for later reuse, GP can generate more effective composite operators.

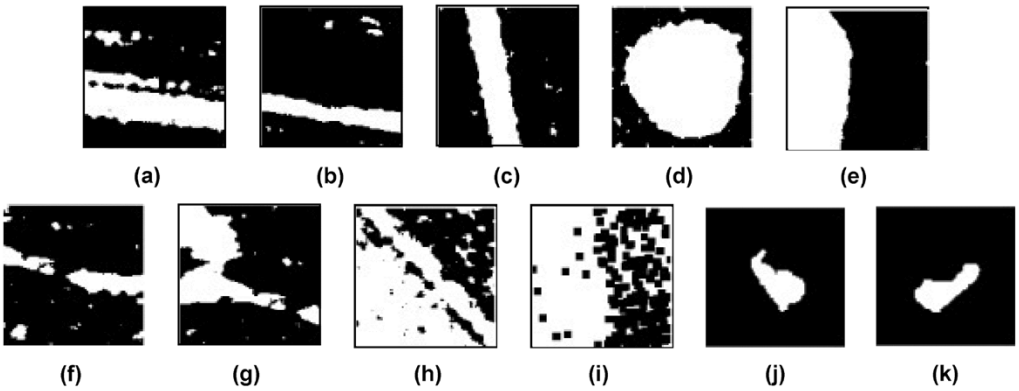

Fig. 18. ROI extracted by the traditional ROI extraction algorithm. (a) Paved road versus field. (b) Unpaved road versus field. (c) Paved road versus grass. (d) Lake versus field. (e) Lake versus grass. (f) River versus field. (g) River versus field. (h) Field versus grass. (i) Field versus grass. (j) T72 tank. (k) T72 tank.

TABLE VII
FITNESS VALUES OF THE EXTRACTED ROIS AND THE CORRESPONDING THRESHOLD VALUES

	Figure 18 (a)	Figure 18 (b)	Figure 18 (c)	Figure 18 (d)
Goodness	0.68 (PFIM15)	0.72 (PFIM15)	0.82 (PFIM15)	0.92 (PFIM3)
Threshold	31	32	31	32.9
	Figure 18 (e)	Figure 18 (f)	Figure 18 (g)	Figure 18 (h)
Goodness	0.95 (PFIM15)	0.62 (PFIM15)	0.84 (PFIM15)	0.82 (PFIM3)
Threshold	30	45	45.5	72.3
	Figure 18 (i)	Figure 18 (j)	Figure 18 (k)	
Goodness	0.71 (PFIM9)	0.86 (PFIM2)	0.86 (PFIM2)	
Threshold	196	99.6	82.2	

TABLE VIII
AVERAGE RUNNING TIME (IN SECONDS) OF THE COMPOSITE OPERATORS AND THE TRADITIONAL ROI EXTRACTION ALGORITHM

	Road	Lake	River	Field	Tank
Composite operator from Smart GP	2.6	1	19	12	2
Composite operator from Normal GP	5	15	33	8	3
Simple ROI exaction algorithm	12.3	10	50.5	32	23

B. Comparison Between Composite and Primitive Feature Images

In order to show the effectiveness of composite operators in ROI extraction, a traditional ROI extraction algorithm is applied to composite and primitive feature images. The traditional ROI extraction algorithm uses a threshold value to segment an image into foreground and background. The region consisting of pixels with value greater than the threshold value is called the bright region and its complement is called the dark region. When primitive feature images are used, the threshold value determines the performance of the traditional ROI extraction algorithm. For a particular threshold value, if the bright region has a higher goodness than the dark region, the bright region is the foreground. Otherwise, the dark region is the foreground. The foreground is the ROI extracted. To find the best threshold value, every possible threshold value is tried by the algorithm and its performance is recorded. In the previous experiments when composite feature images are used, the threshold value is always 0. There is a significant difference between constant and variable threshold values. A variable threshold value is a parameter of the algorithm and it makes the algorithm complicated, since one has to fine-tune it for each image. In this regard, training loses its meaning. As it is well known, algorithms with parameters that need manual tuning are not effective algorithms. With composite feature images, a parameter becomes a constant. The ROI extracted from composite feature images is compared with the ROI extracted from primitive feature images when the best threshold value is used. In order to show the effectiveness of composite features over that of primitive features, the traditional ROI extraction algorithm is applied to all the 16 primitive feature images and the best result from the 16 primitive feature images is reported.

Fig. 18 shows the ROI's extracted by this traditional algorithm when the best threshold value is used. The goodness of the ROI's, their corresponding threshold values and the primitive feature image from which the result is obtained are shown in Table VII. It can be seen that the composite operators learned by GP are more effective in ROI extraction and its performance is better than the best performance of the traditional ROI extraction algorithm on primitive feature images. Table VIII shows the average running time of the composite operators and the traditional ROI extraction algorithm in extracting ROI's from training and testing images. The time is measured in seconds. The composite operators are more efficient, since the traditional algorithm spends a lot of time to determine the best threshold value. For a testing image, the best threshold value found from the corresponding training image is not used, since training and testing images may have different best threshold values. In this regard, training has no meaning.

From the above comparison, it can be seen that composite feature images are more suitable for the object detection (segmentation) task, since in the composite feature images, objects are more different from the background than in primitive feature images or original images. This shows the effectiveness of composite operators learned by GP.

V. CONCLUSION

In this paper, we use genetic programming to evolve composite operators for object detection. To improve the efficiency of GP and address

its well-known code bloat problem, we design a new fitness function based on the minimum description length principle to take the size of a composite operator into the fitness evaluation process. We also design smart crossover and smart mutation to identify and prevent the effective components of composite operators from being disrupted. The new fitness function prevents composite operators from growing too large while at the same time imposes relatively less severe restrictions on the GP search. Our experimental results with real SAR images show that with MDL-based fitness function and smart search operators, GP can learn good composite operators more quickly, thus, improving the efficiency of GP. Compared to normal GP, the composite operators learned by smart GP have better performance on the training and testing images and have smaller sizes, reducing the computational expenses during testing.

REFERENCES

[1] B. Bhanu, D. E. Dudgeon, E. G. Zelnio, A. Rosenfeld, D. Casasent, and I. S. Reed, "Introduction to the special issue on automatic target detection and recognition," *IEEE Trans. Image Process.*, vol. 6, no. 1, pp. 1–6, Jan. 1997.

[2] J. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*. Cambridge, MA: MIT Press, 1994.

[3] W. Banzhaf, P. Nordin, R. Keller, and F. Francone, *Genetic Programming—An Introduction on the Automatic Evolution of Computer Programs and Its Application*. New York: Morgan Kaufmann, 1998.

[4] J. Rissanen, "A universal prior for integers and estimation by minimum description length," *Ann. Statist.*, vol. 11, no. 2, pp. 416–431, 1983.

[5] J. Quinlan and R. Rivest, "Inferring decision tree using the minimum description length principle," *Inform. Comput.*, vol. 80, pp. 227–248, 1989.

[6] Q. Gao, M. Li, and P. Vitanyi, "Applying MDL to learn best model granularity," *Artif. Intell.*, vol. 121, pp. 1–29, 2000.

[7] B. Bhanu and Y. Lin, "Object detection in multi-modal images using genetic programming," *Appl. Soft Comput. J.*, vol. 4, pp. 175–201, 2004.

[8] Y. Lin and B. Bhanu, "MDL-based genetic programming for object detection," in *Proc. IEEE Workshop on Learning in Computer Vision and Pattern Recognition*, Madison, WI, Jun. 2003.