Learning Features for Object Recognition

Yingqiang Lin and Bir Bhanu

Center for Research in Intelligent Systems
University of California, Riverside, CA, 92521, USA
{yqlin,bhanu}@vislab.ucr.edu

Abstract. Features represent the characteristics of objects and selecting or synthesizing effective composite features are the key factors to the performance of object recognition. In this paper, we propose a co-evolutionary genetic programming (CGP) approach to learn composite features for object recognition. 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. On the other hand, CGP can try a very large number of unconventional combinations and these unconventional combinations may yield exceptionally good results in some cases. Our experimental results with real synthetic aperture radar (SAR) images show that CGP can learn good composite features. We show results to distinguish objects from clutter and to distinguish objects that belong to several classes.

1 Introduction

In this paper, we apply genetic programming to synthesize 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 to estimate 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 to be recognized 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 and how to synthesize composite features useful to the recognition from primitive features? The answers to these questions are largely dependent on the intuitive instinct, knowledge, experience and the bias of human experts.

In this paper, co-evolutionary genetic programming (CGP) is employed to generate a composite operator vector whose elements are synthesized composite operators for object recognition. A composite operator is represented by a binary tree whose internal nodes represent the pre-specified primitive operators and leaf nodes represent the primitive features, it is a way of combining primitive features. With each element evolved by a sub-population of CGP, a composite operator vector is cooperatively evolved by all the sub-populations. By applying composite operators, corresponding to each sub-population, to the primitive features extracted from images, we obtain

E. Cantú-Paz et al. (Eds.): GECCO 2003, LNCS 2724, pp. 2227–2239, 2003.

composite feature vectors. These composite feature vectors are fed into a classifier for recognition. The primitive features are real numbers and designed by human experts based on the type of objects to be recognized. It is worth noting that the primitive operators and primitive features are decoupled from the CGP mechanism that generates composite operators. The users can tailor them to their own particular recognition task 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.

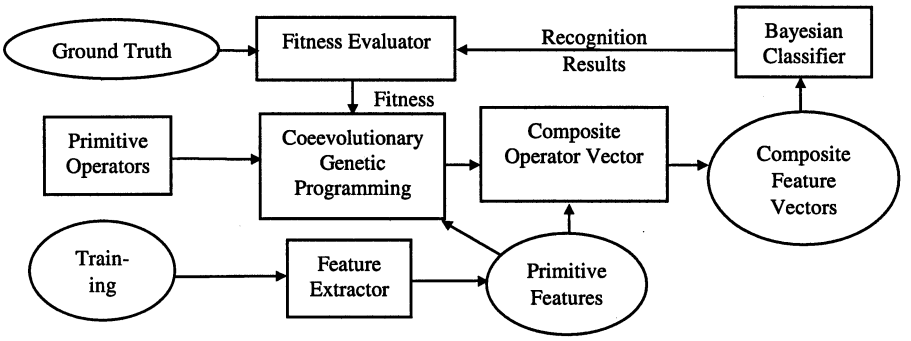
2 Motivation and Related Research

- **Motivation:** The recognition accuracy of an automatic object recognition system is determined by the quality of the feature set. Usually, it is the human experts who design the features to be used in recognition. Handcrafting a set of features requires human ingenuity and insight into the characteristics of the objects to be recognized and in general, it is a very time consuming and expensive process due to the large number of features available and the correlations among them. Thus, automatic synthesis of composite features useful to the recognition from simple primitive features becomes extremely important. The process of synthesizing composite features can often be dissected into some primitive operations on primitive features. However, the ways of combining primitive features are almost infinite and human experts, relying on their knowledge, rich experience and limited by their speed and bias, can try only a small number of conventional combinations. Co-evolutionary genetic programming, on the other hand, may try many unconventional combinations and in some cases these unconventional combinations yield exceptionally good results. Also, the inherent parallelism of CGP and the speed of computers allow much more combinations to be considered by CGP compared to that by human experts and this greatly enhances the chances of finding good composite features.

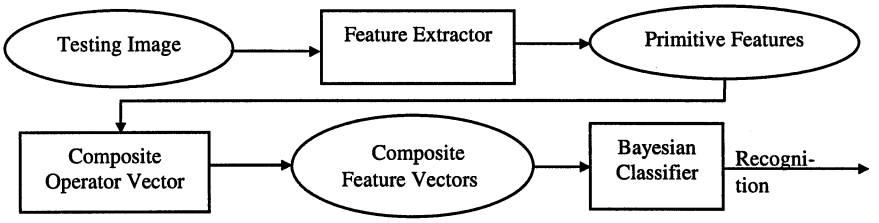
- **Related Research:** Genetic programming (GP) has been used in image processing, object detection and recognition. Poli et al. [1] use GP to develop image filters to enhance and detect features of interest or to build pixel-classification-based segmentation algorithms. Bhanu and Lin [2] use GP for object detection and ROI extraction. Howard et al. [3] apply GP for automatic detection of ships in low resolution SAR imagery. Roberts and Howard [4] use GP to develop automatic object detectors in infrared images. Stanhope and Daida [5] use GP for the generation of rules for target/clutter classification and rules for the identification of objects. Unlike the work of Stanhope and Daida [5], the primitive operators in this paper are not logical operators, but operators that work on real numbers. They use GP to evolve logical expressions and the final outcome of the logical expressions determines the type of object under consideration (for example, 1 means target and 0 means clutter); we use CGP to evolve composite feature vectors for a Bayesian classifier and each sub-population is responsible for evolving a specific composite feature in the composite feature vector. The classifier evolved by GP in their system can be viewed as a linear classifier, but the classifier evolved by CGP here is a Bayesian classifier determined by the composite feature vectors learned from training images.

3 Technical Approach

In our CGP-based approach, individuals are composite operators and all possible composite operators form the huge search space, leading to the extreme difficulty in finding good composite operators unless one has a smart search strategy. The system we developed is divided into training and testing parts, 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 Bayesian classifier is completely determined by the composite feature vectors learned from training images, so both the composite features and the classifier are learned by CGP.



(a) Training — Learning composite feature vectors and Bayesian classifier



(b) Testing — Applying learned composite feature vectors and Bayesian classifier to a test image

Fig. 1. System diagram for object recognition using co-evolutionary genetic programming

3.1 Design Considerations

- **The Set of Terminals:** The set of terminals used in this paper are 20 primitive features used in [6]. The first 10 of them are designed by MIT Lincoln lab to capture the particular characteristics of synthetic aperture radar (SAR) imagery and are found useful for object detection. The other 10 features are common features used widely in image processing and computer vision. The 20 features are: (1) standard deviation of image; (2) fractal dimension and (3) weight rank fill ratio of brightest scatterers; (4) blob mass; (5) blob diameter; (6) blob inertia; (7) maximum and (8) mean values of pixels within blob; (9) contrast brightness of blob; (10) count; (11) horizontal, (12) vertical, (13) major diagonal and (14) minor diagonal projections of blob; (15) maxi-

mum, (16) minimum and (17) mean distances of scatterers from their centroid; (18) moment μ_{20}, (19) moment μ_{02} and (20) moment μ_{22} of scatters.

- **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 1 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.

Table 1. 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).

- **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. The recognition rate of the classifier is the fitness of the composite operator vector. To evaluate a composite operator evolved in a sub-population (see Fig. 2), the composite operator is combined with the current best composite operators in other sub-populations to form a complete composite operator vector where composite operator from the *ith* sub-population occupies the *ith* position in the vector and defines the fitness of the vector as the fitness of the composite operator under evaluation. The fitness values of other composite operators in the vector are not affected. When sub-populations are initially generated, the composite operators in each sub-population are evaluated separately without being combined with composite operators from other sub-populations. After each generation, the composite operators in the first sub-population are evaluated first, then the composite operators in the second sub-population and so on.

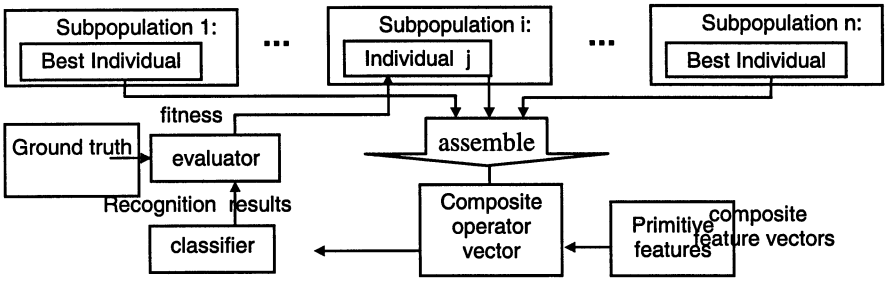


Fig. 2. Computation of fitness of *jth* composite operator of *ith* subpopulation

- **Parameters and Termination:** The key parameters are the number of sub-population 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 sub-population to form the learned composite operator vector to be used in testing.

3.2 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 sub-populations are randomly generated. Although sub-populations are cooperatively evolved (the fitness of a composite operator in a sub-population is not solely determined by itself, but affected by the composite operators from other sub-populations), selection is performed only on composite operators within a sub-population and the crossover is not allowed between two composite operators from different sub-populations.

- **Selection:** The selection operation involves selecting composite operators from the current sub-population. In this paper, we use tournament selection. 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. If the size of one offspring exceeds the maximum size, the crossover is performed again until the sizes of both offspring are within the limit.

- **Mutation:** To avoid premature convergence, mutation is introduced to randomly change the structure of some composite operators to maintain the diversity of sub-populations. Candidates for mutation are randomly selected and the mutated composite operators replace the old ones in the sub-population. 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 selected two subtrees of the composite operator and swap them. Of course, neither of the two sub-trees can be the sub-tree of the other.

3.3 Generational Co-evolutionary Genetic Programming

Generational co-evolutionary genetic programming is used to evolve composite operators. The GP operations are applied in the order of crossover, mutation and selection. Firstly, two composite operators are selected on the basis of their fitness values for crossover. The two offspring from crossover are kept aside and won't participate in the following crossover operations on the current sub-population. The above process is repeated until the crossover rate is met. Then, mutation is applied to the composite operators in the current sub-population and the offspring from crossover. Finally, selection is applied to select some composite operators from the current sub-population and combine them with the offspring from crossover to get a new sub-population of the same size as the old one. In addition, we adopt an elitism replacement method that keeps the best composite operator from generation to generation.

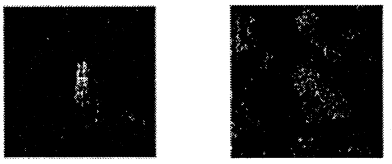
- **Generational Co-evolutionary Genetic Programming:**

0. randomly generate N sub-populations of size M and evaluate each composite operator in each sub-population individually.
1. for gen = 1 to generation_num do
2. for i =1 to N do
3. keep the best composite operator in sub-population 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 sub-population 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 sub-population, combine C_j with those N-1 best composite operators to form a composite operator vecter where composite operator from kth sub-population occupy the kth position in the vector (k=1, ..., 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. 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 composite operator vector consisting of the best composite operators from corresponding sub-populations and evaluate it. If its fitness is above the fitness threshold, goto 10.
endfor // loop 2
endfor // loop 1
10. select the best composite operator from each sub-population to form the learned composite operator vector and output it.

4 Experiments

Various experiments are performed to test the efficacy of CGP in generating composite features for object recognition. In this paper, we show some selected examples. All the images used in experiments are real synthetic aperture radar (SAR) images and they are divided into training and testing images. 20 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 to be noted that the ground truth is used only during training. The parameters of CGP used throughout the experiments are sub-population size (50), number of generations (50), fitness threshold (1.0), crossover rate (0.6) and mutation rate (0.05). The maximum size of composite operators is 10 in experiment 1 and 20 in experiment 2. The constant real number c stored in some primitive operators is from –20 to 20. 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

- **Experiment 1 – Distinguish Object from Clutter:** From MSTAR public real SAR images, we generate 1048 SAR images containing objects and 1048 SAR images containing natural clutter. These images have size 120×120 and are called object images and clutter images, respectively. An example object image and clutter image are shown in Fig. 3, where white spots indicate scatterers with high magnitude. 300 object images and 300 clutter images are randomly selected as training images and the rest are used in testing.



(a) A typical object image

(b) A typical natural clutter image

Fig. 3. Example target and clutter SAR images

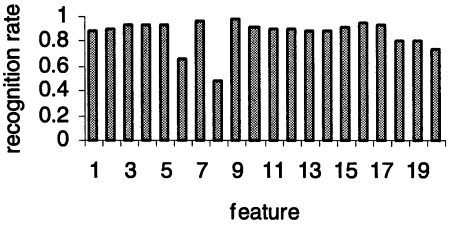


Fig. 4. Recognition rates of 20 primitive features

First, the efficacy of each primitive feature in discriminating the object from clutter is examined. Each primitive feature from 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. 4. 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 μ_{20} (18), μ_{02} (19) and μ_{22} (20) of scatters) are used by CGP; 10 common features (primitive features 11 to 20) not specifically designed to deal with SAR images are used by CGP during synthesis. The number of sub-populations is 3, which

means the dimension of the composite feature vector is 3. The results are shown in Fig. 5, where the horizontal coordinates are the number of primitive features used in synthesis and the vertical coordinates are the recognition rate. 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 all ten runs. Then the number of sub-population is increased from 3 to 5. The same 2, 5 and 10 primitive features are used as building blocks by CGP to evolve composite features. The experimental results are shown in Fig. 6. Table 2 shows the maximum and minimum recognition rates in these experiments, where tr and te mean training and testing and max and min stand for the maximum and minimum recognition rates.

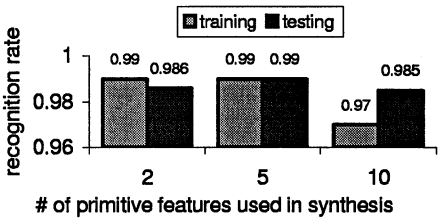


Fig. 5. Experimental results with 3 sub-populations

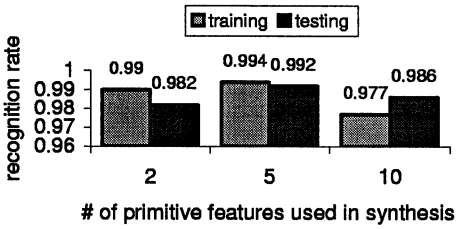
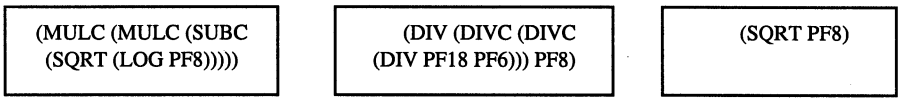


Fig. 6. Experimental results with 5 sub-populations



(a) Composite operator 1 (b) Composite operator 2 (c) Composite operator

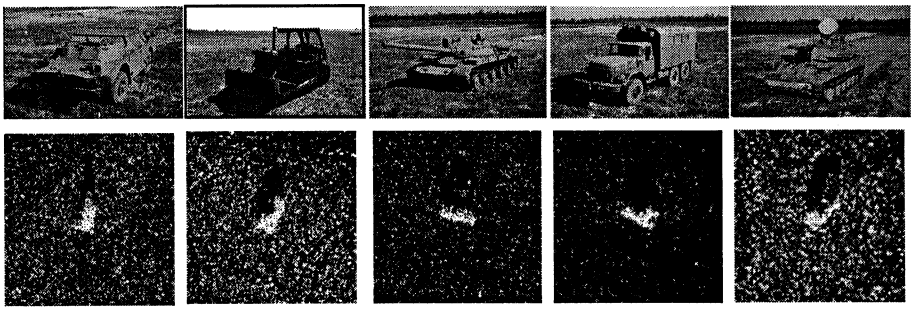
Fig. 7. Composite operator vector learned by CGP

Table 2. The maximum and minimum recognition rate

	3 sub-populations						5 sub-populations					
	2		5		10		2		5		10	
	Tr	Te	Tr	Te	Tr	Te	Tr	Te	Tr	Te	Tr	Te
Max	0.992	0.991	0.995	0.995	0.978	0.989	0.992	0.984	0.995	0.995	0.983	0.992
Min	0.988	0.979	0.978	0.974	0.965	0.979	0.988	0.98	0.993	0.987	0.972	0.979

From Figs. 5 and 6, it is obvious that composite feature vectors synthesized by CGP are very effective in distinguishing object from clutter. They are much better than the primitive features from which they are built. Actually, if both features 6 and 8 jointly form 2-dimensional primitive feature vectors for recognition, the recognition rate is 0.668; if features 6, 8, 18, 19, and 20 jointly form 5-dimensional primitive feature vectors, the recognition rate is 0.947; if all the last 10 primitive features are used, the recognition rate is 0.978. The average recognition rates of composite feature vectors are better than all the above results. Figure 7 shows the composite operator vector evolved by CGP maintaining 3 sub-populations in the 6^{th} run when 5 primitive features are used, where PF*i* means the primitive feature *i* and so on.

• **Experiment 2 – Recognize Objects:** Five objects (BRDM2 truck, D7 bulldozer, T62 tank, ZIL131 truck and ZSU anti-aircraft gun) are used in the experiments. For each object, we collect 210 real SAR images under 15°-depression angle and various azimuth angles between 0° and 359° from MSTAR public data. Fig. 8 shows one optical and one SAR image of each object. From Fig. 8, 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 the object under various azimuth angles, 210 SAR images of each object are sorted in the ascending order of their corresponding 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 are used in testing.



(a) BRDM2 (b) D7 (c) T62 (d) ZIL (e) ZSU

Fig. 8. Five objects and their SAR images used for recognition

1) Discriminate three objects: CGP synthesizes composite features to recognize three objects: BRDM2, D7 and T62. First, the efficacy of each primitive feature in discriminating these three objects is examined. The results are shown in Fig. 9. Feature mean values of pixels within blob (8) is the best primitive feature with recognition 0.73. Three series of experiments are performed in which CGP maintains 3, 5 and 8 sub-populations to evolve 3, 5 and 8-dimensional composite features, respectively. The primitive features used in the experiments are all the 20 primitive features and the last 10 primitive features. The maximum size of the composite operators is 20. The experimental results are shown in Figs. 10, 11 and 12, 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 all ten runs. Table 3 shows the maximum and minimum recognition rates in these experiments.

From Figs. 10, 11 and 12, it is clear that composite feature vectors synthesized by CGP are effective in recognizing objects. They are much better than primitive features used by CGP to synthesize composite features. Actually, if all 20 primitive features are used jointly to form 20-dimensional primitive feature vectors for recognition, the recognition rate is 0.96. This result is a little bit better than the average performance shown in Fig 10 (0.94), but the dimension of the feature vector is 20. However, the dimension of composite feature vectors in Figs 10 and 11 are just 3 and 5 respectively. If we increase the dimension of composite feature vector from 3 to 5 and 8, CGP re-

sults are better. If the last 10 primitive features are used, the recognition rate is 0.81. From these results, we can see that the effectiveness of the primitive features has an important impact on the effectiveness of the composite features synthesized by CGP. With effective primitive features in hand, CGP will synthesize better composite features. Fig. 13 shows the composite operator vector evolved by CGP maintaining 5 sub-populations in the 10^{th} run when all 20 primitive features are used. The size of the first and second composite operators is 20. The size of the third composite operator is 9 and the size of the last one is 15. The fourth composite operator is very simple, just selects the primitive feature 11. The primitive features used by the synthesized 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 directly form a 13-dimensional primitive feature vector for recognition, the recognition rate is 0.96.

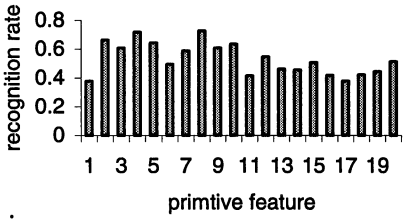


Fig. 9. Recognition rates of 20 primitive features

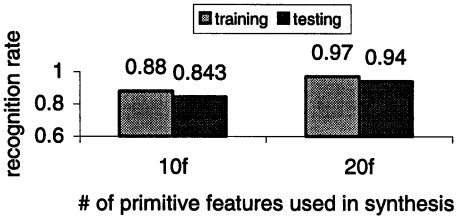


Fig. 10. Recognition rate with 3 sub-populations

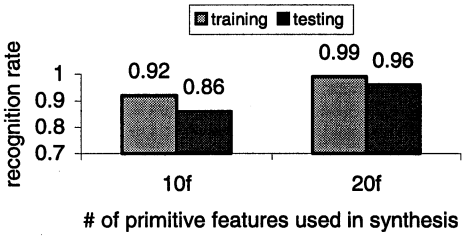


Fig. 11. Recognition rate with 5 sub-populations

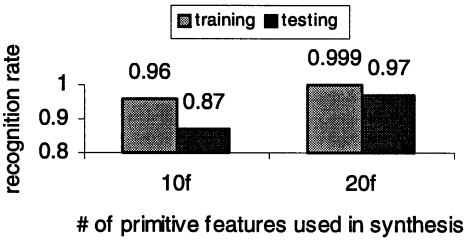


Fig. 12. Recognition rate with 8 sub-populations

(DIV (MULC (SUB (SUB (DIVC (SQRT PF6)) (MULC (SUB PF18 (MULC (SUB PF18 (SQRT PF4)))))) (SQRT PF6))) (MIN2 PF12 PF19))

(a) Composite operator 1

(DIV (MULC (ADD (ADDC (MULC (MUL (MIN2 (ADDC (DIV PF20 PF4)) PF14) PF3))) (LOG (ADDC (DIV PF20 PF4))))) (DIVC PF4))

(b) Composite operator 2

(DIV (MIN2 (SUBC (SUBC PF11)) (MAX2 PF7 PF8)) PF8)

(c) Composite operator 3

(PF11)

(d) Composite operator 4

(LOG (ADDC (LOG (DIV (SUBC (LOG (DIV (SUBC (LOG PF5)) (SUBC PF5)))) (MUL PF2 PF5)))))

(e) Composite operator 5

Fig. 13. Composite operator vector learned by CGP for 5 sub-populations

Table 3. The maximum and minimum recognition rate

	3 sub-populations				5 sub-populations				8 sub-populations			
	10f		20f		10f		20f		10f		20f	
	Tr	Te	Tr	Te	Tr	Te	Tr	Te	Tr	Te	Tr	Te
Max	0.91	0.86	0.98	0.97	0.94	0.88	0.995	0.97	0.98	0.9	1.0	0.98
Min	0.85	0.83	0.95	0.92	0.91	0.84	0.98	0.94	0.95	0.85	0.995	0.95

2) Discriminate Five Objects: With more objects added, the recognition becomes more difficult. This can be seen from Fig. 14, which shows the efficacy of each primitive feature in discriminating these five objects. Feature blob mass (4) is the best primitive feature with recognition 0.49. If all 20 primitive features are used jointly to form 20-dimensional primitive feature vectors for recognition, the recognition rate is 0.81; if only the last 10 primitive features are used, the recognition rate is 0.62. This number is much lower, since the last 10 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 5 and 8 sub-populations to evolve 5 and 8-dimensional composite features for recognition. The primitive features used in the experiments are all the 20 primitive features and the last 10 primitive features. The maximum size of composite operators is 20. The experimental results are shown in Fig. 15. The left two bins in columns 10f and 20f correspond to 5 sub-populations and the right two bins correspond to 8 sub-populations. The bins showing the training results are to the left of those showing testing results. The numbers above the bins are the average recognition rates over all ten runs. Table 4 shows the maximum and minimum recognition rates in these experiments.

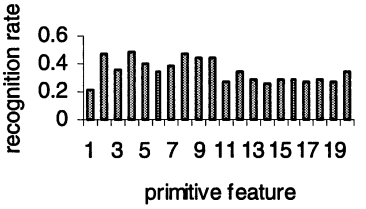


Fig. 14. Recognition rates of 20 primitive features

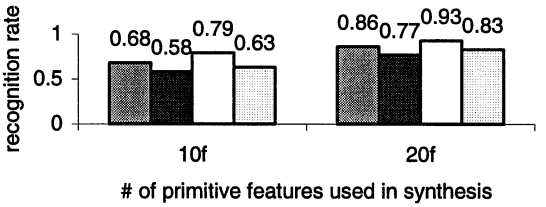


Fig. 15. Recognition rate with 5 (left two bins) and 8 (right two bins) sub-populations

Table 4. The maximum and minimum recognition rate

	5 sub-population				8 sub-population			
	10f		20f		10f		20f	
	Tr	Te	Tr	Te	Tr	Te	Tr	Te
Max	0.71	0.63	0.88	0.8	0.80	0.65	0.94	0.85
Min	0.65	0.55	0.83	0.73	0.75	0.62	0.91	0.80

From Fig.15, 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 10 (0.62) primitive features from which the composite features are built. When the dimension of the composite feature vector is 5, the recognition is not satisfactory when using just 10 common features as building blocks. Also, when the dimension is 5, the average performance is a little bit worse than using all 20 or 10

primitive features, but the dimension of composite feature vector is just one-fourth or half of the number of primitive features, saving a lot of computational burden in recognition. When all 20 primitive features are used and CGP has 8 sub-populations, the composite operators in the best composite operator vector evolved have sizes 19, 1, 16, 19, 15, 7, 16 and 6, respectively. 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 directly form a 16-dimensional primitive feature vector for recognition, the recognition rate is 0.80, which is lower than the average performance of the composite feature vector shown in Fig. 15.

- **Discussion:** The above experiments show that CGP is a viable tool to synthesize effective composite features from primitive features for object recognition and the learned composite features outperform the primitive features or any combination of primitive features from which they are evolved. The effectiveness of composite features learned by CGP is dependent on the effectiveness of primitive features. The usefulness of CGP is that it can evolve composite features that are more effective than the primitive ones upon which they are evolved. To achieve the same recognition rate, the number of composite features needed is smaller than the number of primitive features needed (one-fourth or half), thus, reducing the computational expenses during run-time recognition.

5 Conclusions

In this paper, CGP is used to synthesize 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. To achieve the same recognition performance of primitive features, fewer composite features are needed and this reduces the computational burden during recognition. However, primitive features still have a significant impact on the effectiveness of the evolved composite features. How to let CGP evolve effective composite features using general primitive features is the focus of our future research.

Acknowledgment. This research is supported by the grant F33615-99-C-1440. The contents of the information do not necessarily reflect the position or policy of the U. S. government.

References

1. R. Poli, "Genetic programming for feature detection and image segmentation," in *Evolutionary Computation*, T.C. Forgarty (Ed.), pp. 110–125, 1996.
2. B. Bhanu and Y. Lin, "Learning composite operators for object detection," *Proc. Genetic and Evolutionary Computation Conference*, pp. 1003–1010, July, 2002.
3. D. Howard, S.C. Roberts, and R. Brankin, "Target detection in SAR imagery by genetic programming," *Advances in Engg. Software*, vol. 30, no. 5, pp. 303–311, May 1999.

4. S.C. Roberts and D. Howard, "Evolution of vehicle detectors for infrared line scan imagery," *Proc. Evolutionary Image Analysis, Signal Processing and Telecommunications, First European Workshops*, pp. 110–125, Springer-Verlag, 1999.
5. S.A. Stanhope and J. M. Daida, "Genetic programming for automatic target classification and recognition in synthetic aperture radar imagery," *Proc. Conference. Evolutionary Programming VII*, pp. 735–744, 1998.
6. B. Bhanu and Y. Lin, "Genetic algorithm based feature selection for target detection in SAR images," *Image and Vision Computing*, 2003.